School of Computer Science and Statistics
Trinity College Dublin

AI Approach to Ranking of Search Results and Query Auto-Completions for Enterprise Search

Author:

Colin DALY

Supervisor:

Dr. Lucy HEDERMAN

A Thesis Submitted in Partial Fulfilment of the Requirements
for the Degree of
Doctor Of Philosophy
at
Trinity College Dublin, The University of Dublin

2026

Declaration

I declare that this thesis has not been submitted as an exercise for a degree at this or any other university and it is entirely my own work. I confirm that the work has been completed in full compliance with relevant university policies, including the Academic Integrity policy, the Trinity Policy on Good Research Practice, and the guidance on AI and GenAI in Teaching, Learning, Assessment and Research.

I agree to deposit this thesis in the University's open access institutional repository or allow the Library to do so on my behalf, subject to Irish Copyright Legislation and Trinity College Library conditions of use and acknowledgement.

I consent to the examiner retaining a copy of the thesis beyond the examining period, should they so wish (EU GDPR May 2018).

Signed:_____ Date:_____

Colin Daly

Acknowledgements

First and foremost, I owe immense thanks to my supervisor, Dr. Lucy Hederman. She stepped in at a critical juncture when my research faced significant challenges. Her guidance, unwavering support, and patience were indispensable to completing this marathon project.

This research was conducted with the financial support of Science Foundation Ireland under Grant Agreement No. 13/RC/2106 P2 at the ADAPT SFI Research Centre at Trinity College Dublin. ADAPT, the SFI Research Centre for AI-Driven Digital Content Technology, is funded by Science Foundation Ireland through the SFI Research Centres Programme. By gathering high quality researchers from around the world, I have had the opportunity to work with and learn from many wonderful people.

I also extend my appreciation to Mr. Patrick Magee, Director, IT Services, Trinity College Dublin, who, as industry partner, generously approved the partial funding for this research initiative, and whose backing signals the department's forward-looking commitment to strengthening its capabilities in AI and related technologies.

I would like to express my sincere thanks to Assoc. Prof. David Lillis (University College Dublin) and Prof. Gareth Jones (Dublin City University) for serving as my external examiners and for their insightful, constructive, and highly valuable feedback on this thesis.

This thesis is dedicated to the memory of my parents, Richard and Mary Daly, who both passed away during the course of my research.

Finally, a special word of thanks to Dr. Padma Naik, who has not only guided me through every step of the PhD process, but also someone I am very proud to call a dear friend.

My thanks to you all.

Abstract

Enterprise Search (ES) is an organisation's use of Information Retrieval (IR) technologies to find content from multiple databases (e.g. corporate directories), intranet document repositories and the organisation's web pages. A consequence of the ubiquity of commercial Web Search (WS) is that users have high expectations when interacting with their organisation's ES service. As with WS, ranking is also a major challenge for ES deployments.

In many organisations, the ES service is said to be 'relevance-blind'. Users or employees cannot find the information they seek within an acceptable time because of poor ranking in the search results or inappropriate 'wording' of the initial query. ES is an area of enormous importance for businesses yet has attracted relatively little academic interest, partly due to researchers' difficulties in gaining access to corporate intranets and datasets.

Two methods for improving ranking are Learning to Rank (LTR) and Language Modelling (LM). LTR is the application of supervised machine learning techniques for training a model (with labelled data) to provide the best ranking order of documents for a given query item. LM is a foundational component of Natural Language Processing (NLP) and, when applied to ranking tasks, can learn text features that bridge the gap between query and document vocabulary (semantic matching).

LTR with LM also enables the discovery of the best candidates for Query Auto-Complete (QAC). QAC for ES can inform new staff members about the range of selections available to them and assist in narrowing that selection even before the user has finished typing a query. Query suggestions can steer searchers to use the appropriate organisational jargon/terminology and avoid submitting queries that produce no results.

This thesis investigates whether LTR and LM can significantly enhance the ranking of both search results and query auto-completions in ES. Ranking models are trained and evaluated on real-world ES data from a large third-level academic institution.

A custom Language Model, built using LLM techniques, detects enterprise-specific jargon and is integrated as a feature in the ranking pipeline. LTR is used to weigh feature contributions and create ranking models for a live Enterprise Search service. Offline experiments and online A/B tests show improvements in Mean Reciprocal Rank (MRR) and normalized Discounted Cumulative Gain (nDCG) scores, with up to a 15.3% improvement in MRR for QAC and an nDCG@5 gain of 5.85% for search results.

The overarching contribution is a bridge of the gap between academic AI ranking models and real-world Enterprise Search implementations. Additionally, an LTR-formatted ES ranking dataset is generated and made publicly available to support reproducibility and future research.

Contents

Declaration i

Acknowledgements ii

Abstract iii

Table Of Contents iv

List of Tables vi

List of Figures vi

List of Listings vii

Abbreviations ix

1 INTRODUCTION 1
- 1.1 Motivation and Problem Statement . . . 1
- 1.2 Research Question . . . 7
- 1.3 Research Objectives . . . 7
- 1.4 Research Methodology . . . 8
- 1.5 Research Contributions . . . 9
- 1.6 Thesis Outline . . . 10

2 LITERATURE SURVEY 11
- 2.1 Enterprise Search . . . 11
- 2.2 Ranking of Search Results . . . 18
- 2.3 Ranking of Query Auto-Completions . . . 36
- 2.4 Learning to Rank Method . . . 44
- 2.5 Language Modelling Method . . . 58
- 2.6 The EU AI Act . . . 65
- 2.7 Summary of Gaps in the Literature . . . 65

3 RANKING OF SEARCH RESULTS 67
- 3.1 Introduction . . . 67
- 3.2 Traditional Ranking . . . 69
- 3.3 Other Hand-Crafted Ranking functions . . . 75
- 3.4 Learning to Rank . . . 79
- 3.5 Language Modelling for ES Jargon . . . 90
- 3.6 Experiments . . . 92
- 3.7 EXP-1: Search Results Ranking Approach Comparison . . . 92
- 3.8 EXP-2: Search Results Ablation Study . . . 103
- 3.9 EXP-3: Correlation using CTR and human judgements . . . 109
- 3.10 EXP-4: Inter-Annotator Agreement . . . 114
- 3.11 EXP-5: JARGES Ranking and Recall with LM-generated Synonyms . 118
- 3.12 Validation . . . 124
- 3.13 Conclusions . . . 129

4 RANKING OF QUERY AUTO-COMPLETIONS 131
4.1 Introduction . . . 131
4.2 Traditional Ranking . . . 132
4.3 Other Hand-Crafted Ranking functions . . . 134
4.4 Learning to Rank . . . 136
4.5 Language Modelling for ES Jargon . . . 142
4.6 Experiments . . . 145
4.7 EXP-6: QAC Ranking Approach Comparison . . . 145
4.8 EXP-7: QAC Ablation Study . . . 150
4.9 EXP-8: QACES Online A/B Test . . . 154
4.10 Validation . . . 158
4.11 Conclusions . . . 160

5 DISCUSSION 163
5.1 Revisiting Enterprise Search . . . 163
5.2 Using The University's Dataset . . . 164
5.3 Implications of Experimental Results . . . 165
5.4 Methodological Implications . . . 169
5.5 Position and Organisational Bias . . . 171
5.6 Implementation Challenges and Barriers . . . 172
5.7 Generalisability and Domain Transfer . . . 174
5.8 Scope Limitations . . . 176
5.9 Future Directions and Research Opportunities . . . 178
5.10 Summary . . . 179

6 CONCLUSION 181
6.1 Research Objectives Revisited . . . 181
6.2 Research Questions Revisited . . . 182
6.3 Research Contributions Revisited . . . 183
6.4 Publications . . . 185
6.5 Limitations . . . 186
6.6 Future Directions . . . 187
6.7 Final Comment . . . 187

References 209

Appendices 210

A ES and WS DIFFERENCES 210

B Paper IV 213

C Research Ethics Approval 220

List of Tables

1.1 Inputs, Outputs and Metrics for Ranking Applications 5
1.2 Problem Statement mapping to RQ 10
1.3 GitHub Repositories 10
2.1 Characteristics of popular QAC datasets 40
2.2 Human Relevance Judgements vs Click-through 50
2.3 Popular datasets and corpora for LTR 52
2.4 Collections pay-per-use 52
2.5 Query Independent Features 53
2.7 Semantic matching vs Relevance matching. 62
3.1 Average Number of Words per Query. 83
3.2 Distribution of human relevance judgements in`ENTRP-SRCH`. 87
3.3 Properties of popular LTR & `ENTRP-SRCH` datasets 89
3.4 Query Clustering Hyperparameters 101
3.5 Evaluation scores for Search Results 102
3.6 LTR hyperparameter settings for search results 105
3.7 Comparison of ranking performance. 113
3.8 Inter-Annotator Agreement. 118
3.9 Examples of jargon terms detected by JARGES 122
3.10 Hyperparameters for JARGES LTR 123
3.11 A/B test results for JARGES 124
3.12 RoBERTa decoded jargon terms via contextual synonyms 126
4.1 Top 10 most popular query terms WS and ES 134
4.2 Extracting Terms from URL or Shared Drive Path 136
4.3 QAC logging parameters 138
4.4 Hyperparameter settings 141
4.5 Comparison of top synonyms for two examples of 'fat head' queries . 143
4.6 LTR weightings for features calculated using the RankLib framework. 148
4.7 Traditional versus AI approach to QAC ranking 150
4.8 Offline MRR scores for ES and WS ranking models 150
4.9 A/B test results for ranking models 157
5.1 JARGES versus QACES 171
6.1 Research questions vis-à-vis initial problem, question & objective . . . 183
6.2 Research contributions vis-à-vis problem, question & objective 186

List of Figures

1.1 Enterprise Search: Typical components 2
1.2 The Search Process as a Loop 3
1.3 Query Auto-Completion for the prefix 'machine'. 5
2.1 Query Re-Ranking 20
2.2 Hierarchical Structure 24
2.3 Website Recency 27
2.4 Query Auto-Completion as a re-ranking problem. 39
2.5 Search Demand Curves comparing ES and WS query history. 41
2.6 Google Scholar "Learning to Rank" Publication history 45

2.7 The architecture of Solr's implementation of LTR. 47
2.8 LTR General Framework 48
2.9 How the concepts of AI, ML and DL relate to each other. 60
3.1 Document Recency 72
3.2 Extracting field content from a webpage. 74
3.3 Search Demand Curve for Enterprise Search 82
3.4 TensorFlow word embedding visualisation 84
3.5 An extract of the schema given to judges. 86
3.6 A completed schema. 87
3.7 LETOR-formatted dataset for search results 89
3.8 Architecture of the JARGES algorithm 91
3.9 Comparison of Ranking Performance by Method. 103
3.10 LTR scores with Ablation Study 109
3.11 Strip Scatter Correlation Plot 112
3.12 Bar Chart Explicit versus Implicit Feedback 114
3.13 JARGES Inputs and Outputs 120
3.14 JARGES Sentence Example 121
3.15 LTR ranking model performance with and without JARGES 125
3.16 Impact of dataset size on ranking performance. 127
4.1 Query prefix produces a list of completion choices. 137
4.2 Extract of LTR formatted dataset for QAC. 140
4.3 TF alone versus all ranking features 147
4.4 Relative Term Frequencies for various fields for one prefix 152
4.5 RR representations for MRR calculation 152
4.6 Ablation study for features of QAC ranking models. 154
4.7 Load Balancer used for A/B testing 156
4.8 QAC validation of feature fields. 158

List of Listings

2.1 Feature Definition in Apache Solr. 54
3.1 LinkRank in practice. 73
3.2 URLlength as a ranking function. 77
3.3 Code used for detecting question-form queries in ES and WS datasets 83
3.4 View Count implementation in Apache Solr. 94
3.5 Extracting Most Viewed tuples from Apache HTTP server. 95
3.6 Recency implementation using cURL. 95
3.7 Recency implementation for LTR. 96
3.8 LinkRank Solr schema definition. 96
3.9 LinkRank LTR feature definition. 97
3.10 Boosting individual fields. 97
3.11 Field Boosting LTR feature definition. 98
3.12 Keyword boosting implementation. 98
3.13 Anchor text schema definition. 99
3.14 Anchor text Nutch definition. 99
3.15 Ranking by URL length implementation. 100
3.16 Extracting MFQ from Solr logs. 100

3.17 Enabling the LTR plugin for Apache Solr 106
3.18 LTR Feature Store for Apache Solr 106
3.19 LTR Model for Apache Solr 107
3.20 Uploading the LTR defintions to Apache Solr 107
3.21 Verification of LTR model functionality 108
4.1 Jargon Distance computation in Python. 144
4.2 Implementation of LTR weights to QAC in Apache Solr. 148
4.3 Retrieving and Verifying LTR weights. 159

Abbreviations

AI ..... Artificial Intelligence
AMT ..... Amazon Mechanical Turk
AP ..... Average Precision
BERT ..... Bidirectional Encoder Representations from Transformers
BM25 ..... Best Match 25
C-DSSM ..... Convolutional-Deep Structured Semantic Model
CSE ..... Centre of Search Excellence
CNN ..... Convolutional Neural Network
CTR ..... Clickthrough Rate
DCG ..... Discounted Cumulative Gain
DL ..... Deep Learning
DNN ..... Deep Neural Network
DSSM ..... Deep Structured Semantic Model
ELI5 ..... Explain like I'm 5
EU ..... European Union
ES ..... Enterprise Search
EXP ..... Experiment Number
GB ..... Gradient Boosting
GBDT ..... Gradient Boosted Decision Trees
GDPR ..... General Data Protection Regulation
GPT ..... Generative Pre-trained Transformer
HITS ..... Hypertext-Induced Topic Search
IAA ..... Inter-Annotator Agreement
ICC ..... Intra-class Correlation
IDF ..... Inverse Document Frequency
IR ..... Information Retrieval
JARGES ..... JARGon for Enterprise Search
JD ..... Jaccard Distance
LETOR ..... Learning to Rank (Microsoft format)
LLM ..... Large Language Model
LTR ..... Learning to Rank
LSTM ..... Long-Short Term Memory
MAP ..... Mean Average Precision
MFQ ..... Most Frequently Queried (QAC)
MKS ..... Minimum Keystroke Length
ML ..... Machine Learning
MORO ..... Multi-Objective Ranking Optimisation
MPC ..... Most Popular Completion (QAC)
MRL ..... Mean Recoverable Length
MRR ..... Mean Reciprocal Rank
MV ..... Most Viewed
NAS ..... Network Attached Storage
nDCG ..... Normalized Discounted Cumulative Gain
NIR ..... Neural Information Retrieval
NLP ..... Natural Language Processing
NLU ..... Natural Language Understanding

NLTK ..... Natural Language Toolkit
OBJ ..... (Research) Objective
OpenQA ..... Open Domain Question Answering
PA ..... Percentage Agreement
P-C pairs ..... Prefix Candidate pairs
PS ..... Problem Statement
QAC ..... Query Auto-Complete
QACES ..... Query Auto-Complete for Enterprise Search
Q-D pairs ..... Query Document pairs
RE ..... Ranking Entropy
RFECV ..... Recursive Feature Elimination with Cross-Validation
RNN ..... Recurrent Neural Network
RC ..... Research Contributions
RQ ..... Research Questions
RR ..... Reciprocal Rank
RTQE ..... Real Time Query Expansion
SDC ..... Search Demand Curve
SEO ..... Search Engine Optimisation
SERP ..... Search Engine Results Page
SGE ..... Search Generative Experience
TF ..... Term-Frequency
TF-IDF ..... Term Frequency-Inverse Document Frequency
TREC ..... Text Retrieval Conference
URL ..... Uniform Resource Locator
WAD ..... Web Accessibility Directive
WBS ..... Work Breakdown Structure
WCAG ..... Web Content Accessibility Guidelines
WS ..... Web Search
XFF ..... X-Forwarded-For

1

INTRODUCTION

1.1 Motivation and Problem Statement

A consequence of the pervasive use of commercial/internet Web Search (WS) is that users have high expectations with regard to the ranking of search results. Similarly, the ranking of query auto-completions should match users' intentions. Cleverley and Burnett refer to these elevated expectations as the 'Google Habitus' [1].

Searching for intellectual assets within an enterprise has traditionally been a less satisfactory experience. As far back as 2003, a paper entitled 'the high cost of not finding information' describes how productivity is lost when employees can't find the information they need on the intranet and have to resort to asking for help from colleagues [2]. In a 2011 study, managers in the US, UK, Germany and France say that their internal enterprise search service falls short of expectations and that over half (52%) of surveyed users "cannot find the information they seek within an acceptable amount of time, using their own enterprise search applications" [3]. A 2023 survey by Gartner found that 47% of workers struggle to find the information or data needed to effectively perform their jobs [4]. More recently, Microsoft reported that this figure has risen to 62% [5]. Taken together, these findings indicate that information fragmentation within organisations is worsening despite the proliferation of digital collaboration tools and content management systems. As enterprise data volumes continue to expand across heterogeneous repositories, intranets, and cloud services, employees are spending more time searching for information and less time applying it productively. This growing inefficiency underscores the increasing importance of effective Enterprise Search (ES) systems as a core enabler of organisational productivity and knowledge management. A key contributor to this underperformance is the difficulty ES systems face in ranking documents appropriately for a particular query [6].

In addition to ranking of search results, searchers may also experience difficulties identifying the appropriate vocabulary for effective query formulation [7]. For newcomers, this challenge is compounded by the prevalence of specialised jargon and terminology unique to an organisation. Although much research has been undertaken by commercial search engine providers, ranking performance on smaller and

more focused collections has, to date, attracted less attention [6]. This has led Kruschwitz to state that 'Search has become ubiquitous, but that does not mean that search has been solved' [6].

Enterprise Search (ES) is a service that provides unified and secure retrieval of organisational data. The content components may be internally and/or externally accessible. This may include confidential content such as committee papers, blueprints, and internal messaging boards, as well as published content such as corporate directories, marketing material or web pages intended for public viewing (Figure 1.1). An effective ES service can help users avoid navigating hierarchical menus and hyperlinks as it facilitates the search of assets within an organisation via a single window to the enterprise [8]. ES searches typically comprise queries for persons, usernames, course codes, tracking numbers, purchasing codes, intranet documents or any datum specific to the organisation. ES content is often imbued with organisational terminology/jargon (i.e., words, phrases, expressions, and idioms that are not universally familiar).

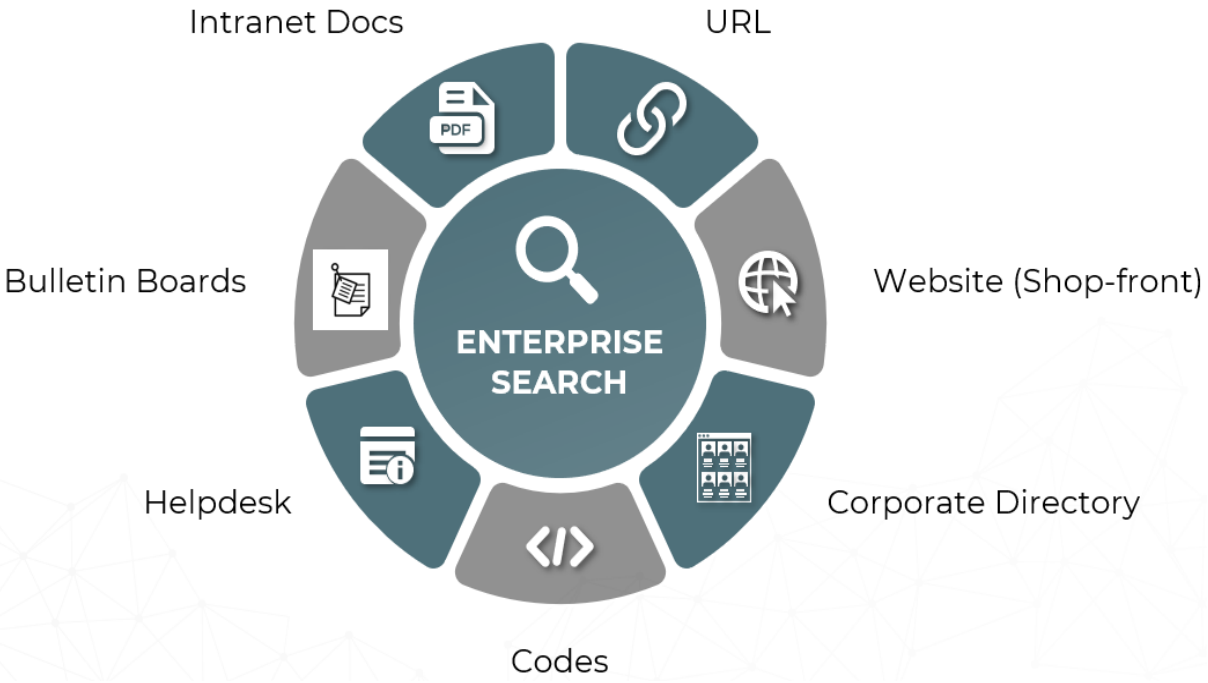


Figure 1.1: Typical Enterprise Search content showing both internal and external-facing components.

As with WS, ranking is also a major challenge for ES deployments [6,9,10]. Ranking is typically applied twice per search:

- The ranking of **Query Auto-Completions** (QAC) for a typed prefix during the Query Formulation phase.
- The ranking of **search results** for a submitted query as presented for the Results Inspection phase.

The search results and QAC ranking applications are complimentary links of a cycle, as shown in Figure 1.2. The query can be refined at the end of the cycle if the user's information need is unsatisfied. In this respect, the search process is often described as a loop [11]. QAC can reduce the number of iterations and help to 'shorten the loop' by enhancing the Query Formulation phase, even before a query is submitted.

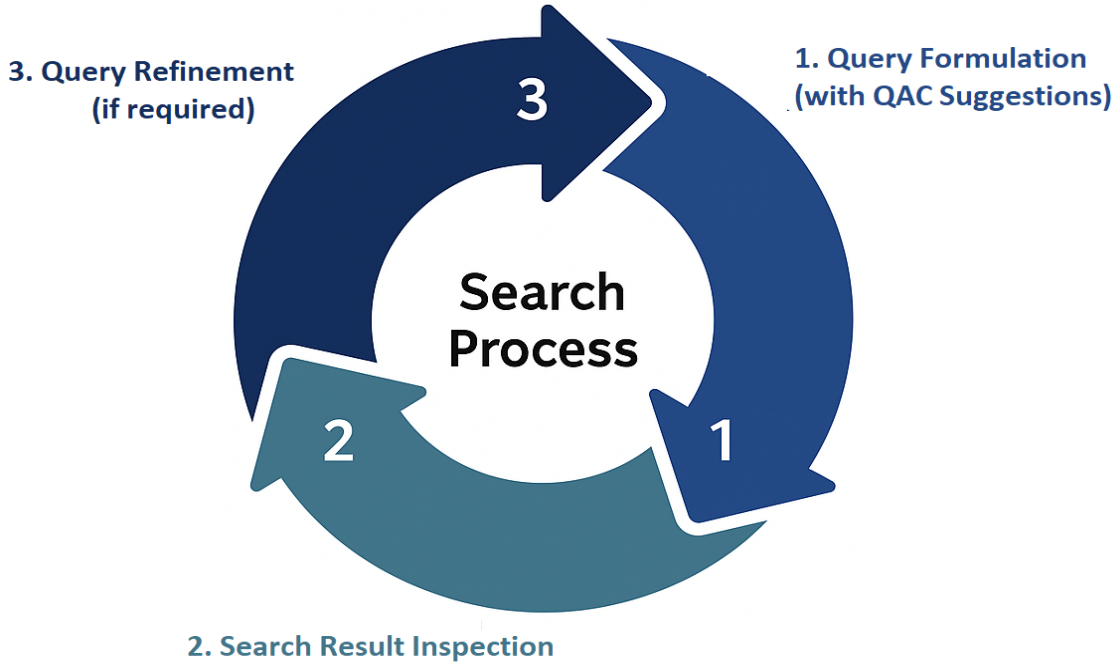


Figure 1.2: The Search Process as a 'loop' and QAC contribution to the Query Formulation link.

1.1.1 Ranking Of Search Results

Ranking refers to ordering results based on the most appropriate and useful information in response to a query. The most relevant web pages are presented at the top of the list. By convention, search engines display just ten results per page. According to a 2025 study, less than six per cent of Google WS users navigate further to the second page [12].

Early search engines relied on similarity matching, where documents containing the exact terms from the user query were retrieved and displayed. Documents were ranked using similarity functions, which produced a relevance score. The BM25 [13] similarity function emphasises important terms within a specific document while downplaying those common across the entire corpus and incorporates other factors like adjusting the relevance score based on the length of the document. Both the Apache Solr and Elasticsearch enterprise search engines are built on Apache Lucene, and since the release of Lucene version 6 in 2016, BM25 has been used as the default similarity function across the Lucene ecosystem [14, 15].

Apart from similarity matching, additional signals of usefulness and importance can boost the relevance score. For example, in WS, link analysis uses the Internet's

hyperlink structure to determine a page's importance. Pages linked to by many other pages, especially those that are important, are considered of greater utility and ranked higher. An example of link analysis is Google's PageRank [16, 17]. In ES, link analysis is less effective, as much of the organisation's content is created in documents without hyperlinks (or even publishing intent). For example, an MS Word document stored on a network share might not contain any hyperlinks.

The accepted metric for evaluating ranking performance for search results is normalised Discounted Cumulative Gain (nDCG). It measures the perceived utility of search results by a user performing sequential browsing of a ranked list [18]. nDCG captures how far an ordered set of search results is from the ideal (annotated) ranking order. The nDCG metric depends on the availability of an agreed 'ideal' ordering of documents, D, for a particular query, Q. This ideal ordering is typically recorded as Q-D pairs annotated by human judges. Precision is not used to directly measure ranking performance, as it merely records the proportion of relevant or non-relevant items in a list - it does not factor the order or degree of relevance. If human judges are unavailable, a click model can be used to record end-user Q-D pair preferences on search results. In WS, click models effectively infer search importance to a document for a given query [19]. Commercial WS providers have researched harnessing click-through data and using it as a surrogate for preference judgements. Its use in Enterprise Search (ES), however, has not been explored.

1.1.2 Ranking Of Query Auto-Completions

Google incorporated QAC into its search engine in 2008. Since then, QAC has gradually become a ubiquitous function in modern search systems. Given a prefix consisting of a number of characters entered into the search box, the user interface proposes alternative ways of extending the prefix to a full query candidate [20] as shown in Figure 1.3. Traditionally, QAC candidates have been ranked using Term Frequency (TF) and query history [21]. In addition to saving typing effort and time [22], QAC helps users formulate their query when they have an intent in mind but not a clear articulation (i.e. 'wording') for their organisation's specialised language and jargon/terminology. Since most organisations have their own local lexicon [23], QAC can play a role in bridging this gap [24].

As with search results, the ranking of QAC candidates in ES exhibits numerous distinctions from WS. For example, query auto-completion suggestions submitted to the engine in ES must produce one or more results. Suggestions that could produce 'no results found' must be avoided (as this would undermine users' confidence in the ES service). This differs from the domain of WS, where larger indices are unlikely to produce suggestions that produce no results. Moreover, searches on an ES service are usually undertaken by 'knowledge workers' attempting to carry out a specific work task. Any delay in retrieving the desired information has a monetary cost for the organisation.

The Mean Reciprocal Rank (MRR) is widely accepted as the principal metric for evaluating QAC ranking performance [20, 25]. Unlike nDCG, only the first 'correct' candidate is considered for MRR. Computing MRR depends on the availability of an agreed 'preferred' ordering of candidates, C, for a particular prefix, P. This ordering

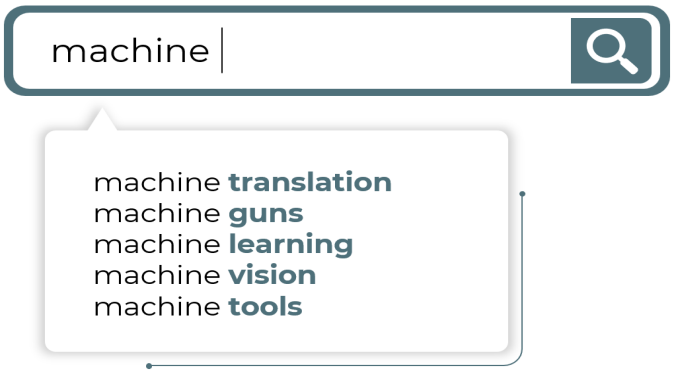


Figure 1.3: Query Auto-Completion for the prefix 'machine' with five candidates presented in the drop-down box. QAC not only saves effort and time for end-users but may also inspire new query possibilities.

is typically recorded as P-C pairs and is usually generated using a click model that records user choices.

Research for the ranking of QAC has been primarily based on datasets created for WS. AOL released a benchmark dataset designed to foster improvements in ranking query auto-completions [26]. The dataset is based on web content from 2006 and includes P-C pair preferences. Mirroring the situation with search results, there are few if any query autocompletion datasets specifically designed for enterprise search applications.

Table 1.1: Inputs, Outputs and Metrics for Ranking Applications

Ranking Application	Input	Output (Ranked List)	Dataset Pairs	Ranking Metric
Search Results	Submitted Query	Documents	Q-D	nDCG
Query Auto-Complete	Typed Prefix	Completion Candidates	P-C	MRR

Table 1.1 outlines the key inputs and outputs for ranking search results and Query Auto-Completions and the principal metrics used to measure ranking performance.

1.1.3 AI Methods

For modern WS engines, the traditional methods have been augmented and in some cases superseded by the following state-of-the-art AI methods, which improve ranking performance for both search results and QAC:

- The Learning To Rank machine learning method is considered "a standard workhorse" [27] for ranking.
- The Language Modelling method involves contextualising a corpus to enable synonym generation and next word prediction [28].

Learning To Rank (LTR) Adding ever-more ranking functions has the inherent problem that it is very difficult and eventually impossible to apportion the 'correct' weighting to each function. For example, the BM25 ranking function can be computed for individual fields (title, content, anchor text, etc.), raising the question of what importance should be assigned to each field. The importance of an anchor text field likely outweighs that of the content field, but by what margin? LTR answers these questions by apportioning a 'weight' to the contribution of each of the ranking functions (or 'features', as they are referred to when using LTR). LTR involves applying supervised machine learning techniques to train a model (with labelled Q-D or P-C data) to provide the best ranking order. The ranked items can be documents in the case of search results or candidates in the case of QAC. LTR differs from classic ML problem types such as classification or regression (which produce a single label or numeric value) because its output is an ordered sequence of items (i.e. a ranked list).

Microsoft and Yahoo have released benchmark datasets designed to foster improvements for LTR algorithms [29–31] in the domain of internet WS. Much of the current LTR research pertains to comparing and evaluating the numerous ranking algorithms and novel features using these datasets. A limitation of this research is that it is specific to web content and has not been applied to enterprise content [8]. A test collection or dataset based on Enterprise Search is hard to come by, as organisations are not inclined to open their intranet to public distribution, even for research purposes [1, 10].

Language Modelling (LM) Bridging the gap between a user's query intention and retrievable content is a more difficult problem than traditional *'similarity matching'*. The problem is sometimes referred to as 'vocabulary mismatch' [32]. LM has the ability to discover hidden meanings, structures, relationships, and features that inform ranking, thus bridging the gap between query and document vocabulary (including jargon/terminology) by interpreting user intent and returning content that matches that intent. Increased retrievable content improves recall and is sometimes called *'semantic matching'*. Google pithily refers to the task as matching 'things, not strings' [33]).

LM is a foundational component of Natural Language Processing (NLP) and specialises in tasks such as synonym generation or auto-completion. Advances in language modelling, particularly through models like word2vec [34], RoBERTa [35], GPT [36], and others, have significantly improved the capabilities of LM/NLP systems, leading to a better contextualisation and understanding of language in an ES corpus.

Little research has been undertaken in the ES domain to investigate or apply the LTR and LM state-of-the-art AI methods to the search results and QAC applications. This leads us to three problem statements:

- **PS-1**: In the field of Enterprise Search, users often see the relevance ranking of search results as poor, especially compared to commercial search engines.
- **PS-2**: A mismatch between publisher and user intent is often encountered in

ES. This leads to documents not being appropriately ranked or recalled for a given query.

- **PS-3**: Users may struggle to formulate their query when they have an intent in mind but not a clear articulation or 'wording' for their organisation's specialised jargon/terminology.

1.2 Research Question

The research question can be broadly framed in a single overarching sentence:

> *To what extent can Learning to Rank and Language Modelling methods, tailored to enterprise content and information needs, boost traditional ES ranking performance for both search results and query auto-completions?*

More specifically, the above can be trisected into the following constituent questions:

- **RQ-1**: Compared to the traditional methods, to what extent can Learning to Rank improve relevance matching and ranking performance of search results for Enterprise Search? This addresses PS-1.
- **RQ-2**: To what extent can Language Modelling address 'vocabulary mismatch' by improving semantic matching for Enterprise Search? This addresses PS-2.
- **RQ-3**: To what extent can LTR and LM methods be used to suggest and improve the ranking of candidates generated for Query Auto-Complete for an organisation's specialised jargon/terminology? This addresses PS-3.

1.3 Research Objectives

Research objectives that arise from the research questions are:

- **OBJ-1**: Develop and evaluate a Learning to Rank model for ES search results and compare it with the effectiveness of traditional ranking methods. This objective addresses RQ-1 and is achieved via experiments EXP-1 and EXP-2.
- **OBJ-2**: Compare the effect of using implicit feedback such as click-through data instead of human judgements in LTR training data gathered from an online ES service. If organisations can safely use CTR in place of costly human annotations, this objective has implications for datasets like the one used in RQ-1 and is tested in experiments EXP-3 and EXP-4.
- **OBJ-3**: Detect jargon/terminology in enterprise content and decode jargon terms for query expansion via synonyms to unlock Semantic Search.This addresses RQ-2 and is achieved via experiments EXP-5 and EXP-8.
- **OBJ-4** Develop and evaluate a ranking model for Query Auto-Completion candidates customised for enterprise content and associated specialised jargon/terminology. This addresses RQ-3 and is achieved via experiments EXP-6, EXP-7 and EXP-8.

1.4 Research Methodology

The approaches and techniques used to investigate the research questions and objectives are outlined in terms of data collection/solicitation, technical approach, and evaluation strategy. The research ethics that governs the methodology is also described here.

1.4.1 Data Collection/Solicitation

In this study, data is gathered for a 'live' ES service of a large third-level academic institution. The data includes the organisation's intranet and public-facing internet components. The data is gathered using two techniques:-

- Soliciting samples of qualitative relevance judgements from expert (human) annotators.
- Extracting quantitative data from the ES system (e.g., use of search log files and indexed documents). This data can be used as a proxy for human relevance annotations.

1.4.2 Technical Approach

The Apache Solr open-source search platform [14] is used to build an Enterprise Search service. On top of this platform, Solr supports an LTR 'contrib module' that can be configured to run ranking models. Solr also supports a 'SuggestComponent', which enables QAC to be implemented in a highly configurable way (**OBJ-4**). Outside of Solr, offline LTR frameworks such as XGBoost [37] and RankEval [38, 39] are used to train and evaluate ranking models based on feature weights **(OBJ-1)**. LM and word embedding technologies like OpenAI's GPT-4 model [36] and RoBERTa [35] are employed for semantic matching and to capture representations of jargon/terminology used in organisations **(RQ-2, OBJ-3)**.

1.4.3 Evaluation Strategy

To evaluate the impact of LTR and LM methods on the ranking performance of a model, two methods are used:

- Conducting offline statistical and ranking experiments with an ES dataset generated from an ES service (**RQ-1,OBJ-2**).
- Conducting online experiments (such as A/B tests) via a load-balancer on a live ES service to select and eventually adopt the best-performing model (**RQ-3,OBJ-4**).

The results of the evaluation of the ES data are compared with the corresponding results of published studies from the domain of WS.

1.4.4 Research Ethics

Experimenting with ES data required a data protection impact assessment to ensure that personal information processing and controlling complied with EU GDPR

(2016) and AI Act (2024) principles, requirements and risks. The subsequent approval of the research proposal involved sign-off from both a Research Ethics Committee and also the IT Services/IT Security departments responsible for the service provision. Conditions for approval were that personal data (and meta-data) should be anonymised or pseudorandomised to protect individuals' identities (GDPR Article 25) and a mechanism to facilitate the rights of data subjects, including the right to opt out of the study (GDPR Articles 7 and 17). A record of all processing activity was maintained (GDPR Article 30). A copy of the Research Ethics Approval form, together with recorded approvals, is included in Appendix C. A proviso was stipulated by IT Security that the name of the enterprise should not be documented in any published material. For this reason, some minor details have been redacted in Appendix C. Additionally, IT Security did not authorise the publication of the raw queries. This restriction does not affect the utility of the released LETOR-formatted dataset, as the standard format is designed to work with anonymised Q-D pairs (the original query strings are not a requirement). Finally, henceforth the organisation will be referred to as simply 'The University'.

1.5 Research Contributions

The overarching intended outcome of this research is a bridge of the gap between state-of-the-art AI ranking models and real-world Enterprise Search implementations, while introducing novel features to detect and decode organisational jargon.

More specifically, this can be broken down into the following constituent contributions:

- **RC-1** LTR Models to rank search results and QAC candidates for ES content and information needs.
- **RC-2** A comparison of the effectiveness of traditional ES ranking methods with ranking performance achieved using LTR and LM methods.
- **RC-3** An Enterprise Search dataset based on document characteristics and search log data from a 'real world' ES service. The dataset is in the LTR format and is publicly available[1].
- **RC-4** Evidence that click-through rate can be used in place of human relevance judgements in training data for ES search results.
- **RC-5** Novel Language Model algorithms that improve ranking and recall for jargon/terminology specific to an organisation (for both search results and QAC).

Table 1.2 provides a mapping of the problem statements to each of their associated research Questions, objectives and contributions.

The research results are disseminated to peer-reviewed venues and the publications arising from this research are outlined in Section §6.4.

[1] https://github.com/ColinDaly75/ES-Ranking

Table 1.2: Problem Statement mapping via Research Problem, Question, Objective and Thesis chapter.

Research Problem Statement	Research Question	Research Objective	Research Contribution	Thesis Chapter
PS-1	RQ-1	OBJ-1	RC-1,2,3	3
PS-1	RQ-1	OBJ-2	RC-3,4	3
PS-3	RQ-3	OBJ-4	RC-5	4
PS-2	RQ-2	OBJ-3	RC-5	3,4

1.6 Thesis Outline

This **chapter one** is an **introduction** to the topics and research motivation while stating the research problem and corresponding questions. Additionally, the objectives and contributions are presented in this chapter.

The remainder of this document is organised as follows.

Chapter 2 presents a **literature survey** and describes the state-of-the-art with gap analyses focused on the research questions above. The core topics of the literature survey are i) Enterprise Search, ii) Ranking of ES search results, iii) Ranking of Query Auto-Completions, iv) Learning to Rank and v) Language Modelling.

Chapter 3 describes the research on using LTR and LM approaches for ranking **ES search results**, addressing RQ-1,2 and OBJ-1,2,3.

The research on using LTR and LM approaches for ranking **Query Auto-Completions**, addressing RQ-3 and OBJ-4, is described in **Chapter 4**.

Chapter 5 is an expanded **discussion**, interpretation and critical analysis of the experimental results (of both search results and query auto-completions) and their implications, generalisability and limitations.

Finally, **Chapter 6** presents a **conclusion** and includes a reflective evaluation as to the extent to which the research addresses the research questions and fulfils the contributions. Several avenues and agendas for future research are suggested.

Supplementary Information:
Additional supplementary information related to the software in this thesis is stored in the repositories outlined in Table 1.3:

Table 1.3: All code used in this document is stored in these GitHub Repositories.

GitHub repository location	**Description**
`https://github.com/ColinDaly75/ES-Ranking`	Search results ranking
`https://github.com/ColinDaly75/QAC_LTR_for_ES`	QAC ranking
`https://github.com/ColinDaly75/JARGES`	JARGES

2

LITERATURE SURVEY

The title of this thesis document is "*AI Approach to Ranking of Search Results and Query Auto-Completions for Enterprise Search*". This title encompasses five key research topics:

- Enterprise Search (ES)
- Ranking of Search Results
- Ranking of Query Auto-Completions (QAC)
- Learning to Rank (LTR)
- Language Modelling (LM)

} AI Approach methods

This chapter describes and analyses the state of the art in each topic with respect to its relation with, and application to, Enterprise Search. A definition (or multiple definitions) is offered for each topic, followed by a discussion of the topic's key challenges, problems and research gaps pertinent to ES.

2.1 Enterprise Search

ES refers to the retrieval of information from within an organisation's internal and external repositories. While small and well-organised organisations might not require a dedicated search function, efficient ES is essential in complex organisations for retrieving diverse and distributed information assets.

2.1.1 Background

ES can be conceived of as occupying a middle ground between Web Search (WS), which indexes content across the open internet, and desktop search, which indexes the contents of a single device [6, 40]. As personal drives are gradually replaced by shared storage on organisation's intranet, desktop search within an enterprise is becoming less relevant [6]. ES aggregates structured and unstructured data from

multiple organisational sources, such as intranet shares, document management systems, emails, social media, and web servers (both internal and external-facing).

Depending on a researcher's field, several related terms are used when studying ES. Synonyms include 'workplace search systems' [41], 'enterprise document search' [42], 'workplace web' [43], 'Enterprise Information Portal' [43], 'Business Portal' [44], 'corporate search' [45] and 'Enterprise AI search' [46]. Lewandowski [47] laments the lack of a consistent and unified terminology in the field of search engines between information scientists and search engine optimisers.

Although scholarly discourse persists regarding the inclusion of external-facing websites within ES systems, prevailing definitions recognise that both internal and public-facing content may be indexed. According to Craswell [10], enterprise content includes intranet pages, email archives and document repositories. Kruschwitz expands this content to also include an organisation's external-facing documents and web pages [6]. In either case, the primary focus of ES systems remains oriented toward serving employees (or members) of an organisation rather than prospective customers or the broader public.

By its very nature, the primary goal of knowledge workers is to access and retrieve information. Examples include programmers, physicians, pharmacists, architects, engineers, scientists, design thinkers, public accountants, lawyers, editors, and academics, whose job is to "think for a living". Accounts of the average time spent looking for a document vary widely. Software engineers are estimated to spend 25% of their workday searching [41]. Design engineers and technical professionals in six high-tech organisations were observed to average 25.6% of their time searching [48]. In an R&D organisation, knowledge workers spend between 10 and 30 minutes daily [49]. Whatever the figure, most people would acknowledge that it is intuitively time-consuming. According to Orbital Media [50], 53% of websites have a search tool in their header, reflecting the fundamental importance of search functionality even outside large enterprises.

Organisational knowledge may include deliverables created explicitly through knowledge work (e.g., patents and reports). It also comprises the codes (rules, formal and informal procedures and policies) and routines (strategies for performing complex tasks) that guide organisational action and decision-making [51]. Ideally, Enterprise Search should also facilitate discovering relationships between entities in the organisation. This involves both understanding user needs in enterprise search and development of appropriate IR techniques [10].

In addition to time loss, any delay in retrieving the desired information or document that describes relatively simple work tasks is also a source of frustration and leads to negative feelings toward the service [42, 52, 53]

An organisation's information assets may be stored in different systems and formats. When employees leave the organisation, important knowledge and information may become irretrievable. Without an efficient ES service, employees may find it difficult to retrieve quickly, collate and cross-reference. This can slow the 'onboarding'

process for new employees. Moreover, it can lead to project delays and decision-making [53].

Managers in the US, UK, Germany and France say that their internal enterprise search service falls short of expectations and that over half (52%) of surveyed users "cannot find the information they seek within an acceptable amount of time, using their own enterprise search applications" [3]. Similarly, many end-users ('enterprise searchers') express low satisfaction with the precision and recall of their workplace search service [52].

2.1.2 Definitions

Enterprise search can be simply defined as finding the information needed from within an organisation [3]. This chimes with White's definition of ES as a service that "enables employees to find all the information the company possesses without knowing where the information is stored" [54].

As illustrated in Figure 1.1, ES is a federated store of workplace information with data gathered from multiple sources, such as intranets, document management systems, e-mail and social media [6, 10] and may also include the organisation's external-facing HTTP web servers [44, 55]. The data can be structured or unstructured [52, 55]. In his book entitled 'Enterprise Search', White defines enterprise search as a service that 'enables employees to find all the information that the company possesses without the need to know where the information is stored' [54]. While this definition does not exclude the corporate website, it clearly prioritises employees over external website visitors as the primary end-users. According to Maiworm et al., the contents of an ES service should simply incorporate 'all available data sources' [53]. White says that there is 'no right answer' to the question of whether 'a search application optimised for internal information will be a good solution for external users of corporate web sites' [54]. Kruschwitz distinguishes ES from WS by the predominance of documents, memos, books and spreadsheets [6]. No definition of ES explicitly excludes externally available website content.

Consequently, it is expedient in this study to define Enterprise Search as covering both publicly accessible and internally restricted content, as this aligns with the hybrid nature of The University's search environment, where staff and students access a shared search service.

2.1.3 Distinguishing Enterprise Search from Web Search

Despite a substantial body of research on WS, ES has not been explored to the same depth [6]. A key starting point is understanding the differences between ES and WS. ES differs from WS insofar as content may be indexed from multiple internal data sources (e.g., corporate directories, SharePoint repositories, intranet file systems), rather than being crawled from publicly accessible web pages. ES datasets therefore contain non-HTML content, legacy files not designed for search, and materials with sparse metadata (e.g., no `<title>` tags, inconsistent headings, or missing hyperlinks). ES also commonly features alphanumeric lookup queries for internal identifiers such as usernames, course codes, tracking numbers, or purchasing

codes (query types for unstructured content that are comparatively rare in WS [56]).

These differences extend beyond content type and format. ES usage behaviour tends to be more navigational, with users seeking a *specific known item* (e.g., a policy document, local jargon terms, or the homepage of a department) [57], whereas WS supports a broader range of exploratory information needs that often include full sentence queries and question answering [58]. Consequently, ES queries are generally shorter than WS equivalents, and question-form queries are significantly less frequent. Moreover, the absence of a rich hyperlink graph in ES means that ranking functions relying on link-based authority signals (e.g., PageRank or anchor-text distributions) are less meaningful, whereas field-aware textual relevance signals and domain-specific terminology often dominate.

Finally, ES environments impose practical constraints that are rarely present in WS. ES relevance judgments cannot be crowdsourced because internal content is not externally visible, resulting in annotation by domain experts who share context and knowledge of organisational terminology. This typically yields higher inter-annotator agreement but at smaller scale [59]. Personalisation, session-based profiling, and location-aware ranking (standard in WS) are constrained by privacy expectations and regulatory compliance (e.g., GDPR), limiting the use of behavioural signals that power modern WS models.

For these reasons, not all WS practices are transferable to ES. As White notes, "the determination of a good answer for intranet search is quite different than on the internet" [60, 61]. The following sections outline how ES/WS differences influence ranking functions, feature engineering, evaluation methodology, and the design of Query Auto-Completion (QAC) systems. Appendix A provides a catalogue of every difference encountered during the course of this research.

2.1.4 Lookup and Exploratory Search

ES Searches are often for known documents (such as this year's college calendar), or other well-structured objects (such as a person's contact details). This is sometimes referred to as a 'lookup search' [62]. Lookup search relies heavily on users' ability to recall and recognise information [52]. In contrast, WS users are more likely to perform exploratory searches, where the information need is broad, evolving, poorly defined and 'exploratory' in nature. This distinction influences query formulation, ranking strategies, and evaluation metrics.

2.1.5 Words per Query

In the field of WS, the number of terms in queries (sometimes referred to as query length) has been steadily increasing over time [63]. Queries are becoming more conversational and often mimic natural language questions. This trend is influenced by several factors, including advancements in search technology, the proliferation of voice assistants, and changing user behaviour. In the context of voice search or AI-powered assistants like Amazon's Alexa, and Google Assistant, queries are more likely to be phrased as questions or full sentences.

Over time, WS users have learned that search engines are capable of understanding detailed and specific queries, leading to more elaborate input. Users therefore feel less restricted by the need to craft keyword queries. WS user expectations and behaviour has gradually spilled over to the field of ES [1].

A 2013 study of search queries by Kruschwitz [64] on a university web site observed an average length of 1.81 query terms and this length will be compared to that of The University's ES service in Section §3.4.2 and Table 3.1. In a 2006 study for WS, the average number of terms per query of the AOL dataset was 2.99 [26].

2.1.6 Query Clustering

Query clustering is the process of grouping similar queries into clusters based on their lexical or semantic similarity. Clustering enables pattern identification of related topics among the queries [65]. For example, in ecommerce, queries like "best smartphones 2021" and "top phones 2021" could be clustered to retrieve overlapping results.

Lexical clustering is achieved via word overlap or surface-level similarity, while semantic clustering uses embeddings (e.g., Word2Vec, BERT, described in Section §2.5) to capture deeper semantic relationships. By understanding shared intent among queries, search systems can better align with user needs and improve the overall search experience. Query clustering is applied to The University's queries in Section §3.4.2, where examples of both lexical and semantic clustering are given for some frequently queried terms.

2.1.7 Search Demand Curve

The Search Demand Curve (SDC) is a graph used to plot the outsize impact that a small number of popular queries has on the overall volume of search activity. The term "fat head" refers to the portion of search queries that are highly popular and repetitive, often dominated by a relatively small number of keywords or search terms. This zone of the curve represents the top-ranked results that receive a significant portion of clicks and traffic [66].

In some cases, researchers or practitioners may use arbitrary thresholds or cutoff points to segment the SDC into different categories, such as fat head, chunky middle, and long tail. However, these thresholds are often chosen based on empirical observations or practical considerations rather than a fixed standard.

The specific percentage of 18.5% is not a universally agreed-upon threshold for defining the fat head segment of the SDC [66]. Instead, the fat head segment is typically characterised by a relatively small percentage of highly popular and competitive search queries that generate the majority of clicks and traffic. The exact percentage may vary depending on the context, industry, and dataset being analysed.

Overall, the fat head segment of the SDC represents the most relevant portion of search queries. It consists of the most frequently queried and high-volume keywords

that further attracts traffic to the highest ranked search results. The SDC for The University's query history is plotted in Figure 3.3.

The SDC helps characterise how users interact with The University's ES service and provides context for the evaluation of ranking models. In ES, a small number of high frequency queries often dominate traffic, meaning that improvements (or regressions) in ranking performance for this segment can have a disproportionate effect on overall user experience. Conversely, long tail queries are sparse and difficult to evaluate, which influences the choice of metrics, the design of experiments, and the interpretation of generalisation performance. Thus, the SDC helps explain which parts of the query distribution our experiments are most likely to affect and how representative the evaluation dataset is of real usage.

Since the segments are expressed as percentages, there is no reason the SDC graph and analysis cannot be applied to both WS and ES, even though typical ES indices are only on the order of hundreds of megabytes. This is tiny compared to WS, where systems operate at petabyte scale (the Google search index is estimated to exceed 100 PB in size [67]). Figure 2.5 compares the shape of the Search Demand Curves between WS and ES.

2.1.8 Jargon/Terminology

Newcomers to an organisation often struggle with unfamiliar internal vocabulary, which can affect their ability to retrieve relevant information. Jargon is enterprise-specific vocabulary that employees/members can understand. It encompasses words, phrases, expressions, and idioms that are not universally familiar or properly understood. While familiar to long-term employees, jargon can impede effective searches by new staff or external stakeholders. This is sometimes referred to as 'vocabulary mismatch' [32]. Query formulation using incorrect terms or phraseology can lead to poor ranking and recall of query results, with a potential reduction in staff productivity.

The same term can have a different meaning outside of the organisation. A search for 'timetable' in WS will probably return departure times for the nearest bus or train. Kruschwitz gives the example that the same search in a third-level education institution might be aimed at lecture timetables (in Autumn) or exam timetables (in Spring) [64]. This demonstrates that a jargon term not only has a different meaning inside an organisation, it may have different meanings over time.

Although excessive use of jargon and terminology in organisations is often perceived as exclusionary, we use the terms here in a positive context for conveying complex ideas, processes, or services among employees/members who share common knowledge of the enterprise. In this context, jargon and terminology facilitate efficient communication.

In addition to jargon, organisations use abbreviations as a shortened form of a word or phrase. Abbreviations include initialisms and acronyms. Initialisms are usually created by omitting certain letters or components while retaining the essential meaning, typically to enhance efficiency in written or spoken communication. Examples

include ‘MBA’ for ‘masters of business administration’ or ‘AR’ for ‘academic registry’. An acronym is a specific type of abbreviation where a pronounceable word is formed from the first letters of other words, pronounced as a single word. An example at The University is ‘SITS’, which is short for the ‘student information and tracking system’. The usage of initialisms and acronyms often depends on context, domain, or convention, and they may require disambiguation, especially in technical, academic and organisational settings [68].

As we will see in §4.5, the challenge of detecting and decoding enterprise jargon/terminology within a corpus lends itself to the fields of natural language processing (NLP) and large language models (LLM). LLMs, such as OpenAI’s GPT (Generative Pre-trained Transformer), are trained on large datasets containing vast amounts of text from diverse sources. Word embeddings can capture semantic relationships between words in text data [69]. While LLMs and embeddings are regularly used in ecommerce [70] and commercial search engines [25], their application for ES has not been sufficiently explored.

As antidote to the use of jargon is the ‘Explain Like I’m Five’ (ELI5) pedagogical and communicative device. It is designed to distil complex concepts into simple, relatable explanations, analogous to how one might explain a topic to a child aged five. The method emphasises clarity and minimalism, aligning with principles from cognitive load theory [71]. However, while ELI5 may be suitable for a one-off explanation, it could quickly become a hindrance in organisations if repetition is involved. Additionally, much of the organisational terminology does not lend itself to simplification. Jargon is here to stay!

2.1.9 Gaps in the Literature

Kruschwitz and Hull, in their 2017 book ‘Searching the Enterprise’ write that ‘Search has become ubiquitous, but that does not mean that search has been solved’ [6]. ES is a growing industry and is now worth hundreds of millions of euros in Europe alone [72]. In spite of this growing commercial importance, ES seems to be widely under-studied by the academic community. Few academic case studies and very few user evaluations are reported [6, 73].

The outstanding challenges in ES include:

- The heterogeneous content sources (e.g., intranets, databases, document systems) make ES content fundamentally different from WS content (as catalogued in Appendix A). Consequently, the ranking signals for ES are unlikely to be the same as for WS.

- The motivation of enterprise searchers is likely to be different to users of ecommerce or commercial WS search engines. ES users use shorter queries with greater emphasis on lookup rather than exploratory searches.

- Academic literature has detailed no mechanism to deal specifically with jargon/terminology and the challenge that it presents for newcomers to the organisation.

Thus, while users expect the sophistication of WS, organisational constraints and enterprise content often inhibit similar levels of search quality in ES (as outlined in PS-1).

2.2 Ranking of Search Results

The ranking of search results is the cornerstone of any Information Retrieval (IR) service, including ES services. A basic definition of ranking is that it determines the order in which documents are presented to the ES users. The most useful and relevant items are ideally presented near or at the top of the list. The success of a search system is thus highly dependent on its ranking performance.

This section reviews foundational and contemporary approaches to ranking search results, with emphasis on their application and limitations within the context of ES.

2.2.1 Traditional Relevance Ranking: TF-IDF and BM25

Early IR systems primarily relied on term matching techniques, where the relevance of a document was assessed based on the overlap of terms between the query and the document text.

Term Frequency–Inverse Document Frequency (TF-IDF) measures the importance of a term within a document relative to its frequency across the entire corpus. Term Frequency (TF) captures how often a term appears within a document, while Inverse Document Frequency (IDF) penalises terms that are common across many documents, under the assumption that rarer terms are more informative.

BM25 [13] is a refinement of the classical TF–IDF model that introduces a probabilistic term-frequency saturation and document-length normalisation. While TF–IDF assumes a linear relationship between term frequency (TF) and relevance, BM25 adjusts the TF component to account for diminishing returns: the gain from each additional occurrence of a term decreases asymptotically. In effect, BM25 can be viewed as a modified TF–IDF weighting scheme in which the TF component is redefined as a non-linear saturation function.

The BM25 relevance score for a document D and query Q can be expressed as:

$$\mathrm{BM25}(D,Q) = \sum_{t \in Q} \underbrace{\mathrm{IDF}(t)}_{\text{global term weight}} \cdot \underbrace{\mathrm{TF}_{\mathrm{BM25}}(t,D)}_{\text{local term weight}}$$

where the modified term-frequency weighting function is:

$$\mathrm{TF}_{\mathrm{BM25}}(t,D) = \frac{f(t,D)\,(k_1+1)}{f(t,D) + k_1\left(1 - b + b\,\frac{|D|}{\mathrm{avgdl}}\right)}$$

and

- $f(t,D)$ is the raw frequency of term t in document D;

- $|D|$ is the document length, and avgdl is the average document length in the corpus;
- k_1 controls term-frequency saturation (typically 1.2);
- b controls document-length normalisation (typically 0.75);
- IDF(t) is the inverse document frequency, e.g., $\log \frac{N-n_t+0.5}{n_t+0.5}$, where n_t is the number of documents containing term t.

This formulation makes explicit that BM25 replaces the linear TF component of TF–IDF with a probabilistically motivated saturation function, producing a score that balances term frequency, document length, and corpus-level discrimination.

The TF contribution is asymptotically bounded by k_1, preventing overly frequent terms from dominating the score. In this respect, BM25 is regarded as a probabilistic approach to relevance estimation. It remains widely used in both web and enterprise search applications, including as the default ranking function in Apache Solr and Elasticsearch, where the default parameters are typically $k_1 = 1.2$ and $b = 0.75$ [74].

BM25F [45] extends BM25 to multiple fields. It treats a document as weighted "streams" such as title, anchor, and body. For each query term, it first aggregates evidence across streams, applies BM25 saturation at that stage, then combines across terms. Some fields are more predictive of relevance than others (e.g., title could be expected to carry more weight than body). BM25F can use stream specific parameters like b_s.

In its simplest form, BM25F is equivalent to BM25 applied to a document where fields are replicated according to their weights. However, a more complex treatment is when BM25F approximates inference within the probabilistic relevance framework for multi-field documents. Each field has its own length and noise. BM25F first normalises per field, weighs the evidence, then applies BM25 saturation after aggregation. This step is non-linear. The weights thus behave like log-odds style coefficients that balance heterogeneous sources under a shared (field-aware) IDF.

BM25F is not employed as an explicit ranking feature in the experimental setup described in Chapters 3 and 4, as the LTR models are configured with field-based features that enable them to learn the relevant weights directly.

However, TF-IDF, BM25 and BM25F are limited by their reliance on surface-level lexical similarity, failing to account for semantic matches or contextual nuances, which are increasingly important in ES scenarios characterised by diverse jargon and varied document formats.

2.2.2 Importance-Based Ranking

Reliance on TF-IDF and BM25 for WS ranking had a major flaw. In WS, a publisher or advertiser was able to promote the ranking of their page simply by increasing the number of times the term is mentioned in the web page!

Hence, in addition to similarity measures like TF-IDF and BM25 (which assess relevance based on term frequency and document structure), importance signals offer an alternative ranking criterion. These signals, which may include factors such as document authority, user engagement metrics, or domain-specific relevance, can enhance ranking by prioritising utility (described in Section §2.2.15) over mere lexical overlap.

LTR frameworks (described in Section §2.4) enable the weighted combination of relevance and importance signals (features) into a unified ranking model, providing considerable flexibility and performance optimisation.

2.2.3 Re-Ranking

Query re-ranking (Figure 2.1) is a technique wherein an initial, computationally efficient sub-query function (A) is executed to identify a subset of matching documents (x). Subsequently, the top k documents of this subset are re-ranked based on the ranking scores generated by a more complex or computationally intensive sub-query function (B).

Ranking function A is typically BM25, whereas the 'B' ranking function is often some proxy measure of popularity or recency (described below).

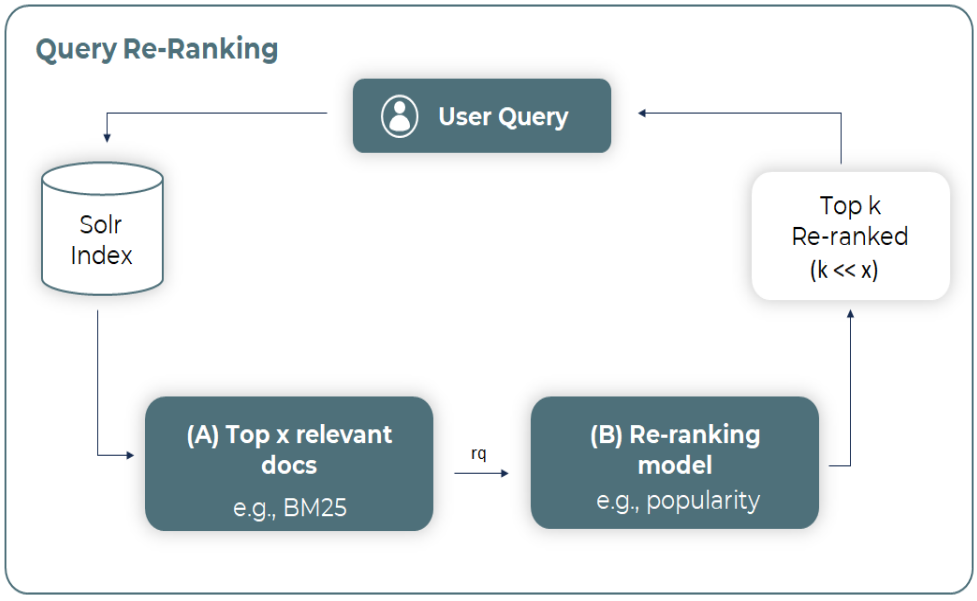


Figure 2.1: Query re-ranking involves processing a regular query for matching x number of documents and then secondary processing (re-ranking) of the top k documents using the scores from a different, more complex query. In Apache Solr, a re-ranking query is specified with the `rq` parameter.

Limiting the application of function B's complex scoring mechanism to a smaller subset of documents significantly reduces the overall computational cost (i.e., compared to applying function B directly to the entire index). However, this approach introduces a trade-off: documents that receive low relevance scores during the initial

function will be excluded from the re-ranking phase, even if their relevance scores under function B would have been high.

Multiple iterations of re-ranking can be employed. This is sometimes referred to as multi-stage re-ranking [75]. Balancing objectives across stages can be challenging, as optimising for one may degrade performance of another (especially as the subset is smaller for each stage). Moreover, latency (query response time) can become an issue as each additional re-ranking stage adds query processing time.

2.2.4 Views

In enterprise environments, document popularity (measured by access frequency) can serve as a useful heuristic for importance. However, reliance on views can introduce bias towards historically popular content, potentially overlooking more current or contextually appropriate documents. The implementation of a ranking function based on view-count for The University's ES service is described in Section §3.7.1.

2.2.5 Link Analysis

Prior to 1998, the online search experience was predominated by relevance ranking (term matching). Relevance scores were computed by algorithms such as TF-IDF. Examining the returned low quality matches was laborious and it was common for users to sift through several pages of search results. As was the case in non-linked collections of old, word of mouth and the advice of a subject expert were common pointers to quality information.

Such was the limitation of ranking based on relevance scores alone, Yahoo! developed and maintained their own hierarchical directories for popular subject areas. Searchers used the 'Yahoo Directory!' table of contents to navigate to a desired subject area and then browse listings selected by Yahoo!.

While hyperlinks existed in earlier hypertext systems [76], what makes the web transformative is its open, internet scale design that allows anyone to publish and interlink documents (without central control). Hyperlinks are a linking feature that uses descriptive anchor text to conveniently direct or recommend a user to another web page. A Webgraph is a visual representation of these relationships. Prior to 1998, webgraphs had been simply used to understand relationships between pages, especially to visualise the quantity of 'inlinks' and 'outlinks' of a particular site. The meta-content of hyperlink structure remained unexplored.

By 2009, when the GOV2 corpus was published (which consists of web pages crawled from the .gov domain and described in Table 2.3), it included ranking features that leverage link analysis. Of the 46 features defined, 4 are based on link analysis (PageRank, Inlink number, Outlink number, and Number of child pages) [77, 78]

Not until the innovation of the Hypertext-Induced Topic Search (HITS) algorithm in 1998 [79], was the potential of webgraphs to improve search results understood. HITS uses mutually reinforcing authorities (pages pointed to by many 'good' hubs)

and hubs (pages pointing to many 'good' authorities). Whereas PageRank (described below) is query independent, HITS is typically query dependent. HITS starts from a small 'root set' of top matches and expands to a 'base set', so the discovered hubs/authorities are tailored to the search topic. This focus could be useful for enterprise or site-level search where topical communities are strong. Unfortunately, Grover notes that HITS is considered too heavy to compute in real-time using Solr [80] (though a practical way would be to compute HITS offline and feed the scores into Solr as fields that could be boosted).

Furthermore, while HITS is query dependent, it can still be combined with term-relevance scores (such as TF-IDF or BM25) by applying HITS to the query-specific base set and then merging its authority/hub scores with the traditional term-matching scores during ranking. In this setup, BM25 selects the candidate documents, while HITS refines their ordering based on link structure rather than used as a relevance ranking on its own [79, 80].

PageRank At about the same time that Kleinberg was developing HITS, Larry Page and Sergi Brin developed, implemented and published a similar algorithm called PageRank[2] [16, 17], which would become the driving force behind the Google Search engine.

According to Google[3], 'the underlying assumption is that more important websites are likely to receive more links from other websites'. For example, if an important website such as CNN.COM links to my personal blog, this is seen by Google as a great endorsement for my blog. However, if it transpires that CNN is also linking to thousands of other blogs, then this will attenuate the weight of the original endorsement.

Mathematically, the PageRank score of page i, denoted $PR(i)$, is defined as:

$$PR(i) = \frac{1-d}{N} + d \sum_{j \in \text{In}(i)} \frac{PR(j)}{\text{outdeg}(j)}.$$

where:

- $d \in (0, 1)$ is the damping factor (typically 0.85);
- N is the total number of pages;
- $\text{In}(i)$ is the set of pages linking to page i;
- $\text{outdeg}(j)$ is the number of outgoing links from page j.

The score produced by PageRank is sometimes referred to as the 'popularity score' [17]. This is unrelated to the relevance score. PageRank measures the intrinsic importance of a page in relation to a collection; it does not measure relevance for

[2]The word PageRank is a double-reference to the author's surname and a web 'Page'

[3]`https://www.google.com/competition/howgooglesearchworks.html`, accessed 7th March 2019

a given query. The popularity score for each page is determined and indexed prior to the execution of a search term. As such, PageRank is a Query-Independent re-ranking function.

The effectiveness of PageRank catapulted Google to become the dominant search engine by the early 2000s [17]. While Yahoo! initially focused on curating a hierarchical directory and later incorporated conventional web search, link-analysis approaches such as PageRank ultimately proved far more effective at handling the web's dynamic and unstructured nature.

Google no longer depends on the PageRank algorithm alone, as the 'social proof' and popularity that it tries to quantify are no longer seen as exclusive measures of importance.

LinkRank In 2009, a new suite of analysis tools called 'LinkRank' was devised as a directed network equivalent of Google's proprietary PageRank [81]. LinkRank is a PageRank-like link analysis program. According to its authors, "LinkRank can be considered as the PageRank of links". It starts by assigning a common score for all URLs. It then creates a global score for each URL based on the number of incoming links and the scores for those links and the number of outgoing links from the page. The process is iterative and scores tend to converge after a given number of iterations. It is different from PageRank in that links between different websites or domains are not included.

LinkRank can be used as an information retrieval ranking algorithm particularly for ranking nodes in a graph based on their connectivity and structural importance. Importance is propagated through the graph, with nodes transferring their own importance scores to others via outgoing links.

Mathematically, LinkRank assigns a score to each directed edge $i \to j$ by estimating the probability that a random surfer located at page i follows the outgoing link to page j [17]. It is defined as:

$$LR(i,j) = PR(i) \cdot \frac{1}{\text{outdeg}(i)}$$

if a hyperlink from i to j exists; otherwise $LR(i,j) = 0$ and where:

- $PR(i)$ is the PageRank score of page i;
- $\text{outdeg}(i)$ is the number of outgoing links from page i;
- $LR(i,j)$ represents the joint probability that a random surfer is on page i and traverses the specific outgoing link to page j.

LinkRank has been incorporated into the Apache Nutch web crawler[4]. It works by first creating a webgraph which stores output scores for each URL in the node

[4]java class is org.apache.nutch.scoring.webgraph.LinkRank as described at `https://cwiki.apache.org/confluence/display/NUTCH/NewScoring`, accessed 20th March 2020

database. LinkRank then combines the anchor text associated with each URL's inlinks and outlinks.

The advantage of PageRank is that it has a view of the entirety of the world wide web, where the Google web crawler is not restricted to a particular domain. LinkRank is conceptually related to PageRank, but can be adapted to various types of networks and link-based structures. As with PageRank, LinkRank can calculate the weight of an inlink between domains and/or documents.

Enterprise websites typically have a 'directed network' nature and are often confined to one specific DNS domain or community of domains. To understand why the LinkRank ranking algorithm is particularly suited for sites consisting of webpages, we first examine the hierarchical structure of The University's website (Fig 2.2). While the precise nature of the hierarchy departmentalisation will be different for every organisation, it is safe to assume that any medium or large enterprise will not have a flat structure.

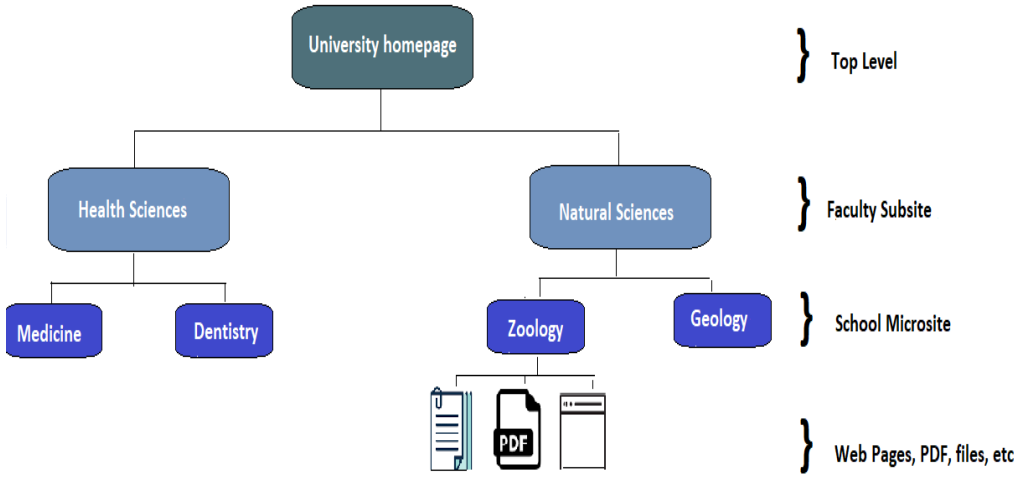


Figure 2.2: The hierarchical (departmental) structure common to many large organisations. The hierarchy at The University follows a faculty / school structure.

An orphaned document is one that is not (or no longer) linked to or from any other documents in the ES index. The isolated document may not be discovered by a crawler [82], and therefore overlooked by the LinkRank scoring mechanism.

While highly effective on webpages, link analysis may have limited applicability in ES, where documents (e.g., reports, spreadsheets) are often orphaned or lack explicit hyperlink structures. Thus, while importance signals can enrich ranking models, they are often less informative in the fragmented, link-poor landscapes typical of enterprise corpora.

LinkRank serves as a versatile algorithm for ranking entities in networks, leveraging the structure of links to determine node importance. Its adaptability and foundation in graph theory make it an essential tool across disciplines such as web search,

social science, and bibliometrics. The implementation of a ranking function based on LinkRank is described in Section §3.7.1.

Microsites Large websites are commonly subdivided into topically coherent regions. In the same way that a newspaper is organised into sections (e.g., world news, finance, sport), with each section having its own editorial focus and visual identity, large organisations structure their sites into distinct domains or 'microsites'. These microsites often have their own homepage, navigation, branding, and content owners (Figure 2.2).

Such structural segmentation is also typical of university websites, where faculties, schools, research centres, and administrative units each maintain their own microsite [83].

As an example, The University's web pages include over 300 microsites. Each microsite has at least one editor. In this case, the role of the editor includes publishing content on behalf of colleagues in their department. The microsite owner must be very familiar with published content, so as to avoid replication and ensure consistent messaging. These microsite editors are subject matter experts and are well placed to 'judge' the ranking order of their pages for a particular term (Section §3.4.4 describes how a representative dataset is created).

This division is relevant for enterprise search because queries and content are frequently scoped, interpreted, and ranked within the context of a specific microsite, and the search engine's indexing and ranking behaviour can vary depending on where content resides within this organisational structure.

2.2.6 Elevation

Query elevation allows for the manual setting of the top result (or results) for a query regardless of the computed rank score. This may be useful when business priorities dictate that particular documents receive the top ranking spot for particular keyword queries. This functionality diverges from the traditional concept of ranking insofar as it operates as a manual override or customised enhancement to regular ranking functions. The Apache Solr documentation describes elevation as 'sponsored search' and 'editorial boosting' [84].

For each keyword query, a predefined query-document (Q-D) pair is specified in Solr's `elevate.xml` file, ensuring that certain documents appear at the top of search results. However, query elevation only works for preconfigured queries and cannot handle unseen data, necessitating regular manual updates.

While this approach is useful for manually prioritising documents to align with organisational objectives, it focuses on result *positioning* rather than ranking. As such, it falls outside the scope of this research.

2.2.7 Multi-Objective Ranking Optimisation

Multi-objective ranking optimisation (MORO) is a methodology where the ranking model optimises for multiple objectives or criteria simultaneously when determining the order of documents in the search results. Whereas ranking usually focuses on a single goal (e.g., maximising relevance to a query), multi-objective ranking seeks to balance or prioritise multiple and sometimes conflicting objectives.

Multi-objective ranking is a powerful approach for tackling complex ranking scenarios where multiple, sometimes conflicting goals must be addressed. By balancing objectives like relevance, diversity, fairness, and monetization, it enables search and recommendation systems to deliver results that align with user expectations and organizational priorities.

However, MORO's success depends on careful management of trade-offs. As with elevation (Section §2.2.6), it poses the risk that these objectives may diverge from the preferences or expectations of the ES end-users. A high-profile example of this came to light during the 2023 US Department of Justice's antitrust case against Google, where concerns were discussed that Google's WS ranking algorithm was "getting too close to the money" as revenues sometimes trumped the end-user preferences [85].

In Section §3.3.5 and in EXP-3, the multiple objectives of human annotations (by microsite editors) and surrogate annotations gathered using click-through data (from end-users) are explored.

2.2.8 Anchor Text

Anchor text refers to the clickable, visible text in a hyperlink. But it also provides context about the linked document's content to the search engines to assist relevance, ranking and recall. In Section §3.7.1, the anchor text data in The University's corpus is used a ranking function. While anchor text is proven to boost ranking in Web Search [86], it may be of limited use with enterprise content, where many documents are simply placed on a drive and contain no inlinks, without ever having had an indexing or search intent, much less anchor text.

2.2.9 Recency

Recency refers to time-based relevance. The assumption is that the newer the document, the fresher the information it contains and ultimately has more utility for the searcher. This is especially important in organisations where multiple near identical versions of a document exist on the intranet. For example, in The University, a search for 'college calendar' should list the current one over previous years.

The 'recip' operator is a mathematical function provided by Apache Solr. It simply applies a reciprocal transformation to a field value. It is commonly used in conjunction with the ms() function, which calculates the difference between the current time and a document's timestamp. The age of a Web page is the time interval between when it was last modified and when it was collected. Recip is defined as follows:

$$recip(x, m, a, b) = \frac{m}{a \cdot (x + b)}$$

where:

x: is the input value, often calculated as the time difference using ms().
m: is the multiplicative constant for scaling.
a: is the slope, controlling how steeply the score decays as x increases.
b: is the offset, controlling the minimum value of the denominator to prevent division by zero.

The default values for the m, a and b parameters are 3.16e-11, 1 and 1 respectively and are geared for WS. This raises the question of whether these defaults are suitable, or require adaptation, when applied to ES. One way to determine this is to examine the website recency plot and compare it to an ES document recency plot. Figure 2.3 plots the number of websites in existence (in billions) against the their creation timeline [87]. The rate website growth was steep until about 2020 when there was a "shift to mobile apps, the consolidation of web services, and the change in how websites are tracked today" [88].

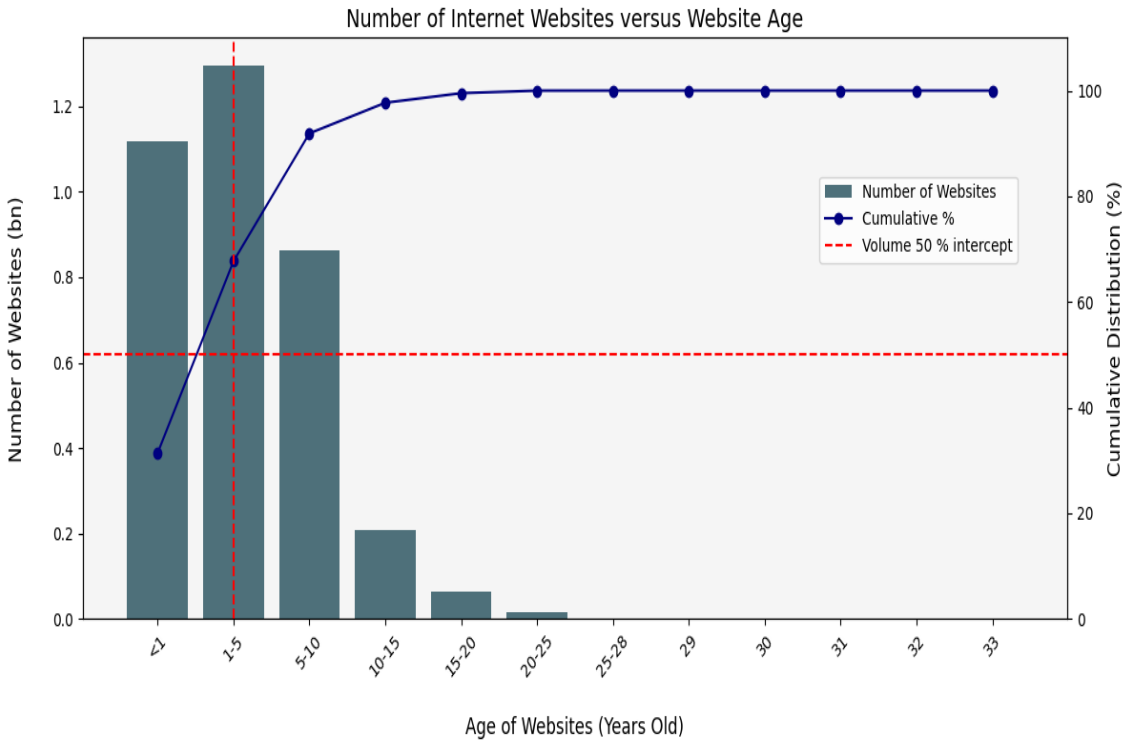


Figure 2.3: Website Recency Plot showing the age (i.e, 'recency') distribution for all websites on the internet. Website age data provided by Semrush [88].

In Figure 3.1 (Section §3.2.3), the shape of this plot, which is based on *websites*, will be compared to that in an organisation's ES index (based on *documents*).

2.2.10 Document Type

Ranking by document type involves prioritising or differentiating documents in search results based on their type, such as format, source, or purpose. Document types can include categories like PDF files, HTML web pages, research papers, news articles, images, videos, or internal organisational documents. By incorporating document type into the ranking process, retrieval systems can better align results with the user's intent or preferences [89]. A simple implementation approach would be to identify file types based on their file extensions (e.g., PDF, xlsx, docx) and using them as features in the ranking process.

Employees of an organisation may prioritise the search for specific types of documents (e.g., contracts, budgeting or accounting data, or meeting notes) over public-facing 'shopfront' HTML pages. Most of these searches would be conducted using a desktop browser that has access to the organisation's intranet. A desktop is likely to have a broad suite of applications installed that allow for automatic opening and reading of the documents.

But for this study, a broad working definition of ES is adopted (as discussed in Section §2.1.2). This encompasses both internal and external facing content (e.g., intranets, file shares, corporate directories and the publicly accessible pages of The University's website). The definition impacts the extent to which ES can be considered as being 'desktop-centred'. For The University, there is a wide range of resources (public and private). ES users in this context include not only staff at their desks but also students, researchers and other stakeholders who are frequently more mobile and may access institutional content from mobile devices.

A smartphone/tablet user performing a search is less likely to click on a document with the '.xlxs' extension (Microsoft Excel) if they do not have the appropriate 'reader' app installed [90–92]. Vassio et al. have documented a rapid expansion of mobile browsing [93]. A 2025 study of global analytics by StatCounter confirms that mobile devices now constitute the majority of web traffic worldwide [94].

Consequently, if an ES search service accommodates both staff and students, effective ranking by document type would would need to identify the user's device. In any case, a ranking function based on document type should not overshadow more critical relevance signals, such as query-term matching or semantic similarity.

2.2.11 Personalisation

Personalisation based on a user's temporal search history (or session history) has proved very effective for completing a user's query prefix in WS as it can infer the user's interests [20]. Personalisation is, however, excluded from the scope of this ES research for several reasons:

- Personalisation and collaborative-based filtering are major areas of investigation in their own right and are typically covered in the domain of recommender systems.

- The IT Services department that manages The University's search service used in this study does not permit the profiling of individual user data without explicit consent. The approval and restrictions granted during the research ethics process are outlined in Section §1.4.4 and in Appendix C.

- According to Chory et al., personalisation potentially comes at the cost of surveillance, which "is often met with resistance from employees as it taps into concerns over workers' privacy rights, due process, and fairness" [95]. Many organisations fear that profiling individual users' search patterns and habits, even if used exclusively for ranking query auto-completions, raises questions about data privacy and security, as well as transparency (users should be informed about how their data is being used for personalisation and given the option to opt-out if they prefer not to have personalised query auto-completions).

- The nature of the relationship between an organisation and its members is not the same as that between an individual and a commercial WS provider (e.g., Google). While a user may not be concerned about sharing behaviour with a third-party organisation like Google, it may be different when asked to share with his/her employer. This is explored in Inman's article entitled "If you let Google have your data, why not the NHS?" [96].

- Identifying the user for anything longer than a single search session would require a user to be identified by logging in. Many ES services are designed for use without first authenticating and authorising the user. Indeed, the query intent behind several search terms is targeted at retrieving the procedure to obtain a login account or password reset!

2.2.12 Location-based Ranking

Location-based ranking in search results refers to the process of ordering and prioritising search outcomes based on the geographic relevance of the content to the user's location. A common example in the field of WS would be 'restaurants near me'. The ES equivalent might be 'printers near me'.

Obtaining the location typically depends on the user 'sharing' a GPS location, an IP address, or a manually entered location. The ranking function would be based on the distance from the user, which would have the effect of boosting the results or QAC suggestions that are geographically closer or more relevant.

An industrial sociologist performed a study of how organisational, social, spatial, temporal, and technical issues relate to the deployment of location tracking systems in an office environment [97]. It presents findings on how workers perceive the experience of the surveillance and predicts resistance, unless transparency and voluntary participation are ensured.

Location-based ranking is, however, excluded from the scope of this ES research for the reason above and also some of the same reasons described earlier for personalisation.

2.2.13 Click-Through Rate

Click-through rate (CTR) is a ratio (or percentage) showing how often people who see a document listing actually end up clicking it. A high CTR indicates that users find the document listings helpful and relevant. It can be thought of as a measure of user engagement with a document for a particular query.

The CTR ratio is usually expressed as a percentage:

$$CTR = \frac{clicks}{impressions} * 100$$

where impressions are the number of users that saw the document on the results page.

Much research has been carried out by WS providers into harnessing click-through data and using it as a surrogate for relevance judgements [98, 99]. Its application in ES, however, has not been explored.

For WS, CTR has two varieties. Organic CTR is where impression occurs when a link appears in the search engine results exclusively due to the ranking score of a user's query. Paid CTR refers to any traffic generated as a result of a paid promotion (overriding the ranking model). As ES does not generally include a sponsored element, all CTR can be considered organic.

Clickthrough data is naturally 'noisy' [98] and the reliability of the subsequent CTR metric may be reduced by so-called 'good abandonment'. This phenomenon occurs when an end-user abandons their search without clicking because their information need is already satisfied by reading the direct answers from the snippet presented on the Search Engine Results Page (SERP). In this case, no click is required or recorded. Zhou et al. have studied this phenomenon for open domain question answering (OpenQA) and concluded that its impact is negligible as 'the majority of contingency answers were likely to invite clicks' if users are intrigued by the snippet [100].

The paper titled "A model to estimate intrinsic document relevance from the click-through logs of a web search engine" by Georges Dupret and Ciya Liao presents a novel approach to interpreting user interactions in search engines by focusing on user behaviour *after* clicking on a document rather than on CTR alone. The authors introduce a 'session utility model', which estimates document relevance based on the cumulative utility a user gains during a search session. This model is contrasted with existing models, such as the examination and satisfaction models, which emphasise document attractiveness and positional biases [101]. Their study finds that the proposed model performs well in predicting user satisfaction for informational queries with high abandonment rates, making it particularly useful for improving ranking functions in non-navigational contexts. Additionally, the model's relevance estimates were incorporated as features in a Learning to Rank framework, yielding measurable improvements in ranking metrics such as nDCG [101].

In Section §3.7.1 and EXP-3, the clickthrough data extracted from The University's

logs is harnessed to produce an ES ranking function for search results.

2.2.14 Evaluation Metrics for Search Results ranking

Gauging human satisfaction is the ultimate measure of utility, but end-user surveys or polls are not usually a practical approach. Surveys can be "time consuming and expensive to do" or "difficult to do well". [102]. Moreover, retrieval behaviour is often complex and difficult to summarise in one number. Hence, any quantitative evaluation metric is just a proxy feedback measure [102].

In the 1950s, early attempts at evaluation in the field of IR highlighted the fundamental problem of subjective judgements. In one evaluation, two separate teams agreed that 1390 documents were variously relevant to a set of 98 questions but disagreed on a further 1577 documents [103, 104]. Metrics can be categorised into two types: online metrics, which look at end-users' interactions with the search service, and principle offline metrics, which gauge precision and recall (both of which are based on the foundational concept of relevance). These are the two principal offline metrics for gauging aspects of Enterprise Search performance [54]. LTR, semantic search, and QAC each use a different preferred metric. Before we delve deeper into specific metrics, we outline the broad concept of 'utility'.

In 1992, Chris Buckley designed and coded a tool called *trec_eval* [105], which reports over 85 figures derived from roughly twenty core metrics, including traditional measures such as precision and recall, as well as rank-based measures such as mean average precision. Rather than being a mere reporting utility, *trec_eval* encodes a shared understanding of what constitutes retrieval quality and makes it possible to compare systems in a transparent, reproducible way. As a result, it has become the de facto standard for evaluating ad hoc retrieval systems. While most experimental work in information retrieval is expected to report results in its terms [106], many of these metrics have since been reimplemented in Python and are now available through widely used evaluation libraries. This allows researchers to script experiments and compute comparable scores directly in code.

2.2.15 Utility for End-Users

Utility in Information Retrieval is simply defined as the expected usefulness of retrieved information to the user [102]. User satisfaction corresponds to the degree to which a retrieval system maximises utility. But this raises the question as to who 'the user' is. For the end-user, utility is typically associated with the relevance of retrieved results. If the end-users' information need is met quickly and with minimal effort, the system is judged positively. Robertson calls this the "utility of the user" [107], and Buckley similarly describes it as the "worth" an individual assigns to the quality of retrieved output [106]. But there are also multiple types of end-users (e.g., staff or students at The University). Similarly, a patent officer is also a kind of end-user, but would be more interested in recall than a casual searcher.

2.2.16 Utility and Other Stakeholders

Apart from the end-user, other ES stakeholders may include content providers, publishers, analytics teams, marketing units or even auditors. An example would be a mediated scenario where librarians issue a query on behalf of patrons. In this case, the end-user is the final recipient, whereas the librarian is the user.

Various stakeholders may define utility differently (for example in terms of their ability to boost, curate, or 'pin' particular documents, services, or products). Thus, utility depends on the stakeholder perspective, and different groups may prioritise different dimensions of system and ranking behaviour. An example from the domain of ecommerce is where the owners of a website might measure utility based on the proportion of users that click on advertised links. The end-user might prefer if advertised links did not pollute the upper section of the results page.

One approach to utility in commercial web search is to create "compelling user experiences that make users want to come back" [85]. In such settings, return visits, clicks, conversions, query growth, and revenue are often used as utility-based metrics [108]. For ES, users are typically employees [6] who constitute a largely captive audience and do not have the option to switch to an alternative internal search system (as ES involves retrieving information from within an organisation's own information repositories). As a result, return-usage metrics are less informative for ES, and utility must instead be understood in terms of task efficiency, relevance, and reduced friction in finding organisational information [109].

Dai et al. have devised a utility-based LTR system which they have named 'U-Rank'. They define utility as the expected weighted clicks of the ranking, where weights are the per-item utility values. U-Rank diverges from the probabilistic ranking principle (PRP) [110] to rank items in a real-world appstore [108]. PRP asserts that a reference retrieval system is based on a ranked list of best-estimate relevance, sorted in decreasing order of probability, using all available relevance data. They argue that a user's attention on items depends on the positions and other 'item attributes'. An attractive thumbnail image, for example, will attract users' attention even if placed lower down on the list. U-rank directly optimises utility by differentiating between 'item-level' and 'query-level' features. Using a mixture of offline and online evaluation, they detected an average improvement of about 20% CTR when using U-Rank over methods based on probabilistic ranking principle [108].

For the purpose of this thesis, which focuses on ranking, utility is understood as the end-users' satisfaction with the order in which results are presented on the search results page.

2.2.17 Precision & Recall

Most information retrieval (IR) evaluation metrics are derived from *Precision* and *Recall*. Precision measures the proportion of retrieved documents that are relevant, while recall quantifies the proportion of all relevant documents successfully retrieved. These metrics are interdependent and must be analysed together, as in-

terpreting them in isolation can lead to misleading conclusions. For example, a perfect recall score of 1 (100%) can be trivially achieved by returning the entire document corpus for every query—but this would result in poor Precision, as most retrieved documents would be irrelevant.

The connection between ranking and recall is fundamental to retrieval system performance. When evaluated as a function of rank:

- Precision typically decreases as more documents are retrieved, since lower-ranked results are less likely to be relevant.
- Recall always increases with rank, as retrieving more documents improves the chance of capturing all relevant ones.

If a system prioritises high recall, it must retrieve a larger set of documents, inevitably including lower-relevance results. This directly reduces precision, particularly at higher ranks, because the pool of returned documents becomes noisier. A search engine optimised for recall might push marginally relevant (or even irrelevant) documents into the top ranks. This trade-off is seen in EXP-5, where a ranking function is created that attempts to increase both ranking and recall simultaneously.

Across multiple queries, precision generally declines as recall increases, reflecting the inherent tension between these metrics. To produce smooth precision-recall curves, interpolation is applied while enforcing monotonicity. However, precision is highly sensitive to the size of the retrieved set. Buckley demonstrates that the precision of the top 100 documents can shift from 0.2 to 0.3 merely by expanding the retrieval pool from 20 to 30 documents [106]. This variability is particularly problematic for ES, where datasets are smaller than those used by commercial WS, making it difficult to achieve high recall without sacrificing precision.

2.2.18 Mean Average Precision

Average Precision (AP) serves as a single-figure metric suitable for comparing and tuning retrieval systems [111]. AP calculates the mean of precision scores at each point where a relevant document appears in the ranked results, assigning zero precision to any relevant documents that are not retrieved.

The Mean Average Precision (MAP) metric extends evaluation by averaging Average Precision (AP) scores across multiple queries. MAP emphasizes the top ranked results, as AP heavily weights early precision [112]. A relevant document at rank 1 contributes more to the score than one at rank 2, though not strictly twice as much as the relationship is logarithmic, not linear. This characteristic aligns well with ES scenarios, where users under time pressure may often examine only the first few results.

The binary nature of MAP (i.e., a document is either relevant or not) potentially obscures nuanced relevance distinctions that might be valuable in certain enterprise contexts, suggesting that MAP should often be used alongside graded-relevance metrics (such as nDCG described below) in comprehensive evaluations.

2.2.19 DCG & nDCG

The use of a Likert-type scale for relevance provides a more nuanced alternative to binary judgements, better capturing the varying degrees of document utility encountered in real world search scenarios. Cumulative Gain (CG) offers a graded counterpart to traditional precision metrics. Indeed, the CG metric reverts to precision, when applied to a simple binary relevance scale.

Discounted Cumulative Gain (DCG) measures the practical utility of ranked results from a user's perspective during sequential browsing [18, 113]. DCG quantifies the deviation from an ideal ranking (as defined by relevance judgements). The normalised variant (nDCG) simply scales this to a [0,1] range by dividing by the maximum achievable gain for each query.

Within ranking evaluation, nDCG has emerged as the standard metric for assessing graded relevance performance [114]. Unlike binary metrics, nDCG is inherently rank-aware. It quantitatively rewards systems that place documents that are more relevant higher in results lists. This proves particularly valuable in ES contexts, where documents can be awarded varying relevance levels by subject matter experts.

The nDCG scores tend to be higher when relevance judgments are based on frequent keywords and core topics rather than rare terms. This occurs for the following reasons:

- Popular / keyword terms generate more consistent relevance signals across judges (see Inter-Annotator Agreement discussion below). Conversely, rarer terms are likely to exhibit higher judge disagreement due to specialised interpretations.
- Core topics typically have clearer relevance boundaries in documentation. These boundaries are often marked within the 'microsite' or departmental hierarchy.
- Systems trained and optimised for common queries naturally perform better on keyword vocabulary

Errors in top rankings (e.g., misplaced highly relevant documents) degrade user experience more severely than equivalent errors in lower positions. The logarithmic discount factor (DC) explicitly models decreasing examination probabilities with rank. This manifests mathematically through nDCG's cutoff-based formulation:

$$DCG_k = \sum_{i=1}^{k} \frac{2^{rel_i} - 1}{\log_b(i+1)}$$

where rel_i is the graded relevance at position i, and $b = 2$ is the conventional discount base. The standard cutoff $k = 10$ aligns with typical SERP pagination, though query-specific tuning (as demonstrated by Karmaker et al. [113] on the 'MSLR-WEB10K' [31] and 'MQ2007' [115] datasets) can optimise nDCG for particular use cases.

The ranking error complement $(1 - nDCG)$ provides an intuitive measure of system improvement potential. Notably, unlike MAP (which evaluates complete rankings), nDCG@k explicitly accounts for truncated user examination. This is a critical distinction for enterprise environments where search depth may vary according to how busy the searcher is or how urgent the information need.

2.2.20 Inter-Annotator Agreement

Inter-Annotator Agreement (IAA) quantifies the degree of consensus between independent raters when categorising or evaluating the same phenomena [116].

The choice of metric for calculating IAA fundamentally depends on the measurement scale employed (nominal, ordinal, etc.). While Percentage Agreement (PA) offers apparent simplicity, it typically proves inadequate for rigorous IAA assessment as it fails to account for chance agreements. Consequently PA often inflates reliability estimates [117]. Cohen's κ remains the standard for nominal data analysis [118, 119], though its native form exhibits two key limitations. Firstly, it treats all disagreements equally. Secondly, it cannot accommodate ordinal scales. These shortcomings become problematic when analysing Likert-scale data, where the magnitude of disagreement matters (i.e. a difference of one scale point like 1 to 2 should be treated as less severe than larger discrepancies like 1 to 3).

To address these limitations, researchers have developed more sophisticated measures. Weighted κ variants and Krippendorff's α [116] incorporate graduated penalty systems that appropriately weight disagreements by their severity. These metrics form part of the broader intra-class correlation (ICC) family, designed specifically for ordinal and interval data where the distance between ratings carries meaningful information.

The interpretation of IAA metrics requires careful attention to their established guidelines. For Krippendorff's α, values $\geq$0.667 permit "tentative conclusions" [117]. For Cohen's κ, values $\geq$0.8 are required for "substantial agreement" [120, 121]

However, the determination of weights for weighted *kappa* remains inherently subjective, with potential for expert disagreement in specific contexts [121]. This methodological variability (encompassing choice of metric, weighting schemes, and interpretation thresholds) necessitates cautious treatment of IAA results. McHugh pragmatically advises calculating both percentage agreement and *kappa*, with the choice between them depending on rater training levels and the likelihood of guessing [122]. In EXP-4, the IAA study employs the PA, κ and α metrics to gauge agreement between two judges for their Q-D annotations.

2.2.21 Gaps in the Literature

This section reviewed the literature pertaining to ranking of search results and revealed several gaps.

- The lack of enterprise specific ranking signals. Traditional ranking functions (e.g., BM25, anchor text) are optimised for WS but fail to account for ES specific signals such as document recency in intranets, internal authority metrics,

or organisational hierarchies. While link analysis (e.g., PageRank) is well studied for WS, its applicability to ES is limited due to sparse hyperlink structures between document in enterprise corpora.

- The over reliance on lexical matching. Current ES ranking methods prioritise term-frequency-based relevance (e.g., BM25) but neglect semantic relationships between queries and documents. This leads to poor recall when users employ colloquial terms or jargon not present in the indexed content.
- The limited integration of implicit feedback. While click-through data is widely used in WS to infer document relevance, its utility in ES remains unexplored.

These gaps underscore the need for ranking models that combine traditional relevance signals with ES-specific features (e.g., departmental structure, document recency) and leverage both explicit and implicit feedback. Addressing these limitations would align ranking performance with the high expectations set by commercial search engines, as outlined in PS-1.

2.3 Ranking of Query Auto-Completions

The application of ranking in ES is not limited to the presentation order of search results; it is also important for optimising Query Auto-Completion (QAC) by prioritising relevant candidates for a typed prefix.

2.3.1 Definition: Query Auto-Completion

QAC is a predictive feature within search systems that offers users a ranked dropdown list of query suggestions as they start typing their query in a search box. QAC facilitates the search process by making it easier for users to finish entering their queries without typing all the letters as it dynamically updates suggestions based on incremental input.

QAC helps users in expressing their information need when they have an intent in mind but may not yet know how best to formulate it as a query [20]. In practice, several suggestions may be acceptable ways of expressing that intent, and QAC therefore offers a shortlist of plausible candidates rather than implying a single 'correct' completion. This enables users to select from a curated list (whichever option appears most appropriate) or continue refining their query until it is submitted.

For evaluation purposes, however, offline studies typically designate one annotated 'golden' candidate as the target completion for a given prefix so that ranking accuracy can be measured. This distinction allows user-facing ambiguity while still enabling formal assessment of system performance.

2.3.2 Evolution of QAC

An early use of auto-completion was in the Unix Shell, where pressing the tab key presented a list of all file names that started with whatever has been typed [123].

Google launched QAC as a beta feature in 2004[5] and formally incorporated QAC into its search engine in 2008 and since then it has gradually become an expected feature of any search service (including Enterprise Search) and has become a ubiquitous component of both commercial WS, ecommerce and, to a lesser extent, ES services.

2.3.3 QAC for Enterprise Search

In ES implementations, Query Auto-Completion serves critical functions that distinguish it from general web search applications:

- QAC prompts query reformulation which helps users avoid submitting queries that produce no results (a potentially common occurrence, as an ES corpus is small compared to those used by commercial WS engines) [6].
- QAC for ES can educate newcomers about the range of selections available to them and assist in narrowing that selection even before the user has finished typing a query. Contextual suggestions can steer searchers to use the appropriate and official organisational jargon/terminology.
- Users of Google's WS service have collectively 'saved over 200 years of typing time per day' [22]. But, unlike WS, searches on an ES service are usually undertaken by 'knowledge workers' attempting to carry out a specific work task. Any delay in retrieving the desired information or document has a monetary cost for the organisation.
- Under the Web Accessibility Directive (WAD), employers should take reasonable steps to meet EU website accessibility standards and requirements to accommodate employees with disabilities [124]. For ES, QAC has the potential to assist disabled personnel in matters of keyboard accessibility and ease of use. QAC is said to reduce keystrokes by 50% for WS users [125]. Similarly, UK government guidelines for website accessibility requirements state that autocomplete may be necessary to conform to Web Content Accessibility Guidelines (WCAG) [126].
- For commercial WS services, the search box typically occupies the centre of an otherwise blank page. Major providers such as Google, Yahoo, Bing, and Baidu offer ten auto-complete suggestions. For ES, the search box is less prominent and has limited real estate to present suggestions. For this reason, many ES services present fewer suggestion candidates. Additionally, more suggestions could cause users (especially on mobile devices) to either begin to ignore suggestions (at which point the additional suggestions become mere noise) or spend an inordinate amount of time reading suggestions (interrupting or even halting the flow of their search session) [127]. Where an ES service presents a restricted number of suggestions, their ranking is more important.

A disadvantage of QAC is that it places a cognitive load and distraction on the end-user whilst typing the query prefix [20]. Traditionally, this overload/distraction has been assuaged by limiting the number of query candidates to a small number

[5] https://googleblog.blogspot.com/2004/12/ive-got-suggestion.html, accessed 15th May, 2023

(typically 5 or 10) or by waiting until the nth character is typed before offering candidates.

2.3.4 QAC Components

Depending on a researcher's field of study, the terminology used to describe QAC varies widely. A user's incomplete input is often referred to as a 'query prefix'. The generated query suffixes or suggestions are 'query candidates' or 'query completions'. Lesser used terminologies used to describe the same or similar functionality include typeahead, query transformation [40], 'search as you type' [61], predictive search, auto-suggest, real-time query expansion (RTQE) [128], query modification suggestions [6] and subword completion [129].

A closely related and overlapping concept to QAC is 'auto-complete suggestion', which is a more general term that often encompasses a broader range of features aimed at assisting users in formulating queries that do not necessarily contain the same starting string of characters. Google differentiates these two concepts by using the word 'predictions' rather than 'suggestions'. For predictions, the priority is to faithfully 'help people complete a search they were intending to do, not to suggest new types of searches to be performed' [22]. In practice, since QAC may also include the related tasks of suggestion, correction (e.g., spelling reformulation) and expansion, the terms QAC and query suggestion are usually used interchangeably [25,130], as is the case in this study.

2.3.5 Ranking Signals for QAC

Users' search history is generally the most likely indicator of current search intent [131,132]. Extracting suggestions from historical log data assumes that current and future query popularity distribution will remain the same as previously observed. A generalised ranking model is required to handle *as-yet-unseen* queries. LTR can be used to combine multiple features with the optimum weighting contribution to a ranking model. When a user types a query prefix, Apache Solr (the search platform used in this study) can retrieve and dynamically score suggestions from multiple sources in real time using a 'weightExpression'. This mechanism underpins the implementation of the QAC methods examined in later experiments.

In the field of QAC, the state-of-the-art research literature outlines two broad approaches for producing optimally ranked query candidates [20]:

- Traditional: This uses learned relationships between hand-crafted features to generate a ranking model. Query popularity, recency (trending), and a user's historic personal preferences are obvious features when generating query candidates. This traditional approach is heuristic and combines domain-specific signals from a corpus and query logs. This approach, where ranking signals are hand-crafted based on relevance or popularity, typically uses experimental or trial-and-error methods to apply weightings to features. The heuristic approach produces ranking models that are relatively transparent.
- Neural Models: Uses context modelling to determine the likelihood that a query would be issued for a given prefix. The second approach employs NLP

or Language Modelling to produce context-aware suggestions [70, 129].

A heuristic query autocomplete model relies on rule-based methods and simple algorithms to suggest possible completions for a prefix. The heuristics are often based on empirical observations and predefined criteria rather than complex statistical models or machine learning algorithms. The features that make up a heuristic model are usually simpler to implement (and understand) and are computationally less intensive compared to a neural method Heuristic models are usually less accurate than those generated using an AI approach, as they cannot leverage the full complexity of data and context [77, 102].

Both of these approaches treat QAC as a re-ranking problem as shown in Figure 2.4. The re-ranking process has also been called 'generate-then-rank two-step framework' [133]. Learning to Rank can combine or 'fuse' the two approaches with any number of features [134, 135] as outlined in §4.7.2.

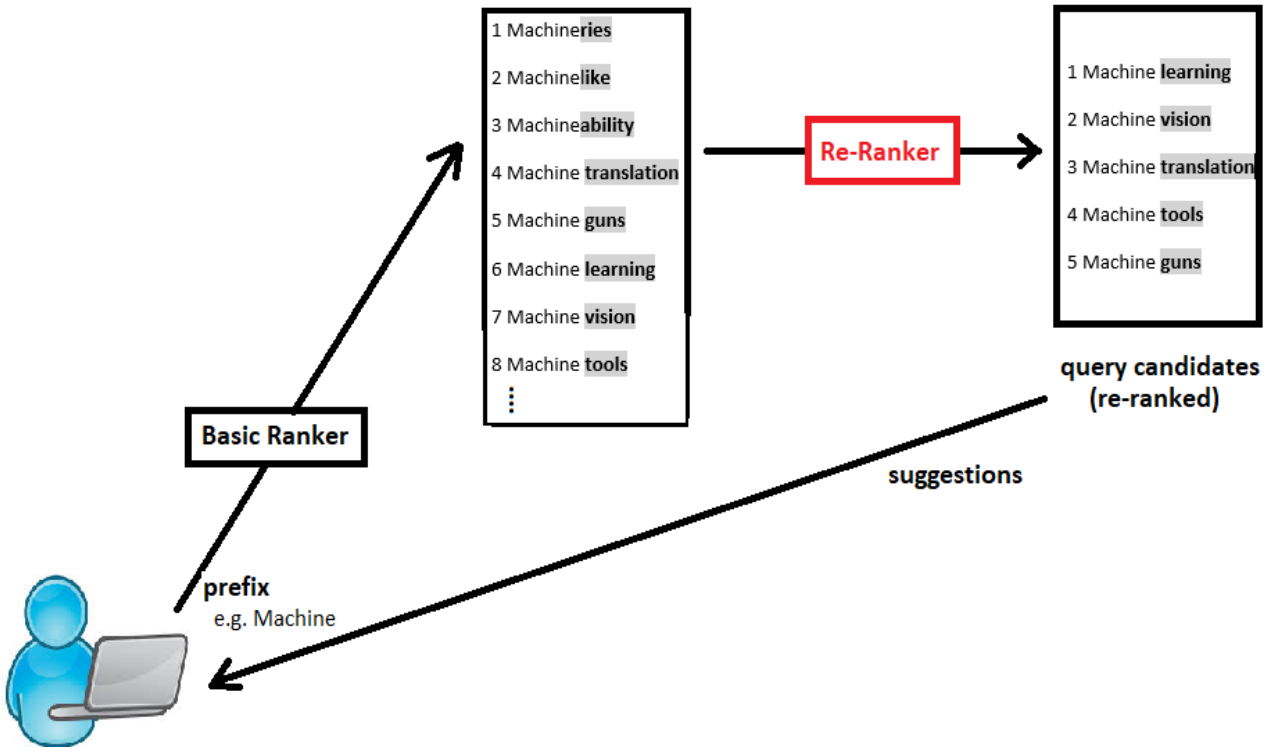


Figure 2.4: Query Auto-Completion as a re-ranking problem. The 're-ranker' transformation (in red) generally use either a Neural or LTR approach to improve on basic ranking or sorting.

Most research focuses on improving the relevance of suggested queries using a ranking model trained and evaluated with data from datasets generated by commercial search engines, such as AOL [26]. In the case of Enterprise Search, where organisations have their own jargon/terminology, the ranking model will need to behave well for unseen prefixes.

In Chapter 4 (Section §4.7.1 in particular), a QAC ranking model will be generated that harnesses optimised weighted signals from multiple ranking functions.

2.3.6 Datasets for QAC

The names and properties of publicly available query log datasets are listed in table 2.1. These datasets record the intended query for given prefixes.

Table 2.1: Characteristics of popular QAC datasets used for evaluating ranking of query candidates.

Properties / Dataset	pub. year	language	queries	URLs
AOL [26]	2006	English	10,154,742	1.6m
MSN	2006	English	8,831,281	4.9m
SogouQ	2008	Chinese	8,939,569	15.1m

Much research for QAC focuses on improving the relevance of suggested queries using a ranking model trained and evaluated with data generated by commercial search engines, such as the 2006 AOL WS dataset [26], which includes over 10 million queries. In §4.3. we demonstrate that the AOL query history is sweeping, centred around popular culture, often archaic or not applicable to the domain of ES.

2.3.7 Historical QAC Data

Relevance judgements in the form of annotated *query-document* pairs are typically required to train a ranking model for documents on the Search Engine Results Page [98,136]. QAC researchers rarely rely on the same editorial effort to manually annotate *prefix-candidate pairs*. More often, QAC relies on the collection of large-scale historical data for QAC tasks [131]. Previously recorded behaviour and queries provide useful information for any user's intent and can be leveraged to suggest completions that are more relevant while adhering to the user's prefix [130].

Most Popular Completions (MPC) is a ranking feature that proposes completion candidates based on user preferences recorded in historical search data. It is a tuple consisting of both the prefix and the associated frequently selected candidates. For this reason, and because of a reluctance of users to provide explicit feedback [6], MPC is typically considered the main indicator of historical relevance [25] and thus considered to be a credible proxy for ground truth in many datasets.

A similar ranking feature extracted from historical logs is Most Frequent Queries (MFQ) [130]. These queries represent the topics or keywords searched for most frequently by the user community. MFQ simply ranks the most frequent suggestions matching the input prefix and is particularly useful when MPC data is sparse.

Figure 2.5 compares the shape of the Search Demand Curves between WS and ES. WS query history, extracted from the AOL dataset between 01st March 2006 and 31st March 2006 (Section §2.5) includes 7,841,462 AOL queries. Two years of ES query history extracted from The University's log data contain 62,044 unique queries.

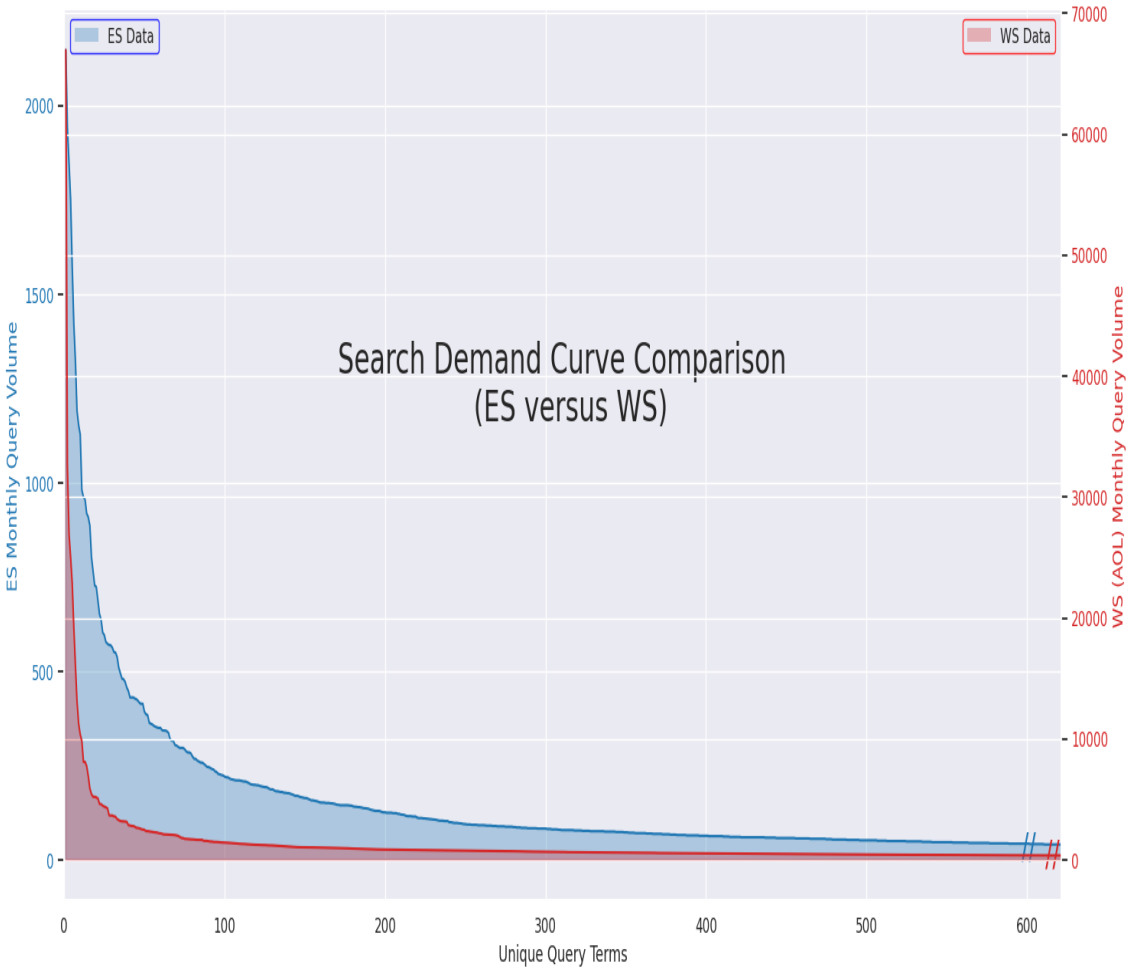


Figure 2.5: Search Demand Curves comparing ES and WS query history..

For the WS dataset, data is recorded over one month, but obviously with greater query volume. Since the Search Demand Curves of ES and WS are similar in shape, it is inferred that the ES data is sufficiently sized (this assumes that the AOL dataset itself was sufficiently sized to begin with).

Personalisation of an individual user's session or recent history enables 'contextual suggestions', which have proved very effective for completing a user's query prefix in Web Search [137]. For example, if a particular user submits a Google search for "Past American Presidents", then if their next query prefix starts with an 'N', the suggestion will be 'Richard Nixon'. The use of personalisation for QAC in the domain of Enterprise Search seems to be rare, possibly because 'profiling' may be prohibited in the organisation (as discussed in Section §2.2.11).

2.3.8 Trending Queries

Queries that have been popular in a recent time period merit a ranking feature to capture temporal behavioural trends. For a number of years from 1999, the operators of the Lycos commercial Web Search engine began publishing a weekly list of the 50 most popular queries submitted. The query term 'Britney Spears' was number two

on the weekly list. The popularity of that term endured, and 'Britney Spears' never fell off the list over the next eight years. This meant that the list was relatively static, and emergent topics were volumetrically drowned out. This has sometimes been referred to as 'The Britney Spears Problem' [138].

A more dynamic list tells us what topic or query is *up and coming* or generating a *buzz* as measured by a sudden abnormal burst of interest. This concept is known as trending [132] and is based on the relative spike in the volume of clustered topic searches in relation to the absolute volume of searches [125]. Unlike the popularity list, the trend list would exclude the constantly popular 'Britney Spears'.

Although trending analysis is most commonly discussed in the context of WS and social media, the concept is also relevant to ES because unexpected surges in internal query behaviour (for example, around new policies, incidents, or organisational announcements) can help administrators understand emerging information needs and evaluate whether the search system surfaces appropriate content during such periods.

2.3.9 Terms extracted from the corpus

The preceding sections describe how historical log data can be harnessed to create candidates. Separately, candidates can also be retrieved from the corpus. Enterprise Search engines, such as Apache Solr [14] are designed for the explicit retrieval of terms within a particular field of a schema, such as title, content body, anchors, footers, etc. A candidate indexed from the anchor text within a corpus is likely to be more important than a candidate from the content field.

2.3.10 Evaluation Metrics for QAC

The evaluation of QAC performance has two general approaches [131], each of which has its own metric:

1. The Mean Reciprocal Rank (MRR) metric focuses on the quality of ranking.
2. The Minimum Keystroke Length (MKS) metric focuses on savings of a user's keystroke effort [139].

Since this study focuses on ranking rather than keystroke effort, we compute MRR, which is widely accepted as the principal metric for evaluating QAC ranking performance [20, 25].
The MRR metric is appropriate for evaluating systems that return an ordered list of candidate responses for a given query prefix. For each prefix, the Reciprocal Rank (RR) is computed by judging relevance across the full list and taking the multiplicative inverse of the rank position of the first suggestion that matches the user's intent (with RR=0 if no suitable suggestions are listed). The system doesn't need to rank a specific 'golden' item first; any relevant item can qualify as the 'first appropriate' one. While datasets often annotate a single 'golden' suggestion as the primary match, the metric accommodates multiple suitable candidates by prioritising the highest-ranked one [70]. This yields a score of 1 if the first relevant suggestion appears in position 1, 1/3 if in position 3, and so on.

The mean reciprocal rank is the average of the RR results for a sample of queries Q:

$$MRR = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{rank_i}$$

where $rank_i$ refers to the rank position of the first relevant document for the i^{th} query. MRR only considers the rank of the first relevant candidate (if there are further relevant candidates, they are ignored).

The MRR metric has two main shortcomings. Firstly, only the rank of the first relevant candidate is considered. If there are further relevant candidates, they are ignored. Secondly, since MRR is the mean of all RR scores, all prefixes are weighted equally. But we know that some prefixes are easier to complete than others. For example, generating candidates for the prefix "machin" is easier than for a prefix like "ma", which would have a vastly larger volume of potential candidates [20].

In the case of search engine results, users are already committed and engaged with the search process. Hence, users are less likely to stop at the first 'correct' document. This is why MAP and nDCG are the preferred metrics for Enterprise Search results, whereas MRR has become the standard measure for evaluating the effectiveness of QAC rankings [20].

Similar to MRR is Mean Recoverable Length (MRL), which is specifically designed for Query Auto-Complete and measures how many upcoming characters the model could complete correctly. MRL also eliminates the need for prefix length sampling for existing rank-based metrics. Kim [129] expresses mean recoverable length as the average of the recoverable lengths of results for a sample of queries Q:

$$MRL = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{RL_i}$$

where RL is the number of characters before the first position where candidates do not have the query.

2.3.11 Gaps in the Literature

The literature on QAC reveals critical gaps when applied to ES. These gaps underscore the disconnect between QAC methods designed for WS and the unique requirements of organisational environments and enterprise content.

- Existing QAC datasets are derived from WS logs and fail to capture ES specific queries (e.g., keywords, internal jargon, alphanumeric codes). This limits the development of ranking models tailored to organisational content.
- Ranking signals for QAC candidates in the literature are derived from WS. ES specific ranking signals are not addressed.
- Relevance blindness and query formulation challenges. WS QAC relies heavily on query popularity (MPC/MFQ) and personalisation (Section §2.2.11), but ES queries often target known items rather than exploratory intent.

- Inadequate Handling of Jargon/Terminology. Current QAC systems lack mechanisms to detect or decode organisation-specific language. Newcomers face barriers in query formulation, as QAC fails to bridge the gap between their intent and internal terminology.

By addressing these gaps, QAC can evolve from a WS-derived convenience tool to a critical ES capability that mitigates vocabulary mismatch (PS-2, PS-3) and, more generally, elevates ranking performance (PS-1).

2.4 Learning to Rank Method

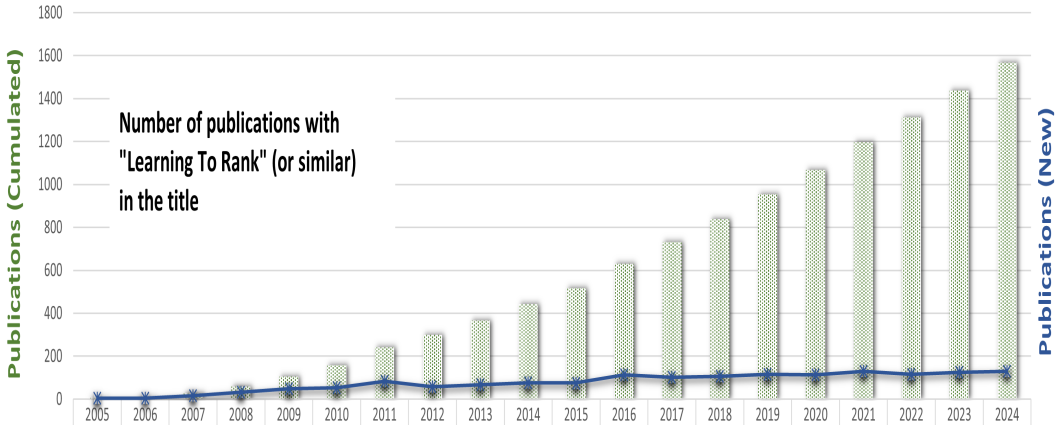
LTR is a supervised machine learning approach that optimises the ranking order of search result documents (or query auto-completion candidates) by training on labelled data [29, 38]. LTR combines multiple ranking functions (referred to as 'features' in the context of LTR) into a unified relevance model. Unlike heuristic ranking methods, which rely on static, hand-crafted rules, LTR adaptively learns ranking functions from data, uncovering complex relevance relationships that are not easily observable through visual inspection.

Traditional (heuristic) ranking methods leverage domain-specific knowledge and empirical observations to design fixed scoring weights (e.g., BM25, TF-IDF). These rules are interpretable, straightforward to implement [140], and can be stacked using query re-ranking [141]. But their rigidity introduces limitations:

- The use of multiple re-ranking functions risks permanently losing relevant items at each pruning stage. Pruning is the discarding of documents early in a multi-stage retrieval pipeline. Pruning is a necessary efficiency optimisation, but it carries the risk that relevant documents may never reach later re-ranking stages if they are filtered out too early. Each re-ranking step only operates on the subset retained by its predecessor, so low scored relevant items can be permanently lost from the recall pool [142].
- Hand crafted rules risk inheriting the assumptions, bias and other 'blind spots' of their creators [134].
- Even if a relevance engineer is very familiar with their corpus content, it is extremely difficult to objectively gauge the weighted ratios between the various ranking functions.

LTR addresses these shortcomings by framing ranking as a machine learning problem. Instead of assigning 'best guess' weightings to scoring functions, LTR models learn to combine (rather than stack) features.

LTR training labels may be explicit (i.e., human-annotated relevance judgements) or implicit (e.g., annotations extracted from clickthrough data). This data driven approach enables LTR to outperform heuristic baselines in complex, dynamic retrieval scenarios.



	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020	2021	2022	2023	2024
Cumulated	5	9	26	58	106	159	241	299	366	442	517	631	732	839	954	1068	1197	1312	1437	1567
New	5	4	17	32	48	53	82	58	67	76	75	114	101	107	115	114	129	115	125	130

Figure 2.6: Number of English-language published research papers with "Learning to Rank" in the title (data source Google Scholar [https://scholar.goggle.com, accessed 18th May 2025])

2.4.1 Evolution of LTR

Before the advent of LTR, multiple ranking functions could be applied using query re-ranking. Apart from being slow and compute-intensive, this had a serious drawback in terms of recall, where each re-ranking stage only operated on the documents passed to it by the previous stage (as described above) [143]. In contrast, modern LTR architectures typically employ a two stage ranking process (an initial retrieval phase that returns a broad set of candidate documents, followed by an LTR re-ranking phase that applies a more sophisticated, feature based model to refine the ordering of this smaller subset).

Before the widespread adoption of LTR, one probabilistic framework attempted to mitigate the above was the *probFuse* method [144], which computed ranking scores for documents in a 'fused result set'. But while probFuse resembled LTR models, it was primarily a data fusion technique rather than a pure LTR algorithm. ProbFuse is thus firmly rooted within the evolution of ranking optimisation and, therefore, is contributor to subsequent LTR frameworks.

Since its introduction in 2005, Learning to Rank has become a hot research topic, no doubt spurred on by commercial Web Search engine providers [145]. Figure 2.6 depicts the recent growth in publications on this topic within the field. This growth was likely influenced firstly by Google making Learning to Rank more accessible with frameworks such as TensorFlow-Ranking [146], and secondly Microsoft and Yahoo releasing datasets specifically designed to foster improvements for Learning to Rank algorithms [29, 30]. It has been reported that major commercial search engines employ ranking models comprised of hundreds of features [143, 147].

In 2024, Baidu still uses LTR, and considers it the "standard workhorse" [27] for ranking search results. Google maintains a highly reticent stance regarding the

specifics of their ranking models (to safeguard its search technology). While they limit disclosures to broad principles, their 2025 public engagements and description of 'RankBrain' suggest that it is also broadly based on an LTR weightings principle [148].

The applications of LTR are not limited to document retrieval, but can also include tasks like question answering, keyphrase extraction, and even translation [77, 108, 149]. Moreover, in a 2023 paper, LTR was reported to be deployed for ranking query auto-completions for a major ecommerce provider [150].

2.4.2 Definitions

Traditionally, LTR has been simply defined as the application of supervised machine learning techniques for training a model to list the best ranking order [149, 151].

Learning to Rank shares some similarities with popular statistical or machine learning problems such as classification and regression. The key difference is in the output of the models. In regression and classification, the task is to predict a label (a value or category, respectively). With LTR, the output is an ordered sequence of items. Unlike regression, an evaluation of the ordered list cannot be based on a distance like 'least-squares' (actual minus predicted).

2.4.3 Architecture

The Solr query re-ranking service allows an initial query (A) for matching documents, which then re-ranks the top k documents, re-scoring them based on a second function (B), as shown in Figure 2.7. The selection of the top k documents is sometimes referred to as 'skimming' [144, 152]. Since the more costly ranking from query B is only applied to the top N documents, it will have less impact on performance than using query B by itself.

The architecture uses re-ranking, and again the potential limitation is that documents which score poorly using the simple query A may not 'make the cut' during the skimming phase, even if those (now discarded documents) have scored highly using function B. A single iteration of re-ranking is used for LTR (this is an improvement over the use of multiple stages of re-ranking).

As shown in Figure 2.8, the key components of machine learning involve inputs, outputs, hypothesis space, and loss functions [77]:

- Output space: this is the output produced by the machine learning framework. In logistic regression [153], this would be, say, a zero or a one. In LTR, the output space is a ranked list.
- Hypothesis space: this defines a class of functions that maps the input to the output space.
- Loss Function: a measure of the degree the generated prediction matches reality (ground truth). According to how many list items are processed in

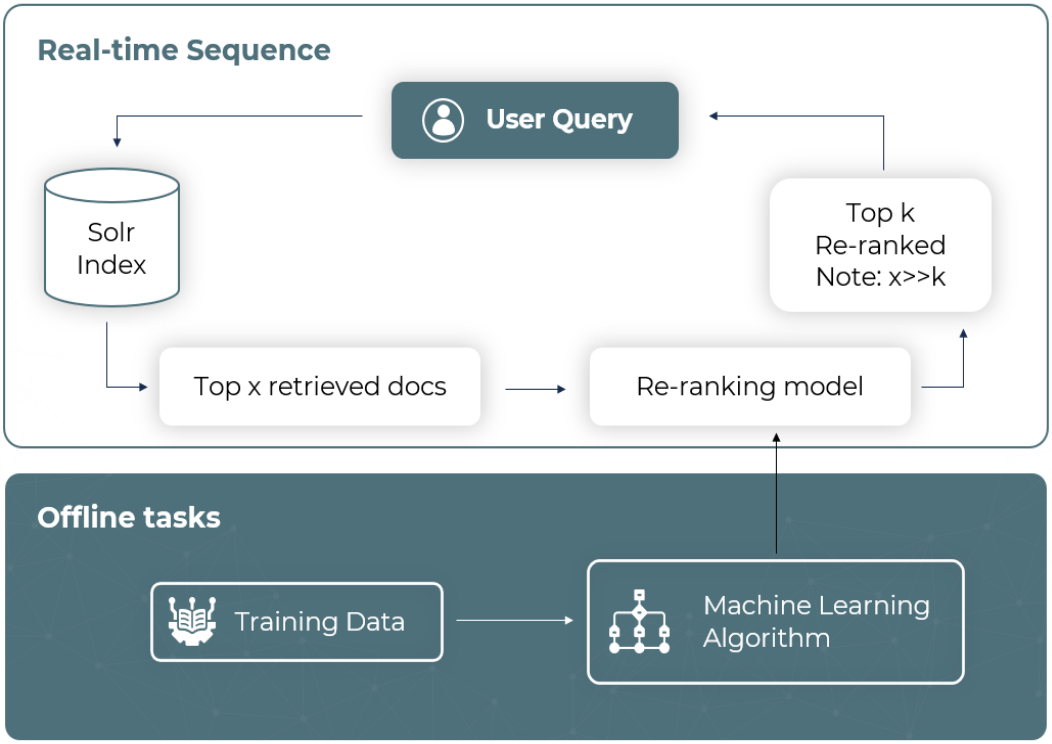


Figure 2.7: The architecture of Solr's implementation of LTR for search results. Re-ranking involves processing a regular query for matching documents and then secondary processing (re-ranking) of the top k documents using the scores from a more sophisticated function. The machine model is trained offline.

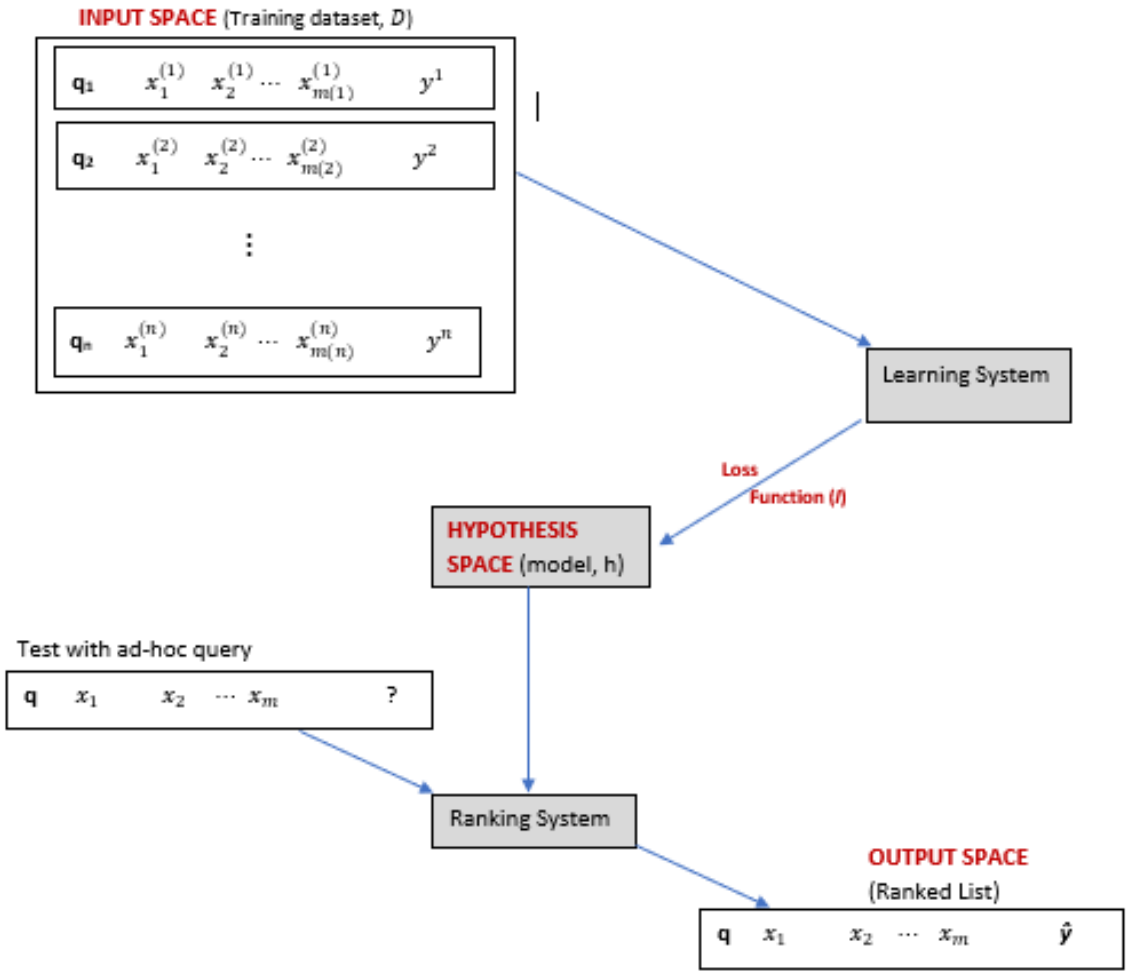


Figure 2.8: A general framework depicting Learning to Rank using Machine Learning (adapted from [77]). Each row in the Training dataset is a trained example which consists of a query q, documents x, and a relevance judgement y.

calculating the ranking loss, learning to rank algorithms can be categorised as point-wise, pair-wise, or list-wise [154]. Much of the research on LTR revolves around the creation and application of optimum loss functions.

Solr has an LTR 'contrib module', that allows for easy integration [155]. Section §3.4 will demonstrate how an LTR model is integrated into The University's live ES service.

2.4.4 Ground Truth

According to the Oxford English Dictionary, 'Ground Truth' refers to 'a fundamental truth; the real or underlying facts'[6]. By this definition, ground truth is the term given to objective (provable) data, which can be used to prove or disprove a research hypothesis. In the domain of information retrieval and machine learning, ground truth refers to the reality we want to model with our supervised machine learning algorithm.

In the subjective discipline of human relevance judgements, the opinion of a small number of expert judges is relied upon for annotations. There are often no hard-and-fast rules to define the objective or ground truth label in all situations. This means that there is an element of subjectivity in the objective. Ground truth cannot be considered a universal truth; evidence for this is any discordance uncovered during IAA studies.

Crowd-sourcing platforms, such as Amazon's Mechanical Turk (AMT), have facilitated explicit labelling by remunerating annotators [156, 157]. While this is appropriate for internet-facing documents or web pages, AMT cannot be used to see the intranet behind an organisation's firewall.

Another coordinated annotation methodology was used in the Text Retrieval Conference (TREC), which employed controlled pools of trained assessors, often with domain expertise (for example, former U.S. military personnel in some tracks), providing greater consistency and reliability in relevance judgements [158].

However, even such a controlled annotation approach would be unsuitable for an enterprise document annotation scenario, as external assessors would not have authorised access to confidential or proprietary organisational material.

An alternative to explicit feedback is to extract implicit relevance feedback from search engine log data. User interactions, such as recorded clicks and dwell time, act as substitutes for annotated judgements. This proxy feedback is referred to as 'click-through data' [159]. Google refer to it simply as 'click data' [160]. The advantage of implicit judgements is that very large quantities of training data can be mined at a low cost. Implicit feedback is 'user-centric', meaning that the bias of expert judges is avoided [161] and can empower organisational ES knowledge bases to be automatically generated from people's activities [51]. The clickthrough data

[6] https://www.oed.com/view/Entry/249130, accessed 19th June 2023

can be converted to CTR, which can be used as a democratic alternative ground truth to the human / experts' institutional approach.

For this reason, some argue that the term 'ground truth' is misleading, disingenuous, and distorts understanding [162].

The 'Gold Standard' is a term commonly applied to medical datasets, where diagnostics can measure objective facts. In the field of IR, researchers generally speak of 'mitigating bias' and subjectivity, but with the understanding that it can never be entirely removed. The term 'ground truth' is used to describe labelling even if the annotations are subjective. In this sense, 'gold standard' is assumed to be the ground truth, and hence the terms are used interchangeably.

Table 2.2 outlines the specific differences between click-through log data and human relevance judgements, as applied to ES. The table illustrates how click-through feedback captures only positive user preferences. Human judgements ought to have greater accuracy as they also include negative feedback (e.g., via a Likert scale).

Table 2.2: Characteristics of human relevance judgements and click-through data as feedback for Enterprise Search.

Characteristic	**Human**	**click-through**
Accuracy	High	Lower
Abundance	Low	High
User preferences	+ve and -ve	+ve only
Domain knowledge	High	Low
Measurement relevance	Absolute	Relative
Principled Approach	Institutional	Democratic
Potential Bias	Organisational bias	Position bias

A practical complicating factor for researchers of implicit feedback is that LTR-formatted datasets (see Figure 4.2 for an example) are generally published separately from associated click-through data. So while the 'Gov' corpus of Microsoft's LETOR 3.0 dataset identifies 64 features per document [31], no implicit feedback data is included. This is understandable, as making public a set of Q-D relevance judgements involves much less disclosure than releasing a full text corpus.

2.4.5 Datasets and Dataset Design

Although most organisations deploy some kind of ES service, few have dedicated relevance annotators to generate training data. As a result, the search system may rely on default similarity algorithms (e.g., BM25 in Apache Solr), operating in a largely 'relevance-blind' manner [61].

Cleverly et al. [1] hypothesise that the general scarcity of academic studies on Enterprise Search environments stems from the difficulties of researchers gaining access to corporate environments. Kruschwitz et al. [64] describe a general reluctance of searchers to provide explicit feedback on the usefulness of results returned for a search query.

A test collection based on Enterprise Search is hard to come by. Identifying the cognitive barriers that deter research in this field is a necessary first step to identifying and mitigating their underlying causes [163]. According to Craswell, an enterprise is not inclined to open its intranet to public distribution, even for research [10]. Furthermore, the corporation may decline permission to publish the results of any research carried out on Enterprise Search. Most researchers with access to an intranet have access only to that of their institution [43]. Specialist providers, whose livelihood depends on their ES products, are unlikely to publish their research. For all of these reasons, the field of Enterprise Search has been overwhelmed by the large volume of datasets coming from internet web search organisations [1] as shown in Table 2.3.

To rebalance this, TREC instigated a new 'enterprise track' in 2005. It began as the successor to the web track, but the key difference is that the 'Enterprise Track' incorporated non-web data (such as intranet repositories, email archives etc) [10]. One of the key conclusions according to the TREC community is test collections should have a minimum of 25 queries and 50 queries as the norm [106]. About the same time, Hawking [55] examined specialist vocabulary in enterprise search systems, though the research achieved limited success owing to dataset restrictions.

The Cranfield experimental methodology, established in the 1960s, set a critical precedent for assessing and benchmarking the relevance of document collections [106, 164]. Aligning with this benchmarking tradition, the release of Microsoft's LETOR dataset provided a significant impetus for Learning to Rank (LTR) research, offering a standardised framework for training and evaluating machine learning-based ranking models [31].

Much of contemporary LTR research focuses on comparing and evaluating diverse ranking algorithms using the LETOR. However, the LETOR dataset is essentially web content, which differs fundamentally from the hierarchical structure, terminology and content typical of enterprise data [8]. Table 2.3 is a summary of popular LTR-formatted datasets, though many of the source corpora are not available, or require payment to use (Table 2.4).

For the reasons outlined above, no publicly available ES dataset is to be found, much less one that includes both explicit and implicit feedback data suitable for correlation analysis and comparison. This research introduces a new LTR dataset, which has been extracted from the The University corpus and search logs, and is described in Section §3.4.4.

When creating a dataset, Rottger [165] warns that when annotation guidelines lack clarity or completeness, designers must clarify or expand them, typically through iterative annotation processes. As a result, quality assurance requires a lot of work, but it follows a structured approach, where inter-annotator agreement functions as a useful (though imperfect) measure of dataset quality.

Another dataset design consideration pertains to the depth or breadth or Q-D pairs [166]. Yilmaz et al. [167] pose the question: "given a fixed judgment bud-

Table 2.3: Characteristics of popular datasets and associated collections used by learning to rank researchers.

Dataset \ Properties	pub. year	collection	queries	documents	features
Gov	2008	Gov	575	56k K	64
OHSUMED	2008	OHSUMED	106	16 K	64
Gov2	2009	Gov2	2476	85 K	46
MQ2007	2009	TREC	1700	69 K	46
MQ2008	2009	TREC	800	15 K	245
MSLR-Web10K	2009	Gov2	10,000	1.2M	136
MSLR-Web30K	2009	Gov2	30,000	3.1M	136
Yandex	2009	Yandex	20267	213 K	245
Yahoo!	2010	Yahoo!	36251	883 K	700
Robust04	2004	TREC	250	500 K	64
ClueWeb-09-Cat-B	2009	TREC	150	34 M	50
ISTELLA-X	2018	Istella	33018	3.4M	220

Table 2.4: While associated datasets are freely available for download, most of the underlying collections or corpora are private or pay-per-use.

Dataset	**Content**	**Availability**
Gov	Ad-hoc	Free
OHSUMED	Medical	Free
Gov2	WS (.gov)	Paid
MQ2007	Ad-hoc	Free
MQ2008	Ad-hoc	Free
MSLR-WEB10K	WS	Free
MSLR-WEB30K	WS	Free
Yandex	WS	Discontinued
Yahoo!	WS	Licensed
ClueWeb09	WS	Agreement req.
Robust04	Ad-hoc	Licensed
ISTELLA-X	WS	Agreement req.

get, is it better to judge as many queries as possible, with fewer judgments per query (shallow judgments) or to judge fewer queries but more documents per query (deeper judgments)?". Their research is based on a large WS provider and they conclude that shallow judgements are more cost effective when using the LambdaRank [168] algorithm. This trade-off looms large in the context of EXP-5, where its consequences may prove decisive and the wrong judgment strategy could critically undermine the results.

Dataset designers at the Baidu search engine [27] present a framework for improving LTR models in large-scale search engines, focusing on active learning strategies to optimise the labelling process of queries and relevant web pages. The authors propose two criteria named Ranking Entropy (RE) and Prediction Variance (PV). These criteria guide the selection of the most informative queries for labelling. RE measures the uncertainty in ranking outcomes, while PV assesses the diversity of

ranking predictions. Combining these criteria aims to balance query selection. Experiments demonstrate that this approach outperforms existing methods (as evaluated by higher accuracy in ranking results with fewer labelled data points).

2.4.6 Signals and Features

In the context of search results, the goal is to combine signals to present the optimal ranking order of documents for a given query [149, 151]. Similarly, in QAC, signals are combined to determine the best order of query candidates for a typed prefix.

When examining a list of search results, various signals become apparent. A signal may serve as an immediate indicator of a document's relevance or importance. For instance, if a URL contains the number '2015' (as illustrated in Table 2.5), it signals to the user that the document is likely over a decade old. These signals can then be transformed into machine-learning 'features' (measurable data points used to predict the relevance or ranking of documents in search results). Features act as effective data representations [169], or what Siebert terms 'scientometric' measures [170]. They may be statistical, axiomatic, or semantic in nature [171] and are combined into a feature vector array. Broadly, features can be categorised as either query-dependent or query-independent [56].

Query-Dependent Features These features assess whether a document is relevant to the query, making them the most common type of relevance signal. They measure the importance of a query term within a document relative to a corpus. Examples include TF-IDF and BM25.

Query-Independent Features At first glance, these features may seem counter-intuitive, as they rank documents irrespective of query relevance. However, certain signals—such as a document's URL structure or historical page hits—can indicate importance without relying on the query term. Table 2.5 provides examples of such features.

Table 2.5: Query Independent Features: The number of slashes in a URL or recorded page hits are both features unrelated to the query term.

URL	Number of Slashes	Page Hits
example.com/Botany	1	512
example.com/Botany/subdir/2015/schedule.pdf	5	6

Zhou et al. [100] argue that an effective ranking model for enterprise search should incorporate recency and authoritativeness (query-independent features) alongside text-matching (query-dependent) features. Similarly, Zilles [145] emphasises combining 'relevance, importance, and preference', integrating query-dependent, query-independent, and labelled features. In Section §3.4.4, a dataset will be created for The University's ES service, and will be comprised of both query-dependent and query-independent features.

Query Intent for ES Enterprise search often involves 'navigational queries', where the user seeks a specific homepage or known site [57, 172]. Such queries typically have one perfect result—the exact website the user intends to find [77, 131]. Features like the *number of slashes* in a URL can aid in homepage discovery, while the presence of a year (e.g., '2015') may indicate outdated content—another example of a query-independent feature.

Feature Selection and Model Efficiency The number of features in ranking datasets varies significantly (see Table 2.3). While the Yahoo! dataset includes 700 features, MQ2007 has only 46. A larger feature set does not guarantee better ranking performance; instead, it increases dimensionality and the risk of overfitting [173].

Feature selection mitigates overfitting without compromising accuracy. Though well-established in classification, it was only applied to ranking in 2007 [173]. Recent advances include graph-based feature selection methods [174] and Recursive Feature Elimination with Cross-Validation (RFECV) [175]. Reducing features improves computational efficiency and can enhance ranking performance by eliminating irrelevant variables.

Derived Features and Ablation Studies A 'derived feature' is constructed from existing variables, often providing greater predictive power than raw data. For example, 'area' (derived from length and width) allows models to bypass redundant calculations. Such features can also help prevent overfitting.

Ablation studies (or step-wise sensitivity analyses) evaluate model performance by systematically removing features, revealing their individual contributions [176]. Simplifying the feature space in advance aids interpretability and clarifies the impact of specific features on ranking outcomes.

2.4.7 Limitations of Feature Engineering

The major limitation of using features in the context of Machine Learning is that each feature is based on a hunch that it represents a significant signal of relevance. The feature is therefore handcrafted based on expert knowledge.

Feature engineering is usually a time-consuming process [169] in terms of identifying, refining, evaluating. But Apache Solr makes the implementation relatively easy. For example, Listing 2.1 shows how urlLength could be coded and uploaded into an Apache Solr 'feature store'.

```
    "store" : "myCombinedFeaturesStore",
    "name":  "urlLength",
    "class": "org.apache.solr.ltr.feature.FieldLengthFeature",
    "params": {
       "field": "url"
```

Listing 2.1: Example of a feaure in Apache Solr ready for uplaod to the 'feature store'. This feature, called urlLength, simply measures the number of characters in the URL.

The assumption is that the urlLength feature was originally a signal for the user's judgement of importance. Only ground truth will tell if this assumption is indeed correct. This feature is a good example of a ranking signal that might make sense on one corpus, but not another.

Similarly, features are engineered for a particular dataset but may be an insignificant or inappropriate fit for others. Note that [135, 169] refer to this over-fitting as 'over-specified' features. According to Yang [177], care should be taken during the feature engineering process to avoid 'naively' combining features. Field-based features (such as those extracted from body and whole document, title and anchor, or title and URL) may be strongly correlated, and their individual contributions to the ranking performance should be isolated and measured. Pearson's correlation should be used to analyse the correlations among the various field features.

Finally, a limitation of feature engineering is that the feature set may be incomplete. This limitation could occur if there are one or more signals that any judge fails to articulate. For example, there may be subconscious signals, such as the UK versus US spelling of a word of the displayed snippet, that prompt the searcher to choose or avoid a document. Undetected signals mean that the codification of the subsequent feature set will therefore be incomplete [135, 169].

2.4.8 Bias in LTR Datasets

Bias is an inescapable artefact of human cognition. This is evident in search behaviour, where the order in which documents are presented profoundly shapes user decisions. Faced with a list of results, users exhibit a depth-first interaction pattern, prioritising efficiency over thorough evaluation. They tend to click the first seemingly relevant document, often neglecting alternatives—a heuristic that saves time but distorts implicit feedback [178].

Eye-tracking studies by Joachims et al. [179] confirm this tendency: users scan results top-to-bottom and "click on sufficiently promising abstracts without first exploring lower-ranked links".

This behaviour introduces 'position bias', a major challenge for interpreting implicit feedback. Clicks on higher-ranked documents do not necessarily indicate greater relevance, nor do skipped items imply irrelevance. As such, equating clicks/non-clicks with positive/negative feedback is problematic [161]. Griffiths et al. [180] caution against using raw "user satisfaction" metrics, while Agarwal et al. [181] emphasise that debiasing click data is a *prerequisite* for reliable analysis. Naively leveraging unprocessed click-through data risks reinforcing systemic biases [160, 181].

Unbiased Learning to Rank (ULTR) methods address this by correcting for position

bias before training ranking models [161]. Recent work demonstrates ULTR's effectiveness in real-world applications [182]. The issue is further exacerbated by user engagement patterns: a 2014 study of Google desktop searches found only 6% of users navigated beyond the first page, intensifying the skew in click data [183].

According to Chapelle et al., explicit human judgements, while costly and time-consuming to obtain, produce cleaner training data by relying on expert annotators [30]. However, these judgements themselves are intrinsically subjective; annotators often disagree on what constitutes appropriate scoring of Q-D pairs. Moreover, explicit relevance labels may misalign with implicit user preferences [184], highlighting the tension between curated (organisational) and *socially proofed* signals. These tensions are on display in EXP-3 and EXP-4 and a means of mitigating them is discussed in Section §5.5.

2.4.9 LTR Loss Functions

The research on advancing LTR algorithmic performance revolves around the creation and application of optimum 'loss functions'. A Loss Function is a measure of the degree to which the generated prediction matches reality (ground truth). LTR algorithms are categorised according to their loss function. According to the number of list items processed in calculating the ranking loss, learning to rank algorithms can be classified as point-wise, pair-wise, or list-wise [154].

- Point-wise: ranking is treated like a classification or regression problem. The scoring function is used to predict the relevance label of a document on its own.

- Pair-wise: ranking is treated as a set of pair-wise preference prediction tasks. An example is LambdaRank or LambdaMART (described below), which Hu describes as the state-of-the-art ranker [185].

- List-wise: this method builds on the pair-wise method, but it takes a set of documents together (not just a pair) and optimises the final ranking metrics. All instances are considered at once. ListMLE [186] is a popular list-wise algorithm, which optimises likelihood loss over cosine and cross-entropy loss. The XGBoost ML library can generate ranking models using point, pair or list-wise loss functions.

2.4.10 LambdaMART

LambdaMART is a 'list-wise' LTR algorithm that trains boosted regression trees using LambdaRank style gradients. It provides an alternative to point-wise and pair-wise LTR models by directly optimising ranking metrics (such as nDCG or MRR) through gradient signals that encode the desired ordering. While LambdaMART is considered as list-wise at the conceptual level, it is pair-wise at the operational level (it uses pair-wise λ gradients to approximate the list-wise objective).

For a query q, consider a document pair (i, j) with relevance labels $y_i > y_j$ and current model scores s_i and s_j. The pairwise training signal is:

$$\lambda_{ij} = \frac{1}{1 + \exp(s_i - s_j)} \cdot \Delta\text{nDCG}_{ij},$$

where ΔnDCG_{ij} is the change in nDCG that would result from swapping the two documents. Each document's gradient is the sum of its pairwise lambdas:

$$\lambda_i = \sum_j \lambda_{ij}.$$

A regression tree $h_t(\cdot)$ is fitted to these gradients at each boosting step:

$$F_t(x) = F_{t-1}(x) + \eta\, h_t(x),$$

with learning rate η. LambdaMART therefore optimises ranking metrics (e.g., nDCG) by training trees on gradients that encode the desired ordering.

The implementation of LambdaMART to rank a) search results and b) query autocompletions is described in Sections §3.8.1 and 4.7.1 respectively.

2.4.11 Click Model Types

A click model is a framework that interprets user interactions to estimate relevance and improve ranking systems. For this research the interactions being interpreted are a) clicks on documents on the search results page and b) candidate selection for QAC. Click models form a spectrum ranging from basic interaction frameworks to sophisticated systems that can mitigate bias. While often used interchangeably, 'click-through models' generally represent the foundational tier, which is a simpler form that establish the basic relationship between user clicks and item relevance.

Basic click models can serve as workhorses for Q-D or P-C annotation pairs, providing insights by tracking raw user interaction patterns and supporting relevance estimation. In WS, click models effectively infer search importance to a document for a given query [19].

The 'cascading model' exemplifies this category, modelling user behaviour as a strict top down sequential process where each result is evaluated individually before click decisions are made [140].

Sophisticated click models build upon these foundations by incorporating behavioural psychology and statistical adjustments to address inherent biases (such as position or presentation bias). This evolution from basic to advanced models reflects the field's progression from simple click counting to nuanced interpretation of implicit feedback signals [20].

In Chapters 3 and 4, basic click models are used to estimate relevance for documents (EXP-3) and QAC candidates (EXP-6).

2.4.12 Per-Query Ranking Models

A custom ranking approach is to dynamically select and apply the most suitable ranking function or model for each individual query [56, 187]. This approach is

grounded in the observation that different ranking functions perform optimally on different query types. Retrieval performance can thus be improved by tailoring the function/model selection at the per-query level.

A key contribution of Peng's paper [187] is the introduction of divergence as a query-specific feature. Divergence quantifies the degree to which a document ranking function modifies the scores of an initial document ranking for a given query. This metric enables the framework to make more informed decisions when selecting ranking functions, thereby enhancing overall retrieval effectiveness.

This custom per-query ranking model could be useful for ES, where queries can include usernames, purchase code, ticket numbers, staff names, or any other diverse artefact [10].

2.4.13 Gaps in the Literature

As outlined in Section §2.4.1 to Section §2.4.5, the evolution of LTR has been shaped by the availability of benchmark datasets, the rise of scalable ML frameworks, and the deployment of LTR in real-world systems by industry leaders such as Microsoft, Google and Baidu.

Despite these advances, practical barriers to deployment at the enterprise level remain. According to [54], the complexity of ES is such that organisation should form a Centre of Search Excellence (CSE), which would include staff with specialist expertise in fields like LTR and LM. As of 2022, Martin White notes a significant shortage of people with skills to support and manage ES [72]. Furthermore, as a first step in deploying AI methods, it must be determined that an AI approach can improve ES search result ranking. Once this is determined, any performance gain informs the discussion of whether state-of-the-art AI ranking methods are 'worth the effort' of overcoming the practical barriers such as acquiring the staff, skill sets and hardware resources.

Moreover, the high 'cost' of human labelling for LTR, the difficulty of LTR model generalisation across domains, and the operational challenges of frequent model retraining to counteract relevance drift all serve as formidable practical barriers to adoption of LTR for ES.

The subsequent section of this literature review investigate the complementary NLP/LM method, and evaluates how it address some of the constraints associated with LTR for both search results and query auto-completions.

2.5 Language Modelling Method

Unlike traditional ranking methods such as TF-IDF or BM25, the LM method infers user intent by learning the probability distribution of word sequences, thereby enabling a deeper, semantic-level understanding of queries and documents. The LM method differs from traditional lexical search (i.e. where the search looks for literal

matches of the query strings). Google pithily refers to the task of bridging the gap between query and document vocabulary as matching 'things, not strings' [33]).

2.5.1 Definitions

An important distinction must be drawn between the overlapping but distinct concepts of NLP, LM, LLM, and Neural Information Retrieval (NIR). NLP encompasses the broader discipline of enabling machines to process, understand, and generate human language, with applications in IR including query understanding, document summarisation, and relevance ranking. LM is a subfield of NLP that focuses explicitly on predicting word sequences and estimating the probability of textual data [188]. NIR refers to the application of artificial neural networks (such as the BERT LM transformer) to learn complex representations from data [189]. An LLM is a type of artificial intelligence system based on deep learning that is trained on massive amounts of text data to understand, generate, and manipulate human language for a wide range of tasks.

This document will take care to differentiate clearly between these four concepts. However, it should be noted that in both academic and industrial contexts, the terms are sometimes used interchangeably, particularly as neural approaches dominate modern language modelling and NLP applications in information retrieval. The umbrella term 'AI approach' is used to cover all of these concepts.

The current relationship and overlap between these concepts in the context of AI, ML and DL is highlighted in Figure 2.9. Learning to rank has traditionally been classified as an ML concept. Word2Vec is not classified as an LM as it does not explicitly predict word sequences or probabilities. Instead, it is used to support LM tasks by capturing semantic word relationships. Instead, Word2Vec is described as 'representation learning', a subfield focused on encoding linguistic features into machine-interpretable forms. While LM has historically been rooted in ML, it has evolved (via BERT and GPT LLM models) to the point where it is now generally classified as DL. ML methods like LTR are more computationally efficient in terms of training than DL methods [69].

Practitioners in the field use phrases like semantic search [61], concept search [11], deep search [188], and AI-powered search [140] interchangeably to describe search engines that comprehend intent.

2.5.2 Evolution

Over the past fifteen years, LM has emerged as a significant advancement within NLP, particularly for improving ranking in semantic search contexts [135, 169, 190].

NLP originated in the 1950s and emerged as a discipline that integrated the areas of linguistics and computer science. The goal of NLP is (and has been) to enable machines to understand and generate human language. The first practical implementation was by IBM in 1954 and involved translating sentences from Russian to English [191].

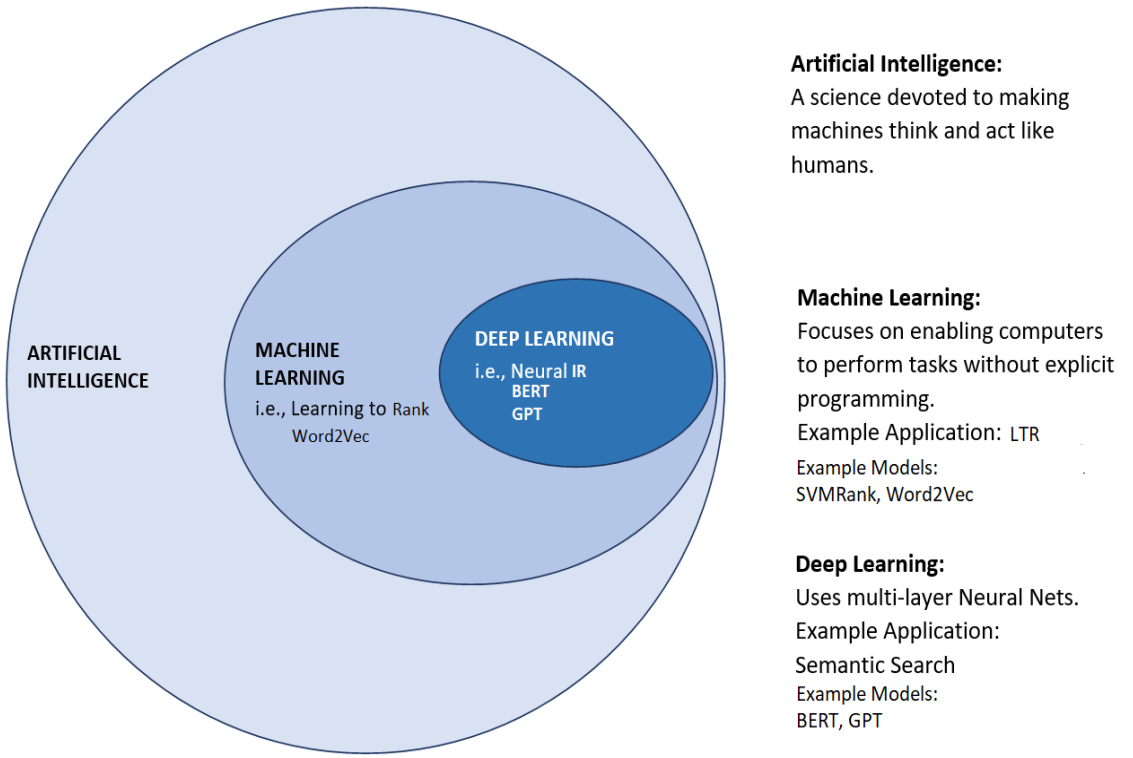


Figure 2.9: How the concepts of AI, ML and DL relate to each other in terms of the methods used in this research.

Word2Vec [34] was introduced in 2013 by a team led by Tomas Mikolov at Google. It revolutionised language modelling by providing scalable, semantically rich word embeddings, enhancing tasks from text processing to early web search ranking. Word2Vec's vector-based representation laid the foundation for contextual language models, shaping modern language modelling. As Word2Vec uses a shallow architecture, it is not classified as a deep learning approach (in Figure 2.9). Nonetheless, it is a neural model that paved the way for deep learning models, such as BERT.

Word2Vec's static embeddings can only assign a single vector per word, ignoring contextual nuances (e.g., 'ticket' can represent a customer support request or an event pass). This limitation prompted the development of transformer-based LM models like Google's Bidirectional Encoder Representations from Transformers (BERT) [192], which can attribute understanding to whole sentences or paragraphs.

Wile BERT was trained on English Wikipedia pages and BookCorpus, it was extended in 2019 to create SciBERT [193]. SciBERT is a pre-trained BERT model specifically designed for scientific text processing. It is trained on a large corpus of 1.14 million scientific papers and therefore has an enhanced understanding of technical and scientific concepts. Similarly, BioBERT [194] is a specialised deep learning model that extends BERT for biomedical NLP, pre-trained on extensive medical literature to enhance understanding of domain-specific language.

In 2019, Facebook developed an improved version of BERT called RoBERTa (Robustly Optimised BERT Approach) [35]. It retains BERT's core architecture of stacked transformer encoder layers, but optimises for byte-pair encryption and is

not restricted to adjacent sentences. RoBERTa achieves better performance on NLU tasks such as sentiment analysis and named entity recognition.

BERT was gradually incorporated into the Google search engine starting in 2019. Among Google users, it marked a paradigm shift toward semantic and intent-driven ranking. This shift was reinforced by subsequent advancements in LLMs like Open AI's GPT [36]. An LLM can contextualise an entire document (or multi-turn conversation), and understands long-range dependencies, nuanced context, and even world knowledge across thousands of tokens. For example, as the GPT-4 LLM can process up to 128K tokens, it can analyse an entire book. LLM's have been integrated into search engines like Bing and Google's Search Generative Experience (SGE) since 2023 [195].

This use of LM for query-document understanding, coupled with the rise of AI generated summaries (described in Section §2.5.6) in WS LLM based search engines has reshaped (i.e. heightened) user expectations for ES.

The classification of the various methods and models in Figure 2.9 is dynamic and has changed over the course of this programme of research. Their classification is liable to change again in the coming years.

2.5.3 Lexical and Semantic Search

Instead of exact term matching, LM/NLP embeddings can be used to find semantically similar terms. The matching problems of LM and traditional retrieval are fundamentally different. In summary:

- LM/NLP is predominantly concerned with the challenge of 'semantic matching' (inferring semantic relations between two pieces of text)
- Traditional retrieval (often referred to as ad-hoc or lexical retrieval) is more concerned with 'relevance matching' (i.e., is a document lexically relevant to the given query?)

A detailed list of difference is outlined in Table 2.7. The differences between these two matching tasks [135] are such that a single AI method cannot accommodate both tasks.

Early deep learning models used Deep Neural Networks (DNN) and Convolutional Neural Networks (CNN) to map Query and Document to a common semantic space [169, 197]. Cosine similarity is then used to calculate a relevance score. Other Neural IR models such as Deep Structured Semantic Models (DSSM) [198] and C-DSSM [199] attempted to map queries and documents into a shared semantic space using techniques like DNNs and CNNs. Cosine similarity between these embeddings was used to estimate relevance. However, these models initially underperformed when evaluated on traditional IR benchmarks such as TREC's Robust04 or ClueWeb09, often trailing behind BM25 and tree-based models [135].

The underperformance was attributed to the fact that early LM-based systems primarily focused on semantic similarity, neglecting non-semantic relevance features

Table 2.7: Semantic matching vs Relevance matching. Neural NLP enables semantic matching, and 'relevance matching' is associated with traditional IR [135]

Retrieval Method	Matching Type	Matching Signals	Terms	Matching Requirements
LM	Semantic	Similarity: relatedness between words, phrases and sentences	Grammatical structures: natural language in sentences or paragraphs	Global: semantic relations between two pieces of text
Traditional IR	Lexical Relevance	Exact: search paradigms prioritize precise matches (exact matches score higher than semantic) [196]	Query term importance: short or keyword-based queries emphasize term matching	Diverse: relevance matching occurs at any document location

such as recency, authority, and user interactions. These features often carry significant weight in enterprise settings where content may have policy, procedural, or compliance implications.

A further limitation of these deep learning approaches is that they do not mimic the holistic human judgement process. Two pieces of text (i.e. in the Query and in the Document) are matched. Non-semantic features (e.g., document popularity or freshness) are excluded. Deep Learning harnesses advances in NLP but sees the challenge primarily as a text or semantic matching task. This distinction is crucial in enterprise contexts. For instance, a document containing relevant policy information should be ranked higher than semantically similar documents that are not topically pertinent. Thus, relevance matching and semantic matching must be integrated, not treated as interchangeable. Relevance matching and semantic matching are complementary building blocks for ranking. Combining the contribution of the relevance matching and semantic matching can be achieved via LTR, which is sometimes described as a 'workhorse' [27] than can bring together the LM and traditional ranking functions.

LMs can decode abbreviations (i.e., initialisms and acronyms) by leveraging their contextual understanding and vast training data, which encompass a wide range of linguistic patterns and domain-specific terminologies. LMs like BERT and GPT, which have been trained on diverse corpora, can infer the meaning of abbreviations based on the surrounding text and semantic relationships [68]. This process typically involves fine-tuning the model on smaller datasets containing the paired abbreviation-expansion examples or employing a masked language modelling approach, where the model predicts the expanded form given the abbreviation and its

context.

2.5.4 Applications of LM in Enterprise Search

The unique characteristics of ES present opportunities that make LM potentially very applicable:

- Enterprises often develop internal lexicons such as **jargon and specialised terminology**. LM techniques such as fine-tuned BERT models can detect these terms and adapt ranking accordingly. The JARGES feature developed in this study (Section §3.11.1) is an implementation of a custom LM method.
- Users may not be familiar with the precise terminology or abbreviations used in official documents. LM helps bridge this **user-query mismatch** by offering QAC synonym generated candidates.
- LM can extend queries to include related concepts. **Semantic expansion** improves recall without (any significant) harming of precision.

The challenge of detecting and decoding enterprise jargon/terminology within a corpus lends itself to the fields of natural language processing (NLP) and LM. Comparative TF-IDF scoring and sparse matrix vectorisation can be used for tasks like plagiarism classification [200] and also to distinguish word or phrase salience patterns between corpora [201]. The TF-IDF divergence calculation is computationally lightweight and can be executed using the standard IT hardware commonly available in most organisations. Word embeddings are another NLP tool that can capture semantic relationships between words in text data through dense vector matrices. LMs, such as Google's BERT [192] captures trained on large datasets containing vast amounts of text from diverse sources. While embeddings and transformers are regularly used in e-commerce [70] and commercial search engines [25], their application for ES has not been sufficiently explored. A 2024 study by Belfathi to classify document genres used BERT to distil 'linguistic attributes' for legal terms and demonstrated elevated ranking performance for specific tasks in the LegalGLUE dataset [201].

The integration of LTR with LM-based features allows systems to capitalise on both exact-match precision and semantic recall. This dual capability is especially beneficial for enterprise use cases that require comprehensive retrieval alongside relevance.

2.5.5 Attention-Based Mechanisms for Ranking

Deep learning approaches, such as attention mechanisms, have been applied to the field of LTR for ranking both image retrieval and text-querying datasets [202]. Using the features of queries and search results as input, neural networks are applied to model ranking probabilities.

Wang et al. [202] use a combination of Recurrent Neural Network (RNN) structures with attention mechanisms and multiple embeddings. They test their model on the

'20 Newsgroup dataset' and rank search results in a listwise approach. MAP and nDCG are used to evaluate their model and achieve favourable results compared to other models such as OASIS and LambdaMART. A limitation is that this model was compared to a limited set of models created from WS corpora (i.e. it was not applied to an ES corpus).

Cross-document interactions represent a departure from traditional ranking concepts. Unlike tree-based approaches for LTR, the cross-document feature score of each individual document is affected by other documents in the collection [203].

Pasumarthi et al. propose a model called Document Interaction Network (attenDIN) with features based on self-attention mechanisms to capture cross-document interactions [203]. They evaluated cross-document interactions of four datasets. Non-neural models such as Gradient Boosted Decision Trees (GBDT) continue to outperform all neural approaches on benchmark datasets. Nonetheless, the cross-document approach outperformed other state-of-the-art neural models.

2.5.6 AI generated summaries

Many of the major WS providers now integrate AI summaries into their interfaces. AI summaries are the automatically generated, concise textual representations of the content found within search results, created by NLP and LM. The summaries aim to enhance user understanding and decision-making by contextualising the relevance of retrieved documents. Users are thus relieved of the need to click ranked pages individually.

In Feb 2023, Microsoft Bing launched AI-powered chat, offering detailed, conversational summaries alongside search results [204]. Similarly, in May 2023, Google introduced Search Generative Experience, using generative AI to provide comprehensive summaries atop the search results page [204].

2.5.7 Gaps in the Literature

While significant progress has been made in applying LM to WS, enterprise-specific applications have been overlooked. This review highlights the importance of customising LM methods for ES content, particularly for handling jargon, improving semantic recall, and aligning with organisational information structures.

LM ranking models may lack transparency, which is potentially a problem for ES deployments in organisations, where interpretability and explainability may be essential considerations (or requirements). This lack of transparency can be somewhat mitigated by LTR, which fuses LM with more traditional ranking functions.

The gap lies in bridging semantic intent with operational relevance through tailored LM techniques that go beyond superficial similarity. No academic literature currently describes the fusion of traditional ranking features with deep semantic embeddings calibrated for enterprise content. This study addresses this gap by developing a domain-adapted 'JARGES' and 'QACES' LM features (described in Sec-

tion §3.11.1 and Section §4.5 respectively) to enhance ranking effectiveness within enterprise settings.

2.6 The EU AI Act

The European Union (EU) AI Act [205], which entered into force on 2024, is an EU-wide regulation that sets out risk management, transparency and reporting obligations for organisations. The provisions of the EU AI Act will become fully applicable in 2026. This transition period allows organisations time to align their AI systems and practices with the new regulatory requirements.

The Act offers a classification for AI systems with different requirements and obligations tailored to a 'risk-based approach'. Under the Act, an information retrieval service, that uses AI methods for ranking, could be categorised either as 'limited risk' (where ranking model is not used for automated decision-making) or 'high risk' (if foundation models like GPT are part of the ranking model).

The Act also imposes different obligations depending on whether the search service is classified as provider or deployer. If an organisation's ES service is developed in-house, it is considered to be a provider under the AI Act and therefore must comply with relevant obligations. When an external (third-party) AI search tool is used, the organisation is classified as a deployer under the Act and is just responsible for confirming the provider's compliance (e.g., ensuring the system has been awarded the CE mark).

The Act obliges 'high risk' service providers to conduct conformity assessments, implement risk management measures, maintain technical documentation, ensure transparency, provide explainable outputs, enable human oversight, facilitate incident reporting, etc. The Act states that non-compliance could result in a fine of up to 6% of the organisation's global revenue. A 'limited risk' service provider has lighter obligations that are primarily focused on transparency and maintaining technical documentation for potential inspection.

Under Article 52 of the Act [205], ES and QAC are generally treated as limited risk (i.e., the transparency and documentation obligations apply, but not the comprehensive requirements for high risk systems).

2.7 Summary of Gaps in the Literature

This chapter has surveyed and detailed the current state of the art for the Learning to Rank and Language Modelling AI methods, in the context of ranking performance for ES search results and query auto-completions. While considerable progress has been made in each field, several important gaps remain, particularly in the context of ES.

Although the **LTR** method is widely used in commercial WS engines, its application to ES has not been sufficiently researched. Existing LTR studies primarily target WS, which benefit from conditions of large-scale user interactions, explicit relevance

judgments, and rich hyperlink structures. These conditions are rarely available in ES. This gap is the foundation of PS-1.

The challenge of organisation-specific language, such as jargon and terminology, is insufficiently addressed in mainstream **LM** research. While transformer-based models such as BERT and GPT have demonstrated exceptional performance in general NLP tasks, few studies have investigated their adaptation to the specialised linguistic characteristics of enterprise corpora or their integration with LTR frameworks for search ranking and query auto-completion. This gap is the foundation of PS-2.

While **QAC** has been studied extensively for WS, relatively little work has focused on query environments like ES, where users may struggle to formulate their query when they have an intent in mind but not a clear articulation or 'wording' for their organisation's specialised jargon/terminology This gap is the foundation of PS-3.

This literature review establishes a foundation for the experiments presented in Chapters 3 and 4, which evaluate the feasibility and performance of LTR and LM methods within a controlled enterprise corpus.

Finally, most existing studies treat search results and QAC ranking as two independent tasks. This thesis argues that these tasks are intrinsically linked in practice and that their integration, especially via shared language features and relevance signals, can lead to improved overall search performance and experience within organisations. This perspective is largely absent from the existing literature.

Addressing these gaps, this thesis explores the use of LTR and LM methods for both search result and QAC ranking, utilising ES content, data, and evaluation strategies designed to reflect real-world organisational constraints. A method for evaluating these fields is discussed, focusing on online methods such as CTR and offline metrics, including MAP, and nDCG.

3

RANKING OF SEARCH RESULTS

3.1 Introduction

Chapter 2 provided a detailed survey of existing knowledge, research, theories and knowledge gaps related to ranking performance in Enterprise Search. In this chapter, theory is put into practice as the role of AI methods in the ranking of ES search results is evaluated (RQ-1, RQ-2).

Ranking of search results is an essential aspect of any search service, but it is of particular importance within organisations for the following reasons:

- Searches on an ES service are usually undertaken by 'knowledge workers' attempting to carry out a specific work task. Software engineers are estimated to spend 25% of their workday searching [41]. Design engineers and technical professionals in six high-tech organisations were observed to average 25.6% of their time searching [48]. In an R&D organisation, knowledge workers may spend between 10 and 30 minutes daily [49]. As outlined in Section §1.1, a 2023 survey by Gartner found that 47% of workers struggle to find the information or data needed to effectively perform their jobs [4]. While accounts of the average time spent looking for a document may vary, any delay in retrieving the desired information must impact productivity or operational efficiency and, therefore, has a monetary cost for the organisation.

- By convention, search engines display just ten results per page. According to Advance Web Ranking [206], less than six per cent of Google WS users navigate further to the second page. A consequence of the pervasive use of WS is that ES users are also not inclined to visit page two, as they have high expectations about the ranking of search results.

- Appropriately ranked search results, based on enterprise content and associated specialised jargon/terminology, can educate new staff members about the range of documents available (even when the query string does not match document content). Surfacing the right information at the right time can assist the onboarding process and steer users to employ the appropriate and official organisational jargon/terminology.

Any poor user experience with ranking search results can be a source of frustration that could lead to negative feelings toward the service and, by extension, toward the organisation [42, 52, 53]. A poor experience could also push searchers to move off the ES service and use WS as an alternative way to perform a search and perhaps inadvertently leak confidential organisational queries to a commercial search engine. While WS can only return publicly available material, users often do not differentiate between public and internal resources when formulating queries, which increases the risk of unintentional disclosure.

This chapter addresses the question as to what extent an AI approach (comprising LTR and LM methods) can be used to improve the ranking of ES search results compared to traditional methods (RQ-1,OBJ-1). Moreover, it examines how LM methods can be used to boost ES-specific content and an organisation's specialised language and jargon (RQ-2). The objective is to develop, implement and evaluate a ranking model for the presentation of search results that is customised for enterprise content and associated specialised jargon/terminology (OBJ-3). As discussed in §1.4, much of the examination is conducted on the 'live' ES service of the large third-level education institution, which receives about 7,000 queries daily. It supplies the data utilised in this study and is herein referred to as 'The University'.

Structurally, this chapter starts with an examination of the various signals used by the traditional (non-AI) approach (§3.2). As the quantity and diversity of the signals grows (Section §3.3), there is an increasing need to apportion 'importance' to each. To this end, an LTR training dataset is created, using both manually annotated judgements as well as click-through data (OBJ-2), and the contribution of each feature to an LTR ranking model is determined. LM is then used to generate an innovative feature called 'JARGES' (described in Section §3.5), which is used to detect and uprank search results containing organisational jargon/terminology.
The experiments are as follows:

- EXP-1: Compares search result ranking performance of traditional and AI approaches.
- EXP-2: An ablation (leave-one-out) study to discover relative contributions of each ranking feature.
- EXP-3: Explores the use of implicit feedback (click-through data) as a substitute for human judgements.
- EXP-4: An Inter-Annotator Agreement experiment to determine agreement between expert annotators and confirm the integrity of The University's dataset.
- EXP-5: Determines whether the JARGES LM detection feature can improve ranking and recall of jargon/terminology on The University's live ES service.

Each experiment is systematically described starting in Section §3.6. Implementation details and challenges (both experimental and organisational) are described immediately after the experimental design. The results of the experiments are presented and tabulated in Sections §3.7.2,§3.8.2, §3.9.2, §3.10.2 and §3.11.2. Finally,

several validation checks are carried out in Section §3.12 to ensure the experiments are rigorous and the results robust.

An integrated synthesis, framing and reflection of these results (from both this chapter and later from Chapter 4) will be discussed in Chapter 5.

3.2 Traditional Ranking

The traditional ranking approach is characterised by explicit, hand-crafted methods and mathematical formulations to estimate the ranking of documents based on functions such as relevance, popularity, authoritativeness, and recency. This approach, sometimes referred to as heuristic, is grounded in classical information retrieval theory and does not depend on machine learning or NLP techniques.

According to Zhou et al. [100], any ranking model for ES should, at the very least, incorporate recency and authoritativeness (query-independent) functions in addition to one or more pure text matching (query-dependent) functions. The following sections describe the most common traditional ranking functions, and how they can be configured and deployed in the context of ES.

3.2.1 BM25 Similarity Matching

As described in Section §2.2.1, BM25 provides state-of-the-art relevance ranking for search queries by balancing term frequency, document length, and inverse document frequency to score and rank documents.

Apache Lucene [207] is a widely adopted open-source search library that serves as the foundation for numerous search engines and applications. Prominent platforms such as Apache Solr [14] and Elasticsearch [208] are both built atop Lucene. The Apache Solr and the Elasticsearch search engine platforms use the BM25 [209] similarity algorithm as the default ranking function.

An out-of-the-box installation of Solr or Elasticsearch relies exclusively on BM25 for ranking search results. Hence, in this research, BM25 is considered the de-facto 'traditional approach' baseline for ES ranking. As described in Section §2.2.1, the two tunable parameters for BM25 are k_1 (controls term-frequency saturation) and b (governs the strength of document-length normalisation). The default BM25 configurations are $k_1 = 1.2$ and $b = 0.75$ [74]. These values are unchanged for the experiments in this chapter; i.e. the same k_1 and b values are used in both traditional and AI approaches. In EXP-1, this performance baseline is recorded prior to evaluating/comparing the impact of the AI ranking approach. Establishing an initial baseline using the traditional approach also informs the discussion of whether state-of-the-art AI ranking methods are 'worth the effort' in the domain of ES (where search may not be at the core of the business, and organisational resources and skills may be limited).

As described in Section §2.4.6, BM25 is categorised as a *query-dependent* ranking signal. BM25 depends explicitly on the specific query text submitted to the engine.

Query-dependent functions are calculated highly efficiently based on the similarity relationship between the query string and indexed terms.

3.2.2 View Count

As described in Section §2.2.4, the total number of views (a.k.a. 'page hits') is often treated as a proxy measure of document popularity or general user interest [56, 210]. A higher view count acts as a *social proof* that a document can satisfy user information needs. Most Views (MV) refers to a sorted list of documents by descending view count.

Each view record is a tuple with a document name and the document's view count. A list of tuples can be extracted from the web server's log files.

The 'most viewed' signal is *query independent* (i.e., a signal of importance ascertained regardless of the submitted query term). Hence, MV cannot be used as a ranking function alone; it is generally pooled with other ranking functions (e.g., as a re-ranking function applied after BM25).

In the field of WS, even unpopular documents are likely to have at least a small number of views. External web crawlers (such as Googlebot) register a page hit. In ES, documents that exist in the index but have no record of viewings in the logs are assigned a default count of zero. This is a potentially common score (since external crawlers cannot access internal ES content). A large number of zero scores may reduce the effectiveness of view count as an ES ranking function.

The key assumption is that the number of views correlates with user satisfaction. However, a repercussion of the MV ranking function is that it reinforces universal perceived utility, which can create a feedback loop. Higher-ranked documents get more views, which further reinforces their rank. The feedback loop can be mitigated by normalising view counts (e.g., dividing by document age, adjusting for visibility or simply resetting the counter periodically) to ensure fairness and surface new, potentially useful documents.

In much of the research literature, the MV ranking feature is referred to as the 'most popular' or 'popularity' ranking function. These terms are potentially misleading, as there are other, potentially equally worthy, proxy measures of popularity (e.g., ranking functions based on link analysis) [211].

A potential limitation of MV is that the view count can be manipulated by anyone wishing to boost the ranking of the document's content. This could be effectuated, for example, by a `cURL` script that retrieved the document header a sufficient number of times. While such SEO manipulation is common in the field of WS, it is unlikely to present a problem within an ES context (where content owners are colleagues within the same organisation).

3.2.3 Recency

For queries where current information is essential, prioritising recent documents enhances the usefulness of search results.

In the context of WS, the timeliness of information is often critical to user satisfaction. Time-critical documents for rapidly evolving topics include breaking news, financial data, event announcements, etc. In the context of ES, users generally do not require or expect this level of immediacy. Nonetheless, an organisation's document updated this year is likely to be more useful for users than one (with the same similarity matching score) that was last updated thirty years ago.

As discussed in Section §2.2.9, Apache Solr has a built-in 'recip' time decay function, wherein the ranking score of a document decreases as it ages. This function ensures newer documents are emphasised while older documents remain accessible based on query-document term relevance matching. Recip has three configurable parameters (governing the timescale and shape of the boost drop-off curve. There is no reason to assume that the default values should be the same for ES and WS. Hence, an analysis of the age distribution profile of The University's corpus is undertaken.

Figure 3.1 plots the number of documents versus document age in The University's ES index. This plot shows that about half of all documents in the corpus (solr index) have been created or modified in the last five years. The red dashed line represents the 50% volume intercept. This age is similar to the internet website distribution plot shown in Figure 2.3 (Section §2.2.9), where 50% of all internet websites were also created over the past five years.

Since the historical time frame of publication dates is similar for both WS and The University's ES index (30 years), and since the cumulative distribution curve follows approximately the same shape, it is assumed that the same default parameter values for m, a and b (steepness, scale and offset respectively, as described in Section §2.2.9) can be used for the recip function.

A prerequisite for recip is that a publication timestamp for each document must be recorded and indexed. The timestamp is extracted using various methods depending on the file type. For a web page, the author may have included publication time/-date. The timestamp of a PDF document on Network Attached Storage (NAS) may be extracted using the Linux 'mtime' command. The details of how the timestamp is extracted from a document, and then indexed into Solr for The University's corpus, are provided in the implementation section (§3.7.1).

Recency as a ranking function requires balancing the relevance of the document's content with its temporal proximity to the query at submission time. Hence, recency is typically pooled with other ranking functions, such as query-document relevance, popularity, authority, etc. The pooling prevents over-prioritisation of new but less useful documents. The weight assigned to recency may vary across domains. For example, legal or academic search contexts often prioritise well-established, peer-reviewed documents, whereas news platforms heavily favour the latest articles.

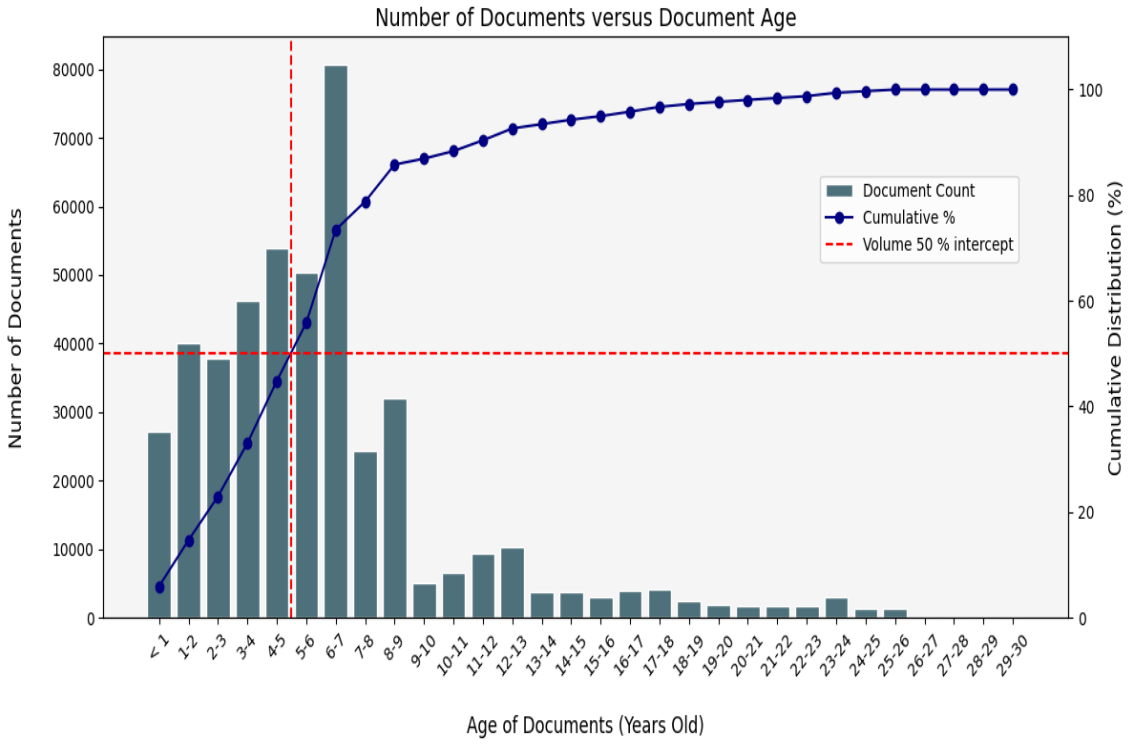


Figure 3.1: Document Recency Plot showing the age (i.e., 'recency') distribution for the documents in The University's ES index. The plot shows that just 5% of indexed documents were created or modified in the past year.

A manipulation risk is that overweighting recency can incentivise frequent, trivial updates to content to improve rankings artificially. This manipulation is widespread in the WS SEO community, but it is unlikely to present a problem in an ES context (where publishers are colleagues within the same organisation and unlikely to be engaged in an intense competition for user attention).

3.2.4 LinkRank

As discussed in Section §2.2.5, LinkRank mirrors the underlying concepts of Google's PageRank. When applied to ES, the LinkRank ranking algorithm leverages the structure of relationships within an organisation's corpus. By analysing the network of links between documents, LinkRank identifies highly referenced and authoritative content.

This ranking feature can be particularly suited to enterprise websites, which typically have a rigorous site structure organised as a site hierarchy expressed broadly through standard navigational links and menus. Links and menus impress upon users a mental model of navigation and provide a template of continuity on each page. The content and systematic structure of typical ES websites often result in only a single existing 'correct' answer for a query (in strong contrast to WS), and users may even have the ability to recognise the specific 'known' document that matches a hierarchical level and query [8].

Figure 2.2 outlined how The University's website is structured with the homepage

as the top level of a hierarchy, with a large number of faculties, departments, and schools constituting mid and lower-level sub-sites and 'microsites' (Section §2.2.5). Since each microsite is largely self-contained and each page generally links back to the microsite's own homepage, this structure aligns closely with the intent of the LinkRank algorithm.

An example of a simple ranking of documents using the LinkRank function is shown in listing 3.1. The example shows that the 'zoology' query term, which has a microsite of the same name, has a higher LinkRank score than pages lower down in the hierarchy. Further details of how the LinkRank score is computed in Solr for The University's website are provided in Section §3.7.1.

```
$ curl ``http://localhost:8983/solr/$CORE/select?\
>q=url:zoology\
>&fl=url,linkrank\
>&rows=10\
>&sort=linkrank desc\
>&wt=csv''

/Zoology/,4.283704
/Zoology/research/groups/buckley/,1.7411453
/Zoology/index.ga.php,1.6279833
/Zoology/contact/,1.6127006
/Zoology/research/groups/jackson/,1.6058393
/Zoology/research/groups/piggott/,1.4902297
/Zoology/research/groups/donohue/,1.4885491
/Zoology/research/groups/oconnor/,1.4876326
/Zoology/research/,1.487632
/Zoology/museum/collection/,1.4862205
```

Listing 3.1: Ranking using LinkRank. A query for the 'zoology' keyword submitted to The University's ES index demonstrates ranking using LinkRank. The URL is shown together with its LinkRank score.

LinkRank is a query-independent ranking function. It must be combined with a similarity-matching (query-dependent) function like BM25. Moreover, while LinkRank may work well for web pages, enterprise content generally includes non-web document types. For example, a PDF document deposited on a network file share, which was never designed with publication in mind, may have no inlinks or outlinks. In this case, the orphaned document's LinkRank score would be zero, thus requiring another alternative (fallback) ranking approach.

3.2.5 Field Boosting

Most documents in a corpus are composed of multiple fields (e.g., title, content, timestamp, URL). Field boosting imbues importance to selected fields during query execution, with the effect of changing the overall ranking score of the document.

For example, Figure 3.2 shows a typical page of The University's website. The page demonstrates the constituent fields and extraction of text (or values) for several fields. Field boosting facilitates prioritisation of similarity matching (i.e., BM25) in the title field over matching in the content field. Solr has a parameter called

'QueryFields' (`qf`) that can be used at query time to assign the boost factor [212]. For example, the query below can be added to the search specification:

```
q="geology"&qf="urlˆ4 titleˆ2 content"
```

This instruction assigns the URL field a boost factor of 4, the title field a boost factor of 2, and leaves the content field the default boost (i.e., 1). In this example, the boost factors make (term frequency with the 'geology' query) matches in the URL field more significant than matches in the title field, which are much more significant than matches in the content field.

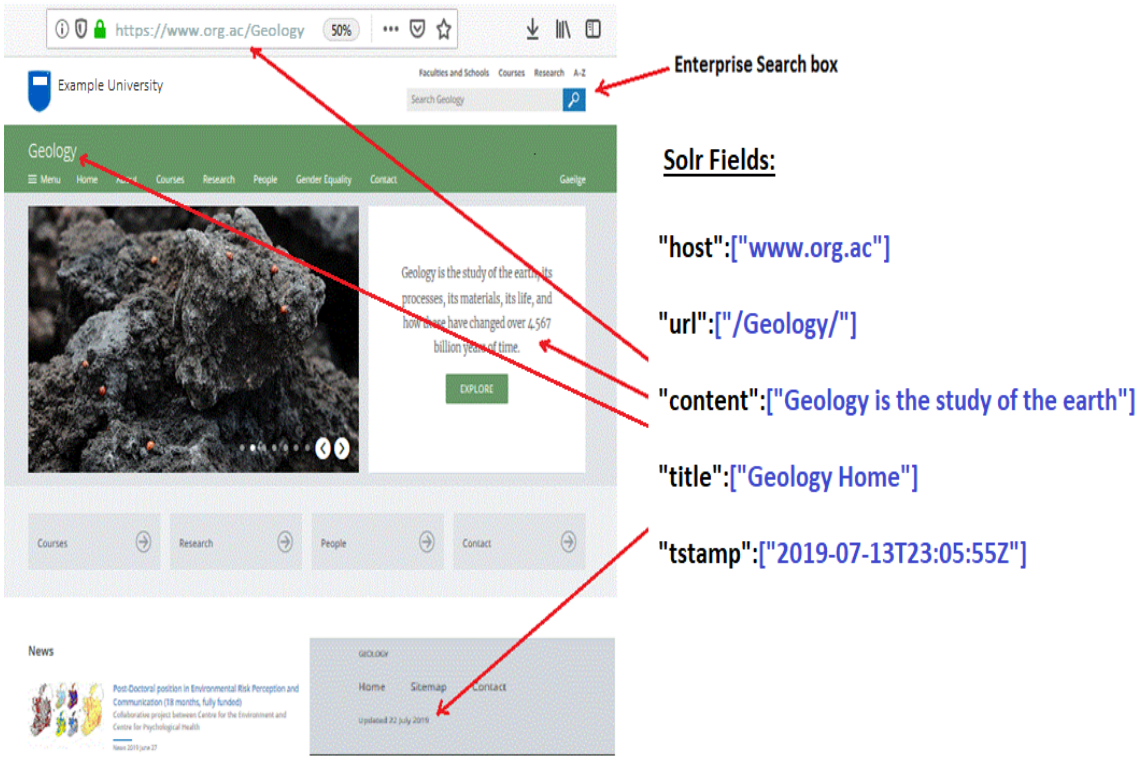


Figure 3.2: Extracting field content from documents of The University's website. The fields are indexed by Apache Solr for ES.

While field boosting is a convenient tuning and tailoring mechanism, it is not necessarily readily applicable to all documents in an ES corpus. Many documents types (e.g., Microsoft Excel and PowerPoint) may have no discernable title field. In this case, all content is indexed into a single field. Consequently, the importance of this ranking function is likely different for ES and WS. Moreover, field boosting does not solve the problem of determining the particular value for each boosting factor. In ES, there may be an intuition of a 'good ranking'. A relevance engineer can spend many hours attempting to estimate the boost factors by *eyeballing* search results. While this (tuning/tailoring) may improve the ranking for one query, it often later transpires that it has the effect of diminishing it for another [6,54]. Practical experience suggests that this trial-and-error approach may not align with the expectations of an organisation's change management committee.

3.3 Other Hand-Crafted Ranking functions

The following ranking functions are also traditional and heuristic but are less common as they require advanced domain knowledge, empirical observations and are often applied explicitly to documents or fields.

3.3.1 Elevation

As discussed in Section §2.2.6, elevation (also referred to as 'query elevation') allows for the manual specification of the top result (or results) for a query regardless of the computed rank score. The elevation function may be useful when business priorities dictate that particular documents receive the top-ranking spot for specific keyword queries. This functionality diverges from the traditional concept of ranking insofar as it operates as a manual override or customised enhancement to regular ranking functions. The Apache Solr documentation describes elevation as 'sponsored search' and 'editorial boosting' [84].

For each keyword query, Q, for which elevation of Document D is desired, a hard-coded Q-D pair is placed in the Solr `elevate.xml` file to explicitly list the document (or documents) that must appear atop the search results. Query elevation cannot work for 'unseen' (i.e., new) documents. An update of the elevate file is required whenever a new document is indexed and needs to be elevated.

While ranking by elevation is a valuable method for manually aligning search results with organisational objectives, it poses a risk when these objectives diverge from the preferences or expectations of the ES end-users. An example of this risk was given in Section §2.2.7 during the review of multi-objective ranking.

A practical use case for elevation is as a *safety net* during the first weeks or months after deployment of an ES service, i.e., before the organisation has a tuned or trained ranking model readied for deployment. Elevation is not a ranking 'function' in the strict sense and it is not used in any of the experiments described in this study.

3.3.2 Keyword Boosting

Keyword boosting refers to the process of prioritising specific keywords (or phrases) in search results. By applying boosts to particular keywords, the search engine ensures that documents containing those keywords achieve a higher ranking score. Unlike the elevation ranking function, keyword boosting does not explicitly force a document to the top of the list. Consequently, as there is no explicit mapping, keyword boosting also works with 'unseen' or new documents as part of a ranking model.

Keyword boosting is useful when business priorities dictate that any document containing (or associated with) the keyword receives attention. Apache Solr has a keyword-boosting function, which can be appended to the Lucene parser or as part of an LTR model. The implementation of keyword boosting for The University's ranking model is described in Section §3.7.1.

3.3.3 Anchor Text

As described in Section §2.2.8, anchor text is the clickable, visible text in a hyperlink that typically provides context about the link's destination. It serves both as a guide to users and a signal to search engines about the content of the linked (target) document.

Anchor text provides a brief description of what the user can expect on the target document. The anchor text may have been composed by a different author, so it may include alternative wording, phraseology or jargon. Consider, for example, a target document whose content includes 'annual leave'. A user unfamiliar with this formal phraseology may instead submit the query 'staff holidays' (using everyday language). If the anchor text associated with a document matches the user's query term, the query is expanded to include both the original term and the corresponding anchor-text phrase. In practice, a document may have several different anchor texts pointing to it, reflecting the variety of ways other pages link to that content. During indexing, all observed anchor text variants are stored in the document's dedicated anchor text field. At query time, a match is triggered when the user's term appears in any of these stored anchor-text strings (after normalisation such as lowercasing and tokenisation). When such a match occurs, each matching anchor text phrase is added as an additional clause to the query, enabling the ranking model to consider both the user's formulation and the anchor-derived variants.

In terms of ranking quality and quantity, anchor text offers the dual benefit of both potentially increasing precision and recall. Ranking precision may be boosted if the query term occurs not just in the target document, but also in the anchor text field. The increased TF boosts the relevance score of the target document. The contextual alternative terms of the anchor text enhance a user's search by adding additional relevant terms (query expansion) to improve recall.

Anchor text is stored in the index as a field associated with the target document, and may be used a ranking function using the TF count for a query (i.e., the same way as ranking by title, content or other field). An example is "research support system 24", which tells us that the string "research support system" string is encoded into anchors a total of 24 times in The University's corpus.

In the context of ES, using anchor text as a ranking function has several limitations. Firstly, anchor text and hyperlinks are predominantly found on web pages. This predominance brings a bias against other document types (e.g., PDF or Microsoft Word) where hyperlinks are less likely to be deployed or against any document type that does not support hyperlinks (e.g., Microsoft WordPad or any `.txt` file). For link based ranking, what matters is whether other documents link to a document. Anchor text has both a source and a target, and ES corpora typically contain far fewer inbound links because much of the content is non-HTML (this will be discussed further in Section §5.7.1).

When implementing anchor text as a ranking function, the terms are mapped to the target document as metadata in the form of a string field in the Solr ES index. Full implementation details are described in Section §3.7.1 and the weighted contribution

of this ranking function for ES content will be measured as part of EXP-2.

3.3.4 URL or Path Length

A concise URL or path can give users a clearer understanding of a document's importance and utility. The length of the URL or path reflects the site's hierarchical structure clearly and intuitively. For example, when viewing search results, a document with a URL path like `/abc/` may signal greater importance and utility than `/abc/def/123/ghi/`. The signal is especially obvious for navigational queries [57] and when the user perceives the site's structure.

In common with LinkRank, ranking by URL or path length is heavily influenced by a site's hierarchical structure. A shorter URL or path is easier to remember, share and link to. This also increases the likelihood of the organisation's publishing community generating backlinks. However, unlike LinkRank, the URL or path length function (which I call `URLlength`) is not exclusively tailored to work for webpages and attendant hyperlinks. The `urlLength` ranking function can be applied to all documents (including orphaned documents) in the ES index.

Apache Solr includes a Java class called `FieldLengthFeature`, which calculates the length of a specific field. When translated into a ranking function, this class is useful in scenarios where field length correlates with document utility. In The University's ES index, `URLlength` is applied to the `url` field, and the implementation details are described in Section §3.7.1.

The five shortest documents by `URLlength` in The University's ES index are shown in listing 3.2. Also included is the document with the longest path. The founding assumption of this ranking function is that ES users find shorter URLs or paths to be more useful.

```
/    1
/az  3
/e3  3
/hr  3
/cao 4
..
/trinitylongroomhub/events-calendar/events/2025/memories-in-stone-a-
   study-of-identity-through-statuary-on-the-island-of-ireland
   -1922-39--little-guests-operation-shamrock-transnational-
   humanitarian-hospitality-and-childrens-lives-in-the-aftermath-of-
   the-second-world-war.php  272
```

Listing 3.2: URLlength as a ranking function. The path is shown together with its calculated URLlength for the five shortest paths, and the longest path.

According to a Semrush analysis of Google search results, "shorter URLs tend to have a slight ranking advantage over longer ones" [213]. The weighted contribution of this ranking function for ES content will be measured as part of EXP-2.

3.3.5 Click-through Rate (CTR)

As discussed in Section §2.2.13, the click-through rate (CTR) reflects how often users click on a document after it appears in search results for a given query. CTR aids in the identification of documents that are perceived as both useful and relevant to the ES user. CTR is calculated as the percentage of clicks to the number of impressions (i.e., the number of times the document is shown on the results page) that are generated for a query [30].

A click model is used to record end-user preferences on search results. In the field of WS, click models are considered as an effective approach to infer search relevance of a document for a given query [19].

Like BM25, CTR is a query-dependent ranking function. This means that the score (i.e. rate) is assigned to a Query-Document (Q-D) pair rather than to the document individually.

Compared with the previously described ranking functions, the implementing of CTR as a ranking function is a major undertaking. In its native form, Solr does not record search session details and typically requires an integration with an external system to log and process CTR data [214]. Moreover, according to Białecki [215], any implementation of CTR as a ranking function should consider the following aspects:

- A mechanism to break out of an 'unbounded positive feedback' loop. The loop occurs where a higher ranking position leads to greater click-through, which, in turn, leads to a higher ranking (similar to the *position bias* phenomena described in Section §2.4.8).
- A decay function should be applied to discount those older documents that were once very popular (e.g., months or years ago) but are now less so. Białecki recommends using a half-life decay model with an adjustable period and rate and an adjustable length of history to affect smoothing [215].
- Clicks on lower-ranking documents can be treated as a correction to the established order and should, therefore, be awarded greater importance than those near the top. Consequently, the increments applied to CTR scores should not follow a linear model.
- A mechanism to reflect trending documents over short periods of time. The CTR framework should react quickly to bursts (topics of the day), hence preventing documents from being 'stuck' at the top by merit of CTR values accrued over a longer period.
- Normalisation of the click-through rate to prevent outliers from disproportionately affecting rankings (Białecki recommends using a log function for normalisation [215]).
- After offline computation of the (normalised, adjusted) click-through rate, the data should be integrated or updated into the ES index. At query execution

time, the score should be fetched from the index to prevent any latency. This refresh should occur in near real-time [214].

Considering all of these aspects, the creation and implementation of a sophisticated CTR framework for ranking purposes would merit its own dedicated investigation and is beyond the scope of the present study. Hence, it was decided not to employ CTR as a *ranking task* for The University's ES service. Instead, basic CTR data was gathered and used as a *training task* (as is often the case for academic studies [101]), where it is used as a surrogate for explicit human judgements for Q-D pairs (EXP-3 gauges correlation between CTR and human judgements).

This section has described several hand-crafted ranking functions. While many of the functions are tuned for The University's corpus, the concept of ranking to boost keywords or fields, etc is useful for any enterprise. An inherent danger of tuning ranking is that one function can impact (or even cancel-out) another.

3.4 Learning to Rank

Re-ranking and combining more and more ranking functions has the inherent problem that it becomes very difficult and eventually impossible to estimate the correct importance of each function.

For each of the ranking functions previously described, two essential questions need to be addressed:

- Eyeballing a list of search results is not a solid approach to measuring ranking performance. How can anyone know if a function results in an improved ranking by drawing nearer to an 'ideal ranking'?
- Intuitively, some ranking functions contribute more than others. How can the correct importance 'weighting' be apportioned to each function?

As described in Section §2.4, the LTR supervised machine learning framework is specifically designed to answer these questions. Using ground truth that includes the ideal Q-D ordering, LTR calculates a weight for the contribution of each of the ranking functions (or 'features', as they are referred to when using LTR).

For example, in field boosting, it was noted that a boost factor can be apportioned to each field. From experience of The University's ES service, it is estimated that importance of the title field likely outweighs that of the content field, but by what weight and by what margin? In experiment EXP-2 (ablation study), ranking for each field will be treated as an individual feature and LTR will be used to compute the feature weighting and evaluate each feature's contribution to overall ranking performance.

Before applying the LTR framework, a prerequisite 'ground truth' of ideal Q-D pairings and an associated dataset is required.

3.4.1 Ground Truth

As described in sections §2.4.4 and §2.4.4, ground truth refers to the ideal ranking to be modelled and is converted into training data for use with a supervised machine learning algorithm. Ground truth requires a relevance label that captures the quality of a Q-D pair. There are two approaches to gathering ground truth labels [149]:

- Explicit (human judgements)
- Implicit (CTR derived from search log data).

The first approach relies on actively soliciting training data by recording user queries and then asking 'judges' (also referred to as human judges, annotators, raters or labellers) to explicitly provide relevance judgments on retrieved documents. The advantage of the explicit human judgement approach is that it produces a high quality clean training dataset. The disadvantage is that it is very expensive and/or time-consuming to acquire, as it requires the use of labellers (who have an expertise in the query topic from within the organisation). The ground truth dataset used in experiments EXP-1, EXP-2, EXP-4 and EXP-5 uses this approach.

The second approach is to extract implicit relevance feedback from search engine log data. User interactions, in the form of CTR, can potentially act as a proxy for human annotated judgements. The advantage of CTR is that large quantities of training data can be mined at a low cost. The ground truth dataset used in experiment EXP-3 uses this approach, where it is correlated and compared with human judgements.

As the world around us changes, the relevance of a document for a given query also changes over time, leading to relevance drift [216]. For example, the topicality of the term 'Ukraine' has changed dramatically over the past years. Enterprises cannot simply 'deploy and forget'. Periodic re-training of Learning to Rank models is required regardless of whether explicit or implicit feedback is used. An advantage of the implicit feedback approach is that it is not necessary to re-solicit judgements from expert annotators.

Prior to commencing the process of soliciting Q-D relevance annotations from human judges, a strategy is needed for efficiently selecting the most representative, popular or appropriate queries. The selected queries will impact on how the judges are chosen.

3.4.2 Query Selection

The queries for annotation are selected to be representative of typical search requests as extracted from The University's search log data. Queries are selected based on a combination of the following criteria:

- Query frequency and associated search demand curve.
- 'Microsite' (subsite) structure associated with the query topic.
- Query Length
- Query Clustering

Query Frequency Query frequency involves mining the search logs for previously submitted queries and counting their frequency. As described in Section §2.3.7, Most Frequent Queries (MFQ) is a sorted list of historical queries representing the topics or keywords searched for most frequently by the user community.

An example is "book of kells 15", which tells us that the query 'book of kells' has been submitted and recorded by the search engine 15 times in the log file.

A closely related concept to MFQ (and query selection for annotation) is the Search Demand Curve (SDC). The SDC plots the search query volume over a given period versus the rank of each query. Using The Universitys ES query logs, Figure 3.3 shows the outsize impact that a small number of frequently submitted queries has on the overall volume of search activity. According to Kritzinger et al., popular search terms make up 30% of the overall searches performed on commercial Web Search engines. Using zoning norms devised by the Search Engine Optimisation (SEO) community in 2011, the 18.5% of searches with the highest occurrence is known as the Fat Head. The next 11.5% is called the Chunky Middle [66]. The Long Tail (x-axis) in Figure 3.3 has been limited for presentation purposes but actually comprises 70% of the total search volume. In The University's ES query logs, it is seen that 65 queries account for 18.5% of all search volume[7].

To maximise representation, the queries selected for Q-D annotation are all within the top 18.5 % of search volume. Consequently, the selected queries tend to reflect a topical information need, as opposed to an exploratory one.

Structure of Microsite According to the Cambridge English dictionary, a microsite is defined as "a small website, usually one advertising a particular product or service for a company". Within the hierarchical context of The University's website (Figure 2.2), the microsite's 'product or service' might include a faculty, department or school. Each of The University's microsites is semi-autonomous of the parent site and has its own sitemap, colour, theming and publisher (or, more commonly, a team of publishers). The publishers are thoroughly familiar with site's content. For example, the history school's microsite is managed by two publishers, who are jointly responsible for all content (pages, documents, forms, records, etc.) under the '/history/' subsite path. Microsites are typically instantiated from one of several available templated structures.

To maximise representation, the queries selected for Q–D annotation were drawn from microsites with diverse structures. The selection process did not rely on a single criterion. Instead, queries were chosen using the multi-stage procedure outlined earlier, which considers (i) query frequency and position on the search demand curve, (ii) the microsite or subsite associated with the query topic, (iii) query length, and (iv) clustering of semantically related queries. These selection criteria ensured that the annotated set captured both high-volume user behaviour and structural variation across the corpus.

[7] https://github.com/colindaly75/QAC_LTR_for_ES

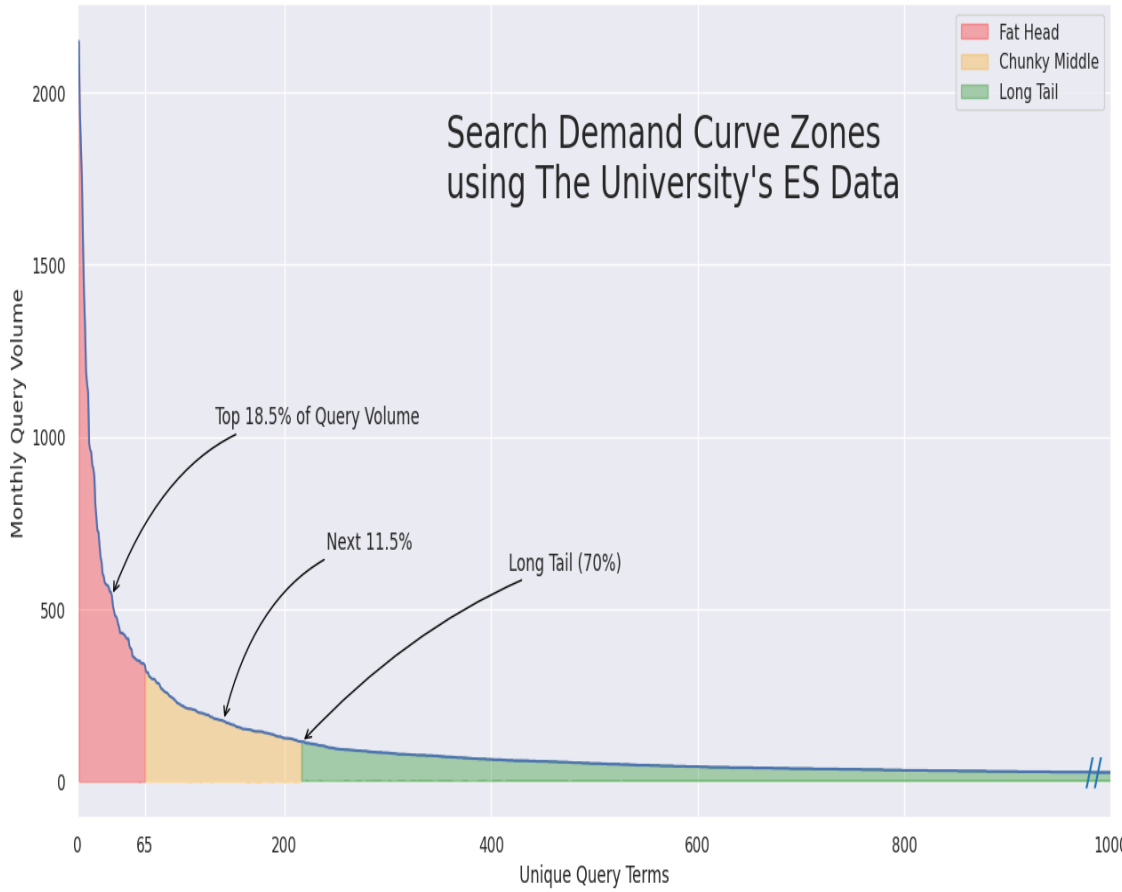


Figure 3.3: Search Demand Curve for The University's Enterprise Search query history, showing the so-called 'Fat Head', 'Chunky Middle' and 'Long Tail' zones. For The University's ES service, the most popular 65 queries account for 18.5% of all search volume.

Words per Query From The University's 2024 Solr log files, the computed average overall number of words per query is 2.187. A similar study of ES search queries by Kruschwitz [64] in 2013 observed an overall average length of 1.81 words per query. While these figures are in broad agreement, the higher count could be explained because query length has been steadily increasing over time [63]. As described in Section §2.1.5, WS users (and by extension of the 'Google Habitus' to ES users) therefore feel less restricted today by the need to craft keyword queries than they did in 2013.

Unfortunately, Kruschwitz's study did not break down the numbers for the various zones of the search demand curve. For The University, the average number of query words in each of the SDC zone is outlined in Table 3.1.

To maximise representation, the queries selected for Q-D annotation range from one to two terms per query in line with the 1.431 average length in the fat head zone.

Users have discovered that some search engines are capable of understanding detailed and specific queries, and this has encouraged more elaborate input. This includes queries submitted in question form. Listing 3.3 compares the number of

Table 3.1: Average number of words per query for each zone of The University's Search Demand Curve.

SDC Zone	Average Query Length
Fat Head	1.431
Chunky Middle	1.477
Long Tail	2.189
Overall	2.187

question form queries (i.e., queries beginning with the words 'who, what, where, when, how, whose, is, are' and/or anything ending with a question mark) for ES and WS datasets. As expected, the WS dataset has a higher percentage of questions than ES. A two-proportion z-test demonstrates that the disparity in question-form queries between the ES (5.3%) and WS (6.7%) datasets is statistically significant ($z = -15.77$, $p < 0.001$) at the $\alpha = 0.05$ significance level.

```
# -----Count total ES queries-----
print("Total␣number␣of␣ES␣queries:", es_df.shape[0])

# Count ES queries phrased as questions
print("Number␣of␣ES␣queries␣in␣question␣form:")
!egrep -e who -e what -e where -e when -e how -e whose -e ^is -e ^are -e $? \
    es-query-history-v25.txt | wc -l

# Total number of ES queries: 62044
# Number of ES questions:     3277
# 5.3% (3277 / 62044) of ES queries are in question form.


# -----Count total WS queries-----
print("Total␣number␣of␣WS␣queries:", ws_df.shape[0])

# Count WS queries phrased as questions
print("Number␣of␣WS␣queries␣in␣question␣form:")
!egrep -e who -e what -e where -e when -e how -e whose -e ^is -e ^are -e $? \
    ws-query-history-v25.txt | wc -l

# Total number of WS queries: 4316675
# Number of WS questions:     289686
# 6.7% of WS (AOL) queries are in question form.
```

Listing 3.3: Code used for detecting question-form queries in ES and WS datasets

Query Clustering As described in Section §2.1.6, query clustering is the process of grouping similar queries into clusters based on their lexical or semantic similarity. Clustering enables pattern identification of related topics among the queries.

Lexical clustering is based on word overlap or surface-level similarity. An easily identifiable example from The University's ES query history is the 'admission' and

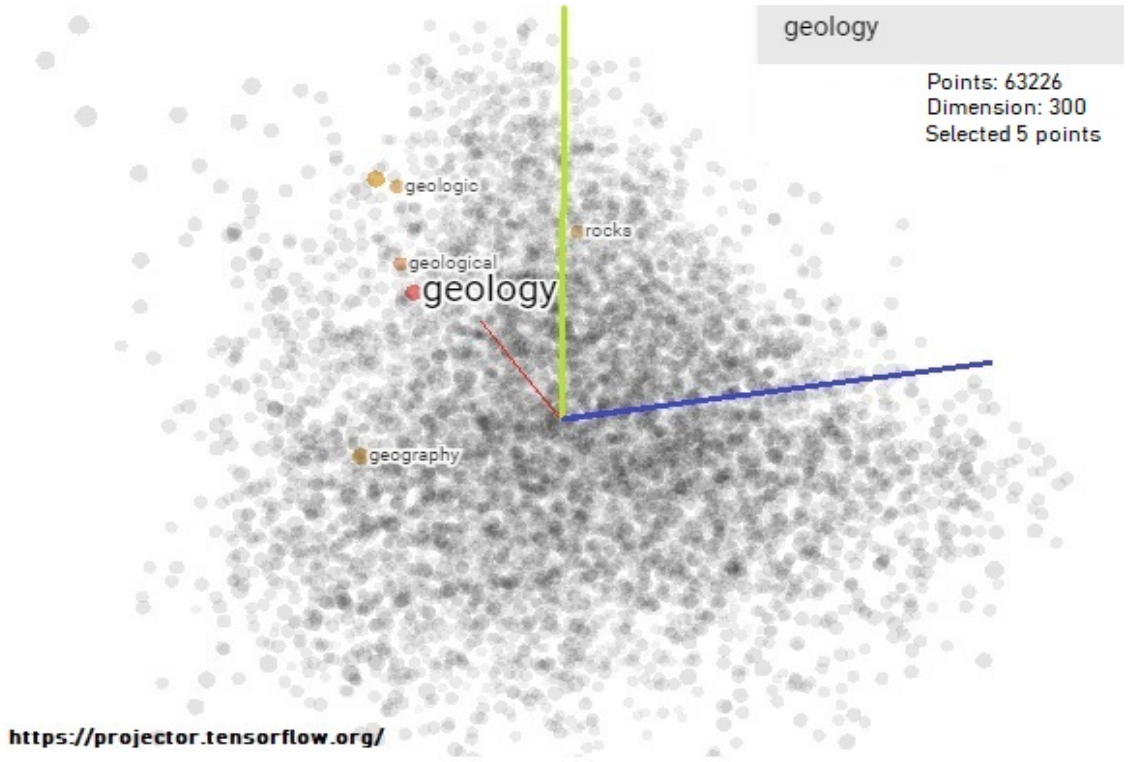


Figure 3.4: TensorFlow word embedding visualisation showing 'proximity' of synonyms for the query 'Geology'.

'admissions' queries (59th and 60th most frequent queries). Clearly, both queries can use the same Q-D annotations.

Semantic clustering involves the discovery of deeper semantic relationships. An easily identifiable example from The University's ES query history is the 'vacancies' and 'jobs' queries (8th and 13th most frequent queries). Intuitively, these queries clearly refer to the same topic. A more complex example is a query like 'geology', where some domain knowledge may be required. Using the Word2Vec NLP technique, the semantic relationship between The University's queries (including those in the fat head, chunky middle and long tail zones of the SDC) can be detected. Figure 3.4 uses Google's TensorFlow tool to visualise the generated Word2Vec model. It shows the four most proximal queries to the topic of geology ['geologic', 'geological', 'rocks', 'geography']. The implementation details are described in Section §3.7.1.

Identifying topics among The University's clustered query history assists with selecting representative and topical query candidates for Q-D annotations.

3.4.3 Judge Selection

The criteria for selecting representative queries for Q-D annotation have been described above. A final criterion/restriction is availability of qualified and willing judges to perform the Q-D annotations.

In the field of WS, crowd-sourcing platforms, such as Amazon's Mechanical Turk(AMT), are often used to facilitate explicit labelling by remunerating annotators [156, 157].

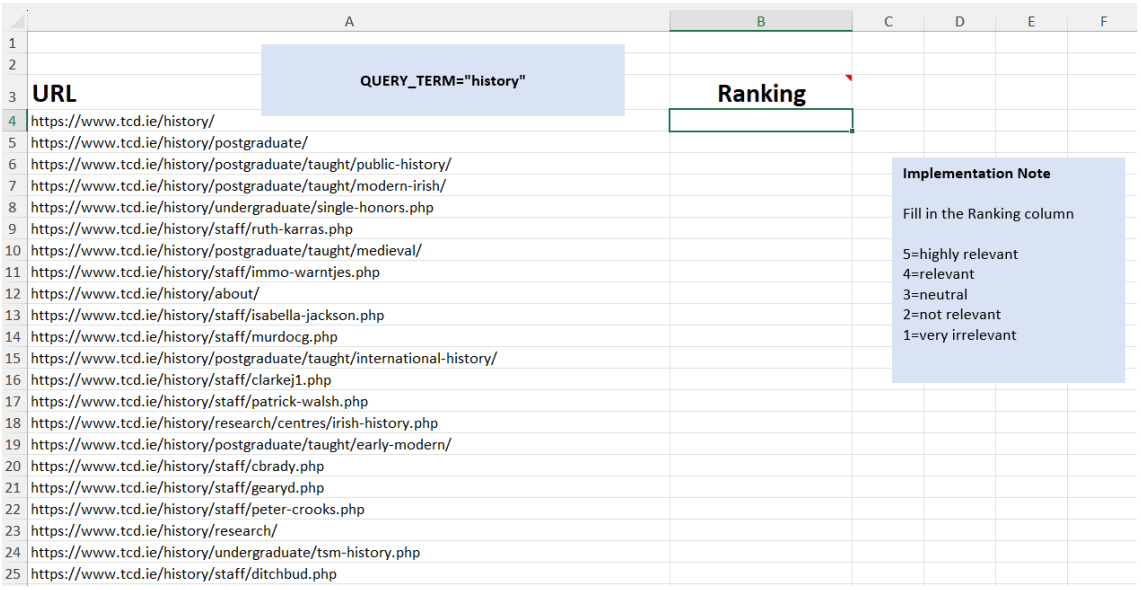
While this is an appropriate method for internet-facing documents or web pages, AMT cannot be used to access the highly technical blueprints, confidential documents, network file shares, etc., behind an organisation's firewall.

For ES, the judge is likely to be an employee or trusted partner of the organisation. Apart from access, the primary criterion employed to select a judge to annotate the Q-D pairs is topic knowledge, which, in turn, is entirely dependent on the query selection process (outlined in Section §3.4.2). In the case of The University, the question of *judge selection* is inherently interconnected with the question of *query selection*, and the two cannot be considered in isolation.

Once a suitable query/topic was chosen, the related microsite editor (The University's corporate services division maintains a list of microsite editors for access control and accountability issues) was tentatively asked if they would be willing to partake in a ranking survey for their microsite. Each microsite has at least one editor/administrator who is a potential judge. In this case, the role of the editor includes publishing content on behalf of colleagues in their department. The microsite owner is obliged to be very familiar with published content, so as to avoid replication and ensure consistent messaging. These microsite editors are uniquely well-placed to 'judge' the ranking order of their pages for their particular topic or subject area. Note that microsite editors are not inter-changeable; an editor of the 'mathematics' site cannot be asked to annotate Q-D pairs for the 'religion' microsite.

Fifteen judges agreed to cooperate with my ranking survey for twenty queries (some judges were able to annotate for more than one query topic). In an organisational context, where judges cannot be specially remunerated, cooperation sometimes comes in the form of a barter. The judges were not remunerated, as would typically be the case when using a platform like Amazon Mechanical Turk for WS annotations. As expected, not all editors initially agreed, but no planned queries had to be excluded as a result. By approaching microsite owners opportunistically and following up as needed, full annotation coverage for the target query set was ultimately achieved, so there was no impact on the composition of the dataset.

Judging Guidelines A custom spreadsheet for each of the twenty queries was emailed to the fifteen judges. It contained two columns called 'URL' and 'ranking' for a given query, as well as an implementation/instructions note. While 'document' is the technical jargon used to refer to the basic unit of indexed content in information retrieval, URL is easier for judges to understand. Similarly, the column marked 'ranking' would have been more accurately referred to as judgements or annotations. Hence the terms 'URL' and 'Ranking' are both named in a simplified sense to avoid any confusion. There were only 5 possible values ranging from highly relevant (5) down to very irrelevant(1). Although nDCG conventionally assumes a zero-based scale (i.e. where 0 denotes non-relevance), a 1–5 scale can be linearly transformed to this convention without affecting relative model comparisons. The URL column listed all documents that matched the query (extracted from the Solr index). The presentation order of the documents was governed by the BM25 rudimentary ranking function.



	A	B	C	D	E	F
1						
2		QUERY_TERM="history"				
3	**URL**	**Ranking**				
4	https://www.tcd.ie/history/					
5	https://www.tcd.ie/history/postgraduate/					
6	https://www.tcd.ie/history/postgraduate/taught/public-history/					
7	https://www.tcd.ie/history/postgraduate/taught/modern-irish/					
8	https://www.tcd.ie/history/undergraduate/single-honors.php					
9	https://www.tcd.ie/history/staff/ruth-karras.php					
10	https://www.tcd.ie/history/postgraduate/taught/medieval/					
11	https://www.tcd.ie/history/staff/immo-warntjes.php					
12	https://www.tcd.ie/history/about/					
13	https://www.tcd.ie/history/staff/isabella-jackson.php					
14	https://www.tcd.ie/history/staff/murdocg.php					
15	https://www.tcd.ie/history/postgraduate/taught/international-history/					
16	https://www.tcd.ie/history/staff/clarkej1.php					
17	https://www.tcd.ie/history/staff/patrick-walsh.php					
18	https://www.tcd.ie/history/research/centres/irish-history.php					
19	https://www.tcd.ie/history/postgraduate/taught/early-modern/					
20	https://www.tcd.ie/history/staff/cbrady.php					
21	https://www.tcd.ie/history/staff/gearyd.php					
22	https://www.tcd.ie/history/staff/peter-crooks.php					
23	https://www.tcd.ie/history/research/					
24	https://www.tcd.ie/history/undergraduate/tsm-history.php					
25	https://www.tcd.ie/history/staff/ditchbud.php					

Figure 3.5: An extract of the schema given to judges. The Microsoft Excel spreadsheet contains the URL (document path), a column where judges annotate a ranking, and an implementation note comprising a key to ranking score.

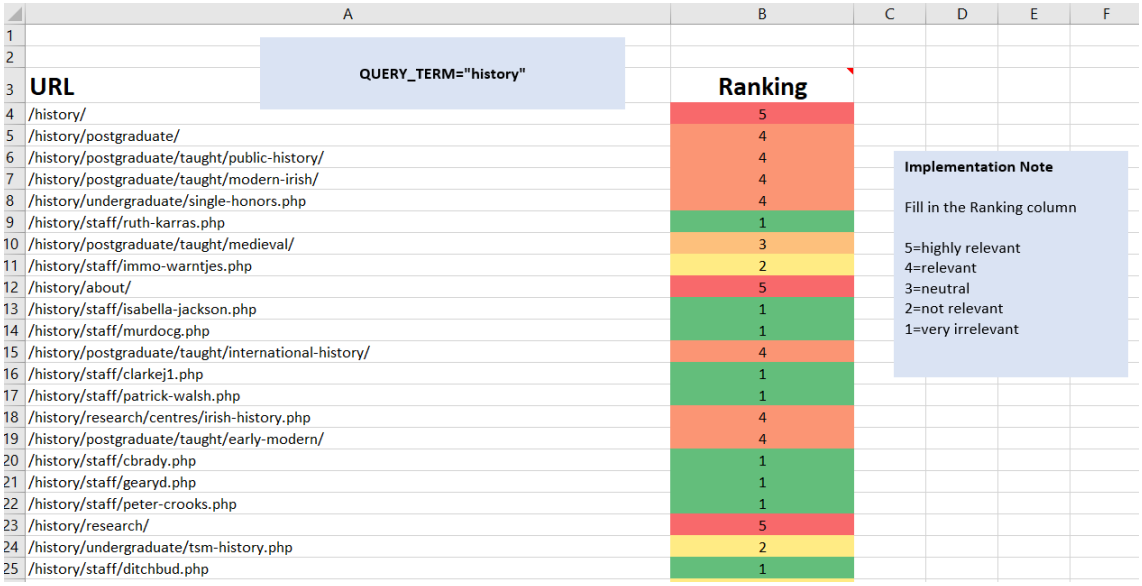
The 'ranking' column was left blank for annotators to complete. Microsoft Excel was chosen as the annotation tool for the following reasons:

- All judges are already familiar with and skilled in using this tool.
- All of The University's desktops and laptops have a pre-installed and licensed copy.
- In the spreadsheet for annotators, rankings were conditionally formatted with a red-yellow-green colour scale as follows: cells with the highest judgement (5) are red; those with the lowest (1) are green. All intermediate judgements are yellowish-orange.

An example of the uncompleted spreadsheet sent to the judges is shown in Figure 3.5. The query term is 'history', and in Figure 3.6 judge has completed the numerical annotation for each document in the 'ranking' column with scores (ranging from five to one).

Annotators were asked to limit the number of '5's awarded (see feedback section below). The distribution of labels in the generated dataset (which I name `ENTRP-SRCH`) is presented in Table 3.2.

Judges' Feedback Over the course of the ranking survey, several clarifications were requested and/or issued. As described in Section §2.4.5, Rotter was able to foresee such issues when he said that "guidelines which are unclear or incomplete need to be clarified or expanded by dataset creators, which may require iterative approaches to annotation" [165].



	A	B	C	D	E	F
1						
2	QUERY_TERM="history"					
3	**URL**	**Ranking**				
4	/history/	5				
5	/history/postgraduate/	4				
6	/history/postgraduate/taught/public-history/	4				
7	/history/postgraduate/taught/modern-irish/	4				
8	/history/undergraduate/single-honors.php	4				
9	/history/staff/ruth-karras.php	1				
10	/history/postgraduate/taught/medieval/	3				
11	/history/staff/immo-warntjes.php	2				
12	/history/about/	5				
13	/history/staff/isabella-jackson.php	1				
14	/history/staff/murdocg.php	1				
15	/history/postgraduate/taught/international-history/	4				
16	/history/staff/clarkej1.php	1				
17	/history/staff/patrick-walsh.php	1				
18	/history/research/centres/irish-history.php	4				
19	/history/postgraduate/taught/early-modern/	4				
20	/history/staff/cbrady.php	1				
21	/history/staff/gearyd.php	1				
22	/history/staff/peter-crooks.php	1				
23	/history/research/	5				
24	/history/undergraduate/tsm-history.php	2				
25	/history/staff/ditchbud.php	1				

Implementation Note

Fill in the Ranking column

5=highly relevant
4=relevant
3=neutral
2=not relevant
1=very irrelevant

Figure 3.6: An extract of the completed schema with conditional colour formatting for annotations.

Table 3.2: Distribution of human relevance judgements in`ENTRP-SRCH`.

Relevance Label	Interpretation	Number	Percentage
5	highly relevant	147	5.78%
4	relevant	184	7.23%
3	moderately relevant	359	14.17%
2	irrelevant	1639	64.45%
1	utterly irrelevant	214	8.42%

For some judges, the objective of the survey was not sufficiently understood. All judges universally understood the act of soliciting judgements on a document per query as an endeavour by IT Services to improve ranking results. However, some judges were labouring under the impression that their annotations would lead to a static mapping in precisely the order that they had prescribed. These judges were later surprised that their 'judgements' were not precisely implemented per their *instruction*. The more technologically savvy judges understood that their annotations would be diluted when merged into a larger pool of data, which would, in turn, be used for *training* a general model.

In another example of feedback from judges, it was found that the conditional colour formatting in the spreadsheet is helpful in overcoming a confusing numbering system. 'Number 1' might have been incorrectly interpreted as a 'top score'. Our numbering system operates in reverse (5 is the best score, 1 indicates least relevance/importance). The advantage of using a reverse numbering system is that it limits judgements between 5 down to 1. If a forward numbering system was used, a 1 would be used for the most relevant/important, but this could sink to 10 or 100 for the least relevant. In this respect, the employed reverse numbering system has the advantage of imposing natural bookends. Alternatively, the MS Excel 'data validation' (a drop-down list of restricted numbering) could have been employed.

One judge appeared offended by the suggestion that not every document was highly relevant, initially rating each one as a '5'. It was later clarified that this approach undermined the purpose of the survey. Upon discussion, the misunderstanding was traced to a related initiative where microsite editors were instructed to remove outdated documents. The annotator had assumed that any document rated less than highly relevant could eventually be marked for deletion. After reassurance, the judge adjusted their ratings accordingly.

Another common area of confusion among judges is why 0 (zero) is not presented as an allowable judgement (e.g., to represent a document that the judge would prefer not to show in search results at all). It was clarified that each document is already found to have a minimum degree of relevance according to the BM25 similarity retrieval function.

Judges' Inter-Annotator Agreement As described in Section §2.2.20, Inter-Annotator Agreement (IAA) examines agreement among independent observers who rate, code, or assess the same phenomenon.

Throughout the selection process, it was assumed that a single judge would be sufficient to annotate relevant documents within the microsite under their responsibility. This is a departure from the usual practice in Web Search, which involves having multiple annotators as a quality control. The difference is that judges in the field of ES come from within the organisation, are likely to be topic experts and participate based on goodwill. Van der Lee describes this as distinguishing 'between expert-focused and reader-focused evaluation' [217].

This assumption is tested in an IAA experiment EXP-4 where the agreement between the labels of two independent judges is measured. It is designed to answer the question 'are the annotations in the `ENTRP-SRCH` dataset reliable, given that a large portion of the Q-D pairs were rated by single judges?'. The implementation challenges of IAA for organisational and ES contexts are discussed in Section §3.10.1 and the experimental results are presented in Section §3.10.2.

3.4.4 LTR Dataset Construction

When the 15 judges have completed the annotations for a total of 2544 Q-D pairs, covering the 20 queries, a formatted dataset is created. As described in Section §2.4.5, the LETOR format stipulates a relevance judgement in the first column, a query identifier (`qid`) in the second column, followed by a numbered feature vector array. The University's new dataset is named `ENTRP-SRCH`, and an extract is shown in Figure 3.7. Features 1 to 8 refer to BM25, recency, isAbout, isContact, view count, URLlength, LinkRank, and CTR, respectively. Each of these features is derived from the ranking functions described in sections §3.2 and §3.3.

The properties of the `ENTRP-SRCH` dataset are summarised and compared with popular LTR datasets for WS in Table 3.3. Since `ENTRP-SRCH` is much smaller than the Microsoft and Yahoo! WS datasets, a validation test was carried out to confirm that it is sufficiently sized (see Section §3.12.2).

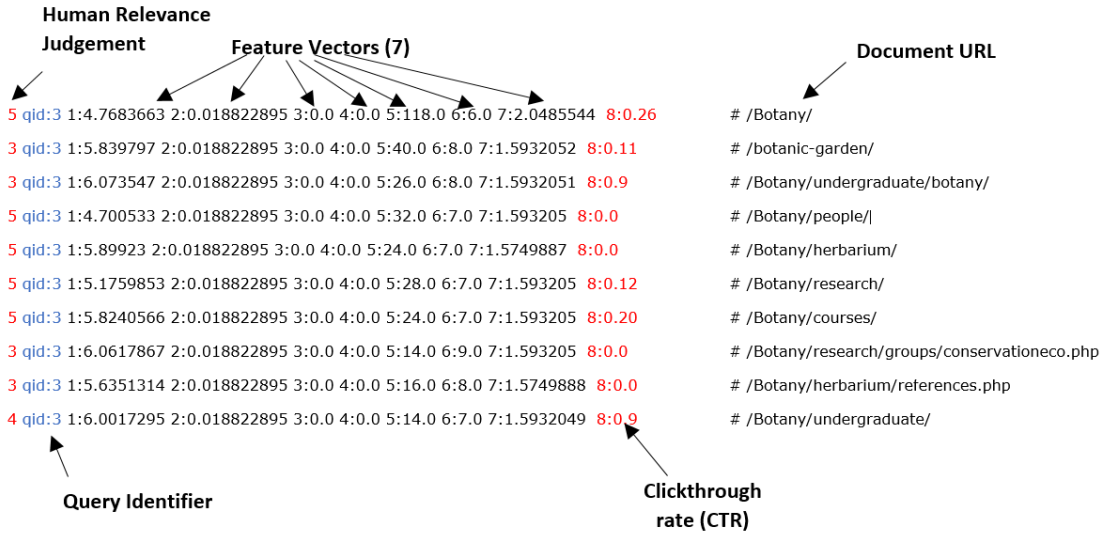


Figure 3.7: The LETOR-formatted dataset. Each row represents a query-document pair. This dataset contains explicit relevance judgements (first column) and the calculated click-through rate (in red). In the example, the query with id '3' includes a value of '8:0.26' for the first document/row, meaning that feature 8, which is the CTR, has a value of 26%. The '5' in the first column indicates that this document has been labelled 'highly relevant' for this query.

An anonymised version of the `ENTRP-SRCH` dataset is publicly available for download from GitHub. The anonymised version redacts the comment column (i.e., the part after the '#' containing the document path), which is not needed for the functioning of the LTR framework.

Table 3.3: Comparison of the properties of popular LTR datasets and The University's `ENTRP-SRCH` dataset.

	Microsoft	**Yahoo!**	**`ENTRP-SRCH`**
Publication year	2010	2010	2022
Documents	3771K	883K	2544
Queries	31531	36251	20
Avg. doc per query	119	24	127
LTR Features	136	700	8
No. of Reviewers	Not Stated	Not Stated	15
Clickthrough records	None avail.	None	375
Corpus Type	WS	WS	ES

The `ENTRP-SRCH` dataset will be employed in experiments EXP-1, EXP-2, EXP-3, EXP-4 and EXP-5.

3.4.5 LTR Model

At this point, the LTR-formatted ES dataset has been created and is ready to be used to generate a search results ranking model. This model will be used in EXP-1 to

compare an LTR model against traditional ranking approach and address RQ-1. The ranking model is generated using the XGBoost (Extreme Gradient Boosting) [37] implementation of the LambdaMART list-wise ranking algorithm. The Rankeval framework [38,39] performs comparably to XGBoost and is employed to cross-verify nDCG scores and validate weightings.

As outlined in Section §2.4.9, a list-wise loss function examines the entire ranked list rather than individual pairs or items. The LambdaMART algorithm uses gradient boosting to build an ensemble of decision trees. With each new tree, it improves the ranking by reducing errors in previous predictions. LambdaMART optimizes list-wise ranking measures, making it well-suited for tasks where the position of the first matching item in a ranked list is of prime importance (as is the case in a list of search results). No single fixed set of training queries is used as all queries participate in model training through k-fold Cross Validation (CV) (each fold uses a different subset of queries for training and reserves a disjoint subset for evaluation). CV is discussed in §3.12.

The implementation of LambdaMART within the XGBoost LTR framework, together with the hyperparameters used, is detailed in Section §3.8.1.

In this section, the background methodology to create a dataset for use with the LTR frameworks has been described. The generated dataset, based on The University's corpus, is used in Section §3.6 where a series of LTR experiments are undertaken. The resultant weightings output by the LTR model are described in Section §3.8.2. The following section describes the background methodology for another AI method, namely language modelling, in the domain of enterprise search.

3.5 Language Modelling for ES Jargon

This section examines how LM methods can be used to boost ES-specific content and an organisation's specialised language and jargon (RQ-2). As discussed in Section §2.1.8, ES content is often imbued with organisational terminology/jargon (i.e., words, phrases, expressions, and idioms that are not universally familiar or have different meanings inside and outside of the organisation).

3.5.1 JARGES LM feature for ES

To address this problem, a ranking model that integrates Learning to Rank (LTR) and Language Modelling (LM) is proposed. The LM component is called 'JARGES' (JARGon for Enterprise Search), and is designed to detect and decode jargon in ES queries and corpora (OBJ-3). The detection and decoding stages of JARGES are demonstrated in Figure 3.8. This study aims to evaluate the effectiveness of this approach in improving search result rankings, particularly for content rich in organisational terminology. To test this hypothesis, a quantitative evaluation of the performance of an LTR ranking model with and without JARGES is performed. The LTR-formatted `ENTRP-SRCH` dataset (2,544 human-annotated Q-D pairs) [136]

is used. Subsequently, a qualitative analysis of the decoded jargon terms via their contextual synonyms is performed.

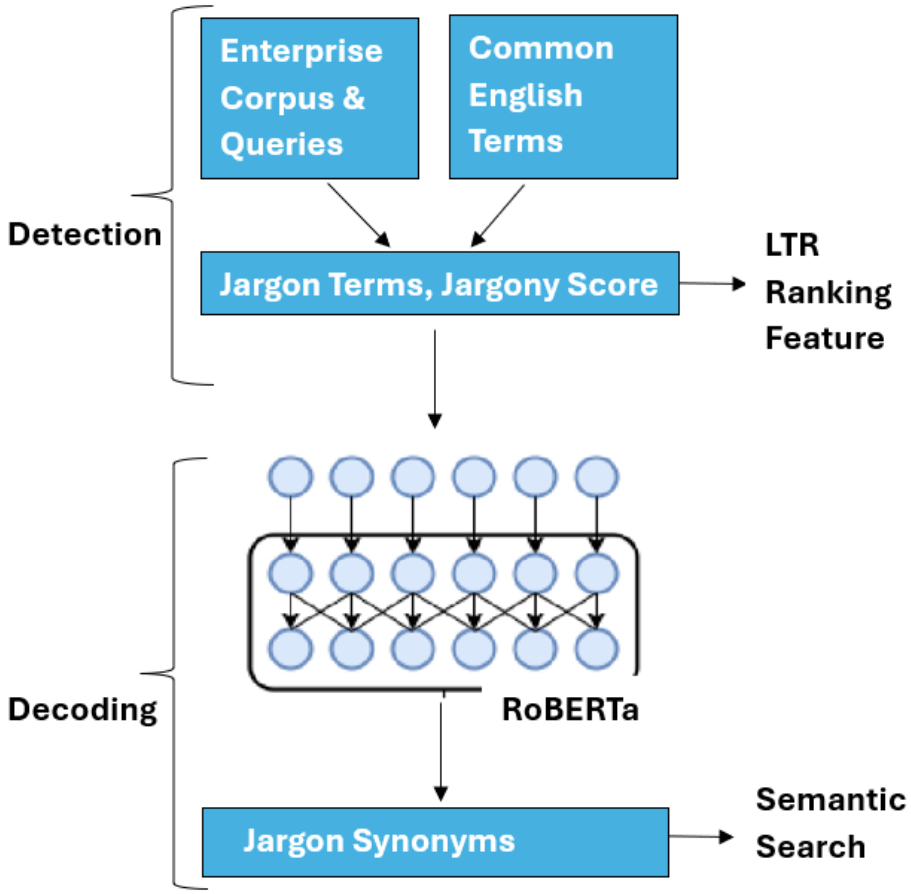


Figure 3.8: Architecture of the JARGES algorithm, including the detection (classification) and decoding (synonym generation) stages.

The challenge is to detect enterprise specific words, phrases, expressions, and idioms that diverge from those universally familiar or understood outside of the organisation. This divergence is measured by computing the absolute difference in TF-IDF scores for the terms.

The mathematical formulation of JARGES works by assigning a *jargony score* to each term t by measuring the degree to which its TF–IDF salience in the enterprise corpus diverges from its salience in a general English reference corpus. The score is defined as:

$$J(t) \;=\; \Big|\,\mathrm{TFIDF}_{\mathrm{ES}}(t) \;-\; \mathrm{TFIDF}_{\mathrm{REF}}(t)\,\Big|.$$

A term is classified as jargon when its divergence exceeds the minimum threshold δ:

$$J(t) \;\geq\; \delta,$$

where δ was set to 15% in our experiments (which is the same as the threshold adopted in Belfathi's study [201], full implementation details described in Section §3.11.1). The resulting ranked list of jargon terms, used as an LTR feature, is

obtained by sorting all terms by descending jargony score:

$$\mathcal{J} \;=\; \mathrm{sort}_t\Big(J(t)\Big).$$

For the decoding stage, contextual synonyms for each detected jargon term $t \in \mathcal{J}$ are generated using a RoBERTa model fine-tuned on the enterprise corpus:

$$\mathrm{Syn}(t) = \mathrm{RoBERTa}^{FT}(t),$$

where $\mathrm{RoBERTa}^{FT}$ denotes the fine-tuned model.

These synonyms are subsequently ordered using the jargony scores $J(t)$ so that more specific, higher-divergence terms receive greater priority during query-time synonym expansion.

The implementation details of JARGES are described in 3.11.1 and experiment EXP-5 includes a quantitative and qualitative analysis of its contribution. The results are presented in Section §3.11.2.

3.6 Experiments

Five experiments were designed to determine the extent to which LTR and LM methods can be used to improve the performance of the ranking of search results for an organisation's content and specialised jargon/terminology (RQ-1, RQ-2).

Each of the experiments listed below is designed using a consistent structure (description, hypothesis, baseline, mechanism, sufficiency of data, domain knowledge, limits, and expected outcomes). The experiments are intended to be implemented independently, but they are presented in a logical sequence (for example, EXP-2 explores questions that emerge from EXP-1). EXP-3 and EXP-4 investigate whether the `ENTRP-SRCH` dataset exhibits the properties required for the subsequent stages of analysis. The results are subsequently presented in Sections §3.7.2, §3.8.2, §3.9.2, §3.10.2 and §3.11.2.

3.7 EXP-1: Search Results Ranking Approach Comparison

The objective of this experiment is to compare search results ranking performance between a traditional approach and an AI approach in the domain of ES (OBJ-1). For the traditional approach, the search result ranking is configured to use a combination of BM25 and field boosting functions. The performance of this approach is then compared with an LTR ranking model generated using the eight features (described in Section §3.4.4, and will again be a central focus for EXP-2).

Apart from determining whether an AI approach can improve ES search result ranking, the outcome informs the discussion of whether state-of-the-art AI ranking methods are 'worth the effort' (as discussed in Section §2.4.13). Moreover, this

experiment helps to gauge whether the current approach is sufficiently well-founded to justify moving on to the next stages of the study.

A description of the `ENTRP-SRCH` dataset, the dataset itself, and a study which includes the experiment outlined below were published in Paper I "Learning to Rank: Performance and Practical Barriers to Deployment in Enterprise Search" [136] and the associated code is available on GitHub.

- Hypothesis: The hypothesis is that the search results ranking performance, as measured by nDCG and MAP, improves using the LTR model, generated with eight features, versus using a combination of BM25 and field boosting ranking functions.

- Baseline: Two baselines are employed as part of the traditional approach:

  - BM25 is Lucene's default 'out-of-the-box' ranking function for search results, which is why it is chosen as the basis of the traditional approach for this experiment.
  - Combined with this, field boosting is also applied as a re-ranking function. Four fields used are for boosting (title, content, URL and anchor).

  A consistent processing pipeline is applied across all ranking approaches, including tokenisation, stopword removal, and stemming. This pipeline uses a customised configuration (e.g., edits to the stopword list), but these settings are fixed throughout the study and therefore treated as controlled variables. Both the traditional baselines and the LTR models operate over exactly the same preprocessed text, ensuring that differences in ranking behaviour arise from the ranking approaches rather than from variation in text analysis.

- Mechanism: As described in Section §3.4.5, the LTR ranking model is generated using the XGBoost framework and the LambdaMART list-wise ranking algorithm. The model produced from eight features. No single fixed set of training queries is used as all queries participate in model training through k-fold cross-validation (each fold uses a different subset of queries for training and reserves a disjoint subset for evaluation). This LTR model represents the AI approach and is compared with the traditional baselines. In all cases, ranking is measured using nDCG and MAP metrics.

- Sufficiency of Data: The `ENTRP-SRCH` dataset consists of 20 Queries and 2544 Query-Document (Q-D) pairs with manually annotated relevance judgements. A subsequent validation test involving a 'learning curve graph' (Section §3.12.2) provides evidence that the dataset size is sufficiently sized for the intended purpose.

- Domain Knowledge: As described in Section §3.4.3, microsite (subsite) owners were solicited to judge the relevance of a document for a given query. The microsite owners must be familiar with published content to avoid replication and ensure consistent messaging. These microsite owners are well placed to 'judge' the ranking order of their pages for a particular term. A subsequent IAA experiment (EXP-4) is performed to validate the integrity of the annotations.

- Limits: If the AI approach yields only a marginal difference relative to the traditional baseline, any potential benefit must be considered in terms of the practical implementation effort and costs for ES organisations (where search may not be at the core of the business, and organisational resources and skills may be limited).
- Expected Results: An improvement is to be expected when using the AI approach. Not only does the `ENTRP-SRCH` dataset incorporate both BM25 and field boosting features as well as many other ranking features, LTR can optimise the weighting of each. More specifically, it is expected the LTR ranking model will prove to perform better than BM25 combined with field boosting, which in turn, will perform better than using BM25 on its own.

The techniques to implement the field boosting (traditional ranking function) and LTR are described in Section §3.7.1 and Section §3.8.1 respectively. The results of EXP-1 are presented in Section §3.7.2.

3.7.1 Implementation (EXP-1: Search Results Ranking Approach Comparison)

This section details the practical application of the ranking functions described in sections §3.2 and 3.3, as well as the implementation of methods described in the experimental design section (§3.6).

View Count The implementation of the view count ranking function involves i) the creation of a so-called 'external' field in Apache Solr and ii) the generation of a list of document view count tuples.

The field and field type configuration within the Apache Solr schema.xml file are shown in Listing 3.4. The ExternalFileField class allows for data to be read from an external file.

```
<fieldType name="externalPopularityScore"
           keyField="id"  defVal="1"
           stored="false"  class="solr.ExternalFileField"/>
 ...
<field name="popularity" type="externalPopularityScore"/>
```

Listing 3.4: View Count implementation in the schema.xml file of Apache Solr.

The generation of a list of document view count tuples is relatively easy. Listing 3.5 shows a simple bash shell command that utilises native tools like awk, sed and sort to extract the top 10 most viewed documents from The University's active Apache HTTP server log file. The listing is an example from an active log file. For the creation of the `ENTRP-SRCH` dataset, six months' worth of web server logs were concatenated. This list of tuples is stored in a file called `data/external_popularity`. The file is updated nightly via a scheduled 'cron' job. Updates to this file do not require a restart of Apache Solr.

```
$ cat www_access_log | awk '{ print $7 }' |\
grep \.php$ | sort |\
uniq -c | sort -rn | head -10

  54898 /check_filesystem.php
  18302 /check_db1.php
    754 /wp-login.php
    578 /xmlrpc.php
    568 /study/international/how-to-apply/entry-requirements.php
    426 /study/404/index.php
    332 /students/orientation/visiting-exchange/module-enrolment.php
    330 /museums/index.php
    320 /study/apply/admission-requirements/undergraduate/index.php
    312 /study/international/scholarships/postgraduate/gexpg.php
```

Listing 3.5: An example shell command that extracts MV tuples from an active Apache HTTP server log file. Note that the first few tuples are outliers as they are generated by operational monitoring/health checks

The results of using view count as a traditional ranking function are presented in EXP-1 results, and the results of its application as an LTR feature are presented in EXP-2 results and EXP-3 results.

Recency Many Web Content Management Systems (WCMS) have an integration to register a new or modified document's publication timestamp directly to a search index. In the case of ES, where documents are typically distributed across heterogeneous systems (corporate directories, web pages, document shares, ticketing systems, etc.), the implementation of ranking by recency of publication date is not straightforward. The separate repositories/systems are unlikely to have an integration to automatically update the ES index.

To record the publication date of documents in The University's ES corpus, several custom scripts are created to extract the timestamps from various document types. The timestamps are then stored in Solr's index. The Python `beautifulsoup` library is used to scrape the 'pubdate' from documents and register it within the index.

However, some ES document types, such as records from the corporate directory, do not present any timestamp data (or metadata). Other documents, with extensions such as TXT or CSV, do not have built-in metadata for a creation date. While such timestamps could, in principle, be inferred from filesystem metadata using additional tools (e.g., Bash or MS PowerShell scripts), this supplementary processing was not undertaken in the present study. Documents with no easily extractable timestamp are assigned the median timestamp value in the document's pubdate field.

```
curl 'https://\$SOLR_SERVER:8983/solr/$CORE$/update?commit=true'
     -H 'Content-type:application/json'
     -d '[{"id":"/Botany/vacancies/", \
     "pubdate":{"set":"2020-06-24T00:00:0Z"}}]'
```

Listing 3.6: Recency implementation using cURL to update the 'pubdate' field in the Solr index.

Listing 3.7 shows how recency can be defined as a feature in an LTR model. Note that the arguments passed to the `recip` function represent the x,m,a and b values described in Section §2.2.9.

```
{
  "store": "myCombinedFeaturesStore",
  "name":  "documentRecency",
  "class": "org.apache.solr.ltr.feature.SolrFeature",
  "params": {
    "q": "{!func}recip( ms(NOW,pubdate), 3.16e-11, 1, 1)"
      }
}
```

Listing 3.7: LTR feature definition for recency implementation.

The results of using recency as a LTR feature are presented in EXP-2 results.

LinkRank As described in Section §2.2.5, Apache Nutch is an open source web crawling and indexing framework that utilises a plug-in architecture, providing flexibility to incorporate custom scoring algorithms such as LinkRank. LinkRank first creates a webgraph that stores output scores for each document in a node database. It then combines the anchor text associated with each document's inlinks and outlinks. The code to achieve this has been uploaded to GitHub and here is an outline of the steps involved:

1. A webgraph data structure is generated on a segment-by-segment basis.
2. LinkRank calculates the score based on the webgraph.
3. Score Updater updates the score from the webgraph back into the crawldb.
4. The data is then dumped in readable format into a file.
5. The linkdb mini database is created.
6. The contents of linkdb are dumped to create an anchors file.
7. The complete list of solr documents is then dumped into file.
8. Finally, the linkrank field in Solr is updated with a numerical score.

A field called linkrank is created in Apache Solr to store the score for each document, as shown in Listing 3.8. The `pdoubles` field type (which can store double-precision floating-point numbers) is used as the LinkRank scores have a precision of 8 and a scale of 5 (e.g., 210.07489).

```
<field name="linkrank" type="pdoubles" stored="true"
            indexed="true" required="false"/>
```

Listing 3.8: LinkRank schema definition for Apache Solr.

The LinkRank scores are updated weekly (via a weekend scheduled cron job). This update frequency was chosen for the following reasons:

- While a target document may change, the number of inlinks is unlikely to change dramatically over a shorter period of time.
- For The University's ES architecture, the Apache Nutch crawler runs on the same server as Apache Solr. The eight step process above is computationally expensive and in danger of creating a search latency if there is competition for CPU and memory resources.

Listing 3.9 shows how LinkRank can be defined as a feature in an LTR model.

```
{
  "store" : "myCombinedFeaturesStore",
  "name":  "linkRank",
  "class": "org.apache.solr.ltr.feature.FieldValueFeature",
  "params": {
    "field": "linkrank"
      }
}
```

Listing 3.9: LinkRank LTR feature definition.

The result of using LinkRank as an LTR feature, including the weights allocated to it relative to other features, is presented in EXP-2 results. LinkRank also forms part of EXP-1 and EXP-3 (where it is used indirectly as part of the `ENTRP-SRCH` dataset).

Field Boosting Field boosting can be performed either at index time or at query time. Boosting at index time would mean the boost is computed once and stored with the document. Index time boosting means that the boost factor can not be adjusted later (a re-indexing would be required).

In Section §3.2.5, an example was given to show how query time field boosting could be implemented in its simplest form as arguments passed to to the Solr `qf` parameter.

Another way to use field boosting as a traditional ranking function is to hard code it into the Solr `solrconfig.xml` or `params.json` files as shows in Listing 3.10.

```
<field name="title" type="text_general" boost="2" />
<field name="url" type="text_general" boost="4" />
<field name="content" type="text_general" boost="1" />
<field name="anchor" type="text_general" boost="3" />
```

Listing 3.10: Boosting individual fields.

In the context of LTR, a field *boost factor* is treated as just another LTR *weighting*. Listing 3.11 shows how field boosting can be defined as features in an LTR model. The `titleOrginalScore` is a BM25 score for the title field. The boost is converted to a weight which is determined by the LTR framework and defined in the Solr `myCombinedModel.json` file (described in Section §3.8.1).

```
{
  "store" : "myCombinedFeaturesStore",
  "name" : "titleOriginalScore",
  "class" : "org.apache.solr.ltr.feature.OriginalScoreFeature",
  "params" : { "field": "title"}
},
```

Listing 3.11: Field Boosting converted to an LTR feature definition.

The result of using field boosting as part of the traditional approach is presented in EXP-1 results and as an LTR feature, including the weights allocated to it relative to other features, is presented in EXP-2 results.

Keyword Boosting Keyword boosting can be applied at query time or at index time [218]. Apache Solr uses the caret symbol (^), followed by a numeric boost factor, appended to a query term. This method was previously used in the University's ES service as part of a traditional ranking approach (though it is not included in the baseline for EXP-1).

After transitioning to an AI-based approach, keyword boosting can be incorporated directly as an LTR feature. Listing 3.12 illustrates how terms such as 'about' or 'contact' can be modelled within Solr's LTR feature definition file. Such terms frequently correspond to canonical organisational pages (e.g., 'About Us' pages, contact directories, or governance statements), which users often expect to appear prominently for certain queries. Incorporating these terms as features simply enables the LTR model to learn when these canonical pages should be surfaced, rather than relying on manual, static boosting rules.

```
{
    "store" : "myCombinedFeaturesStore",
    "name" : "isContact",
    "class" : "org.apache.solr.ltr.feature.SolrFeature",
    "params" : {
      "fq": ["{!terms f=title}contact"]
        }
  },
  {
    "store" : "myCombinedFeaturesStore",
    "name" : "isAbout",
    "class" : "org.apache.solr.ltr.feature.SolrFeature",
    "params" : {
      "fq": ["{!terms f=title}about"]
        }
  },
```

Listing 3.12: LTR feature definition for keyword boosting implementation. The two features below will boost any document with the words 'contact' or 'about' in the title field.

The result of using keyword boosting as an LTR feature, including the weights allocated to it relative to other features, is presented in EXP-2 results.

Anchor Text Ranking via an anchor text field and the linkrank ranking function are conceptually interrelated; both functions exploit hyperlink information as a proxy for document authority and relevance.

In practice, the linkrank function is implemented before the ranking by anchor text. As described in 3.7.1, the creation of an anchor text file is a by-product of the linkrank function creation.

A new string field called 'anchor' is created in The University's Solr index to hold a list of anchor text descriptors (Listing §3.13).

```
<field name="anchor" type="text_general"
            stored="true" indexed="true"
            multiValued="true"/>
```

Listing 3.13: Anchor text field definition in the Solr schema.

An example of anchor text used for ranking is 'accommodation 33', which tells us that the term accommodation is deployed in 33 separate pages as a descriptive anchor text (pointing to a particular page or group of pages).

An analysis of The University's generated anchor text file reveals extensive use of non-descriptive phrases like "click here", "read more", "learn more" or simply "next page". Each of these anchor texts could be considered a wasted opportunity to provide additional context to a query term and boosting to a document. Consequently, such generic phrases are filtered out of the anchor field during preprocessing (to reduce noise and improve the quality of the anchor-based feature).

Listing 3.14 shows how Apache Nutch can be configured to update the anchor text field in Apache Solr as part of the crawling process.

```
<name>plugin.includes</name>
  <value>parse-(pdf|tika)|protocol-http|
   urlfilter-(regex|validator)|
   index-(basic|anchor)|
   indexer-solr|
   scoring-opic|urlnormalizer-(pass|regex|basic)
   </value>
```

Listing 3.14: Anchor text field definition in Apache Nutch. The anchor word in the configurion ensures that the anchor text field is updated.

Anchor text is a regular field in Apache Solr (not unlike the title or content fields). It is used as a method in the traditional ranking approach, and the result is presented in EXP-1 results. The result of using anchor text as an LTR feature, including the weights allocated to it relative to other features, is presented in EXP-2 results.

URL or Path Length An example of how the length of the path of a document could be used as a ranking function was given in Section §3.3.4.

Apache Solr has a `FieldLengthFeature` that provides a quantifiable measure of a document field's length. It can be integrated into the ranking LTR definitions as shown in listing 3.15. This feature is computed for each returned document at query time. Field length is an important component in traditional ranking models like BM25, where longer documents are often penalised to prevent length bias. The LTR framework can learn the impact of FieldLengthFeature on a particular field.

```
    "store" : "myCombinedFeaturesStore",
    "name":  "urlLength",
    "class": "org.apache.solr.ltr.feature.FieldLengthFeature",
    "params": {  "field": "url"  },
```

Listing 3.15: LTR feature definition for ranking by URL length implementation.

The result of using path length as an LTR feature, including the weights allocated to it relative to other features, is presented in EXP-2 results.

Query Frequency Most Frequent Queries (MFQ) data is used to analyse the character of the queries within The University ES service. This analysis is necessary for selecting queries for Q-D annotation.

MFQ is easily extracted from The University's Solr search logs. An extraction demonstration is shown in Listing 3.16. It shows a simple bash shell command that utilises native tools like awk, sed and sort to extract the top 10 most frequent queries from an active solr log file.

```
$ cat /var/log/solr/solr.log |\
> grep -o -P '(?<=q=).*(?=&def)' |\
> awk '{print tolower($0)}' |\
> sort  | uniq -c | sort -nfr |\
> awk '{for (i=2;i<=NF;i++) printf("%s ",$i)} {print "\t" $1}' |\
> sed 's/ \t/\t/g' | head -10

fees            36
phd             27
vacancies       23
jobs            22
library         19
calendar        18
insurance       16
medicine        15
book of kells   15
scholarship     13
```

Listing 3.16: An example shell command that extracts MFQ from a Solr log. The output of the command shows the top ten most popular queries and their frequency count.

MFQ is used as input to create the `ENTRP-SRCH` dataset, which is used in EXP-1, EXP-2, EXP-3 and EXP-4.

Query Clustering A problem with extracting raw MFQ data in the manner described above is that it does not distinguish between contextually similar words and phrases (e.g., jobs and vacancies in the above listing). As described in Section §3.4.2, query clustering is the process of grouping similar queries into clusters based on their lexical or semantic similarity using the word embedding technique.

Word embedding involves converting The University's corpus into into a numerical representation where each word is represented as a dense vector in a high-dimensional space. These vectors capture semantic relationships among words and phrases. Here's an outline of the process employed:

- A dumpfile containing The University's corpus is read line by line, each of which is tokenised using the Python `gensim simple_preprocess` library.
- The Word2Vec embedding algorithm uses shallow neural networks. The hidden layer weights become the word vectors. The Word2Vec model is trained on the tokenized corpus, resulting in dense vector representations for each word in the vocabulary. Once trained, each word in the vocabulary is associated with a vector.
- The k-means clustering (vectorisation method) is then applied to the set of word embeddings (representing the query). The model is trained so words appearing in similar contexts have similar vector representations. The skip-gram model is employed to maximise the probability of the word or phrase context.
- The number of clusters (num_clusters variable) is adjusted until there are about five objects for each of the keyword/phrase queries (this is sometimes referred to as the 'elbow method'). At a minimum, the cluster for the 'vacancy' query should contain 'vacancies' and 'jobs'.
- The tensorflow embedding projector is then used to verify and visualise the model (as shown in Figure 3.4.

The above steps are scripted in a Python notebook called `es-query-clustering.ipynb` which can be viewed in the GitHub repository.

Table 3.4 shows the hyperparameters used during the Word2Vec query clustering process of The University's corpus.

Table 3.4: Query Clustering Word2Vec and kmeans hyperparameters.

Hyper Parameter	Value
model	Word2Vec
vector_size (dimensions)	100
n_init	auto
window	5
min_count	1
workers	4
num_clusters	2000

As for MFQ, query clustering is not used directly in the experiments. Instead, it is an input (a Solr field consisting of a query-count tuple) to create the ENTRP-SRCH dataset, which is part of EXP-1, EXP-2, EXP-3 and EXP-4 word2vec and gensim combined hyperparamter table.

CTR collection As described in Section §2.2.13, a basic 'cascading' click model treats user behaviour as a sequential scanning process from top to bottom. Users examine each item one by one and decide whether to click.

When an end-user submits a query to The University's search engine, they are presented with a ranked list of documents. The click model, as implemented in the Apache Solr search logs, records only a single click per query instance. Specifically, the logging captures:

- the submitted query and its corresponding timestamp;
- the ordered list of the top fifty documents presented on the search engine results page;
- the position of the document clicked by the user (and its timestamp), recorded as a single-click event.

This logging configuration does not capture multiple clicks or more complex interaction patterns; only the first (or sole recorded) click is available for analysis, which should be considered a limitation of the click model used in this study.

The Apache Solr log4j module [219] is customised to capture the above items. Sessions where a query is submitted but no document is clicked within a 60-second time limit are excluded. The results of using the click-through model are presented in EXP-3 results.

3.7.2 Results (EXP-1: Search Results Ranking Approach Comparison)

As described in section Section §3.7, this experiment compares the search result ranking performance for the traditional and AI approaches. The traditional approach is represented by i) BM25 (on its own) and ii) BM25 with field boosting (4 fields). The AI approach is represented by LTR using the LambdaMart algorithm with The University's eight-feature ENTRP-SRCH dataset.

Table 3.5 shows the results for the three tested methods. The nDCG values for each of the three methods are additionally graphically represented in Figure 3.9.

Table 3.5: Evaluation scores for three ranking methods as applied to the ENTRP-SRCH dataset. The metrics used are nDCG@{3,5,10} and MAP. Best performing scores are in **bold** font.

%	nDCG@3	nDNG@5	nDCG@10	MAP
BM25	57.54	60.55	55.38	40.33
BM25 combined with field boosting	94.08	92.80	81.18	60.29
Learning to Rank (8 features)	**99.08**	**98.65**	**98.24**	**80.13**

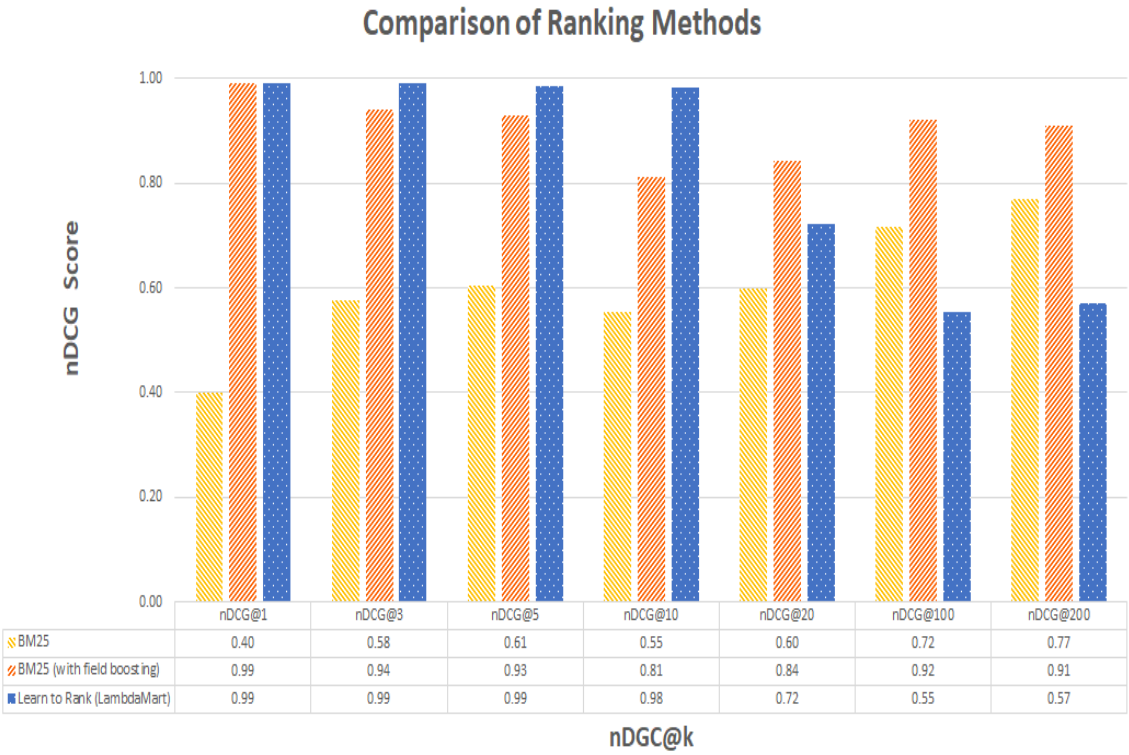


Figure 3.9: A comparison of the performance of Ranking Methods, as measured by nDCG, for the `ENTRP-SRCH` Enterprise Search dataset.

For nDCG@5, the LTR method performs 5.85% better than BM25 with field boosting. While it was expected that the LTR method would perform better, it is surprising that BM25 with field boosting can achieve such a high score. This could be explained by the fact that many of the queries are topical rather than exploratory in nature. BM25 with field boosting performs 32.3% better on using BM25 its own. This was in line with the expectations outlined during the experimental design phase. Surprisingly, the performance advantage of the LTR method (over BM25 with field boosting) is reduced and even reversed for nDCG@k, when k surpasses 20 and 100, respectively.

An ancillary outcome of this experiment is a greater understanding/appreciation of the additional implementation steps and effort required to adopt the AI approach. Measuring implementation effort is a subjective task, and there may be no way to represent it in absolute terms. Chapter 5 includes a discussion on whether the AI approach is 'worth the effort' for organisations, when the limited performance gain is taken into account.

3.8 EXP-2: Search Results Ablation Study

The objective of this study is to determine the relative weights for each of the LTR features (OBJ-1) in the `ENTRP-SRCH` dataset. The ablation process involves removing a feature, which allows direct observation how its absence affects model performance, confirming its true importance. An ablation study also provides insights into the redundancy, and interactions between features.

The experiment/study outlined below was also published as part of Paper I "Learning to Rank: Performance and Practical Barriers to Deployment in Enterprise Search." and, as for EXP-1, the associated code is available on Github.

- Hypothesis: The hypothesis is simply that the ranking features are of differing importance. Specifically, this study discovers the ranking features that make the biggest contributions to the model's performance. This 'experiment' can be regarded more as a study than an experiment in the strict sense.

- Baseline: For an ablation study, each collective feature set serves as a baseline before the systematic removal of a single feature. The performance of one feature is marked against 'all the rest'.

- Mechanism: This involves removing one feature from each iteration and performing the LTR training and testing steps again. The ranking performance differential is measured using the nDCG metric for each feature. If a large decrease in nDCG scores is observed, the ablated feature is very important for search results ranking model.

- Sufficiency of Data: The `ENTRP-SRCH` dataset consists of 20 Queries and 2544 Query-Document (Q-D) pairs with manually annotated relevance judgements. A subsequent validation experiment involving a 'learning curve graph' (Section §3.12.2) confirms that the dataset size is sufficiently sized.

- Domain Knowledge: The procedures and protocols of an ablation study are well-defined. No particular domain knowledge is required as the ES dataset has already been created. However, knowing which features are likely to make big or small contributions helps estimate the expected results (see below).

- Limits: The ablation study focuses on quantitative changes in performance. It provides no insights into why a feature may contribute positively or negatively. Hence, the results may be difficult to interpret without some domain knowledge.

- Expected Results: As outlined in Section §3.2, in ES, there is often an intuition of a 'good ranking'. In the traditional approach, a relevance engineer can spend many hours attempting to estimate the field boost factors by *eyeballing* search results. An LTR model, generated using eight features, is less transparent, and may produce more unpredictable results. Nonetheless, the practical trial-and-error experience of the traditional approach informs expectations for the LTR ablation study.

The techniques to implement the LTR framework and compute weights are described in Section §3.8.1. The results of EXP-2 are presented in Section §3.8.2.

3.8.1 Implementation (EXP-2: Search Results Ablation Study)

LTR weightings XGBoost implements LTR by solving ranking problems using the pointwise, pairwise and listwise loss functions (as described in Section §2.4.9).

For EXP-2, XGBoost is configured to optimise the 'listwise' (`rank:ndcg`) loss function. While listwise is computationally more expensive than pairwise, this is hardly a factor for the `ENTRP-SRCH` dataset, which has just 2544 Q-D pairs and eight features. Moreover, the pairwise loss function is designed for simpler tasks that have, for example, just binary relevance annotations (the `ENTRP-SRCH` annotations are in the range of [1-5]).

Table 3.6 lists the hyperparameters used by the XGBoost LTR framework to rank The University's ES search results.

Table 3.6: Hyperparameter settings used to evaluate ranking performance for the Learning to Rank ranking method.

Parameter	Value
Framework	XGBoost
Algorithm	LambdaMART
rank	ndcg
loss function	listwise
max_depth	10
num_round	15
eta	0.3

XGBoost is built atop the Gradient Boosting (GB) ML algorithm, widely used for supervised learning tasks and the 'max_depth' parameter is the maximum depth of the GB tree.

The learning rate (designated by 'eta') is set to 0.3. In the XGBoost gradient-boosted decision tree model, eta controls the contribution of each successive tree to the final model. A higher value for eta typically accelerates convergence but may lead to overfitting, while a lower value requires more boosting iterations (specified by num_round) to achieve comparable accuracy. The num_round parameter determines the total number of boosting rounds (i.e. how many trees are sequentially added to the ensemble). A high num_round value increases training time and computational cost, but allows the model to refine feature weights more gradually. Since this experiment was conducted offline to compute feature weights, it was not necessary to use an aggressive learning rate. As discussed in Section §2.2.19, nDCG and MAP are accepted as the de facto metrics (i.e. the 'rank' parameter) for measuring the ranking performance of search results.

The calculation of the XGBoost LTR weights is scripted in a Python notebook called `lambdamart-xgboost.ipynb` which can be viewed in the GitHub repository. The notebook also shows a visualisation of the model in the form of a decision tree (using the `xgb.plot_tree` function), which shows where 'splits' at nodes are conditional on the inputted features in the feature vector for each query.

The weightings are presented in EXP-2 results.

Model Integration with Apache Solr Once the LTR weightings are computed, the next step is integrating them with a model in the ES search engine. Apache Solr has a plugin called LTR [220] which can be enabled via the `solrconfig.xml` file by adding lines as shown in Listing 3.17.

```
<searchComponent name="ltr"
class="org.apache.solr.ltr.search.LTRQParserPlugin" />
<requestHandler name="/select" class="solr.SearchHandler">
  <lst name="components">
    <str name="component">ltr</str>
  </lst>
</requestHandler>
```

Listing 3.17: Enabling the LTR plugin for Apache Solr.

A feature definition file specifies the features used by the LTR model. Each feature represents a ranking function such as those described in sections §3.2 and §3.3. An extract of the Apache Solr feature definition json-formatted file used by The University's ranking model trained with the `ENTPR-SRCH` dataset is shown in Listing 3.18.

```
 $ cat myCombinedFeatures.json
[
  {
    "store" : "myCombinedFeaturesStore",
    "name" : "originalScore",
    "class" : "org.apache.solr.ltr.feature.OriginalScoreFeature",
    "params" : {}
  },
  {
    "store" : "myCombinedFeaturesStore",
    "name":  "documentRecency",
    "class": "org.apache.solr.ltr.feature.SolrFeature",
    "params": {
      "q": "{!func}recip( ms(NOW,pubdate), 3.16e-11, 1, 1)"
        }
  },
...
  {
    "store" : "myCombinedFeaturesStore",
    "name":  "rawHits",
    "class": "org.apache.solr.ltr.feature.SolrFeature",
    "params": {
      "q": "{!func}popularity"
        }
  },
  {
     "store" : "myCombinedFeaturesStore",
     "name":  "urlLength",
     "class": "org.apache.solr.ltr.feature.FieldLengthFeature",
     "params": {
        "field": "url"
        }
  },
```

```
  {
    "store" : "myCombinedFeaturesStore",
    "name":  "linkRank",
    "class": "org.apache.solr.ltr.feature.FieldValueFeature",
    "params": {
      "field": "linkrank"
        }
  }
]
```

Listing 3.18: An extract of the LTR feature definitions for Apache Solr. Features definitions include for recency, rawhits, field length and LinkRank.

The Apache Solr model definition is shown in Listing 3.19. Each of the defined features is allocated a weight value in this file. Note that the urlLength feature is allocated a negative value in the model. This value reflects the inverse relationship between ranking importance and the length of the URL. The complete definition files are available in the 'Solr-Files' folder on the GitHub repository.

```
$ cat myCombinedModel.json
{
  "class" : "org.apache.solr.ltr.model.LinearModel",
  "name" : "myCombinedModel",
  "store" : "myCombinedFeaturesStore",
  "features" : [
    { "name" : "originalScore" },
    { "name" : "documentRecency" },
    { "name" : "isAbout" },
    { "name" : "isContact" },
    { "name" : "rawHits" },
    { "name" : "urlLength" },
    { "name" : "linkRank" }
  ],
  "params" : {
    "weights" : {
      "originalScore" : 60.0,
      "documentRecency" : 10.0,
      "isAbout" : 20.0,
      "isContact" : 20.0,
      "rawHits" : 5.0,
      "urlLength" : -35.0,
      "linkRank" : 15.0
    }
  }
}
```

Listing 3.19: An extract of the LTR model definitions for Apache Solr. The model includes weights for each of the features.

The Solr managed schema API is used to upload these definitions to create a ranking model called `myCombinedModel` , as shown in Listing 3.20.

```
 $ curl -k -XPUT ''$SERVER':8983/solr/$CORPUS/schema/feature-store' \
--data-binary "@/home/solr/combined/myCombinedFeatures.json" \
-H 'Content-type:application/json'
```

```
 $ curl -k -XPUT ''$SERVER':8983/solr/$CORPUS/schema/model-store' \
--data-binary "@/home/solr/combiSolr-managededModel.json" \
-H 'Content-type:application/json'
```

Listing 3.20: Uploading the LTR defintions to the Apache Solr feature and model store.

The ranking model can be verified via the command line with the `rq` (re-rank query) parameter, which calls the LTR function (`!ltr`), as shown in Listing 3.21.

```
$ curl ''$SERVER':8983/solr/$CORPUS/select?q="*:economics"&\
rq={!ltr%20model=myCombinedModel%20reRankDocs=500\}\
&fl=url&rows=20&wt=csv'

/Economics/
/Economics/people/
/Economics/undergraduate/
/Economics/postgraduate/msc-economics/
/Economics/undergraduate/what/
/Economics/undergraduate/joint-honours/
/Economics/postgraduate/
/Economics/postgraduate/research-degrees/
/Economics/postgraduate/msc-econ-policy/
/Economics/SER/
```

Listing 3.21: Verification of LTR model functionality.

Finally, the `params.json` file is used to make the rq function above the default parameter applied to all queries. This is a helpful override as it means that no coding modifications are required at the web front-end.

3.8.2 Results (EXP-2: Search Results Ablation Study)

As described in Section §3.8, this experiment determines the relative weight for each of the LTR features in the `ENTRP-SRCH` dataset.

Figure 3.10 shows the breakdown of the contribution of each of the individual features in the Learning to Rank method.

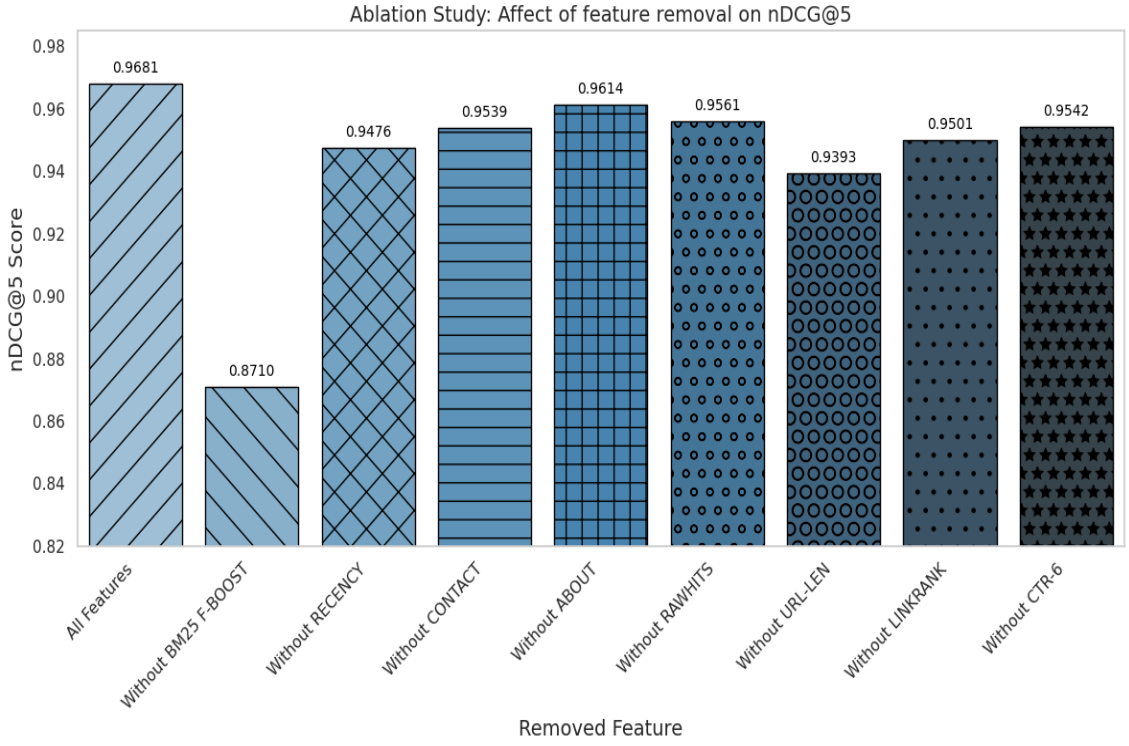


Figure 3.10: As more learning-to-rank features are added, the nDCG@5 score increases. The final nDCG score is 0.9681; the most significant contributing feature is BM25 with Field Boosting (i.e. the 'BM25 F-BOOST' combined feature, without which the nDCG@5 score drops to 0.8710).

As expected, the BM25 feature is the most significant contributor. Field boosting also performs well, but may have performed better had every document been split into fields (many documents types such as Microsoft Excel and PowerPoint have no discernible field structure, as described in Section §3.2.5). The performance contribution of the recency feature was smaller than expected. As described in Section §3.7.1, this may be because some ES document types, such as records from the corporate directory, do not present any timestamp data (or metadata).

3.9 EXP-3: Correlation using CTR and human judgements

An issue that the ES community has not addressed is whether click-through data, used as standard in WS, is also a good choice for organisations. This experiment will test this assumption and is designed to achieve OBJ-2.

The labelling of Q-D pairs has traditionally relied on human relevance judgements to create training data for ranking models. However, soliciting and gathering feedback in the form of relevance judgements is expensive, time-consuming, and may be subject to annotator bias.

Although most organisations are expected to deploy an ES service, few have relevance judges (annotators) lined up to create training data. This scarcity motivates this investigation into the application of implicit feedback methods designed to replace human judgements. If ES click-through data relevance feedback correlates

with human relevance judgements, judges could be dispensed with and abundant click-through data could be relied on exclusively.

This experiment was first published in Paper II "Enterprise Search: Learning to Rank with Click-through Data as a Surrogate for Human Relevance Judgements" and the associated code is available on Github.

- Hypothesis: The hypothesis is that the click-through rate has a statistically significant correlation with human relevance judgement scores in ES.
- Baseline: The human relevance judgements, which occupy column 1 of the `ENTRP-SRCH` dataset, is the baseline (as shown in Fig 3.7). Corresponding click-through scores (column 8) are to be tested for correlation with this baseline. Subsequently, the ranking performance is compared, using the two alternate 'ground truths'.
  - Human relevance judgements is the baseline ground truth to be tested and is the one traditionally associated with learning to rank.
  - The click-through rate is used as an alternative (surrogate) ground truth. The LTR model's custom features, as listed in section 3.4.4, are likely better at matching human relevance judgements over CTR (since the features were initially engineered to match the requirements of human relevance judgements).
- Mechanism: As described in Section §3.3.5, a basic click model is deployed on The University's ES service to gather click-through data. The collection period is three-months and there are approximately 7,000 queries per day. The correlation coefficient (which is a measure of the strength of the relationship between CTR and ordinal human relevance judgements) is calculated. Since the human relevance variable is ordinal in nature (a Likert scale) with a non-normal distribution of judgements, Spearman's rho is the appropriate correlation metric. The performance of the subsequent LTR ranking model trained on the CTR data is then compared against a model trained on human annotations (i.e. using two alternative 'ground truths' as input for the LTR method). As is typical in the field of search result ranking, the metric for ranking performance is nDCG.
- Sufficiency of Data: While the `ENTRP-SRCH` dataset contains 2544 human judgements, the CTR-based subset contains only 375 datapoints. This smaller number does not come from the full query log (which contains thousands of daily queries), but from the intersection of two requirements: each CTR Q–D pair must correspond to (i) a query prefix for which a click was recorded in the Solr logs, and (ii) a document that also appears in the `ENTRP-SRCH` judged dataset. Only those pairs that satisfy both conditions are retained, which explains why the CTR subset is considerably smaller than the raw query volume. Moreover, since end-users are not inclined to scroll to 'page 2' of the search results, this limits the number of CTR records. A statistical power test is conducted to confirm the CTR data is sufficient. Statistical power is governed by the sample size, effect size, significance level, and the variability in the data. The power calculation suggests that only 64 CTR Q-D pairs would

have been needed to achieve a desired power of 0.8 and still achieve statistical significance.

- Domain Knowledge: Domain knowledge plays an essential role in analysing any discrepancies in correlation. For example, one of the query terms in our training dataset is 'english'. The annotator for the English query is employed by the Department of English and, therefore, assigned preferential judgements based on his department's perspective. The end-users, many of whom are prospective students from non-English speaking countries, are less interested in literature and instead wish to ascertain the minimum English language requirements for entry to the student register.

- Limits: As discussed in Section §3.3.5, CTR scores ideally have sophisticated corrections for problems like unbounded positive feedback, a decay function for older documents, and log normalisation to mitigate outliers etc. In this experiment, these corrections are not applied and CTR is used in its simple form.

- Expected Results: Studies in the field of WS and ecommerce have shown that click-through data can be harnessed as a surrogate for relevance judgements [19, 98, 159, 221, 222]. Similarly, in ES, it is expected that a degree of positive correlation will be discovered between the two ground truths. It is also expected that the LTR model's custom features, as listed in section 3.4.4, are better at matching explicit (human) rather than implicit (CTR) feedback, since the features were initially engineered to match the requirements of human relevance judgements.

The techniques to implement CTR collection and relevance correlation are described in sections §3.7.1 and §3.9.1 respectively. The results of EXP-3 are presented in Section §3.9.2.

3.9.1 Implementation (EXP-3: Correlation using CTR and human judgements)

Relevance Correlation Figure 3.11 is a strip plot that graphically presents a degree of correlation between the human relevance judgements and the CTR. It is obvious from the plot that the documents labelled with a higher relevance score by expert human annotators tend to be more clicked by end users. The calculation of the correlation score is scripted in a Python notebook called es-ltr-explicit-implicit-feedback.ipynb, which can be viewed in the GitHub repository. The correlation is presented in the results section of EXP-3 (below).

3.9.2 Results (EXP-3: Correlation using CTR and human judgements)

As described in Section §3.9, the first step of this experiment is to determine whether the click-through rate has a statistically significant correlation with human relevance judgement scores in ES. The second step involves a comparison of the ranking of performance using the two alternate 'ground truths'.

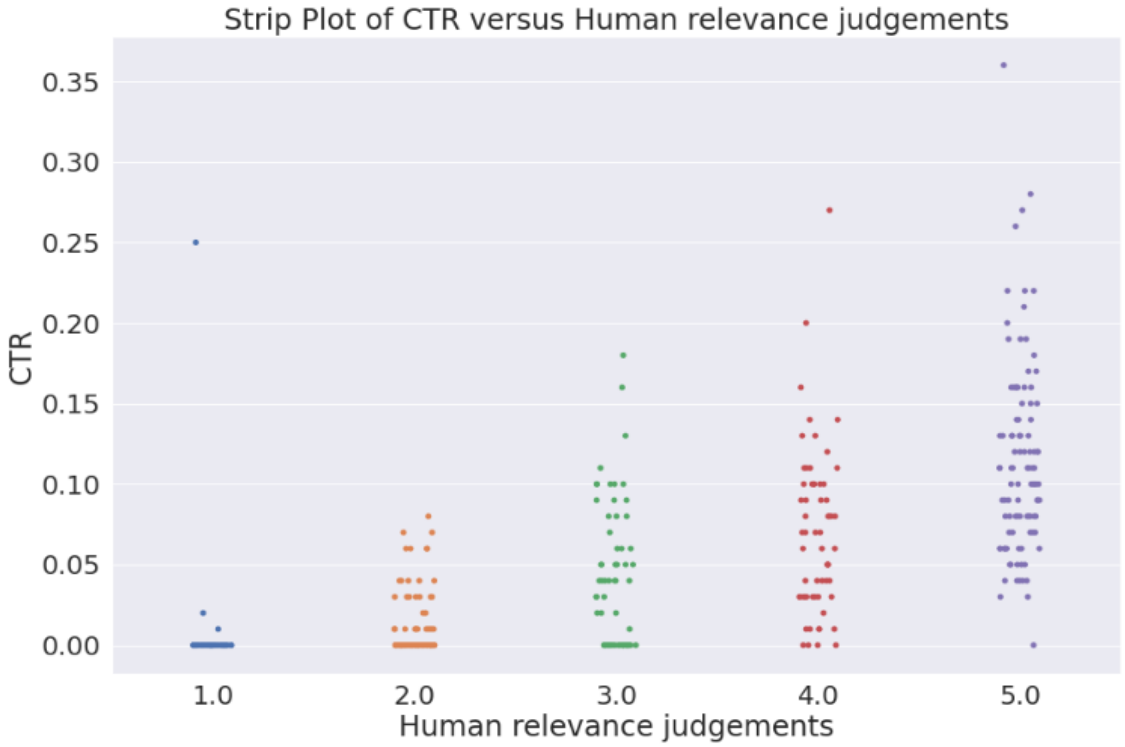


Figure 3.11: A strip scatter plot showing correlation points between click-through rate (CTR) on the y-axis and human relevance judgements on the x-axis. Those documents that received a higher relevance judgement tend to have recorded more click-through activity.

Correlation and Outlier Analysis

Using the CTR/human judgements data that was used to create the strip plot in Figure 3.11, a correlation score of $\rho = 0.704$ was calculated. Most statisticians consider a score of 0.7 or above as a 'strong correlation' [223]. A significance test using the paired t-test gives a p-value of less than 0.01, proving that the Spearman correlation is statistically significant.

Figure 3.11 shows a slight divergence from a linear correlation plot (i.e., a straight line), insofar as there seems to be an excess of points 'under the line'. For example, where the human relevance judgement score is 4 ('relevant', but not 'highly relevant'), many points have a low CTR rate. Analysis of the respective documents suggests that this non-linearity may be caused by few end users navigating to pages 2 or 3 of the search results (the ranking model displays a preponderance of 'highly relevant' documents on Page 1). This is in line with the 'position bias' confounding problem outlined in Section §2.4.8 (i.e. reports that less than 6% of end users navigate beyond Page 1 of the Search Engine Results Page).

End users are not the only feedback party subject to bias. 'Organisational bias' occurs when factors such as strategic focus and team organisation influence data selection to a point where selection is no longer based on individual merit. Some instances of disagreement over pages judged to be 'irrelevant' by expert annotators

are seen but have been awarded a high click-through rate by end users. For example, the homepage of a VIP / celebrity individual in the organisation has been annotated as 'irrelevant' (for the given query). This judgement dampens favouritism and diminishes the democratic preferences of end users. In this case, the experts are subject to organisational bias, where they overlook an individual's popularity in favour of an institutional outlook.

A further example of a pronounced contradiction between annotators and end-users was detected. One of the query terms in our training dataset is 'English'. As mentioned in Section §3.9, the annotator for this query is employed by The University's Department of English and, therefore, issues preferential judgements based on his department's perspective. The end-users, many of whom are prospective students from non-English-speaking countries, are less interested in literature and instead want to ascertain minimum English language requirements for entry to the student register. This correlation disparity/outlier for the 'english' query highlights that even gold standard annotation can be subject to biased annotation or inconsistency if tokens are ambiguous [224].

Comparing the fit of features

If the ground truths in the LTR model are alternated, such that either CTR scores or human relevance scores are used to train the data, there will be a change in how well the features combine to 'fit'. The nDCG values for both feedback methods are shown in Table 3.7 and graphically represented in Figure 3.12.

Table 3.7: Comparison of ranking performance (nDCG) for human relevance judgements (explicit annotator feedback) versus query preferences extracted from click-through (implicit feedback).

Cutoff	Relevance Judgements	Click-through
ndcg@1	**1**	0.997
ndcg@3	**0.987**	0.970
ndcg@5	0.941	**0.964**
ndcg@10	0.787	**0.963**
ndcg@20	0.567	**0.853**
ndcg@100	**0.335**	0.271
ndcg@200	**0.313**	0.061

The results show that the LTR model's custom features (described in section 3.8) are approximately an equal fit at matching explicit and implicit feedback. For example, the nDCG@3 ranking performance using relevance judgements is just 1.6% higher than when click-through data is used. For nDCG@5, nDCG@10 and nDCG@20, using CTR as ground truth achieves higher scores than human judgements.

The results for nDCG@{100,200} can be disregarded. This is because there are almost no CTR feedback data points in this range. CTR feedback in this range would involve an end-user clicking the 'next page' ten or twenty times respectively.

Overall, neither feedback mechanism can be said to produce a clear winner. This

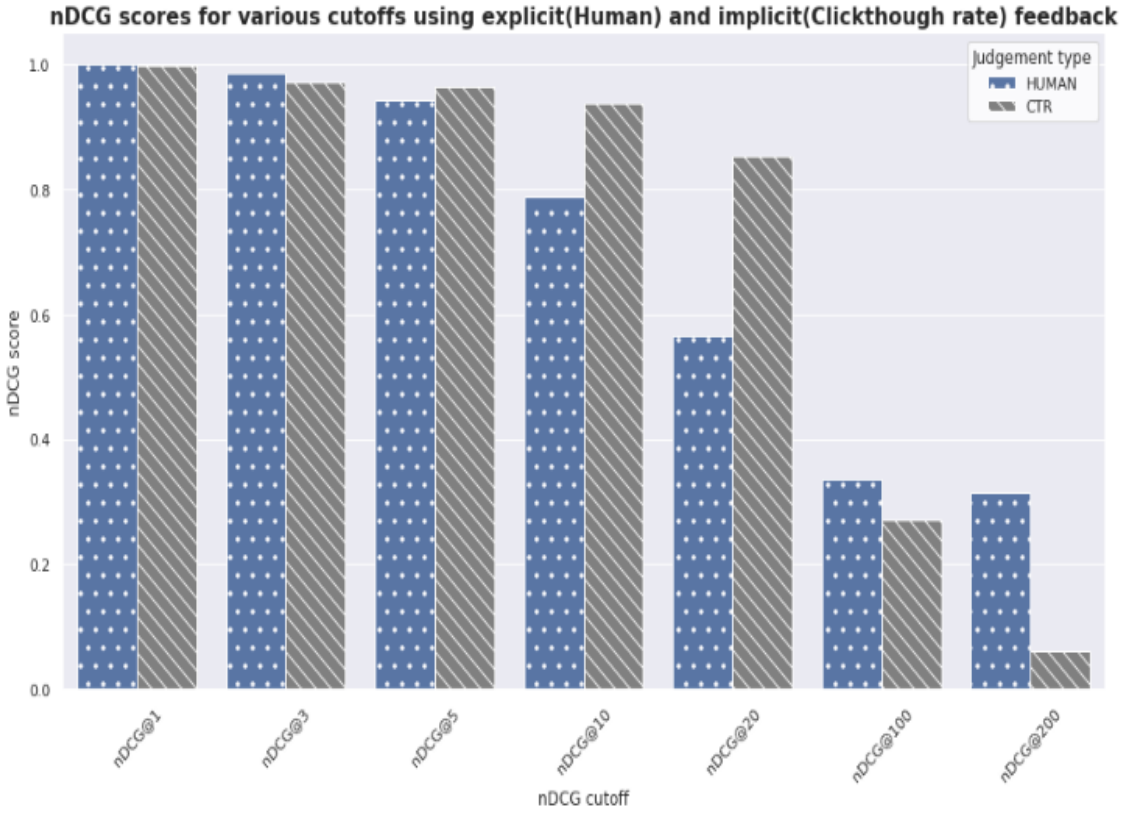


Figure 3.12: A bar chart showing the ranking performance differences between explicit and implicit feedback for various nDCG cutoffs.

is unexpected, as it was anticipated that human relevance judgments would yield superior results. This expectation was based on the fact that the features were originally designed to align with human relevance judgment ground truth.

In Chapter 5, the ES implementation trade-offs between human relevance judgements and implicit feedback are discussed. This includes an analysis of both end-user and expert annotator potential preferences and biases.

3.10 EXP-4: Inter-Annotator Agreement

As described in Section §2.4.4, the multi-step pipeline for finding high-agreement Amazon Mechanical Turk raters, described by Zhang [157], cannot be used in the context of ES. This is because ES content comprises highly technical blueprints, confidential documents, network file shares, etc., much of which is located behind an organisation's firewall.

During the `ENTRP_SRCH` dataset construction process, a significant portion of annotated documents were evaluated by a single rater for each query. Across the 20 query topics included in the dataset, a total of 2,544 documents were assessed by only 15 raters.

The raters are all microsite editors, whose role includes publishing content on behalf of colleagues in their department. They are necessarily familiar with published

content (preventing replication and ensuring consistent messaging).

It is assumed, therefore, that these microsite editors are well placed to 'judge' the ranking order of their documents for a particular query. This experiment tests the agreement between two independent raters (who are not part of the original group of fifteen, and who are rating new Q-D pair that are not part of the ENTRP-SRCH dataset) to verify this approach's validity.

Inter-annotator Agreement (IAA, Section §2.2.20) examines agreement among independent observers who rate, code, or assess the same phenomenon. If consensus in the ratings of a target is low, it suggests that the decision to rely on a single rater was incorrect. Agreement metrics comes with their own value interpretation guidelines. With Krippendorff's coefficient, reliabilities with $\alpha \geq 0.667$ are "acceptable for tentative conclusions" [117] whereas Cohen's κ should be ≥ 0.8 for "substantial agreement" [120, 121].

- Hypothesis: The hypothesis is that there is 'acceptable' agreement (as defined above) of Q-D pair annotations between two independent microsite editor raters, as measured by Percentage Agreement (PA), Cohen's κ, and Krippendorff's α.

- Baseline: As with the original fifteen raters (i.e., that were used to create the `ENTRP-SRCH` dataset), two new independent raters were chosen based on their familiarity and experience with the query topic. Agreement is then measured between the two sets of Q-D annotations.

- Mechanism: The two new raters selected are microsite co-editors for the same subsite. They were given guidance identical to those of the original fifteen (Section §3.4.3). To ensure independence between the raters, neither rater was shown the Q-D annotations of the other.

- Sufficiency of Data: The κ and α metrics are presented with a Confidence Interval (CI). The CI is computed using the 'bootstrapping' statistical procedure. Bootstrapping involves re-sampling a single dataset to create many simulated samples. This process allows for the calculation of standard errors, confidence intervals, and hypothesis testing [225, 226]. If the standard error is big, the final scores could span a range of interpretations that could indicate anything from good to poor agreement [122]. The number of Q-D pairs must therefore be large enough to produce a small standard error. The annotations range ([1-5]) can be considered as interval data for computation of Krippendorf's α. The experiment compares the ranking of 565 documents, corresponding to the full set of items returned for one specific keyword query within the microsite's index.

- Domain Knowledge: In this case, the experiment is designed to examine how domain knowledge is shared between two independent raters, both of whom have an expert level of knowledge of topics within their own microsite.

- Limits: This experiment measures agreement between two raters for a single microsite. However, it is likely that agreement between two raters for one

microsite would differ from that for another microsite. For example, the IAA score between two expert judges annotating Q-D pairs for 'religion' would likely be different from two other experts annotating Q-D pairs for 'mathematics'.

- Expected Results: Observations from the field of *Web Search* suggest that humans find it hard to agree in relevance ranking tasks [6]. In the case of *Enterprise Search*, annotators are likely to come from within the organisation. The subject matter of the documents is specific to the enterprise, and therefore, only individuals with domain knowledge are well placed to volunteer relevance judgements [10]. Hence, a higher level of consensus between the (like-minded, but independent) experts, such as The University's microsite co-editors is expected.

The implementation details for EXP-4 are described in 3.10.1 (below) and the results are presented in Section §3.10.2.

3.10.1 Implementation (EXP-4: Inter-Annotator Agreement)

Inter-Annotator Agreement As described in Section §3.10, the IAA experiment (EXP-4) tests the agreement between two microsite editors. Each is asked to independently annotate 565 Q-D pairs. Each pair is allocated a [1-5] rating. As with the judges used during the creation of the `ENTRP-SRCH` dataset, they were asked to limit the number of '5's (highly relevant) awarded. This paucity is because most end users will focus on the top of the ranked list of the first search results page. The University's search logs suggest that less than 2% of users navigate to page 2 or deeper.

The three IAA agreement metrics (discussed in Section §2.2.20) are employed in the experiment:

- **Percentages of Agreement (PA)**. This metric is generally not an adequate measure of IAA since it does not correct for agreements that would be expected by chance and, therefore, would overestimate the level of agreement [117].
- **Cohen's** κ. This statistic is regularly used when dealing with nominal (unordered) data [118, 119]. It is also of limited use, as the annotator ratings are inherently ordered.
- **Krippendorff's** α. This statistic has the advantage that it works for various data types (nominal, ordinal, interval, and ratio) and accounts for chance agreement, making it more versatile and robust compared to Cohen's κ [117]. An alpha value of 1 represents a perfect agreement, while 0 means no agreement beyond chance.

Since the PA metric is an empirical measure (rather than a statistical estimate), a calculation of the confidence interval (CI) is not required. The κ and α metrics do require a CI calculation, not least because the standard error may be so big that the final score spans a range of interpretations that could indicate anything from good to poor agreement [122]. The confidence interval of the mean of the records is

calculated using the `scipy.stats.bootstrap` Python library configured with 1,000 simulated samples.

The meaning of values in a Likert scale must be considered before choosing a statistical coefficient.

- The *Nominal Level* coefficients treat all disagreement equally (e.g., *1* is as different from *5* as it is from *2*).
- The *Ordinal Level* treats neighbouring disagreements less severely than disagreements spanning several ranks. For example, if the first annotator awarded *5* and the second *4*, this would be a higher agreement than if the first gave *5* and the second gave *3*.
- The *Interval Level* is similar to the ordinal level, but with the additional assumption that all points on the scale are equidistant, i.e. the distance between *1* and *2* is equal to the distance between *4* and *5*.

The calculation of the coefficients (and their corresponding confidence intervals) for several data types (nominal, ordinal and interval levels) is scripted in a Python notebook called `inter-annotator-agreement-in-enterprise-search.ipynb` which can be viewed in the GitHub repository. The level of agreement between the expert annotators is presented in EXP-4 results (below).

3.10.2 Results (EXP-4: Inter-Annotator Agreement)

During the annotator selection process (Section §3.4.3), a key assumption was that The University's microsite editors were well placed to 'judge' the ranking order of their documents for a particular query. For a given query, many documents were therefore judged by a single rater. As described in Section §3.10, EXP-4 is designed to test the agreement between two independent raters and verify that this assumption was valid.

Table 3.8 records the results of the IAA experiment. The experiment compared the Likert scale ranking [1-5] by two independent judges over 565 documents.

The PA score of 84.25% is very high and suggests a high level of agreement, even if chance agreement is discounted. The number of documents in our experiment (565 documents) works to minimise the contribution of chance agreement (this size provides a sufficiently large sample to yield a stable estimate with reduced variance). The high PA score may reflect that the raters selected are experts for their query topic, and little guessing is applied to annotations.

Regardless of how the spacing of measurement levels in the Likert survey is interpreted, the calculated ordinal and interval α scores (0.703 and 0.730, respectively) are both greater than Krippendorff's minimum requirement of 0.67, above which 'tentative conclusions' can be drawn [117].

Table 3.8: Inter-Annotator Agreement results for nominal, ordinal and interval data types.

Coefficient Name	Score
Nominal Level	
Percent Agreement	84.25%
Ordinal Level	
Cohen's κ	0.626 ± 0.061
Krippendorff's α	0.703 ± 0.061
Interval Level	
Cohen's κ	0.707 ± 0.087
Krippendorff's α	0.730 ± 0.087

Similarly, the ordinal and interval weighted Cohen's κ scores (0.626 and 0.707 respectively) fall into the interpretation category of 'substantial agreement' (which ranges from 0.61 to 0.80) [116].

Thus, the agreement recorded by the α and κ scores suggest that it can be *tentatively concluded* (in line with Krippendorff's guidance and phraseology) that the hypothesis is supported (i.e., the single rater approach adopted for Q-D annotation per topic does not invalidate the `ENTRP-SRCH` dataset).

3.11 EXP-5: JARGES Ranking and Recall with LM-generated Synonyms

Newcomers to an organisation often struggle with unfamiliar internal vocabulary, which can affect their ability to retrieve relevant information. Enterprise Search (ES) systems frequently underperform when queries contain jargon or terminology that is specific to the organisation. This experiment introduces 'JARGES', a novel feature for detecting and decoding jargon for ES. It is designed to enhance a ranking model combining Learning to Rank (LTR) and transformer-based synonym expansion.

This experiment was first published in Paper IV "JARGES: Detecting and Decoding Jargon for Enterprise Search." [227] and the associated code is available on GitHub.[8].

- Hypothesis: The hypothesis is that the addition of the JARGES feature to a baseline ranking model results in a statistically significant improvement of ranking performance. To test this hypothesis, a quantitative evaluation of the performance of an LTR ranking model with and without JARGES is conducted. Additionally, a qualitative analysis of the decoded jargon terms via their contextual synonyms is conducted.

- Baseline: Our baseline ES LTR ranking model is generated using eight features as part of the `ENTRP-SRCH` dataset. These features include BM25, recency, doc-

[8] `https://github.com/ColinDaly75/JARGES`

ument hits, linkrank and click-through rate and are described fully in Section §3.2.

- Mechanism: A ninth feature, representing JARGES, is added to the dataset to test its impact. Table 3.10 describes the hyperparameters used for the LTR calculation. A/B test comparing the ranking performance for two models generated and evaluated using the `ENTRP-SRCH` dataset. Before commencing the A/B test, an A/A test is performed. Also known as a null test, the A/A test is used to establish trust in our experimental platform. The null test involves splitting the search requests into two pools, as in a regular A/B test, but where B is identical to A. If the scripts to record search requests and compute metrics from the logfiles are functioning consistently, a t-test would be expected to prove that any difference in nDCG results is not statistically significant [228]. The first model (A) incorporated eight features from our base model, while the second (B) additionally included the JARGES feature. In addition to the A/B test, a 'leave-one-out' ablation study is conducted to evaluate the contribution of individual features. This method systematically removes one feature at a time to measure its impact on the overall model performance.

- Sufficiency of Data: The LTR-formatted `ENTRP-SRCH` dataset (2,544 human-annotated Q-D pairs, for 20 of The University's most queried topics) is used to train, test and evaluate the ranking model.

- Domain Knowledge: A popular PDF document in the corpus is entitled 'Jargon Buster' and describes 126 commonly used jargon terms within the institution. The file acts as an independent reference for assessing both the precision and the comprehensiveness of the decoded jargon terms (i.e. synonyms)

- Limits: A potential limitation of the JARGES algorithm is its reliance on TF-IDF score divergence between common terms and corpus-specific terms. A situation can occur when the divergence is too small for detection, even where a term is obviously jargon. Resolving this issue would require an alternative algorithm that differentiates terms based on semantic meaning instead of TF-IDF scoring.

- Expected Results: JARGES is designed to enhance a ranking model combining Learning to Rank (LTR) and transformer-based synonym expansion. The ranking model is evaluated using the `ENTRP-SRCH` dataset. A positive outcome would be a statistically significant increase in nDCG scoring when JARGES is applied.

The techniques to implement the JARGES algorithm are described in Section §3.11.1 , and the results of EXP-5 are presented in Section §3.11.2.

The experiments that have been described in this section (EXP-1 to EXP-5) are designed address the first two research questions of this thesis (RQ-1 and RQ-2). Chapter 4 will describe experiments that are designed primarily to address RQ-3.

3.11.1 Implementation (EXP-5: JARGES Ranking and Recall with LM-generated Synonyms)

This section outlines the methodology employed to create the JARGES feature, integration with an Apache Solr ES service, and the subsequent evaluation of the LTR ranking model. The ranking model is trained and evaluated using the `ENTRP-SRCH` dataset, and incorporates offline A/B testing to assess ranking performance.

JARGES. The JARGES algorithm is designed to detect and decode jargon. It processes three input files and generates two output lists, as depicted in Figure 3.13. These outputs are tailored to enhance distinct components of Enterprise Search (ES):

- A ranked list of the organisation's **detected** jargon terms, ordered by their 'jargony' score. This list serves as a ranking function for integration into an LTR model.
- A list of synonyms for each **decoded** jargon term from within the organisation. This output enables query expansion in the search engine, thereby improving recall.

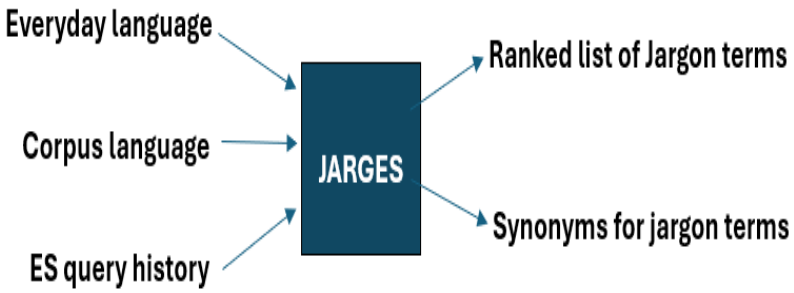


Figure 3.13: The three inputs for the JARGES algorithm, which outputs two lists (for ranking and recall respectively).

In this study, the `ENTRP-SRCH` LTR dataset is used, which includes 2544 human-annotated Q-D pairs of twenty most frequent queries of a large third-level education institution. One popular PDF document in the corpus is entitled 'jargon buster' and describes 126 commonly used jargon terms within the institution. The full code for the JARGES ranking and recall feature has been published to GitHub[9].

Detection. The JARGES feature is centred on the relative unusualness of words, such as those used in organisational jargon/terminology. Figure 3.14 is a demonstration of how JARGES works when applied to a real sentence in the organisation's corpus. Detection works by harnessing TF-IDF (Term Frequency, Inverse Document Frequency). TF-IDF is a core NLP statistical technique and was chosen as it is better at detecting semantic importance than raw word counts [229]. TF-IDF is used to identify important (single and multi-word) terms in an enterprise corpus and

[9] `https://github.com/colindaly75/JARGES`

compare their scores against general English frequencies. A substantial difference between the two scores indicates that the term may be jargon.

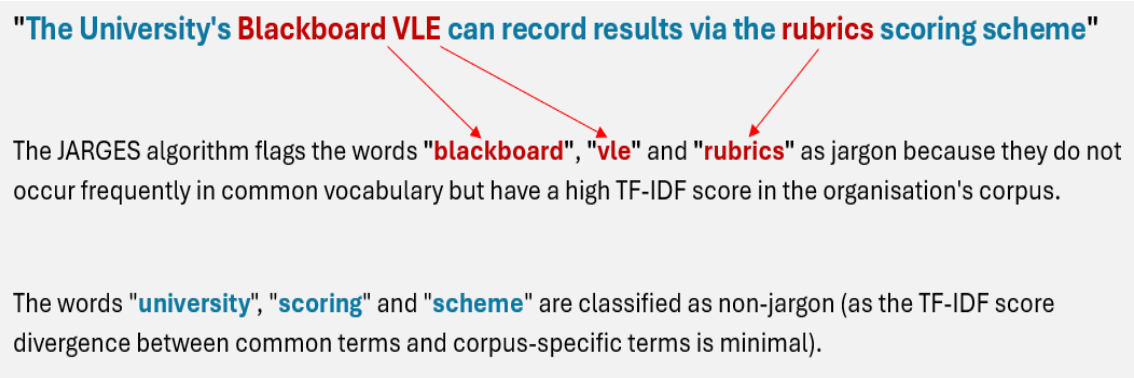


Figure 3.14: An example of jargon / non-jargon classification by JARGES when applied to a sentence in the corpus.

ES Corpus Vectorisation. The TfidfVectorizer Python library [230] is used to convert a collection of raw text document into a matrix of TF-IDF features. It splits and tokenises the document's words into n-grams and then computes the TF-IDF score for each n-gram in a document. The text is vectorised into a numerical matrix where each row represents a document and each column represents a word's TF-IDF score.

Detecting Divergence. The challenge is to detect enterprise specific words, phrases, expressions, and idioms that diverge from those universally familiar or understood outside of the organisation. This divergence is measured by computing the absolute difference in TF-IDF scores for the terms.

SpaCy is an open source Python NLP library for fast text processing that includes a small, pre-trained statistical model for English called 'en_core_web_sm'. The model is trained on web and news text data. SpaCy has been shown to work well for standardised English (e.g., formal writing, news, technical documents). For this reason, it is employed as our reference source of TF-IDF for common English.

Jargony. Moreover, the magnitude of the divergence between the common score and the corpus score represents a measure of 'jargony' (i.e. how much jargon is likely imbued in the term). For example, TfidfVectorizer computes the TF-IDF score of the word 'blackboard' in common English (i.e. en_core_web_sm) is X, whereas the score recorded from the organisation corpus is Y. In the event where a term does not occur in common English, but occurs frequently in the organisation's corpus, this too is treated as jargon (e.g., the 'SCSS' term shown in Table 3.9).

Refining the List. The size of the sorted list, which includes both the jargon term and its corresponding jargony score, is determined by the min_divergence parameter. In the case of our corpus, this was set to be 15%, meaning that any jargon term with a TF-IDF divergence smaller that this is discarded. This is the same as the

Table 3.9: JARGES classification. The terms in purple font were successfully classified as jargon.

Jargon Term	Wikipedia-api Definition	Meaning within Organisation
Blackboard	A reusable writing surface on which text or drawings are made with sticks of calcium sulphate or calcium carbonate, known, when used for this purpose, as chalk. Blackboards were originally made of smooth, thin sheets of black or dark grey slate stone.	Blackboard is an abbreviation of 'Blackboard Learn', which is the organisation's Virtual Learning Environment (VLE). Students can use Blackboard to access lecture notes, online assignments and other activities.
Atrium	One of the two upper chambers in the heart that receives blood from the circulatory system. The blood in the atria is pumped into the heart ventricles through the atrioventricular mitral and tricuspid heart valves.	Event space and meeting place in the Dining Hall building which is often used by student societies.
SCSS	No definition.	Initialism for 'School of Computer Science and Statiscics'.
Botany Bay	An open oceanic embayment, located in Sydney, New South Wales, Australia	A residential square located behind the Graduates' Memorial Building.
Hilary	Hillary Diane Rodham Clinton (née Rodham;[a] born October 26, 1947) is an American politician, lawyer and diplomat.	Hilary is the second of The University's three annual semesters, running from January to April.

threshold adopted in Belfathi's study [201]. Moreover, on visual inspection, the terms below this threshold did not intuitively appear jargon-like. This list is further refined by filtering the jargon terms with those query terms previously submitted to the search engine (this is the ES query history input shown in Figure 3.13). Two years of ES log data included 62,044 unique query terms. The resultant list (named refined-jargon-list.txt) consists of overlapping terms that are a) likely to be jargon and b) have a history of being queried. For our corpus, the refined list consisted of 547 records.

Blind Spot. A key limitation of the JARGES algorithm is its reliance on TF-IDF score divergence between common terms and corpus specific terms. A situation can occur when the divergence is too small for detection, even where a term is clearly jargon. For instance, as illustrated in the last row of Table 3.9, the jargon term 'Hilary'. The top Wikipedia-api definition presented changes the spelling from Hilary to Hiliary (i.e. Clinton). The term Hilary exhibits comparable prominence in both sources, thereby preventing its identification as jargon. Resolving this issue

would require an alternative algorithm that differentiates terms based on semantic meaning instead of TF-IDF scoring.

Decoding. In the decoding stage, the RoBERTa language model is adapted for synonym generation by fine tuning it on the organisation's corpus. RoBERTa is used to predict plausible alternatives for the previously detected jargon terms. Synonyms can enhance semantic search by enabling the system to recognise and retrieve conceptually related terms beyond exact keyword matches.

The synonyms are output to a text file in a format that can be understood by the search engine. A practical consideration for Apache Solr is that the more specific terms should be listed in the file before general ones [14]. The synonyms are therefore ordered by their jargony score (computed in the detection phase). To further prioritise specific over more general synonyms, the Solr `SynonymGraphFilterFactory` is used because it handles multi-term and large synonym sets more effectively than the default `SynonymFilterFactory`. At query time, when a user enters a generic term, the filter injects the corresponding jargon variants as additional clauses, while the original (generic) term retains the highest weight. In other words, the aim is to add jargon terms to non-jargon queries (i.e., generic $\rightarrow$ jargon), not to expand jargon into generic vocabulary. This preserves the user's intent, improves recall over organisation-specific content, and avoids swamping results with overly broad matches. The synonyms of the jargon query terms should align with the description in the organisation's 'jargon buster' PDF document.

JARGES LTR Ranking Model. Our ES LTR ranking model is generated using eight features as part of the `ENTRP-SRCH` dataset (i.e., the same feature set used in EXP-1). These features include BM25, recency, document hits, linkrank and click-though rate and are described fully in [231]. A ninth feature, representing JARGES is then added to the dataset to test its impact. Table 3.10 describes the hyper-parameters and validation used for the LTR calculation.

Table 3.10: Hyperparameters used in the Learning to Rank experiment.

Parameter	Value
Dataset	ENTRP-SRCH
Algorithm	GradientBoosting
n_estimators	100
max_depth	3
learning_rate	0.1
rank	nDCG
random_state	42
n_splits	5
num_features	8 (9 incl. JARGES)
Validation method	Cross-Validation (CV)
CV Folds	5

3.11.2 Results (EXP-5: JARGES Ranking and Recall with LM-generated Synonyms)

As described in Section §3.11, JARGES is a novel feature for detecting and decoding jargon for ES. It is designed to enhance a ranking model combining LTR with LM and transformer-based synonym expansion.

An A/B test was conducted to compare the ranking performance for two models generated and evaluated using the `ENTRP-SRCH` dataset. The first model incorporated eight features from our base model, while the second additionally included the JARGES feature. The ranking scores for both models are displayed in Table 3.11. The results indicate no significant percentage change between the two.

Table 3.11: A/B test results for ranking models with the percentage change in nDCG score after implementation of the JARGES feature.

Feature	**nDCG@10**
Base LTR model	0.9646 ± 0.001
With JARGES	0.9639 ± 0.001
Percentage change	0.0007%

A 'leave-one-out' ablation study to evaluate the contribution of individual features was also conducted. This method systematically removes one feature at a time to measure its impact on the overall model performance. As shown in Figure 3.15, removing the JARGES feature from the baseline model has a negligible effect on ranking performance.

Both experiments showed that there is no statistically significant improvement for nDCG@10 at the 5% level ($p > 0.05$). This is attributed to the `ENTRP-SRCH` dataset's limited query diversity (many of the queries are more topical rather than exploratory in nature, as described in Section §3.4.2) and scarcity of jargon rich Query-Document pairs, which deflated the performance score of the JARGES feature.

The 'Jargon Buster' PDF file acts as a independent reference for assessing both the precision and the comprehensiveness of the decoded jargon terms (i.e. synonyms). Of the 126 jargon terms defined in the PDF, 53 also appear in our generated list. Furthermore, a cursory inspection of the synonyms generated for these terms appears to reflect the correct semantic context. Some examples are shown in Table 3.12.

3.12 Validation

The experimental designs outlined in Section §3.6 include several intrinsic validation checks for robustness, stability and generalisability. For example, EXP-3 and EXP-4 work to assure the validity of the `ENTRP-SRCH` dataset. In this section, several additional diverse validation checks are described.

3.12.1 Model

Overfitting can occur if a statistical model fits *too closely* against its training data (including noise and irrelevant information). The model learns the noise, generates

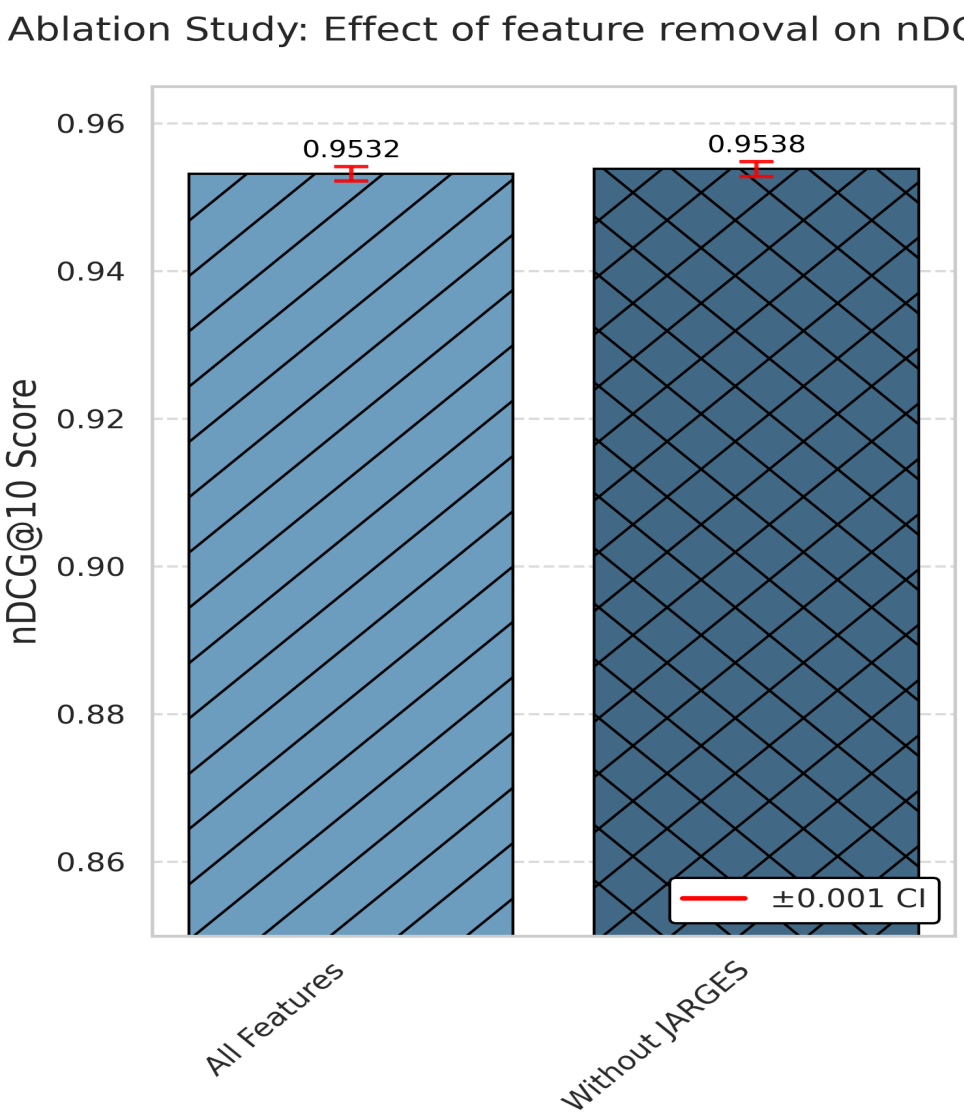


Figure 3.15: LTR ranking model performance with and without JARGES.

spurious correlations, and fits the training set too closely. The model is said to be 'overfitted' if it is unable to generalise well to new data. The best method to detect overfit models is by testing the machine learning models on more data and with a comprehensive representation of possible input data values and types. To facilitate this, several validation methods are employed and compared.

- Basic Validation
- Holdout Validation
- Cross Validation
- Generalisation using new 'real world' data

Basic Validation The validation method generates a numerical estimate of the difference in predicted and original responses (i.e. training error). A limitation of this method is that it only approximates how well our model does on data used

Table 3.12: Some examples of correctly decoded jargon terms via their contextual synonyms generated by RoBERTa.

Jargon Term	Synonyms	Meaning per 'Jargon Buster'
pav	pavilion, pavilion bar	Formally, the Pavilion Bar. The University's student bar, managed by the Sport Union, located at the eastern end of College Park.
tangent	climate entrepreneurship, social data science	The ideas workspace, providing courses and events centred on business and innovation.
forum	cafe, restaurant	College-operated eatery located in the Business School. Here you'll find hot and cold lunch offerings, barista coffee, and - often - pop-ups run by local businesses.

to train it. Validation does not indicate how well the model will generalise to an independent/ unseen data set. The typical 80–20 rule of splitting data into training and validation can be vulnerable to accidentally ending up in a perfect split that misleadingly boosts model accuracy. A high error rate during the model accuracy test may indicate overfitting.

Holdout method The shortcomings of Basic Validation can be overcome by removing a part of the training data and using it to get predictions from the model trained on the rest of the data. The error estimation then tells how our model is doing on 'unseen data'. Although this method does not take any overhead to compute and is better than basic validation, it still suffers from issues of high variance. It is unsure which data points will end up in the validation set, and the result might be entirely different for different sets.

Cross Validation Since our `ENTRP-SRCH` ES dataset is relatively small (in comparison to WS datasets), there is a limit to the size of the holdout partition. If the holdout data is sizeable, it may reduce the effectiveness of the training and validation sets. The CV method allows us to repeatedly and randomly split the dataset into training, validation, and holdout partitions. The CV method involves partitioning the dataset into K equally sized subsets or sample sets called 'folds'. The training process consists of a series of iterations. During each iteration, the steps are:

- Keep one subset as the validation data and train the machine learning model on the remaining K-1 subsets.
- Observe how the model performs on the validation sample.
- Score model performance based on output data quality.

After the last iteration, the model is finally retrained on the complete dataset using the optimisation learned from CV (hence ensuring efficient use of the limited Q-D pairs).

Generalisation using new 'real world' data While validation asks the question "am I building the model correctly?" during training , generalisation asks "does my model work in the real world?" during the post-deployment phase. As the validation metrics (i.e., CV scores) are high, the model is likely to generalise well. Generalisation using new/unseen data, and the related concept of domain transfer are discussed in Chapter 5.

3.12.2 Size of Dataset

There are many factors that determine the minimum size of a dataset, not least of which is the purpose for which the dataset was designed. The more complex the underlying relationship between the variables, the more data would be needed to estimate it. Additionally, the more noise corrupting the data, the more data one is likely to need to average out the effects of the noise. The greater the size of the dataset, the less likely overfitting (see Section § 3.12.1) will be a problem for the subsequent model generated using the dataset.

Learning curve graphs can be employed to estimate the required size of a dataset. As the dataset size increases, the performance initially increases until the model saturates. The minimum sufficient dataset size occurs when adding more data does not significantly improve performance. When the model performance saturates, it can be assumed that the general population and the training dataset now have similar underlying distributions. Thus, the cost of further computation and storage has diminishing rewards.

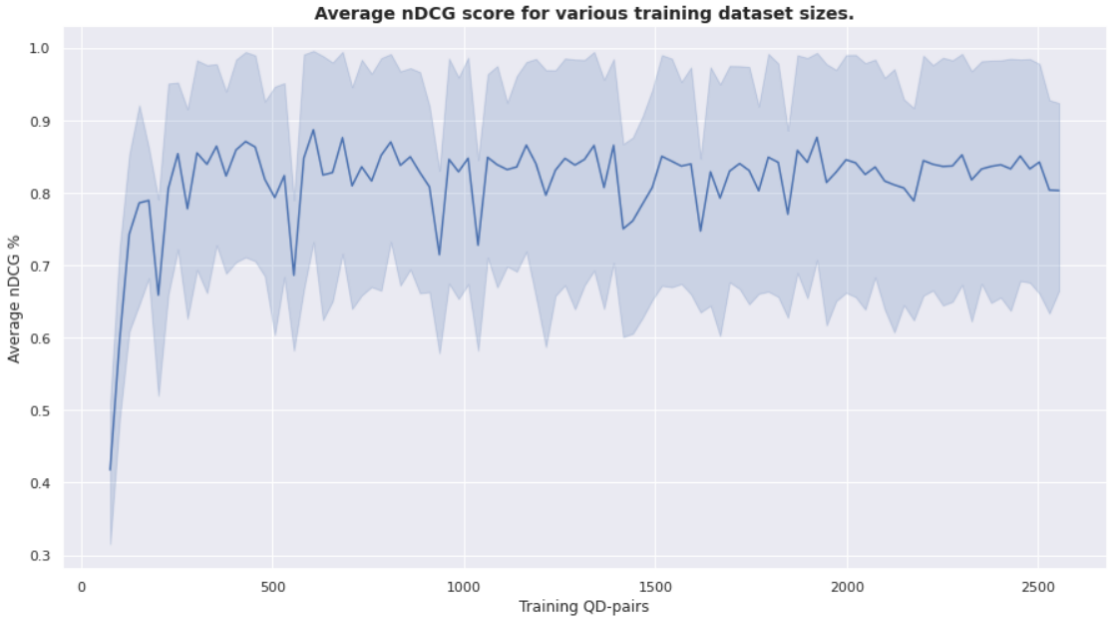


Figure 3.16: Impact of dataset size on ranking performance. The ranking performance, as measured by the average nDCG@k score (where k = 1,3,5,10 and 20) on the y-axis, is seen to plateau after the first 1000 training Q-D pairs (x-axis).

Figure 3.16 is a learning curve that plots nDCG performance for various training dataset sizes. The plot shows the ranking performance growing rapidly and then

reaching a plateau. As the size of training data (number of Q-D pairs) increases, the ranking performance improves following a power law. But, after about 1000 Q-D pairs, the ranking performance plateaus (based on visual inspection). While the curve initially exhibits some turbulence, it is seen to 'smoothen' with increasing Q-D pairs. This would suggest that the size of `ENTRP-SRCH` dataset (2554 Q-D pairs) is more than sufficient for the given ranking task in EXP-1, and is not a limiting factor. For reproducibility purposes, the code used to generate this plot is available online [10]. The sizing study is conducted as an auxiliary experiment and described below.

- Hypothesis: The null hypothesis is that, there is no requirement to further expand or otherwise adjust the `ENTRP-SRCH` dataset, for the purpose of ranking the most frequently queried ('fat head') topics. The alternate hypothesis is that a bigger dataset would lead to improved ranking performance.

- Baseline: A learning curve that plots nDCG performance for various training dataset sizes. The plot is designed to estimate the dataset size required for a target performance value. The plot shows the ranking performance growing according to a power law and then reaching a plateau.

- Mechanism: Learning curve graphs can be employed to estimate the required size of a dataset (for a given task).

- Sufficiency of Data: As training data size (the number of Q-D pairs) increases, the ranking performance improves following a power law. But, after about 1000 Q-D pairs, the ranking performance plateaus.

- Domain Knowledge: For this experiment, there is little requirement for any knowledge of the domain of ES. The knowledge is intrinsic to the dataset.

- Limits: The assumption is that the task for which the dataset is employed is to optimise ranking for the most frequently queried terms (e.g., EXP-1, EXP-2). This does not include EXP-5, which assumes a very different task (ranking jargon terms). Apart from this, overfitting can occur if a statistical model fits exactly against its training data (including noise and irrelevant information). The sizing experiments assume validation is undertaken as a pre-requisite.

- Results versus Expected Results: As the dataset size increases, the performance initially increases until the model saturates. The minimum sufficient dataset size occurs when adding more data does not significantly improve performance. When the model performance saturates, it can be assumed that the general population and the training dataset now have similar underlying distributions. Figure 3.16 shows that the experimental results closely match these expectations.

[10] https://github.com/ColinDaly75/LTR-dataset-sizing-ES

3.13 Conclusions

This chapter has explored the application of AI methods, specifically LTR and LM, in the context of ranking search results for ES. The research questions addressed are RQ-1 (To what extent can Learning to Rank improve relevance matching and ranking performance of search results for Enterprise Search?) and RQ-2 (To what extent can Language Modelling address vocabulary mismatch by improving semantic matching for Enterprise Search?).

The primary objectives were to evaluate whether these AI methods offer measurable improvements over traditional ranking methods such as BM25 and field boosting (OBJ-1 and OBJ-2). Through a series of structured experiments, the effectiveness of these methods was assessed using the nDCG and MAP ranking performance metrics.

The results of EXP-1 demonstrate that the LTR-based ranking approach outperforms traditional methods, achieving higher nDCG scores at various cutoff points (e.g., nDCG@3, nDCG@5, nDCG@10). The ablation study in EXP-2 revealed that BM25 and field boosting remained strong contributors to ranking effectiveness, underscoring the importance of combining traditional ranking functions with AI-driven approaches. Surprisingly, the recency feature (i.e., how recent is the document's creation or modification timestamp) contributed little to the LTR model. This may be because many ES documents do not have any ascertainable timestamp. Documents with extensions such as TXT or CSV do not have built-in metadata for a creation date. In these cases, the median value was *imputed* for the document and stored in the Solr index. The median was chosen as it represents the central tendency and is less sensitive to outliers than the mean. The imputations may have the effect of diluting the contribution of the recency feature.

The findings of EXP-3 indicate that utilising CTR data, as a surrogate for human relevance judgments in LTR models is a valid approach, but introduces potential biases related to organisational and position effects. Notably, this finding means that a dataset could be created, based on CTR data, that is sufficiently diverse and large enough to carry out diverse ranking tasks (i.e. capture the nuances of enterprise specific terminology).

The IAA experiment (EXP-4) revealed that while human relevance judgments exhibited moderate agreement levels, variability among annotators highlighted subjective differences in document assessment. This underscores the need for diverse annotator perspectives or alternative feedback mechanisms, such as implicit user interactions, to enhance the reliability of training data for LTR models in ES. The decision to depend on a single judge, even if they are a recognised subject matter expect, to award annotations for a topic or subject area was questionable. A better approach would have been to seek out two judges, each willing to participate and annotate for their topic. This IAA test is a validation experiment; it was introduced latterly to confirm the integrity of the `ENTRP-SRCH` dataset in spite of this single-rater-per-topic limitation.

EXP-5 was designed to address OBJ-3. It showed, however, that the JARGES fea-

ture yielded no significant quantitative improvement over the baseline ranking performance (nDCG@10 = 0.964, $\Delta = 0.001$, $p > 0.05$). This failure is likely due to the dataset's lack of jargon-rich pairs and highlights the need for larger ES datasets derived from diverse, real world click-through data or other implicit feedback to detect subtle ranking signals. Had the `ENTRP-SRCH` dataset been designed with more queries (perhaps traded against having fewer judgments per query), it is suspected that the experiment would have proved a quantitative improvement in ranking scores. The qualitative analysis of the generated synonyms (using the 'Jargon Buster' reference file) reveals promising results for recall as a foundation for semantic search.

Despite these general quantitative and qualitative improvements, the adoption of AI-based ranking methods introduces additional complexities. The implementation effort required for training, feature engineering, and maintaining LTR models necessitates expertise in machine learning, relevance tuning, and statistical analysis. Moreover, while the AI-enhanced ranking model provides clear advantages in precision and recall, its real-world applicability must be weighed against the availability of organisational resources and the need for periodic retraining.

In summary, this chapter provides quantitive evidence supporting the use of LTR (research contribution RC-1) for ES search result ranking. In terms of LM and NLP, the JARGES algorithm exhibited some promising qualitative results which partially address the 'recall' element of RC-5. However, organisations considering these AI methods should carefully evaluate the trade-offs between performance improvements and the associated implementation complexity. Future research may focus on further expanding the dataset using CTR data, whilst mitigating both organisational and position bias in implicit feedback annotations.

The conclusions and observations drawn from the ranking of search results cannot be examined in isolation from those of QAC candidate ranking (described in Chapter 4). Hence, a detailed and holistic synthesis of the experiments and associated results will be undertaken and discussed in Chapter 5.

4

RANKING OF QUERY AUTO-COMPLETIONS

4.1 Introduction

Whereas Chapter 3 explored the role of AI methods in the ranking of ES *Search Results* (RQ-1, RQ-2), this chapter examines the ranking of *Query Auto-Completions* (RQ-3, and RQ-2). QAC is an important aspect of any search service, but it is of particular importance to the field of Enterprise Search for the following reasons:

- Query Auto-Completions (QAC) prompts query reformulation which helps users avoid submitting queries that produce no results (a potentially common occurrence, as an ES corpus is small compared to those used by commercial WS engines) [6].
- QAC for ES can educate new staff members about the range of selections available to them and assist in narrowing that selection even before the user has finished typing a query. Contextual suggestions can steer searchers to use the appropriate and official organisational jargon/terminology.
- Users of Google's WS service have collectively 'saved over 200 years of typing time per day' [22]. But, unlike WS, searches on an ES service are usually undertaken by 'knowledge workers' attempting to carry out a specific work task. Any delay in retrieving the desired information or document has a monetary cost for an organisation.
- QAC is said to reduce keystrokes by 50% for WS users [125]. For ES, QAC also has the potential to assist disabled personnel in matters of keyboard accessibility and ease of use.

As is the case with the ranking of ES Search Results, any poor user experience with QAC can be a source of frustration that could lead to negative feelings toward the service and organisation [42, 52, 53].

This chapter addresses the question as to what extent LTR and LM methods can be used to improve the ranking of candidates generated for Query Auto-Complete for

ES-specific content (RQ-3) and an organisation's specialised language and jargon (RQ-2). The objectives are to develop and evaluate a ranking model for the presentation of QAC candidates that is customised for enterprise content (OBJ-4) and associated specialised jargon/terminology (OBJ-3). To this end, aspects of QAC ranking performance on The University's ES service (using a combination of offline and online experiments) are developed, deployed, base-lined and evaluated.

Structurally, this chapter adopts the same section headings as Chapter 3. It starts with an examination of the various signals used by the traditional (non-AI) approach (§4.2). As the quantity and diversity of the signals grow (§4.3), LTR is used to apportion 'importance' to each (§4.4). LM is then applied in the form of the newly developed 'QACES' feature (described in Section §4.5), which is used to detect and uprank organisational jargon/terminology. The experiments are as follows:

- EXP-6: Compares QAC ranking performance of traditional and AI approaches.
- EXP-7: An ablation (leave-one-out) study to discover relative contributions of each QAC ranking feature.
- EXP-8: Determines whether the QACES LM detection feature can uprank jargon/terminology candidates.

Each experiment is systematically described and, wherever possible, carried out both on an ES and a WS dataset (§4.6). Implementation details and challenges (both experimental and organisational) are described in Section §4.7.1, §4.8.1 and §4.9.1. The results of the experiments are presented and tabulated toward the end this chapter in Sections §4.7.2, §4.8.2, and §4.9.2. Finally, to ensure the experiments are rigorous and the results robust, several validation checks are reported out in Section §4.10.

An integrated synthesis, framing and reflection of these results (from both this chapter and Chapter 3) will be discussed in Chapter 5.

4.2 Traditional Ranking

As outlined in Section §2.3.2, Google incorporated QAC into its search engine in 2008. Search engines began incorporating LTR in the mid-2000s. Word embeddings (e.g., Word2Vec) did not come to prominence until 2014. Google's Transformer breakthrough for Language Modelling was in 2017 [232]. Considering these timelines, the question arises as to how ranking performed before the advent of AI. Hence, before evaluating the impact of the AI ranking approach for QAC, an experiment is conducted to analyse how QAC ranking can perform using traditional and heuristic (non-AI) ranking functions for an ES service.

Establishing an initial (minimum) baseline using traditional ranking functions also informs the discussion of whether state-of-the-art AI ranking approaches are 'worth the effort' in the domain of ES (where search may not be at the core of the business, and organisational resources and skills may be limited).

Traditional QAC ranking functions are based on corpus content or users' collective search history. Two common traditional functions are ranking by a) term frequency and b) query frequency.

4.2.1 Ranking by Term Frequency

In the context of QAC, Term Frequency (TF) is the number of times an indexed term containing the prefix appears in the corpus. The assumption is that a higher TF value signals higher candidate relevance for a given prefix.

A TF value can be computed for all fields (combined) within a corpus or, at a more granular level, it can be scored for individual fields. For example, a corpus may have fields such as title, footer, content, path, anchor text, etc.

Unlike WS, however, the effectiveness of Term Frequency as a relevance signal may be reduced in the domain of ES. This is because, within an ES corpus, many words appear multiple times as part of branding within the header, title, or footer fields. For example, the name of The University, the university's maxim, address, contact details, etc, may appear on almost all pages or documents in an ES corpus. This repetition may be part of The University's consistent branding policy but has the impact of diluting the originality (i.e. reducing the relevance) of the term.

In Chapter 3, TF-IDF and BM25 were used to overcome this limitation when ranking *Search Results* (which is a document retrieval task). The IDF element of TF-IDF is a measure of how common or rare a term is across all documents in the corpus. A term that appears in many documents has a lower IDF, indicating that it is more common (i.e. less original). However, the concept of assigning a weight to words that are more unique and relevant to specific documents rather than those common across all documents is founded on *Query Document (Q-D)* pairs. Unfortunately, since QAC is based on *Prefix-Candidate(P-C)* pairs, no individual document is considered. Hence, neither TF-IDF nor BM25 is applicable to the field of QAC. TF, when used alone, is unlikely to rank QAC candidates in a sequence that matches users' intent.

4.2.2 Ranking by Query Frequency

Whereas TF relates to counting the number of occurrences of a term in the *corpus*, Query Frequency is about mining the *query logs* for previously submitted terms and counting their frequency. With the Query Frequency ranking function, more frequent queries appear higher in the suggestion list.

As described in Section §2.3.7, Most Frequent Queries (MFQ) [130] is a sorted list of historical queries representing the topics or keywords searched for most frequently by the user community and simply ranks the most frequent suggestions matching the input prefix. MFQ consists of a table of queries with their frequency of occurrence extracted from the query logs and can be applied as a ranking feature. Chapter 3 also documents how the MFQ list can be easily extracted from the University's log data, and how it can be used produce a search demand curve.

The accumulated Search history (i.e. of all users collectively) is said to be a likely

indicator of current search intent [131]. Hence, even when used alone as a ranking signal, MFQ is likely to closely rank QAC candidates in a sequence that matches users' intent.

In EXP-6, the traditional QAC ranking methods described above will act as a non-AI baseline for ranking performance. This will then be compared to the results obtained using an AI-based approach on The University's ES service.

4.3 Other Hand-Crafted Ranking functions

In addition to the traditional ranking functions above, a number of other functions can be created to rank the completions. The list of ranking functions below can be described as 'heuristic' because they are carefully hand-crafted based on domain knowledge, known user behaviour, empirical observations, predefined criteria and information retrieval principles. These ranking functions do not employ complex statistical models or machine learning algorithms. The contribution of each of the functions described below will measured in EXP-7.

- *aolFeature* This ranking function is generated from a list of pertinent queries extracted from the AOL WS dataset (20 million search queries from about 650,000 users collected between May and July 2006). It is a table of queries with their frequency of occurrence extracted from the AOL dataset. An example is "music downloads 517", which tells us that the query 'music downloads' has been submitted to the AOL search engine 517 times. To use this feature in ES, the WS list must be filtered to offer only 'sensical' candidates. Hence, the usable list for The University's dataset is filtered down to just 1740 matching candidates, which is 0.009% of the AOL total. This highlights how the AOL WS queries differ markedly from the types of query expected for ES, which are of a much narrower focus, as shown in Table 4.1.

Table 4.1: The top 10 most popular query terms for the 2006 AOL WS compared with The University's ES query history. The two columns contain entirely distinct sets of terms.

Rank	AOL Web Search	Enterprise Search
1	google	scholarship
2	ebay	fees
3	yahoo	library
4	mapquest	phd
5	yahoo.com	medicine
6	google.com	psychology
7	myspace.com	erasmus
8	internet	courses
9	myspace	vacancies
10	www.google.com	law

- *trendingFeature*. This ranking function incorporates trending queries within an enterprise, helping users stay updated with searched topics that are both recent and common. These queries can potentially uncover useful information

such as 'what are our ES users collectively working on, curious about and thinking about today'. A trending query is a search term or keyword phrase that is currently experiencing a significant increase in search volume over a short period of time (as described in Section §2.3.8). This sudden rise in interest may be due to breaking news, recent events, or seasonal changes. To measure an abnormal spike, one must first determine what a normal baseline volume would be. This type of calculation lends itself to z-scores (To measure an abnormal spike, one must first determine what a normal baseline volume would be. This type of calculation lends itself to z-scores (i.e. quantifying how far an observation deviates from the baseline mean [233]). In this case, the burst of popularity against the backdrop of the historical average (including its standard deviation) is computed. This feature consists of a table of queries with their computed z-score. A higher z-score indicates that the query is more 'trending'. Using The University's ES data, an interesting example is "erasmus 24.9". The 'erasmus' query, with a z-score of 24.9, is a student exchange program, which allows students to study, intern, or train in another EU country for a certain period. Applications open for Erasmus at certain times during the academic year, and this might explain any temporary surges in query volume for this topic. The code used to compute z-scores for trending queries is published on github[11].

- *anchorTextFeature*. The 'anchorText' is the link label that content providers use to describe a document. This feature was generated using the LinkRank algorithm (similar to Google's PageRank) as described in Section §2.2.5. In the field of Web Search, this feature can be expected to have a high weighting coefficient as it tags meaningful descriptive text and adds context to a document. In ES, LinkRank may be less effective, as many documents are created without publishing intent (e.g., MS Word documents placed on an intranet drive). This feature consists of a table of terms with their frequency of occurrence. An example is "research support system 24", which tells us that the phrase 'research support system' is encoded into anchors 24 times in The University's corpus (these anchors originally pointed to a specific page or group of pages). Considerable filtering was required to remove non-descriptive terms and repetitive labels such as 'Next page', 'Previous Page', 'Home', etc.

- *titleFeature*. This term frequency ranking function uses a list of QAC candidates with their corresponding frequency retrieved from the field of The University's enterprise corpus. An example is "communication 333", which tells us that the term 'communication' occurs 333 times in the title field.

- *contentFeature*. This term frequency ranking function uses a list of candidates with their corresponding frequency retrieved from the content field of the enterprise's corpus. The ranking is derived from the sorting of the frequency value.

- *pathFeature*. This term frequency ranking function is based on terms extracted from the path part of the URL or shared drive path field in the corpus. The path part of a URL or drive path potentially contains a rich source of QAC

[11] https://github.com/ColinDaly75/QAC_LTR_for_ES

completions, as shown in Table 4.2. To extract terms and n-gram phrases from the URL field, LM and NLP techniques are used. The NLTK (Natural Language Toolkit) and spaCy library are used to tokenise the text. The CountVectorizer utility from sklearn is then used to extract n-grams from the path[12]. Table 4.2 gives an example of how one URL can be 'mined' to reveal many rich terms and n-gram phrases for use with QAC. The hierarchical and departmentalised nature of an enterprise means that there is likely be a higher TF score for a top level sub-directory.

Table 4.2: Extracting Terms from URL or Shared Drive Path.

URL Path Segment	Recognized Search Terms
Full Path: `/itservices/skills-development/online-learning/linkedin-learning`	
/itservices/	it, services, itservices, it services
/skills-development/	skills, development, skills development
/online-learning/	online, learning, online learning, on line learning
/linkedin-learning/	linked, in, linkedin, learning, linked in,
	linkedin learning, linked in learning

4.4 Learning to Rank

As described in Section §2.4, layering more and more ranking functions has the inherent problem that it is very difficult and eventually impossible to apportion the 'correct' weighting to each function. For example, it was noted that the TF ranking function can be computed at a granular level for individual fields. This raises the question as to what importance should be assigned to each field. The importance of the anchor-text field likely outweighs that of the content field, but by what margin? The LTR AI supervised ML method answers these questions by apportioning a 'weight' to the contribution of each of the ranking functions (or 'features', as they are referred to when using LTR).

XGboost is an open-source Python LTR framework [37] based on ensembles of decision trees and is employed to train the model and determine the ranking performance. It can also calculate each feature's contribution (e.g., via MAP or MRR) to the overall ranking efficiency. RankEval [38, 234] is employed to cross-verify MRR performance scores and validate weightings.

The coming sections (§4.4.1 to §4.4.4) describe how logging is configured to capture users' autocomplete session behaviour, and how this collective historical behaviour is then converted into a QAC dataset, which is used to train a baseline ranking model using LTR-weighted features.

4.4.1 LTR Ground Truth

LTR is the application of machine learning techniques to labelled data to train a ranking model (i.e. supervised learning). Ground truth consisting of a labelled

[12] `https://github.com/ColinDaly75/LM_N-Gram_TF_Extraction`

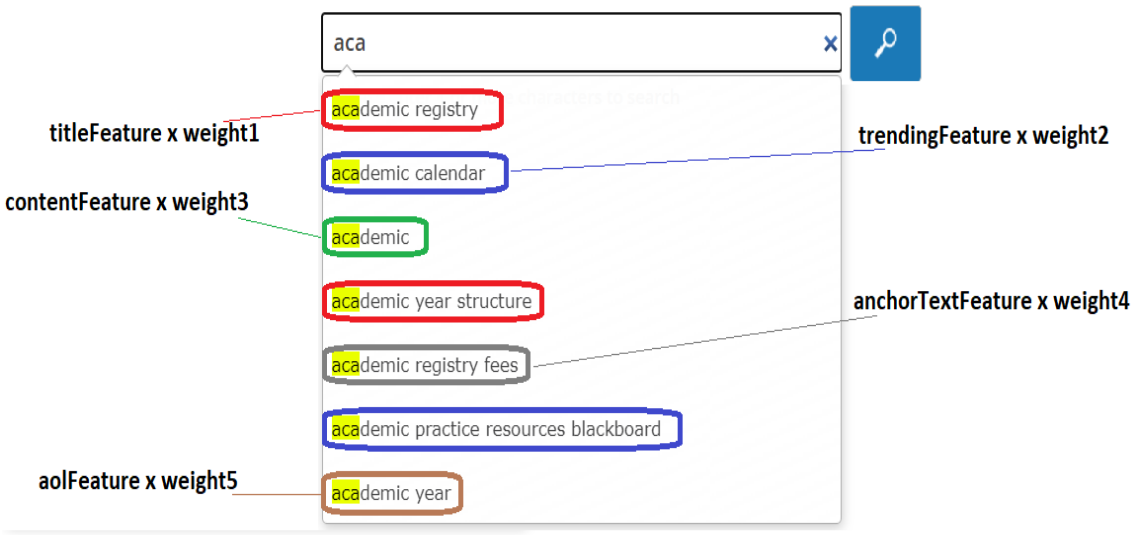


Figure 4.1: The typed "aca" prefix presents a list of completion choices via a ranked list of prefix-candidate pairs. Candidates are generated from various sources/features, each of which is 'weighted' using LTR.

dataset is therefore required. Each input (prefix, P) is paired with a correct output (candidate, C) to create *P-C pairs*. Each prefix is likely to have many potential candidates.

But neither TF nor MFQ can be used as ground truth as they rank candidates independently of the prefix. In other words, TF and MFQ are signals of candidate importance that are ascertained regardless of the typed prefix.

As described in Section §2.3.7. the Most Popular Completions (MPC) [235] ranking function overcomes this limitation by matching completion candidates based on user preference for a particular prefix. MPC is thus a prefix-dependent ranking signal.

MPC is also more sophisticated than TF and MFQ because it depends on user interaction data. The system records which completed query candidate users select after typing a particular prefix. MPC is not computed by a predictive model; it is derived directly from Solr's query and click logs, which capture the frequency with which users choose each completion following a given prefix.

A less practical alternative to a click model would be to ask a group of annotators to explicitly judge prefix-candidate pairs. This is generally considered an impossible task [6], as annotators cannot be expected to guess or estimate which candidates are best suited for prefixes of various lengths. Relevance judgements in the form of annotated *query-document* pairs are typically required to train a ranking model for documents on the Search Engine Results Page (SERP) [98, 136]. QAC researchers rarely rely on the same editorial effort to annotate *prefix-candidate pairs* manually. More often, QAC relies on collecting large-scale historical data, such as MPC, for QAC tasks [131]. Previously recorded behaviour and queries provide helpful information for any user's intent and can be leveraged to suggest more relevant completions while adhering to the user's prefix [130].

In terms of implementation MPC presents some challenges (as described in Section §4.7.1), but it is typically considered the gold standard of historical relevance [25, 235]. Yossef et al. claim that MPC carries 'the wisdom of the crowds' [236] and can be regarded as an approximate maximum likelihood estimator [25, 130]. MPC is thus considered a credible proxy for ground truth in QAC LTR datasets.

The QAC dataset for the LTR AI method is therefore based on MPC data created from The University's ES log files using a click model. The process is described in the next section.

4.4.2 Log Collection for QAC

QAC session data can be visualised as consisting of two dimensions. The horizontal dimension consists of prefix keystrokes. The vertical dimension consists of candidate drop-down selection. According to Li [237], collection of this 2-D session data constitutes a 'high-resolution QAC log'.

By default, the log files on the Apache Solr Enterprise Search platform [14] only collect submitted queries. To enable detailed capture of users' session behaviour (and MPC data), the Solr log4j module[13] is modified to record the suggestion candidates for a prefix, the selected candidate (if any), and finally, the submitted query. Table 4.3 lists the recorded parameters for each query session. This high-resolution QAC log approach mirrors that used by Google and Yahoo [25].

Table 4.3: QAC session logging parameters to capture suggestion candidates for a prefix, record the user's selection, and the submitted query.

Parameter	ES QAC Dataset Example
user id (anon)	qtp1209411469-21
session id	33418 (rid)
timestamp	2024-01-14 16:25:37.697
prefix	"aca"
top suggestion candidates (by MFQ)	academic registry, academic calendar, academic year structure, academic registry fees, academic year, academic practice, academic resources
user selection	"academic registry"
submitted query	"academic registry"

The QAC session commences when the first letter is typed into the search box. At this point, the session id is allocated and the timestamp recorded. The autocomplete minChars is set to '2'; this means that QAC candidates are presented and recorded only after the second letter is typed. This setting was chosen to avoid placing an excessive demand on the Apache Solr framework. For example, in The University's ES corpus, there are 94226 completion candidates starting with the letter 'a', 'ac' has 12180 completion candidates and 'aca' has 2193 completion candidates.

[13] `https://cwiki.apache.org/confluence/display/solr/SolrLogging`, accessed 16th Jan 2024

User identification is anonymised in accordance with the approval restrictions granted during the research ethics process (Section §1.4.4 and Appendix C). Similarly, no IP address information is gathered in the search log files. The timestamp is gathered to identify timed-out sessions. Based on knowledge (estimation) of typical user behaviour, the timeout has been set at 60 seconds.

The session ends when:

- the user selects one of the presented QAC candidates, or
- the user clicks the 'submit' button, or
- the user hits return (equivalent to clicking submit), or
- the timeout is exceeded (session fragment is ignored and no valid QAC session is recorded)

This enhanced logging, together with offline scripts to extract and shape the MPC prefix-candidate pairs into tabular form, are the constituent parts of my basic 'click model'. This method is similar to that carried out in Chapter 3 where another basic click model ('click-through') was used to record and annotate end-user Q-D pair preferences for *Search Results*.

The enhanced logging for QAC has no discernible impact on the responsiveness or latency of The University's live ES service. Storage was added to the back-end Linux servers to host both the enhanced logging file size and extended file retention.

4.4.3 QAC Dataset construction

MPC consists of user preference data (extracted above) for P-C pairs. Enumerated annotations {1-5} are allocated to the candidates for a given prefix, representing {utterly irrelevant, irrelevant, moderately relevant, relevant, highly relevant}. The annotation scores are computed as a percentile of the MPC score. The scores are therefore normally distributed with an approximately equal number of each for a given prefix. Figure 4.2 shows an example extract of The University's QAC dataset for a single prefix, which has been formatted for use with the XGBoost [37] or RankLib [238] LTR framework. The first column in the dataset is the so-called 'relevance label' [31]. The second is the prefix id (pid). Some LTR frameworks require this to be called qid (query id, which reflects that LTR is historically and primarily used for training Q-D pairs). Columns three to eleven are the feature vectors. Each feature value is proceeded by a feature number and a colon. For example, in the first row, "9:12" means that the ninth feature has a value of 12. This means that the candidate term 'open day' occurs twelve times within URLs within the corpus.

The candidate must contain (i.e. start with) the prefix string. Each feature is necessarily based on the candidate (the prefix itself may have no meaning). This differs from feature classification in the domain of *search results*, which can have both query dependent and query independent features (§2.4.6).

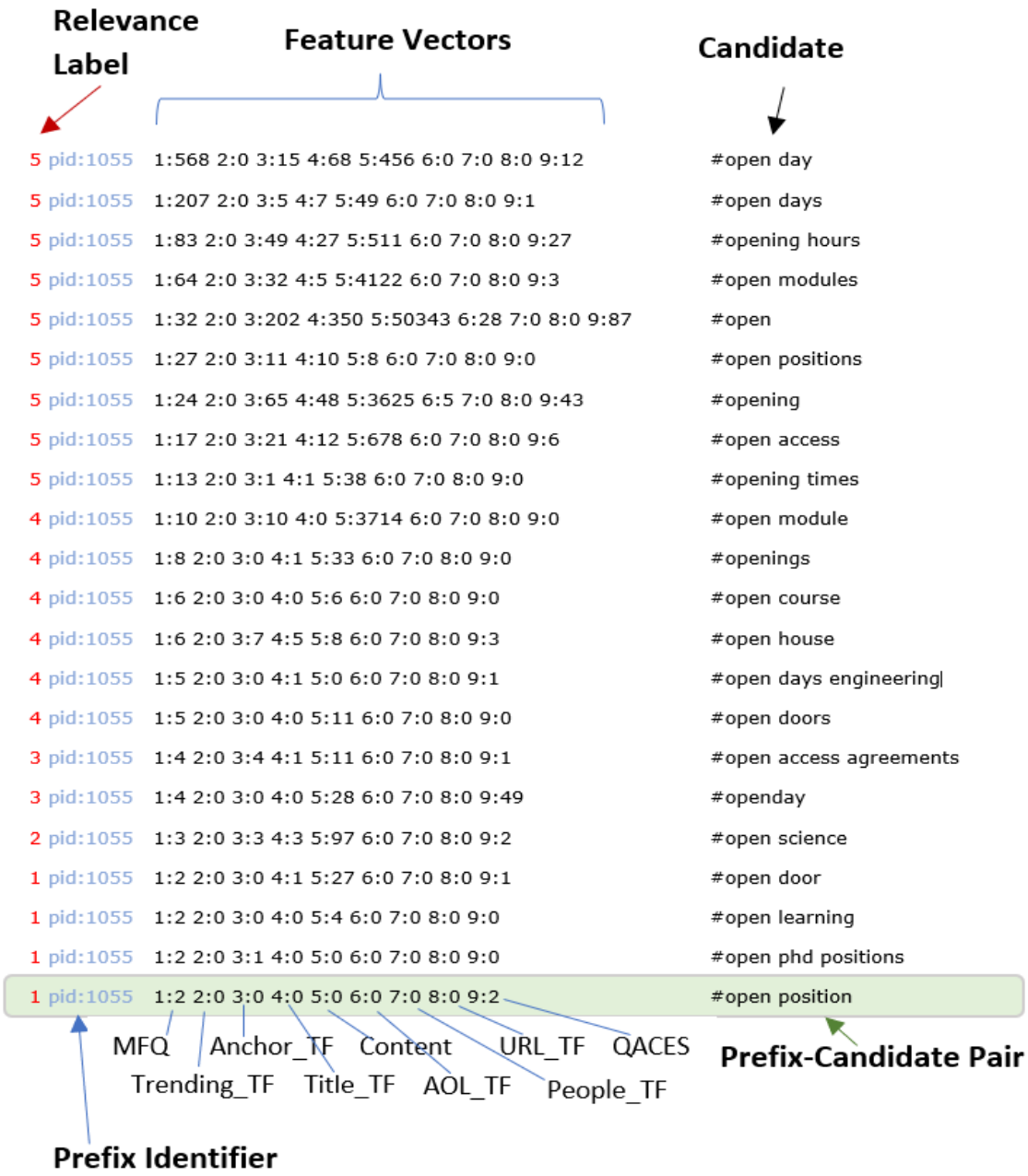


Figure 4.2: An extract of The University's LTR-formatted dataset including a sample of the prefix-candidate pairs for the "open" prefix (id 1055). Each candidate has an associated judgement for a particular prefix (generated using MPC). Columns labelled 1:n to 9:n represent an array of nine feature vectors representing MFQ, Anchor_TF and other ranking functions described in Section §4.3.

The dataset's creation was subject to approval by both an ethics committee and The University's IT Services department. A condition was that personal data (and metadata) should be anonymized or pseudorandomized to protect individuals' identities (GDPR Article 25). It is based on log data collected over six months between June and December 2024. This period recorded 902,412 prefix-candidate pairs. This is relatively small compared to the AOL WS dataset (17,521,031 P-C pairs, collected over the course of one month). Therefore, in Section §4.10, a learning curve graph is used to validate the data quantity gathered over this time period to ensure that

The University's QAC training dataset is sufficiently sized.

4.4.4 LTR Model

At this point, the LTR-formatted ES QAC dataset has been created and is ready to be used to generate a QAC ranking model. The ranking model is generated using the XGBoost (Extreme Gradient Boosting) implementation of the LambdaMART list-wise ranking algorithm. As outlined in Section §2.4.9, a list-wise loss function examines the entire ranked list rather than individual pairs or items. Table 4.4 lists the hyperparameters used.

Table 4.4: Hyperparameter settings used to evaluate ranking performance for the Learning to Rank ranking method.

Parameter	Value
Algorithm	LambdaMark
Framework	XGBoost
max_depth	10
rank	MAP
num_round	10
eta	0.5
Validation method	Cross-Validation (CV)
CV Folds	5

The LambdaMART algorithm uses gradient boosting to build an ensemble of decision trees. With each new tree, it improves the ranking by reducing errors in previous predictions. LambdaMART optimizes list-wise ranking measures, making it well-suited for tasks where the position of the first matching item in a ranked list is of prime importance (as is the case for QAC).

XGBoost is a highly efficient and scalable implementation of the Gradient Boosting (GB) ML algorithm, widely used for supervised learning tasks. The XGBoost 'max_depth' parameter is the maximum depth of the GB tree.

The learning rate (designated by 'eta') is set to 0.5. A higher value for eta results in higher computation speed, but would require a higher num_round value. Since this is an offline experiment for computing feature weights, it was not necessary to use a high value.

As discussed in Section §2.3.10, MRR is accepted as the de facto metric for measuring QAC ranking. However, for determining LTR feature weights, the MAP or nDCG metrics are generally used. Moreover, some LTR frameworks (like RankLib [238]) do not natively include MRR as an evaluation metric, requiring manual implementation. Hence, MAP enables cross-comparison between weights derived from different LTR frameworks.

In this section, the background methodology to create a dataset for use with the LTR frameworks has been described. These methods are used in Section §4.6 where

a series of LTR experiments are described. The resultant weightings output by the LTR model are described in Section §4.7.1.

The next section similarly describes the background methodology for the use of language modelling in enterprise search.

4.5 Language Modelling for ES Jargon

As discussed in Section §2.1.8, ES content is often imbued with organisational terminology/jargon (i.e., words, phrases, expressions, and idioms that are not universally familiar or have different meanings inside and outside of an organisation). To address this, a QAC ranking feature explicitly designed for ES is introduced. This feature, which harnesses both LLM and LM techniques, is centred on the relative unusualness of words, such as those used in organisational jargon/terminology.

4.5.1 QACES LLM feature for ES

The task of detecting enterprise jargon/terminology terms depends on comparing semantic divergence. I call this feature 'QACES' (Query Auto-Complete for Enterprise Search). It is computed using the synonyms of terms in the real world and the closest terms in an enterprise corpus. Where these differ significantly, the term is considered jargon. Once the jargon terms have been detected, they can be used as a feature in the ranking model.

Whereas JARGES (described in Section §3.5) focuses on improving the ranking of search results through the detection and interpretation of organisation-specific terminology, QACES addresses a distinct problem: the generation and ranking of query auto-completion candidates during user input.

Table 4.5 outlines the interacting LLM and LM components of the QACES feature. The first step is to use LLM to identify synonyms (i.e. nearest terms) in the real world of the most common query terms in The University's search logs. The next step is to use LM (via corpus vectorisation) to identify the within-corpus synonyms of these same terms. The two lists are then compared to discover discrepancies.

Extracting LLM near-synonyms LLMs are trained on enormous datasets containing vast amounts of text from diverse sources. The near-synonyms extracted from an LLM model are therefore assumed to represent *common parlance*.

OpenAI's GPT-4 model [36] includes an API called 'client.chat.completions.create'. This is used to produce a list of ten English language 'LLM nearest' terms for each query in the 'fat head' zone of The University's Search Curve (this is the second column of Table 4.5). A potential alternative tool to GPT-4 would have been WordNet [239]. WordNet was not used because frequency data are not independently available, making it impossible to determine the *nearest* terms.

The API was used to generate the near-synonyms (as opposed to using the graphical interface). This prevents data leaking [240] which could feed back into the GPT-4 model. The temperature parameter was set to both 0.5 and 1.0 (changing this

parameter did not significantly modify the list of nearest terms). The prompt was "create a flat, json-formatted, sorted, unnumbered list of the top 10 nearest (semantically) words or phrases for each of the words in the following array". One of the strengths of GPT-4 is that it is conversational in nature, and does not tend to repeat itself. For this reason, the prompt was very specific. A factual inquiry about top ten near-synonyms, as measured by frequency in the LLM model, would not be expected to fluctuate. Nonetheless, it was a struggle to achieve repeatable lists of near-synonyms on each run. The detailed wording of the prompt was necessary to achieve repeatable results. The output array included near-synonyms for all of the query terms in the 'fat head' of The University's Search Demand Curve.

Extracting near-synonyms using ES Corpus Vectorisation Word embeddings represent terms as dense vectors where similar words are closer together in vector space. Word2Vec [34] is used to learn representations based on their contextual usage in the ES corpus. This produces word and phrase vectors where vectors close together in vector space have similar meanings based on context. This becomes a list of 'Corpus Nearest' terms for each query in the 'fat head' zone of the Search Curve (this is the third column of Table 4.5).

Table 4.5: A comparison of the top synonyms for two examples of 'fat head' queries. The 'Divergent Terms' are those that commonly feature in the Enterprise Corpus but are not part of LLM's common vocabulary. The 'JD Distance' is a measure of jargony.

'Fat Head' Query	**LLM Nearest (X)**	**Corpus Nearest (Y)**	**Divergent Terms ($\widetilde{X} \cap Y$)**	**JD Distance (Jargony)**
scholarship	grant, bursary, fellowship, financial aid, award, stipend, tuitions assistance, academic fund, educational grant, study grant	foundation scholarship, scholarship examinations, entrance scholarships, scholarship exams, schols, visiting scholar	schols	0.65
id card	Identification Card, Identity Card, Personal Identification, Photo ID Card, Driver's License, Passport, Employee Badge, Student Card, Membership Card, Official Documentation	id card, student card, student id, tcard	tcard	0.9

Detecting Jargon/Terminology Jargon/terminology consists of enterprise-specific words, phrases, expressions, and idioms that diverge from those universally familiar or understood outside of an organisation. In Set Theory, these divergent terms can represented by the set of elements both in Y and not in X:

$$Divergent\ Terms = \widetilde{X} \cap Y$$

where X is the set of LLM generalised terms and Y is the nearest neighbour terms with the ES corpus. The fourth column in Table 4.5 lists the divergent terms.

Jargony Jaccard Distance (JD), also known as the Jaccard similarity coefficient, is a measure commonly used to calculate the similarity or dissimilarity between two sets of words such as those in columns 2 and 3 in Table 4.5. JD is particularly useful in scenarios where the presence or absence of elements is more important than their actual values. In this case, the Jaccard distance represents a measure of divergence. A higher distance suggests the divergent terms are more 'jargony' (i.e. more unlikely to be understood outside the enterprise). The calculated JD score is presented in the final column of Table 4.5).

In set theory mathematics, JD is defined as the size of the intersection divided by the size of the union of the sample sets, as per the following formula:

$$JD = \frac{|\widetilde{X} \cap Y|}{|\widetilde{X} \cup Y|}$$

This formula can be practically represented as a function in Python using the code in Listing 4.1. Note that, for interpretation purposes, the jargony score is one minus the JD similarity score. Jargon is thus a measure of dissimilarity (i.e., the bigger the number, the bigger the distance, the more jargony the candidate).

```
def jaccard_jargony(list1, list2):
    # inverse of jaccard, i.e. big number means big distance
    set1 = set(list1)
    set2 = set(list2)
    intersection = len(set1.intersection(set2))
    union = len(set1.union(set2))
    #print (set2-set1)   # the words in list2 that do not exist in list1
    number = (1 - (intersection / union))
    decimal_value = decimal.Decimal(number)
    rounded_number = decimal_value.quantize(decimal.Decimal('0.00'))
    return rounded_number
```

Listing 4.1: The Jargon Distance dissimilarity function and computation in Python (3.1.4).

As will be described in EXP-8, the hypothesis is that adding this jargon feature to the LTR ranking model significantly increases the QAC ranking performance, as measured by Mean Reciprocal Rank (MRR).

4.6 Experiments

Three experiments are designed to determine the extent to which LTR and LM methods can be used to improve the ranking of candidates generated for Query Auto-Complete for enterprise content (RQ-3) and an organisation's specialised jargon/terminology (RQ-2).

Each of the experiments listed below is designed with a consistent structure (description, hypothesis, baseline, mechanism, sufficiency of data, domain knowledge, limits and expected results). The experiments are intended to be implemented independently but are presented in a logical sequence (EXP-7 elaborates on the finding of EXP-6, etc). The results are subsequently presented in Sections §4.7.2, §4.8.2 and §4.9.2.

These QAC experiments were published in Paper III "'Learning to Rank for Query Auto-Complete with Language Modelling in Enterprise Search." and the associated code is available on GitHub.

4.7 EXP-6: QAC Ranking Approach Comparison

This experiment compares QAC ranking performance between a traditional approach and an AI approach in the domain of ES. For the traditional approach, the QAC ranking is configured to use the TF ranking function. This is then compared to the ranking performance when both of the LTR and LM methods (i.e. the QACES feature) of an AI approach are applied. In both cases, ranking performance is measured using MRR. Apart from determining whether an AI approach can improve ES QAC ranking, the outcome informs the discussion of whether state-of-the-art AI ranking approaches are 'worth the effort'. Furthermore, this initial result provides validation that the approach is sound before later advancing to EXP-7 and EXP-8.

- Hypothesis: The hypothesis is that QAC ranking performance, as measured by MRR, improves when AI methods (LTR and LM) are used instead of using the TF ranking function alone.
- Baseline: TF is the default 'out-of-the-box' QAC ranking function, which is why it is chosen as the 'traditional approach' baseline. In Apache Solr, if the `SuggestComponent` is enabled, the "main index is used as a source of terms and weights"[14]. Experiments are to be performed on The University's QAC dataset, which uses MPC as the ground truth for ranking annotations.
- Mechanism: As described in Section §4.4.3, a click model was created to record user preference data and subsequent MPC ground truth, which is used to create The University's LTR ES dataset. The experiment is carried out in an offline setting (i.e., on an instance of Solr that is not serving live requests) using the static dataset. Ranking is measured using MRR metric (as described in Section §4.8.1). No single fixed set of training queries is used as all queries participate in model training through k-fold cross-validation.

[14] `https://solr.apache.org/guide/solr/latest/query-guide/suggester.html`, accessed 18th Aug 2024

- Sufficiency of Data: As described in Section §3.12.2, the *learning curve graph* demonstrated that even the 2544 Q-D pairs of the `ENTRP-SRCH` dataset were more than sufficient to train a model to rank search results for 'fat head' queries. For The University's QAC dataset, log and click data were recorded over six months and a total of 902,412 P-C pairs were recorded. As this is an order of magnitude greater, it is inferred that the data size is sufficient for the QAC ranking task.

- Domain Knowledge: Knowledge of The University's corpus and ES service is a pre-requisite for building the hand-crafted field ranking functions for various fields and LM feature used to generate the LTR ranking model. Amongst the features used are the ranking functions described in Section §4.3.

- Limits: If the improvement using the AI approach is marginal (versus traditional TF), it must be balanced against the real-world implementation efforts in an Enterprise Search QAC deployment. But measuring implementation effort is subjective; there is no easy way to represent it in quantitative terms. Another limitation is the assumption that the traditional approach uses just a single ranking function (TF in this case). This is not necessarily correct, as there is no reason why additional re-ranking functions could not be applied after TF (e.g., recent and trending queries). However, in such a case, without using LTR, it would be impossible to determine the correct weighting to apply to each function.

- Expected Results: An example for a single prefix shown in Figure 4.3 compares the QAC ranking when switching between the TF ranking function and all ML ranking features combined using LTR weightings. Even without domain knowledge (i.e. knowing that 'calendar' is a very popular search query at The University), the AI approach (on the right hand side) seems to produce better candidates. Hence, it can be expected, even from this single but apt example, that the ranking performance achieved when using the combined LTR and LM approaches to exceed that of the traditional approach (i.e., TF ranking function on its own).

4.7.1 Implementation: (EXP-6: QAC Ranking Approach Comparison)

This section details the practical application of the experiment designed above (Section §4.7).

MPC Log Collection The collection of Most Popular Completions (MPC) data, which serves as a proxy for ground truth in the training dataset, took longer than expected. After a three month period, it was observed that many of search sessions in log files had recorded 'null' entries for submitted queries. Two possible reasons for this are outlined below.

Firstly, the MPC null records may have occurred because the desired candidate was simply not listed for the typed prefix. If the initial ranking is poor (i.e. prior to the application of the AI approach), and since The University's ES service only offers seven candidates (k=7), the candidate may not not 'make the cut'.

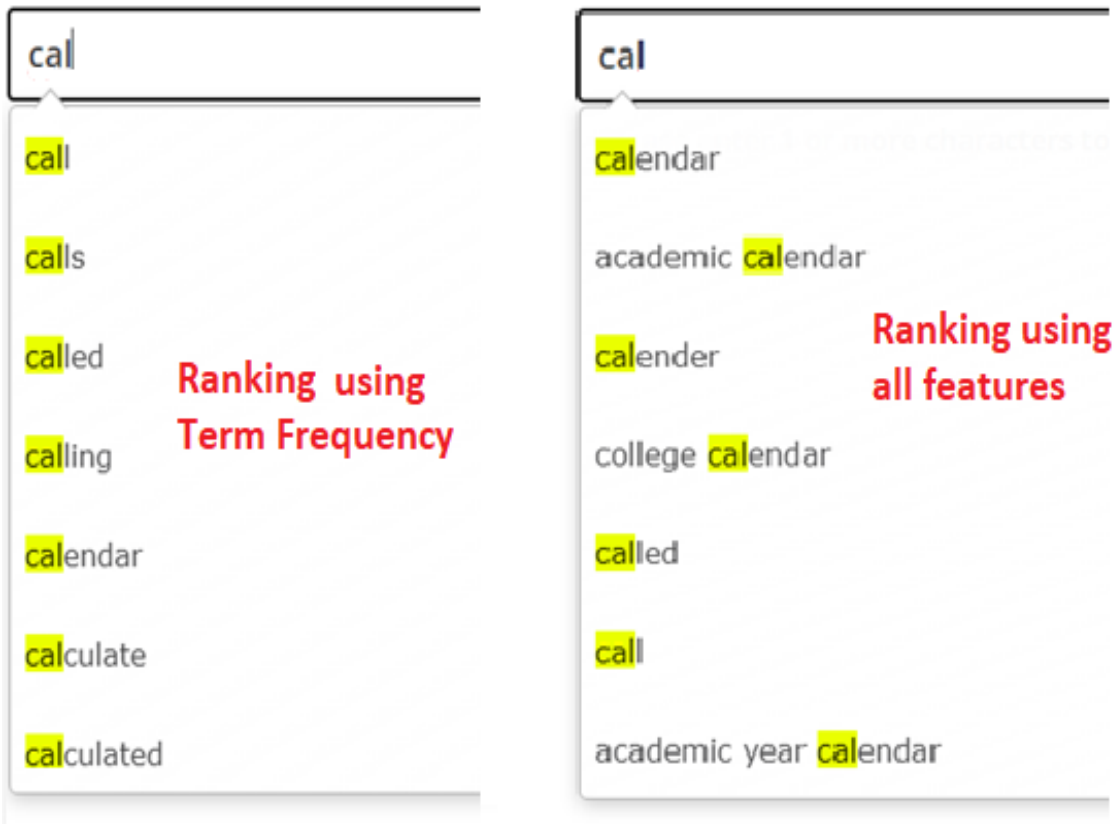


Figure 4.3: An example of QAC candidate ranking for prefix 'cal' using Term Frequency (left) ranking versus all ranking features combined using LTR and LM (right).

Secondly, it may be because many search users fail to engage with the QAC service or to select any suggested completions, even if the correct one is listed. In this scenario, the MPC click model would not record any data (since no candidate was selected). This concept is discussed further in Section §4.9.2.

The MPC null record problem was mitigated using the following twin-track approach:

- The ranking performance, which initially depended exclusively on TF, was improved using the MFQ ranking function. As the ranking improved, a drop in the number of MPC null sessions was observed.
- The MPC data was collected over a six month period (twice the originally planned collection period).

Sensical Suggestions Before using the QAC ranking functions (described above) to present candidates on a live ES service, an important step is the removal of suggestions that would produce no search results if selected. An incorrect suggestion may be considered more damaging than no suggestion, as it would undermine users' confidence in the ES service (it suggests that the corpus and QAC indices are out of sync). This step is sometimes referred to as producing 'plausible completions' or filtering out of 'nonsensical' suggestions [130]. For example, many suggestions

extracted from the 2006 AOL WS dataset (Table 4.1) cannot be used. Even the top AOL queries, such as 'mapquest' and 'myspace' do not produce search results within The University's ES corpus.

Pre-Processing Further pre-processing steps involve converting all queries, prefixes, and completions to lowercase. Full stops, other punctuation, and diacritics were removed. Queries and suggestions with more than eight words, or 50 characters (whichever was the bigger) were removed or truncated. These restrictions are driven by the limited real estate space dimension available to the enterprise search box and the requirement to avoid line-breaks within a candidate.

LTR QAC weightings implementation Table 4.6 shows the RankLib computed weighting associated with each of the ranking features.

Table 4.6: LTR weightings for features calculated using the RankLib framework.

Feature Number	Description	Weight
suggestion_weight_1	MFQ	101
suggestion_weight_2	Trending	8
suggestion_weight_3	Anchor TF	50
suggestion_weight_4	Title TF	80
suggestion_weight_5	Content TF	1
suggestion_weight_6	AOL	3
suggestion_weight_7	People	1
suggestion_weight_8	QACES	20
suggestion_weight_9	URL TF	45

The Apache Solr 'weightExpression' parameter of the 'DocumentExpressionDictionaryFactory' dictionary implementation is used to score the suggestions. The 'AnalyzingInfixLookupFactory' Lookup Implementation allows for suggestions where the starting string does not necessarily match the query prefix. These numeric weights were calculated offline using the RankLib framework [39] and cross-verified using XGBoost [37] (as described in Section §4.4.4). Figure 4.1 depicts how each feature weighting contributes to ranked suggestions in the ES search box. The solrconfig.xml file is published on github[15].

```
<searchComponent name="suggest" class="solr.SuggestComponent">
<lst name="suggester">
  <str name="name">suggester_all</str>  <!--combines all sources -->
  <str name="lookupImpl">AnalyzingInfixLookupFactory</str>
  <str name="highlight">false</str>
  <str name="dictionaryImpl">DocumentExpressionDictionaryFactory</str>
  <str name="field">suggestion_term_all</str>
  <int name="rows">10</int>
  <str name="weightExpression">(
   (suggestion_weight_1 * 101) +  <!-- MFQ -->
   (suggestion_weight_2 * 8)   +  <!-- Trending -->
   (suggestion_weight_3 * 50)  +  <!-- Anchor TF -->
```

[15] https://github.com/colindaly75/QAC_LTR_for_ES

```
    (suggestion_weight_4 * 80)  +  <!-- Title TF -->
    (suggestion_weight_5 * 1)   +  <!-- Content TF -->
    (suggestion_weight_6 * 3)   +  <!-- AOL -->
    (suggestion_weight_7 * 1)   +  <!-- People -->
    (suggestion_weight_8 * 20)  +  <!-- QACES -->
    (suggestion_weight_9 * 45)     <!-- URL TF -->
    )
    </str>
    <str name="sortField">suggestion_weight_1</str>
    <str name="sortField">suggestion_weight_2</str>
    <str name="sortField">suggestion_weight_3</str>
    <str name="sortField">suggestion_weight_4</str>
    <str name="sortField">suggestion_weight_5</str>
    <str name="sortField">suggestion_weight_6</str>
    <str name="sortField">suggestion_weight_7</str>
    <str name="sortField">suggestion_weight_8</str>
    <str name="sortField">suggestion_weight_9</str>
    <str name="suggestAnalyzerFieldType">text_en</str>
    <str name="buildOnStartup">true</str>
  </lst>
</searchComponent>
```

Listing 4.2: Implementation of LTR weights to QAC in Apache Solr. The teal numbers applied to the suggestion_weight lines are those weights generated by LTR

QAC Offline Implementation The offline experiments (EXP-6 and EXP-7) use 'curl' scripts to retrieve and score the ranked list of candidates for a given prefix; no click model is used.

A cloned Solr instance was required to conduct offline experiments, as utilising The University's live production environment was not feasible due to the following constraints:

- The live service handles a constant stream of incoming queries and associated completions. The Solr log files would end up with a mixture of live data as well as the experimental data (e.g., completions for 'fat head' queries). My basic click model cannot distinguish the difference.
- Each time EXP-7 is run or re-run, it would pollute the live log files and hence impact the subsequent online experiment (EXP-8).

The results of EXP-6 are presented in Section §4.7.2.

4.7.2 Results (EXP-6: QAC Ranking Approach Comparison)

As described in Section §4.7, this offline experiment compares the QAC ranking performance for the traditional and AI approach. The results of the comparison experiment are shown in Table 4.7.
Per the expectations outlined during the design stage, the AI approach's MRR ranking performance is substantially improved over the traditional approach. For

Table 4.7: Traditional versus AI approach to QAC ranking for 'fat head' queries for k=7. TF is used in the traditional approach; LM and LTR are used in the AI approach.

Ranking Approach	MRR@7 (%)
Traditional (TF only)	0.59
Using LTR and LM	**0.68**

k=7 in The University's ES QAC dataset, the AI approach's performance is 15.3% higher than using the traditional approach.

This performance gain suggests that the AI ranking approach (comprising LTR and the QACES LM feature) is indeed quantitatively 'worth the effort'. Based on the ranking performance literature for ecommerce [241] and Web Search, a performance gain was expected. This is now also proven for the domain of Enterprise Search. Additionally, the results reassure and confirm that the research is on the right track before proceeding to EXP-7 and EXP-8.

The MRR scores for selected k-value cut-offs for the three Search Demand Curve zones are listed in Table 4.8. The scores for the AOL WS and ES datasets are included for comparison with the University's ES dataset. Across all zones, the AOL scores are higher. It may be that this may be due to AOL using personalisation features. Singh et al. have achieved QAC MRR scores in the region of 0.485 for ecommerce site search [242], but any comparison is complicated as they omitted to break down scores into Search Demand Curve zones.

Table 4.8: Offline MRR scores for ES and WS ranking models, with a breakdown for the sections of the Search Demand Curve. The ES model comprises the LTR and LM methods.

Dataset / Zone	**MRR @1**	**MRR @7**	**MRR @10**
ES dataset			
Fat Head	0.60	0.68	**0.68**
Chunky Mid	0.32	0.34	0.36
Long Tail	0.21	0.24	0.24
All Zones	0.29	0.33	0.34
AOL WS dataset			
Fat Head	0.72	0.75	**0.78**
Chunky Mid	0.29	0.36	0.36
Long Tail	0.23	0.25	0.29
All Zones	0.33	0.36	0.36

4.8 EXP-7: QAC Ablation Study

.
LTR feature weightings provide a static view of importance and are sometimes misleading due to correlated (multi-collinear) features or noise in the data. Ablation involves removing a feature, which allows direct observation how its absence affects

model performance, confirming its true importance. The study is carried out for MRR@{1,3,7 and 10}. An ablation study also provides insights into the redundancy, and interactions of features.

- Hypothesis: The hypothesis is simply that the ranking features are of differing importance. Specifically, this study discovers the ranking features that make the biggest contributions to the model's performance. This 'experiment' can be regarded more as a study than an experiment in the strict sense.
- Baseline: For an ablation study, each collective feature set serves as a baseline before the systematic removal of a single feature. The performance of one feature is marked against 'all the rest'.
- Mechanism: This involves removing one feature from each iteration and performing the LTR training and testing steps again. The ranking performance differential is measured using the MRR metric for each feature. If a large decrease in QAC MRR scores is observed, the ablated feature is very important for the model.
- Sufficiency of Data: This experiment uses the same dataset as that used in EXP-6 where a learning curve graph is employed to estimate the required size of the ES QAC dataset size.
- Domain Knowledge: The procedures and protocols of an ablation study are well-defined. No particular domain knowledge is required as the ES dataset has already been created. Knowing which features are likely to make big or small contributions helps estimate the expect results (see below).
- Limits: A limitation of this study is that feature contributions in The University's ES dataset cannot be compared with those in the AOL WS dataset. This is because the AOL dataset only presents MPC and MFQ data (it does not include feature definitions or a ranking model).
- Expected Results: Figure 4.4 displays how a prefix and associated candidates are sorted as Term Frequencies for a sample selection of fields. From this example, it is expected that MFQ will be the dominant contributor to the ranking model. The example also suggests that it may be expected that the Anchor TF (a link label that content providers use to explicitly describe or 'tag' a document, see §4.3) will be a more prominent contributor than Content TF.

4.8.1 Implementation (EXP-7: QAC Ablation Study)

This section details the practical application of the experiment designed above (Section §4.8).

MRR Metric calculation As described in Section §2.3.10, the calculation of the RR metric is a prerequisite for MRR. RR is the multiplicative inverse of the rank of the first correct answer; 1 for first place, 1/3 for third place, or zero if the submitted query is not present in the ranked list. A file called mrr-list is created by the mrr.sh script, which parses the solr log files, collecting the position of selected candidate

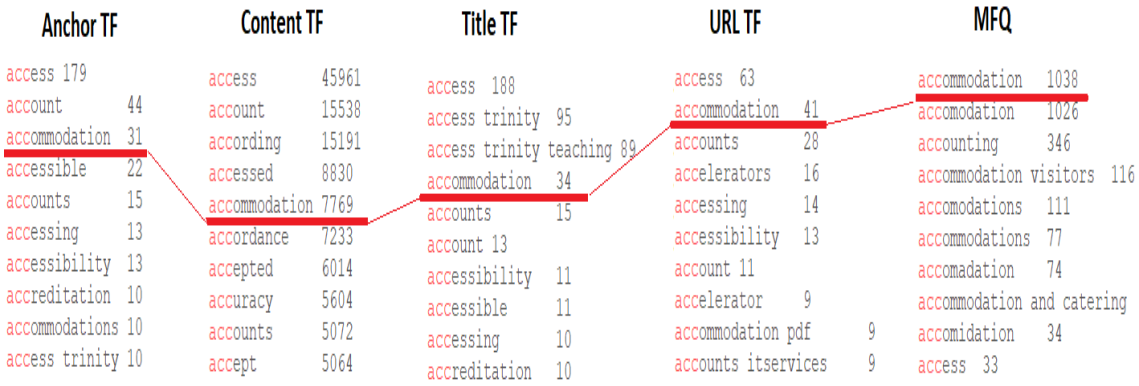


Figure 4.4: Relative Term Frequencies for various fields. The example prefix is 'acc', and the target (golden) candidate is 'accommodation'. This side-by-side graphic gives us an approximate expectation as to which fields are expected to be of most importance for the ranking model.

for a given prefix and assigns an RR score. For example, the [0,0,0,0,0,1] boolean representation means that, for a user's prefix, the correct candidate (sometimes referred to as the 'golden' or 'target' candidate) was positioned 6th in the list. Other examples of RR representations are shown in Figure 4.5. The graphic shows that the geo prefix shows the target candidate in second from the top of the list, therefore producing an RR encoding of [0,1].

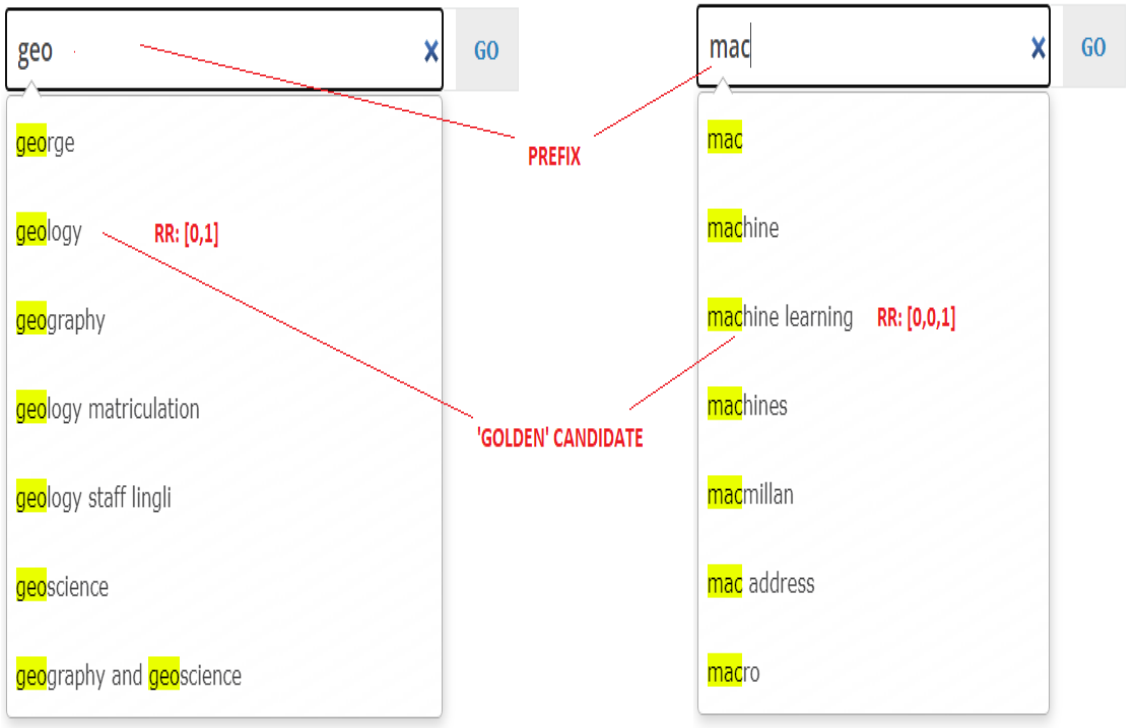


Figure 4.5: RR representations for MRR calculation. Example prefixes are 'geo' and 'mac'. According to the ground truth, the correct or 'golden' candidates are 'geology' and 'machine learning', respectively.

In the offline study, the MRR scores are broken down for the three zones of The

University's ES Search Demand Curve (Figure 3.3). The MRR scores are computed after three prefix keystrokes (i.e. p=3). For the purpose of the experiments, it does not matter the value of p, so long as the same value is used throughout. The SDC breakdown helps us distinguish the QAC ranking performance for the most popular queries versus more obscure queries in the long tail zone.

For the online study, MRR is computed for the users' last keystroke since this is where the user selects a suggested candidate for submission to the search engine. Chang [131] refers to this horizontal measure as MRR@last (which is a little confusing as MRR@5 is the accepted notation for the MRR score for the top five in the vertically ranked list). A score of zero is used where no candidates are offered or in the case where users fail to select any of the offered candidates.

MRR@k is calculated, where k is the number of offered completion candidates. I present results for k=10 (as is the norm) for the offline 'curl' experiments and I use k =7 for the online experiment (because The University's ES service presents a maximum of seven candidates.)
The results of EXP-7 are presented in Section §4.8.2.

4.8.2 Results (EXP-7: QAC Ablation Study)

The ablation analysis was undertaken to better understand the contribution of each of the different features of the QAC ranking model (for MRR@{1,3,7 and 10}). Figure 4.6 shows their relative contribution totals.

MFQ is the most important ranking feature, as its removal from the model results in a sharp decrease in MRR scores. This suggests that the collective query history is the best indicator of user intent.

Term Frequency is the next most important contributor. For this feature, all document fields (title, anchortext, content, etc) are combined into a single feature. This enables a comparison with Apache Solr, where this is the default, stand-alone, out-of-the-box QAC ranking function.

The AOL feature, which extracts pertinent completion candidates (and their associated frequencies) from the AOL WS dataset, makes the smallest contribution to the model. Perhaps this is because many of the suggestions in the feature are deprecated or simply because the feature was not generated from The University's ES index.

Surprisingly, the 'trend' feature (which extracts trending queries within an enterprise) did not fare better. Perhaps there was no sudden rise in interest (e.g., due to breaking news, recent events, or seasonal changes) in The University on the day that the dataset was created.

The contribution of the QACES jargon/terminology feature is similar to that of the trend feature. A big contribution was not to be expected, as few queries are expressed using jargon vocabulary.

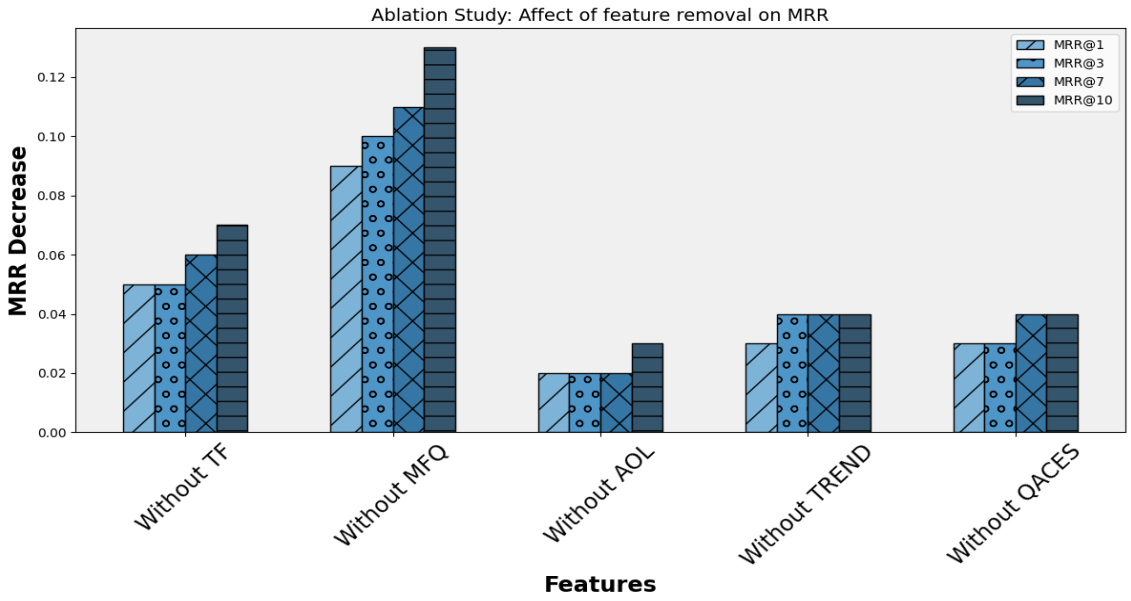


Figure 4.6: Ablation/leave-one-out analysis showing the contribution of individual features to the MRR performance across the QAC ranking model.

4.9 EXP-8: QACES Online A/B Test

While the ablation study in EXP-7 is useful for investigating model internals for offline evaluation, an online test can validate changes in a real-world, user-centric environment. This experiment evaluates the ranking performance contribution of the QACES LM feature by comparing two ranking models in an online environment. The first model (designated 'A') includes non-LM features only. The second ('B') also includes the QACES LM feature. The A/B test is undertaken on The University's live ES service.

- Hypothesis: The hypothesis is that adding the QACES LM feature to a LTR ranking model that uses non-LM features increases the QAC ranking performance, as measured by Mean Reciprocal Rank (MRR).
- Baseline: The LTR ranking model consisting of non-LM features is the baseline (i.e., the 'A' in the online A/B test) as per the mechanism outlined below.
- Mechanism: Before commencing the A/B test, an A/A test is performed. Also known as a null test, the A/A test establishes trust in the experimental platform. This null test involves splitting the search requests into two pools, as in a regular A/B test, but where B is identical to A. If the scripts to record search requests and compute metrics from the logfiles are functioning consistently, a t-test is expected to prove that any difference in MRR results is not statistically significant [228].

  Having established the integrity of the experimental platform, the A/B test is subsequently undertaken to compare the ranking performance of QAC candidates using two pools:

  – A: This is the *control pool* and encompasses all of the non-LM features described in Section §4.3.

  - B: This *treatment pool* includes all of A's features *plus* the additional QACES feature.

- Sufficiency of Data: Since the data in group A is independent of the data in group B, and the groups are normally distributed, an *unpaired two-sample t-test* is used to validate statistical significance with α set to 0.05. The t-test is computed based on the standard deviation of both populations (i.e. 'group'). If the p-value is less than α, the improvement is statistically significant. Data is gathered over a three month period initially, and with the potential to use a longer period if necessary.

- Domain Knowledge: The QACES feature is generalisable as it detects jargon/terminology independently of domain knowledge. As per the earlier example in Section §4.5, the algorithm behind the QACES feature detects that candidates like 'schols' and 't-card' are enterprise-specific words, phrases, expressions, and idioms that diverge from those universally familiar or understood outside of The University.

- Limits: The volume of data required to capture sufficient prefix-candidates that QACES detect as jargon requires time to collect. This period is difficult to estimate in advance, and three months is chosen to start with. The A/B test depends on users typing prefixes that produce candidates classified as jargon/terminology.

- Expected Results: I expect that adding the QACES feature will increase model performance. As jargon/terminology is only a small fraction of QAC suggestions overall, I expect that the contribution will be modest. Since the XGBoost LTR framework includes regularisation techniques with cross-validation, adding the QACES feature can only make a zero—or positive contribution to a model's performance.

The results of EXP-8 are presented in Section §4.9.2.

4.9.1 Implementation (EXP-8: QACES Online A/B Test)

This section details the practical application of the experiment designed above (Section §4.9).

QAC Online Implementation A load balancer is configured as shown in Fig 4.7. It consists of two servers running the Apache Solr application in an active/active configuration, with a 50% traffic allocation to both the Control and Treatment pool. The load balancer has a session persistence (stickiness) parameter enabled so that the suggestion candidates are presented by the same back-end server that executes the submitted query. Stickiness enables the load balancer to consistently route the same client's requests to the same back-end server based on the IP address. This ensures that each log file has a complete record from the first keystroke until the query is submitted.

In addition to enabling the stickiness parameter, it was discovered that X-Forwarded-For (XFF) header forwarding was also required to enable sticky sessions via Apache ProxyPass.

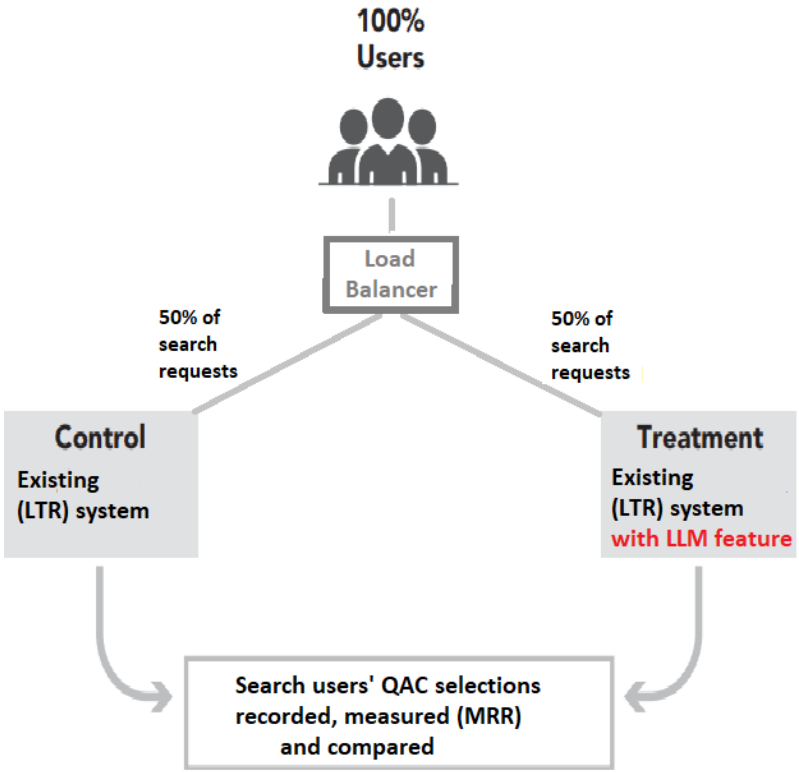


Figure 4.7: Load Balancer in active/active mode used for A/B testing.

Before proceeding the the A/B test, an A/A (null) test was undertaken to establish trust in the load-balanced platform. Both of the Apache Solr instances behind the load balancer were configured identically (i.e. using the same features and weights). The `mrr.sh` script recorded the MRR scores on a day-by-day basis for 30 days on both pools. The standard deviation was computed. After several days, it was noted that the MRR scores increasingly converged to produce near identical results on both instances. A two-tailed independent sample t-test was used to demonstrate that any slight difference between the MRR scores were not statistically significant.

Having established the integrity of the test platform, the A/B test is subsequently undertaken. Figure 4.7 shows how the load balancer is configured to send half of incoming search sessions to the Control pool and the other half to the Treatment pool. The result of the A/B test are presented in Section §4.9.2.

Implementation Resources Deploying the AI ranking model to a production ES service requires additional resources and considerations beyond those associated with offline experimentation with established datasets.

Many organisations, where search is not at the core of the business, may not have access to a 'relevance engineer' with knowledge of AI ranking methods. Relevance engineering generally falls outside of the generic skill sets of an IT Services department. In my case, I served in several organisational roles, including as relevance engineer, systems administrator, network administrator and as a sponsor of the deployment 'change' via the local change control process. This resource would not have been necessary if a traditional ranking approach had been adopted.

As the world around us changes, the relevance of a candidate for a given prefix also changes, over time leading to relevance drift. This can be mitigated only by periodic re-training of Learning to Rank models [216]. In deployments managed by an IT Services group, the project and budgetary structures are generally designed to 'deploy and forget'. Enterprises are thus in danger of not taking such ongoing maintenance costs into account when choosing machine learning methods. The prolonged commitment would not have been necessary if a traditional ranking approach had been adopted.

During the course of the deployment of the ranking models, the EU AI Act 2024 was published. The provisions of the act will become applicable in 2026. As discussed in Section §2.6, the act stipulates that public institutions must adhere to the same standards as private entities, including conducting conformity assessments, implementing risk management systems, and ensuring transparency in AI system operations. Consequently, the decision to use LTR and LM methods means that there will be further (as yet unplanned and uncosted) work in the future. This obligation would not have been necessary if a traditional ranking approach had been adopted.

The general qualitative question of the additional implementation effort required for the AI approach (for both QAC and search results) is discussed in 5.6

4.9.2 Results (EXP-8: QACES Online A/B Test)

Two ranking models were compared. The first (A) included heuristic features only. The second (B) included everything from (A), but additionally included the QACES feature. The ranking score for each ranking model is presented in Table 4.9. The A/B test was undertaken on The University's live Enterprise Search service. The model was tested for initially for three months on 100,000 queries, which resulted in a statistically significant increase in MRR score by +3.8% with a p-value < 0.05.

Table 4.9: A/B test results for ranking models with the percentage change in MRR score after implementation of the QACES feature.

Feature	**MRR@10**
Heuristic only.	0.219 ± 0.01
With QACES.	0.227 ± 0.02
Percentage change.	+3.8%

The observed MRR@10 score in the online evaluation is **0.227**, which is substantially lower than the value of **0.34** obtained in the offline study. This difference is to be expected because the online metric reflects real user behaviour suggestion abandonment. Such behaviour can lower click-based evaluation scores. The online metrics do not ignore this behaviour; they incorporate it directly.

A possible cause is that search users type with speed and urgency and opt not to take the time to engage with the QAC interactive search offering, even where a candidate was an exact match to the query. Li et al. refer to this as 'skipping' in WS [235].

In the context of ES, I call this phenomenon 'QAC abandon' and speculate that it frequently occurs in the case of navigational searches (i.e. returning users who already have a good idea of what document they are looking for).

QAC abandon is not recorded in the offline study (since the submitted query exactly matches the candidate, the logic in the scoring script assumes that the candidate was selected). Conversely, for the online evaluation, an RR score of zero is awarded if the matching candidate is not positively selected. One idea to obtain a relative sense of QAC abandon would be subtract the online MRR score from the offline score, but only for those P-C pairs that exist in both.

4.10 Validation

The experimental designs outlined in Section §4.6 include several intrinsic validation checks for robustness, stability and generalisability. For example, in EXP-8, a robustness A/A test is performed before proceeding to the A/B test. Moreover, both offline and online experiments are used to cross-check results.

An additional and practical verification and validation test to ensure that all of the various features (i.e. fields in the Apache Solr index) are being utilised by QAC is to add a unique 'tracking' string that immediately conveys the source of the candidate. For example, "qactracker[1..n]", where n is the number of features as shown in Figure 4.8.

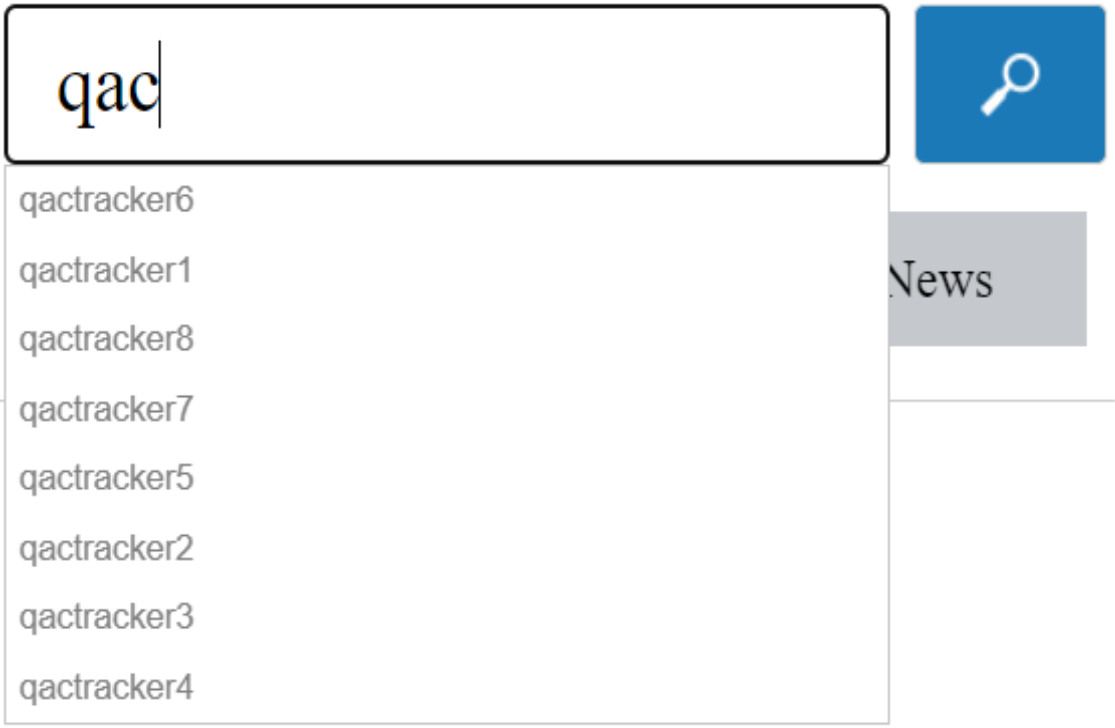


Figure 4.8: QAC validation. For the prefix 'qac', each feature is represented by the 'qactracker' candidate followed by a number that coveys the source feature (field).

This tracker validation method performs a simple gauge of both the recall and ranking aspects of QAC for ES:

- Recall: the instance number appended to each string tells us the origin of the QAC candidate. If there are nine features, there should be nine candidates for the appropriate prefix.

- Ranking: the order in which the candidates are ranked is a reflection of the weightings that are assigned via LTR. In Figure 4.8, feature number 6 (identified by qactracker6) has a relatively higher LTR weighting than, say, feature number 3.

The speed and simplicity of this tracker method are appropriate for an Enterprise Search environment, where search may not be the core focus of an organisation and where it is unlikely that there is a dedicated Centre of Search Excellence (CSE) comprised of staff with specialist expertise in fields like LTR who are constantly monitoring the QAC subsystem [54]. A bash script called `qactracker.sh` is used to populate the fields of the Solr index. While the script is custom-written for Apache Solr, it should be easily adaptable for any IR platform. The code for `qactracker.sh` and the accompanying 'readme' file are published on GitHub[16].

The QAC tracker tool has two important limitations. Firstly, if the ranking model contains more features than the number of candidates presented via the drop-down (usually ten or seven), those features with a lower LTR weighting would not be visible. This is easily overcome by typing the full prefix length (e.g., qactracker17). Secondly, the presented order of the tracking strings merely conveys the *relative* ranking order; it does not give any *absolute* information regarding the LTR weighting.

To retrieve the absolute weighting for each feature, a more sophisticated validation and verification of the LTR implementation and operation is required (Listing 4.3).

```
$ curl http://localhost:8983/solr/$CORE/suggest?\
> q=\${QAC_STRING}\
> &suggest.count=10\
> &suggest=true\
> &sort=weight+asc\
> &suggest.dictionary=suggester_all\
> &wt=json\
> | grep -e term -e weight
            "term":"qactracker6",
            "weight":600,
            "term":"qactracker1",
            "weight":101,
            "term":"qactracker8",
            "weight":80,
            "term":"qactracker2",
            "weight":8,
            "term":"qactracker3",
            "weight":50,
            "term":"qactracker5",
            "weight":1,
            "term":"qactracker7",
            "weight":1,
            "term":"qactracker4",
```

[16] https://github.com/ColinDaly75/ES-Ranking

```
            "weight":80,
            "term":"qactracker9",
            "weight":45,
```

Listing 4.3: REST API command to query the Solr Index to retrieve and verify the LTR weighting (in teal font) for each feature.

The weight figures above can be validated against the LTR weightings generated offline by the RankLib framework. Moreover, the order in which `qacktraker` terms are presented in Listing 4.3 should match the ranking order in Figure 4.8.

4.11 Conclusions

Chapter 3 showed the extent to which LTR and LM techniques, tailored to enterprise content and information needs, can boost the ranking and recall of ES search results.

Similarly, in this chapter, the custom requirements for QAC when tailored to enterprise content and context are outlined, followed by a series of experiments (again using LTR and LM techniques) to determine the extent of ranking performance improvements.

The first experiment (EXP-6) demonstrates that the AI approach produces better ranking scores than the traditional approach. Although prior research in e-commerce and web search suggested this performance improvement, it has now been empirically demonstrated for Enterprise Search. EXP-6 advances research contribution RC-2.

A natural shortcoming of EXP-6 is the inability to answer the subjective and qualitative question, 'Is it worth it?'. In other words, although the QAC ranking model performance gain is substantial using AI, it must be weighed against the additional skills, resources and effort required for an organisation to implement (and periodically maintain) it.

EXP-7 is an ablation study to determine each feature's MRR contribution. The study also demonstrated the implementation mechanism of a feature-rich QAC ranking model on the Apache Solr platform. The ranking model comprised features derived from the ES corpus, the ES log data and data from an external WS dataset. The features are crafted and described in terms of suitability for enterprise content. Feature weightings were computed using the XGBoost and RankLib LTR framework. The ablation study shows that MFQ and TF are the most important ranking features. EXP-7 advances research contribution RC-1.

The final experiment (EXP-8) proved that, in an A/B test, adding the QACES LLM-based jargon/terminology ranking feature to the baseline heuristic LTR model resulted in a statistically significant improvement to QAC ranking performance (MRR score) on The University's live ES service. This experiment advances RC-5.

EXP-8 uncovered several challenges. Apart from the repeatability issue when using the GPT-4 LLM, the online experiment depended on users submitting queries that produced 'jargony' QAC candidates. Moreover, an analysis of the search logs suggested that many searchers opted not to take the time to engage with the QAC

'interactive search' offering, even when the golden candidate is offered. In WS, this is sometimes referred to as 'skipping' [235]. For ES, I call this phenomenon 'QAC abandon' and it offers an interesting new direction for the field of auto-completion.

Utilising offline (EXP-6, EXP-7) and online experiment (EXP-8) allows for a thorough evaluation a model's performance in both controlled and real-world scenarios, ensuring robust and user-aligned improvements.

The conclusions and observations drawn from the ranking of QAC candidates cannot be examined in isolation from those of Chapter 3. For example, the QAC ranking should complement and reinforce the Search Results ranking. Similarly, when ranking both Search Results and QAC completions, there are inherent limitations to using the 'wisdom of the crowds' as training data. Hence, a detailed and holistic synthesis of the experiments and associated results will be undertaken and discussed in Chapter 5.

5
DISCUSSION

This chapter contextualises and synthesises the research and its findings, emphasising their significance, implications, and limitations. It also examines how the results contribute to existing knowledge and identifies directions for future research.

It starts with revisiting the context of Enterprise Search (ES) and the prerequisite creation of a new dataset. The discussion then integrates experimental findings from the evaluation of Learning to Rank (LTR) and Language Modelling (LM) methods in ES for both search result ranking and Query Auto-Completions (QAC) ranking. The chapter also highlights key challenges encountered during implementation, including dataset constraints, biases, and practical barriers to adopting the AI approach for organisations. Finally, in Section §5.9, some future research directions and opportunities are explored.

5.1 Revisiting Enterprise Search

In §2.1.2, it was noted that academia has many definitions for Enterprise Search. At the beginning of this study, the broadest definition of ES was adopted, encompassing both internal and external-facing content, such as intranets, file shares, corporate directories, document management systems, as well as content more typically associated with Web Search (WS), such as pages on an organisation's website.

When this research commenced six years ago, this definition was expedient as it matched the hybrid content available on The University's site. While the primary differences between ES and WS content pertain to file types, hyperlink structure, size and jargon/terminology, this study also documents numerous other smaller distinguishing factors (as recorded in Appendix A).

An unexpected effect of this longitudinal study has been a growing awareness of the importance of the ES service to The University. The initial recruitment and engagement of judges for Q-D annotation promptly identified a designated individual responsible for the ES service. Although the research focused on ranking, the initial outreach meant that there was now an answer to questions like 'Who should I talk to if I want my directory indexed?', 'Why are my pages not showing near the

top of the search results?' or 'Why does it take several days for new content to be indexed?'. While these questions confuse the roles of 'relevance engineer' and 'search engineer' [61], there may be little room for distinction in busy organisational environments.

It is expected that this awareness (of ranking, the AI approach, and the ES service in general) and enhanced collaboration experienced at The University would also apply to other organisations. So perhaps a more practical revised definition for ES would transcend content type, to also include this collaborative aspect between parties responsible for search and content provision.

5.2 Using The University's Dataset

As outlined in Section §2.4.5, numerous extensive and publicly accessible datasets are available for WS. However, as consistently emphasised throughout this study, ES and WS content exhibit fundamental differences.

Test collections based on ES content are hard to come by [10]. This may be due to security or privacy concerns. Organisations might decline permission to publish the results of any research carried out on their ES service. Most researchers with access to an intranet have access only to that of their own institution [43].

For all of these reasons, it was considered necessary to create a new dataset (`ENTRP-SRCH`) with 2544 Q-D pairs, and specifically designed to improve ranking of the most frequently queried topics. Moreover, as the dataset is formatted as a feature vector array for use in LTR frameworks, it contains no sensitive organisational or personal information and has been approved for publication for all to use (Research Contribution RC-3).

Had the study depended exclusively on a previously published dataset, our ranking model might have been a 'black box' with opaque reasoning behind ranking and feature importance. Using a 'home-grown' dataset allowed for more transparent inner workings, leading to understandable explanations for feature importance, ranking behaviour and enabling comparison with the traditional ranking approach.

The scale and composition of the datasets used in this research were necessarily constrained by practical and ethical considerations. As the study was conducted within a live organisational environment, the volume of data that could be collected was limited by institutional approvals, privacy safeguards, and the need to avoid intrusive monitoring of individual users. Resource constraints (particularly in terms of personnel, funding, and time) also influenced the achievable dataset size and scope of annotation. While larger-scale datasets may have enabled broader generalisation, the dataset assembled for this work represents what was realistically and ethically feasible within the context of a single organisation ES service. These limitations are common in applied information retrieval research conducted under real-world conditions and should be interpreted as reflecting a balance between methodological rigour and organisational responsibility.

Another drawback of creating an in-house, human annotated, relatively small dataset, with a specific design purpose, is that it may not perform well for other ranking tasks. For example, the 'JARGES' feature, which was designed to detect and up-rank jargon/terminology (EXP-5), failed to demonstrate a statistically significant increase in ranking performance. Had the dataset been designed with a greater breadth (i.e., the 'shallow judgement' dataset design strategy recommended by Yilmaz et al. [167]), it is possible that JARGES would have resulted in an improved ranking score. This dataset limitation was avoided for the QAC experiments, as Most Popular Completions (MPC) data was used as the 'ground truth' for annotations of the Prefix-Candidate (P-C) pairs. This generated QAC dataset contained 902,412 P-C pairs, and EXP-8 proved that this was sufficiently diverse to capture an improvement in ranking performance for jargon terms.

The generalisability and domain transfer of experimental findings based on `ENTRP-SRCH` to other domains and contexts are discussed in Section §5.7.

5.3 Implications of Experimental Results

The implications of the experimental results are discussed in terms of search result ranking and QAC ranking before 'bringing it all together' to discuss the overall combined implications.

5.3.1 Implications for Search Result Ranking

EXP-1 compared the ranking performance of traditional and AI approaches using LTR. The experiment was designed to answer RQ-1 by meeting OBJ-1. For nDCG@5, the LTR method performs 5.85% better than BM25 with field boosting (the full results of EXP-1 are presented in Section §3.7.2). A prerequisite for measuring performance was the annotation of Q-D pairs and the subsequent creation of the publicly available `ENTRP-SRCH` dataset, which fulfils Research Contribution RC-3. For search results ranking, the LTR model consistently achieved higher nDCG@k and MAP scores where the cutoff, k, is ≤ 10. This affirms the LTR model's ability to perform better on the first page of search results (RC-1, RC-2). Surprisingly, the performance advantage of the LTR method is reduced and even reversed when k is 20 and 100, respectively. For most users, however, scores for $k \geq 10$ may not be critical. According to a 2014 Google organic desktop search study, it was found that just 6% of end-users navigated to pages two and three [183, 206]. Moreover, an analysis of The University's search logs indicates that this figure is less than 2%.

EXP-2 was an ablation study conducted to detail the contribution of each feature in the LTR model. This experiment elaborates on EXP-1 and was designed to further investigate RQ-1. The results of EXP-2 are presented in Section §3.8.2. As expected, BM25 and field boosting were the biggest contributing features. Surprisingly, the recency feature (i.e., how recent is the document's creation or modification timestamp) contributed little to the LTR model. This may be because many ES documents types do not have any ascertainable timestamp. As described in Section §3.7.1, several approaches were used to extract timestamps. However, documents with extensions such as TXT or CSV do not have built-in metadata for a creation

date. In these cases, the median value (calculated using timestamps of all documents in the index) was imputed to the document. This imputation strategy seems acceptable when a small amount of data is missing. For The University's ES content, an imputed timestamp was assigned to nearly 30% of documents, potentially diluting the contribution of the recency feature.

More generally, the diverse nature of ES content means that similar limitations are encountered for LinkRank (e.g., an MS Word document stored on a network share might not contain any hyperlinks) and field boosting (e.g., MS Excel documents generally do not have distinguishable title and content fields).

EXP-3 explored the use of implicit feedback (click-through data) as a substitute for human relevance judgements and was designed to meet objective OBJ-2. The results of EXP-3 are presented in Section §3.9.2. EXP-3 was not the central focus of Chapter 3, and can be considered as an auxiliary experiment because it supplements the main study by reinforcing the approach adopted to build the `ENTRP-SRCH` dataset. Statistical analysis showed a 'strong correlation' between CTR and human relevance judgement, meaning organisations can safely use CTR in place of costly human annotations (RC-4). Perhaps more interestingly, a subsequent analysis of correlation outliers revealed examples of position bias and organisational bias. These biases are discussed further in Section §5.5.

EXP-4 examined the assumption that The University's microsite editors were sufficiently qualified to assess the ranking order of their documents for a given query, to the extent that only a single rater per topic was employed. The EXP-4 Inter-Annotator Agreement (IAA) results are presented in Section §3.10.2. Similar to EXP-3, EXP-4 can be considered a confirmatory experiment as it was conducted to assess the reliability of the `ENTRP-SRCH` dataset (which was created to meet OBJ-1).

The IAA results reveal that while human relevance judgements exhibited moderate agreement levels, variability among annotators highlighted subjective differences in document assessment. This underscores the need for diverse annotator perspectives (or alternative feedback mechanisms like CTR), to enhance the reliability of training data. The practical barriers to soliciting and enrolling annotators for ES are described in Section §5.6.

EXP-5 used the JARGon for Enterprise Search (JARGES) innovative LM algorithm / feature, which was designed to detect and uprank search results containing organisational jargon/terminology. The experiment attempted to answer RQ-2 by meeting OBJ-3. The results of are presented in Section §3.11.2. Unfortunately, the results show no statistically significant improvement for nDCG@10 at the 5% level ($p > 0.05$). In an accompanying qualitative study, the 'Jargon Buster' PDF file was used as an independent reference for assessing both the precision and the comprehensiveness of the decoded jargon terms (i.e. synonyms). Of the 126 jargon terms defined in the PDF, 53 also appear in our generated list. Furthermore, a cursory inspection of the synonyms generated for these terms appears to reflect the correct semantic context. This qualitative analysis of the generated synonyms reveals promising results for recall as a foundation for semantic search.

The lack of a quantitative ranking improvement is attributed to the ENTRP-SRCH dataset's limited query diversity and scarcity of jargon rich Query-Document pairs, which deflated the performance score of the JARGES feature. This attribution is based on the quality of the generated synonyms for jargon terms (the final step of the algorithm). The obvious way to prove this would be to expand the dataset, so that it is sufficiently diverse and large enough to capture the nuances of enterprise specific terminology (this is discussed further in Section §5.9).

5.3.2 Implications for QAC Ranking

EXP-6 compared the QAC ranking performance of traditional and AI approaches using LTR and LM. The experiment was designed to answer RQ-3 by meeting OBJ-4. The results of EXP-6 are presented in Section §4.7.2. As per expectations (based on the literature for ecommerce and Web Search), the AI approach produces significantly higher ranking (MRR) scores than the traditional approach (RC-1).

The MRR scores achieved using the ES model trained on The University's data are about 10% lower across the board (fat head, chunky middle, long tail) than those achieved for WS (AOL model), as shown in Table 4.8. At first glance, this is disappointing, particularly as the AOL model was generated in 2006. Nonetheless, as has been constantly emphasised throughout this study, ES and WS are two very different domains; comparing results between the two might be like comparing apples and oranges.

EXP-7 was an ablation study conducted to determine the performance contribution of each feature. This experiment elaborates on EXP-6 and was designed to further investigate RQ-3. The results of EXP-7 are presented in Section §4.8.2. The ablation showed that Most Frequent Queries (MFQ) is the most important ranking feature for QAC. This suggests that the collective query history is the best indicator of user intent.

The 'trend' feature (which extracts trending queries within an enterprise) generated a small contribution to the overall MRR score. Perhaps there was no sudden surge in interests (e.g., due to recent or seasonal events) within the organisation during the period when the dataset was created. More likely, an organisation's task-focused searchers are unlikely to exhibit the meme-like cultural behaviour typically generated by users for/on social media platforms.

EXP-8 was an A/B test to measure the effectiveness of the 'QACES' LLM-based jargon/terminology innovative ranking feature. The experiment was designed to answer RQ-2 by meeting OBJ-4. The results of EXP-8 are presented in Section §4.9.2. The addition of the QACES feature resulted in a small but statistically significant improvement to QAC ranking performance (MRR score) on the organisation's live Enterprise Search service (RC-5).

Perhaps the most significant challenge uncovered during EXP-8 was 'repeatability' when using the GPT-4 LLM for near-synonym extraction. This points to GPT-4 not being the appropriate tool for the experiment. This is discussed in Section §5.4.

Finally, the search logs of The University's ES service recorded that many searchers opted not to take the time to engage with the QAC 'interactive search' offering, even when the golden candidate was offered. This phenomenon was uncovered when comparing the results of online and offline studies. Within WS systems, this user action is sometimes characterised as 'skipping' [235]. In the context of ES, I call this phenomenon 'QAC abandon', and it is potentially a serious problem for ES where QAC helps users avoid submitting queries that produce no results (a potentially common occurrence, as an ES corpus is small compared to those used by commercial WS engines). QAC abandon offers an interesting new direction for the field of auto-completion (e.g., why do searchers abandon? Is it only the very fast typists who abandon? Or does abandon correlate with latency in the presentation of candidates?. Could the abandon rate be used as a new metric, perhaps in conjunction with MRR?).

5.3.3 Combined Implications

Before the emergence of artificial intelligence, the academic community had, for decades, extensively explored traditional ranking approaches in the field of information retrieval. Out of an abundance of caution, independent experiments were carried out for a) search results and b) QAC. These experiments ensure the robustness of the assumption that LTR and LM can yield superior ranking scores.

EXP-1 and EXP-6 can be seen as a pair of experiments demonstrating that the LTR method outperforms the traditional approach. Similarly, experiments EXP-5 and EXP-8 reinforce each other as a pair that shows how LM can be used either to improve ranking or recall of jargon/terminology. While LTR and LM are routinely used in commercial search engines [25] and ecommerce [70], their application and appraisal for ES had hitherto not been sufficiently explored. It is hoped/anticipated that this study will elevate the adoption and integration of AI methodologies in the field of ES to match existing adoption in WS and ecommerce.

The field of search result ranking dates back to the early information retrieval systems of the 1960s. The field of QAC is relatively newer and dates back to the late 1990s. This difference in the maturity of the two fields may be why the performance gains achieved in the QAC ranking have been higher than those achieved for the search results ranking (EXP-1), where decades of additional research have reduced the potential room for improvement.

5.3.4 Relationship Between JARGES and QACES

Although both JARGES and QACES operate within the broader ES ecosystem, the two features are designed to address fundamentally different problems and are therefore not interchangeable. Their distinct objectives, inputs, and outputs position them at different stages of the search pipeline, with JARGES functioning as a semantic search enhancement and QACES as a query auto-completion framework.

JARGES is concerned with identifying and interpreting the organisation specific terminology that emerges within an enterprise corpus. Its approach is term centric. By measuring the divergence between TF–IDF weights in the enterprise corpus and

a reference corpus of general English, it quantifies the degree to which individual terms are specific to the organisation's internal discourse. Terms that exceed a divergence threshold are designated as *jargon*, and a separate decoding stage uses a fine-tuned transformer model to generate contextual synonyms for these terms. JARGES does not generate query completions, nor does it rank suggestions for a given prefix. The JARGES role is to augment the linguistic signals available to the downstream retrieval component.

By contrast, QACES is designed as a query auto-completion framework. Its focus is prefix centric. Given a partially typed query, the goal is to generate and rank a set of completion candidates that reflect likely user intent. The output of QACES is a ranked list of candidate queries presented interactively to the user during typing.

Mathematically, JARGES operates over the term space ($t \in \mathcal{V}$), where $\mathcal{V}$ denotes the enterprise vocabulary, to model linguistic divergence within the corpus. In contrast, QACES operates over the query auto-completion space ($p, c \in \mathcal{Q}$), where p represents the user-typed prefix and c candidate completion, estimating the conditional ranking $P(c \mid p)$ that reflects the likelihood of each completion candidate being selected.

The two features are consequently not interchangeable. Their problem formulations differ as JARGES ranks Q-D pairs, whereas QACES ranks P-C pairs. Their inputs and outputs are distinct, as are their evaluation criteria. JARGES is assessed using intrinsic measures (e.g., whether it captures genuine organisational terminology) and extrinsic effects on retrieval metrics such as nDCG. In contrast, QACES is a feature for QAC, and is therefore evaluated through the MRR metric.

Despite these differences, the two features are complementary, as they both have the goal of dealing with jargon within organisations and their respective roles within the ES pipeline are distinct but mutually reinforcing.

5.4 Methodological Implications

This section reflects on the research methods used in the study and their broader significance for the field of ES.

The research questions and objectives specified in Chapter 1 are, at first glance, entirely quantitative in nature. However, the challenge of deploying AI methods like LTR and LM in organisations (where search may not be at the core of the business, and organisational resources and skills may be limited) necessitates some accompanying qualitative analysis. An example of fusing quantitative and qualitative methods is to be found in EXP-1 and EXP-6, where quantitative improvements in ranking performance are judged against the additional effort, skills and resources required to implement them. This trade-off is captured in the subjective question 'Is it worth it?' as mentioned in Section §3.7.2 and §4.7.2 and discussed further as a 'barrier' to AI-approach adoption in Section §5.6.

In a 2024 paper, Baidu described LTR as its "standard workhorse" [27] with respect to the ranking of search results. Not only does LTR have no theoretical upper limit on the number of features that it can train on, it can also incorporate LM-based features like it would for a regular feature. This study, therefore, agrees with the workhorse designation and takes it a step further, where LTR is used not just for search results but also for ranking QAC candidates.

The use of the Search Demand Curve (SDC, §3.4.2) for query and prefix selection is perhaps an unconventional method more likely to be found in commercial SEO studies than in information retrieval literature. In this study, the SDC is used twice. Firstly, it is used as a criterion for optimum query selection for Q-D annotation. Secondly, it is used for categorising Most Popular Completions (MPC) for query auto-completions. The rarity of using the SDC is probably because most ranking studies involve off-the-shelf datasets, where the original selection method of queries for Q-D annotation (or prefixes for P-C annotation) is an obscured or long-forgotten detail.

To maximise representation, the queries selected for Q–D annotation are drawn from the top 18.5% of the SDC search volume. Focusing on this segment ensures that annotation effort is concentrated on queries that collectively account for the majority of user activity. However, this choice necessarily under-represents rarer, long tail queries, which are often more heterogeneous and potentially more challenging for a ranking model. This was not an objective in itself but a practical constraint: long-tail queries occur infrequently, making them difficult to judge reliably, and many lack sufficient click or usage data to support model training.

The problem of repeatability using GPT-4 for synonym extraction for the QACES feature in EXP-8 might have been foreseen. One of the attractions of GPT-4 is that it is not inclined to repeat itself (even with low-temperature settings). In hindsight, using Google's BERT LLM may have been more appropriate. BERT is primarily designed for *understanding* and works on a masked language modelling objective [243]. Moreover, BERT is not a generative model (like GPT-4) and is likely to be more repeatable under controlled experimental conditions. The emergence of generative AI has stirred significant enthusiasm within the research community, and it is possible that the selection of GPT-4 for EXP-8 was influenced, at least in part, by this prevailing excitement. Future research might involve replacing GPT-4 with BERT and re-conducting EXP-8.

Two complementary methodological innovations were deployed to tackle jargon at different places in the search loop. First, JARGES treats jargony terms as a contrastive-salience signal (a high TF-IDF in the enterprise corpus but low TF-IDF in a general corpus). Matching documents can be boosted at result ranking time. Second, QACES uses a set-dissimilarity signal (Jaccard distance) to prioritise domain-relevant completions at the prefix stage, nudging users towards in-vocabulary phrasing before they submit the query. These two innovations act at different places (document ranking vs candidate ranking) and rely on different signals (TF-IDF differential vs Jaccard), as summarised in Table 5.1.

Table 5.1: Contrasting methodology used for the JARGES and QACES LM features.

Aspect	JARGES	QACES
Signal definition	Jargony as *contrastive salience*: high TF-IDF in the enterprise corpus and low TF-IDF in a general-language corpus.	Jargony as *set dissimilarity*: larger Jaccard distance between candidate/prefix token sets and a reference vocabulary/context.
Jargony score	A number derived from subtracting one TF-IDF score from another.	A number representing the Jaccard distance between sets.
Reference data	Two corpora (enterprise vs. general) for a contrastive score.	A reference token set (e.g., enterprise lexicon, accepted completions, or context tokens).
Interpretability	"Frequent here, rare in general English" (linguistic salience story).	"Low overlap with generic tokens" (distance/overlap story).
Place in 'Search loop'	Document ranking for a given query.	Candidate ranking for a given prefix.

5.5 Position and Organisational Bias

While auxiliary EXP-3 was designed to explore the use of implicit feedback (click-through data) as a substitute for human relevance judgements, the analysis of correlation outliers revealed examples of position bias and organisational bias.

5.5.1 Position Bias

As described in Section §2.4.8, users surveying search results will likely click on the first sufficiently promising abstracts without further exploration of links below that result [178,179]. Results are clicked on not necessarily because they are more relevant, but because they appear earlier in the list. This innate time-saving mechanism is known as position bias.

Failure to mitigate position bias in click data can lead to 'naïve' models that have been trained to learn *position* rather than *relevance* [179]. This omission would have the effect of reinforcing the existing ranking positions. However, as the `ENTRP-SRCH` dataset was constructed with explicit Q-D annotations, this scenario did not arise. However, position bias detection in the CTR data is interesting as it implies that a complete dependence on CTR for relevance feedback (i.e., ground truth) would require bias mitigation, such as inverse propensity scoring. While much research has been carried out on de-biasing in the domain of WS, little has been carried out for ES, which could be an opening for potential future study.

5.5.2 Organisational Bias

Organisational bias is inherent to the institutional approach when employing the organisation's subject area experts for Q-D annotations(§2.4.8). It stems from the collective attitudes, cultural norms, and structural policies that inadvertently influence employees to favour certain documents above others.

In EXP-3, the example given was that of the homepage of a VIP / celebrity individual in the organisation, where the page was annotated as 'irrelevant' (for the given query). This judgement dampens favouritism and diminishes the democratic preferences of end users. In this case, the expert judges are subject to organisational bias, where the 'wisdom of the crowds' is discarded in favour of an institutional outlook. The wisdom of the crowds cannot be depended on if the crowds are not wise to new documents, products (in the case of e-commerce), or jargon/terminology, if members of the crowd are new to the organisation.

A potential mitigation for organisational bias would be to use a hybrid annotation scheme, where Q-D scores are formed by a mixture of CTR and expert judgements. For example, each feedback method could have a 50/50 input to the final annotation score. Alternatively, the ratio could be an adjustable variable, much like the 'temperature setting' in generative AI. This ratio would likely have the effect of dampening both position and organisation bias, and could be an interesting avenue of future research.

5.6 Implementation Challenges and Barriers

The challenges and barriers associated with implementation can critically influence the decision to adopt the AI approach or cling to the traditional ranking approach.

Implementation challenges and barriers hold particular significance in the domain of ES. Unlike commercial search providers or ecommerce sites, an ES service search is unlikely to be a central business objective, and the allocated organisational resources and skills may be limited.

Transitioning from traditional ranking methods to LTR and LM introduces multiple challenges beyond algorithmic ranking performance. Below is a summary of the main challenges and barriers identified during this research.

- Academic literature pays scant attention to the amount of effort required to **construct the machine learning pipeline** in its totality. The pipeline includes tasks such as identification of relevance signals, converting signals to ranking functions and subsequent features, labelling, extracting features from a corpus, data pre-processing, and training and evaluating a model.

- The effectiveness of LTR is contingent on **high quality feature engineering and selection**. Constructing the LTR features requires expertise in both machine learning and enterprise search, which may not be readily available in all organisations. According to White [60], a Web search engine like Google relies on 20,000+ engineers to look after the system, and in ES, there is "rarely more than one lonely person with the responsibility for supporting the search application and making sure that it is tuned to meet user requirements".

- To create an LTR dataset using explicit feedback, a lot of effort is required to **solicit Q-D human relevance judgements** from subject matter experts in the organisation. In the field of ES, crowd-sourcing platforms, such

as Amazon's Mechanical Turk(AMT), cannot be used because external annotators cannot access the highly technical blueprints, confidential documents, network file shares, etc., behind an organisation's firewall.

- As the world around us changes, the relevance of a document for a given query also changes over time, leading to relevance drift. Unlike traditional methods (like BM25), models based on LTR and LM require periodic retraining to mitigate relevance drift. Large-scale enterprises, such as Google, frequently update their ranking models based on user feedback, whereas many smaller organisations operate under a 'deploy and forget' model that may not sustain optimal performance over time. It is possible that these smaller organisations have not planned sufficiently for these **ongoing maintenance costs**.

Prior to the introduction of BERT in 2018, and the transformer revolution, language models were trained on enormous datasets without leveraging pre-trained models or transfer learning. Training these early models required massive computing power (requiring resources such as high-end GPUs and large bandwidth memory). Smaller organisations may not have had easy access to such resources. This is less of a barrier nowadays. For example, in this study, where LM was used for synonym generation and detection of domain specific jargon/terminology, no compute-intensive resources were required. The large pre-trained, embedding models (Word2Vec and BERT) were fined-tuned on The University's corpus. The LM fine-tuning was performed on computational infrastructure that is readily accessible to most organisations. Similarly, the implementation of LTR in this study (utilising the XGBoost ML library with a listwise LambdaMART loss function and eight input features) required minimal compute resources.

Measuring implementation effort in an organisation would involve counting the time (in person-hours, assuming availability of skilled engineering staff) required to execute a project, associated tasks and Work Breakdown Structures (WBS), risks and process change for the deployment of the AI approach. In addition to the initial implementation effort, an estimation of the annual ongoing maintenance effort would also be required. These measurements may be subjective, and the emergence of a person-hours figure might imply an unwarranted degree of precision and accuracy. These efforts compound the already substantial energy requirements and consequent environmental consequences of modern AI training procedures. A natural limitation of the scope of this study is that the numerous additional steps that are required for the AI approach versus the traditional approach are merely outlined; no attempt is made to count person-hours, costings, etc.

The broader adoption of AI across other business functions in organisations is inevitable. According to the Harvard Business Review, the question is no longer whether AI will reshape business processes, but how quickly and extensively it will do so [244]. The challenges listed above will therefore need to be addressed in any case. Implementing AI methods for search result and query auto-completion ranking offers a pragmatic entry point that serves as a relatively low-risk introduction and a strategic bridge toward future AI integrations.

5.6.1 Legislative Barrier

As described in Section §2.6, the EU AI Act will become fully applicable in 2026. Any search service using AI methods must align its AI systems and practices with the new regulatory requirements.

Under the Act, ES and QAC services are generally categorised as a 'limited risk' (as the ranking models are not used for automated decision-making). But even providers of limited risk services are obliged to consider conformity assessments, risk management systems, bias mitigation documentation, and ensuring transparency in AI system operations. This minimum adherence is required regardless of the designated risk categorisation [205].

No similar compliance, documentation or regulatory requirements are applicable when using the traditional ranking approach. Smaller enterprises might prioritise traditional methods (e.g., BM25 with field boosting) over the AI approach. Therefore, the Act, which purports to "promote the uptake of human-centric and trustworthy AI", could inadvertently act as a barrier for smaller organisations considering the transition from traditional to AI approaches for ES.

In summary, this section outlines the main implementation challenges and barriers associated with adopting the AI approach. The challenges are listed and summarised to provide an indication of the additional effort required. Unfortunately there can be no straight reply to the subjective question 'is it worth it?', as the answer is entirely dependent on the priorities, capabilities and resources of the individual organisation.

5.7 Generalisability and Domain Transfer

While this study demonstrates measurable ranking improvements of LTR and LM methods over traditional ranking methods on The University's custom ES service, the generalisability and domain transfer of these findings to other contexts and domains warrants careful consideration. This section discusses the challenges and opportunities related to generalisability and domain transfer.

For this discussion, generalisability is defined as the extent to which the findings of a study can be broadly applied using unseen data drawn from the same distribution as the ES training data. Domain transfer deals with crossing contextual boundaries into different but related domains (e.g., legal search, healthcare search, WS).

5.7.1 Generalisability

As described in Section §3.12, the performance of the ranking model was validated using basic, holdout and cross-validation techniques. These techniques also have the benefit of ensuring that overfitting is avoided.

Since The University's ES ranking model is deployed on a 'real world' corpus that changes from day to day, another technique is to use new unseen 'real world' data. These changes present opportunities to 'eyeball' how well the ranking model performs on new data. For example, a new microsite called 'Ukraine' was created several

months after the initial deployment of the LTR ranking model, coinciding with the onset of the Ukraine war, which was widely covered in the news at the time. The new data was not labelled, and was never part of the ENTRP-SRCH dataset, so the performance of the ranking model for a specific query can not be formally validated. However, even with some basic subject matter knowledge, it was possible to observe that the ranking model generalised quite well for a cluster of Ukraine-related queries.

The extent to which such generalisation holds depends on whether newly created microsites exhibit characteristics similar to those represented in the training data. For instance, if a microsite were composed entirely of MS Excel documents (which typically lack hyperlinks, anchor text, and well-defined structural fields), the model would be expected to perform poorly. Although this study focuses on a university setting (where content is naturally segmented into faculty, school, and administrative microsites), the same structural pattern is common across many higher-education institutions [83]. As such, the features and ranking approach developed here may have applicability to other universities that maintain similar microsite architectures.

5.7.2 Domain Transfer Challenges

A recurring theme of this study is the incompatibility of models and datasets developed for WS when applied directly to the domain of ES (appendix A presents a comprehensive list of differences between ES and WS). These differences constrain the transferability of ranking models trained on open WS datasets (e.g., the Microsoft and Yahoo LETOR benchmark datasets [29, 30]), as such models rely on assumptions about content structure, user behaviour, and feedback loops that do not hold in ES contexts.

Even within the field of ES, content varies widely across organisations. A key consideration in deploying the AI approach for ranking models is the transferability of findings beyond the experimental setting. While this study successfully demonstrated LTR and LM effectiveness in one ES environment, certain results may be specific to the dataset and organisational context. For instance, the reliance on domain-specific features, such as jargon-aware ranking signals, may limit applicability to organisations with different content structures or user behaviours.

ES systems in other industries (e.g., legal, healthcare, manufacturing) have distinct content types (e.g., contracts, arguments, patient records, technical manuals) and structures (e.g., hyperlinks in intranets vs. file shares). Features like URLlength or LinkRank may lose relevance in non-web-based ES systems (e.g., stand-alone document repositories). Domain-specific feature engineering (e.g., 'publication date' for legal court records) could replace or complement the existing features.

The JARGES and QACES LM features are tailored to detect organisational jargon. Jargon is, by definition, specific to an organisation. Their effectiveness depends on the prevalence of such terminology in the target domain. For example, in a healthcare ES system, medical terminology might require a different LM fine-tuning approach compared to academic jargon. On the other hand, some organisations are taking proactive steps to prevent the proliferation of jargon. As described in Section §2.1.8, the 'explain like I'm 5' (ELI5) pedagogical and communicative device

is the latest organisational trend that encourages employees to explain concepts in jargon-free terms anyone could understand [71]. This would limit the effectiveness of the JARGES and QACES LM-based features.

It is somewhat paradoxical that, although feature engineering can significantly improve in-domain performance, it may constrain transferability unless the engineered features are available or can be reliably reproduced in the target domain.

While the dataset's size (2544 Q-D pairs) is sufficient for its intended purpose at The University (namely search result ranking, optimised for the most frequently queries topics), it may not scale to larger or more diverse enterprises. By contrast, the methodology for dataset creation (e.g., leveraging microsite editors as judges or the use of CTR data) could be replicated in other organisations to build domain-specific datasets. The public release of `ENTRP-SRCH` allows future researchers to validate and adapt their approach.

In conclusion, the research demonstrates that the AI approach can significantly improve ranking in ES, but transferability depends on addressing domain-specific challenges. While the core methodologies (e.g., LTR for feature weighting, LM for jargon/terminology detection) are transferable, the application of the `ENTRP-SRCH` dataset to new domains warrants careful consideration.

A potential avenue for future study could focus on creating more adaptable frameworks to fit broader ES contexts, ensuring that the AI approach's ranking model can be readily applied in other organisations.

5.7.3 User Behaviour Differences

The study notes that ES users (e.g., knowledge workers) exhibit different search behaviours than WS users (e.g., query length, navigational intent). The 'QAC abandon' phenomenon observed and described in EXP-8 might vary in domains with different user urgency levels (e.g., customer support vs. academic research or even a team of auditors).

5.8 Scope Limitations

The possibilities for research in information retrieval for ES are nearly limitless. However, due to practical constraints, the scope of this study has been restricted. Exclusions encompass ranking functions like personalisation, location-based ranking, and AI generated summaries.

5.8.1 Personalisation Ranking Function

As discussed in Section §2.2.11, personalisation and collaborative-based filtering are major areas of investigation in their own right and are typically covered in the domain of recommender systems. While personalisation is not within the scope of this study, it has been very effective for ranking in the field of WS [21, 137, 245].

Profiling of individual user or session data was not permitted for this programme

of research by The University (in accordance with research ethics and data protection requirements). While many enterprises employ various forms of monitoring or telemetry for operational, security, or compliance purposes, such practices are generally governed by internal policies and employment contracts. In contrast, for research purposes, the profiling of individual search behaviour raises distinct ethical and legal concerns relating to consent, transparency, and the proportionality of data use under GDPR. Consequently, only aggregated and anonymised query history and selection data was used in this study.

The nature of the relationship between an organisation and its members/employees is not the same as that between an individual and a commercial WS provider like Google. While a user may not be concerned about sharing their behaviour patterns with a third-party organisation like Google, they may feel differently when asked to share this with their employer [96].

5.8.2 Location-based Ranking Function

As described in Section §2.2.12, the ranking of search results based on the user's location is called 'location-based ranking'. The concept can also be applied to QAC, which would involve tailoring the ranking of prefix-candidate pairs to consider the user's geographical location. In ES, this function could be used for the upranking of nearby organisational services, such as offices, printers or first aid defibrillators.

As per personalisation, this function would depend on users disclosing personal data (location). While users may be content to share their location (and location history) with the major commercial WS providers, it is generally considered a breach of privacy and trust to grant it to their employer/organisation. This is probably due to the potential for misuse and the inherent power imbalance in the employer-employee relationship. Location-based ranking is not within the scope of this study, though it has proven to be very effective in the fields of WS and ecommerce [30, 169, 246]

5.8.3 AI Generated Summaries

Many of the major WS providers now integrate AI summaries into their interfaces. As described in Section §2.5.6, AI summaries are automatically generated concise textual representations. The summary affects ranking insofar as the content is presented atop the search results page. These summaries are typically generated using transformer-based language models combined with extractive summarisation techniques.

The conservative nature of smaller enterprises may delay adoption of generative AI until the dangers of factual inaccuracies and hallucinations is reduced. For example, for The University's ES service, IT Services management have raised concerns that AI generated content might replicate or reinforce biases present in training data, potentially exposing the institution to reputational and legal risks. Furthermore, enterprises may be at a loss to understand, much less explain why certain content was included in AI summaries.

Just as most advancements in WS eventually make their way to ES, AI summaries will likely soon also become standard in ES. Microsite editors would be well-placed, as subject matter experts, to gauge and validate summaries. By leveraging LTR and LM methods in search results and QAC ranking today, the organisation would be well-prepared for implementation of (more disruptive) AI summaries in the future.

5.9 Future Directions and Research Opportunities

While this study has demonstrated the viability of LTR and LM approaches for ES ranking, several areas warrant further exploration:

- Future research might include **replacing GPT-4 with BERT** and re-conducting the jargon detection experiment (EXP-8). As discussed in Section §5.4, BERT is primarily designed for understanding. It is not a generative model (like GPT-4) and is likely to be more repeatable under controlled experimental conditions.

- While this study proved a correlation between human relevance judgements and CTR feedback, it would be interesting to use a **hybrid ES dataset**. For example, each feedback method could have a 50/50 input to the final annotation score. Alternatively, the ratio could be an adjustable variable much like the 'temperature setting' in generative AI (as discussed in Section §5.5.2). Comparing the performance of the subsequent ranking models could be an interesting avenue for a future study. A hybrid dataset would also serve to mitigate position and organisation bias identified in EXP-3.

- A limitation of the `ENTRP-SRCH` dataset, which centres on just twenty of the most frequently submitted queries, was apparent in EXP-5 (JARGES). The use of click-through data in place of human judgements for Q-D pair annotation would facilitate larger and more diverse ES datasets that are better able to capture the nuances of enterprise specific terminology. Finally, it would be interesting to perform a longitudinal study to gauge the JARGES impact on semantic and exploratory search, based on query expansion and recall on a real-world ES system.

- Many real-world searchers opted not to take the time to engage with the QAC 'interactive search' offering, even when the golden candidate is offered atop the list. This **QAC abandon** phenomenon presents some interesting new directions for the field of auto-completion (e.g., why do searchers abandon? Is is just the very fast typists who abandon? Does abandon correlate with latency in the presentation of candidates?).

- Employing generative AI to produce **AI summaries** for ES would mark a significant leap forward. Introducing AI summaries for ES would be an exciting, but perhaps sensitive/controversial avenue for further study and experimentation (as discussed in Section §5.8.3), which would likely be subject to a thorough multi-stakeholder research ethics approval process. A smoother implementation of this service can be expected where the organisation's ES infrastructure already employs LTR and LM methods.

5.10 Summary

This chapter synthesised experimental findings, discussed their implications for ES ranking, generalisability, and practical adoption challenges. While LTR and LM methods have proven potential in improving ranking performance, their integration requires careful consideration of implementation complexity, dataset generation, and ongoing maintenance requirements. The scope limitations of this study are noted. Future research could include replacing GPT-4 with BERT for jargon detection, using a mixture of human and CTR data to create a larger and de-bias dataset, investigating causes of QAC abandon, and an exploration into the use of AI summaries for ES.

6

CONCLUSION

This research set out to address challenges in Enterprise Search (ES) by investigating the applicability and effectiveness of Learning to Rank (LTR) and Language Modelling (LM) methods to improve ranking performance for both search results and query auto-completions (QAC).

The motivation for this study stemmed from the persistent gap between user expectations, shaped by commercial web search (WS), and the generally poor performance of ES systems, particularly for ranking search results, auto-completion candidates, and addressing the vocabulary mismatch that is compounded by the prevalence of specialised jargon and terminology unique to an organisation.

This concise chapter complements the comprehensive discussion in Chapter 5 by systematically evaluating the fulfilment of the Research Objectives (OBJ), addressing the Research Questions (RQ), and assessing the outcomes of the Research Contributions (RC) originally defined in Chapter 1.

6.1 Research Objectives Revisited

The following provides a summary of how each of the four research objectives was achieved.

- **Fulfilment of OBJ-1**

  > *Develop and evaluate a Learning to Rank model for ES search results and compare it with the effectiveness of traditional ranking methods.*

  This objective was fully accomplished. Experiments (EXP-1, EXP-2) demonstrated that LTR improved ranking performance (as measured by nDCG) compared to traditional methods like BM25 and field boosting. The ablation study (EXP-2) identified the contribution of each feature in the LTR model.

- **Fulfilment of OBJ-2**

Compare the effect of using implicit feedback such as click-through data instead of human judgements in LTR training data gathered from an online ES service.

This objective was fully accomplished. EXP-3 showed a strong correlation between click-through data and human judgements, validating CTR as a viable surrogate for training data. The experiment also uncovered biases related to organisation and position effects.

- **Fulfilment of OBJ-3**

  Detect jargon/terminology in enterprise content and decode jargon terms for query expansion via synonyms to unlock Semantic Search.

  This objective was partially accomplished. While the JARGES LM feature (EXP-5) was introduced to detect jargon/terminology, evaluation using the `ENTRP-SRCH` dataset did not exhibit a statistically significant improvement in ranking performance. In spite of this, the qualitative analysis of the generated synonyms revealed promising results for recall as a foundation for semantic search.

- **Fulfilment of OBJ-4**

  Develop and evaluate a ranking model for Query Auto-Completion candidates customised for enterprise content and associated specialised jargon/terminology.

  This objective was fully accomplished. EXP-6 and EXP-7 demonstrated that a ranking model generated using LTR improved QAC ranking (as measured by MRR). Furthermore, the A/B test in EXP-8 confirmed the positive impact on ranking performance of QACES in a live ES service.

6.2 Research Questions Revisited

With the research objectives accomplished, the original research question can be revisited and evaluated.

As outlined in §1.2, the overarching research question pursued in this thesis is:

To what extent can Learning to Rank and Language Modelling methods, tailored to enterprise content and information needs, boost traditional ES ranking performance for both search results and query auto-completions?

This overarching question is divided into three underlying research questions, each targeting a distinct problem in ES: (PS-1) the poor relevance of ranked results compared to WS, (PS-2) the challenge of semantic mismatch between query intent and document language, and (PS-3) the difficulty users experience in formulating queries using organisation-specific jargon.

- **Answer to RQ-1**

Compared to the traditional methods, to what extent can Learning to Rank improve relevance matching and ranking performance of search results for Enterprise Search?

This question has been fully answered. EXP-1) showed that LTR significantly outperformed traditional methods (e.g., BM25 + field boosting), with measurable gains in nDCG (e.g., a 5.85% improvement for nDCG@5). Feature ablation (EXP-2) confirmed that BM25 and field boosting are the dominant contributors.

- **Answer to RQ-2**

  To what extent can Language Modelling address 'vocabulary mismatch' by improving semantic matching for Enterprise Search?

  This question has been fully answered. LM methods (e.g., JARGES/QACES) bridged vocabulary gaps by detecting divergence in jargon and terminology (Tables 3.12 and 4.5). EXP-5 showed a qualitative improvement in search result recall via synonyms for jargon terms. EXP-8 showed improved ranking when LM methods were added to the LM model for QAC.

- **Answer to RQ-3**

  To what extent can LTR and LM methods be used to suggest and improve the ranking of candidates generated for Query Auto-Complete for an organisation's specialised jargon/terminology?

  This question has been fully answered. The QACES feature (EXP-8) enhanced MRR in live A/B tests, while LTR-weighted features (e.g., MFQ, anchor text) outperformed TF alone (EXP-6).

Table 6.1 summarises how each of the three research question is tied to a specific research problem/objective and experiments.

Table 6.1: A mapping of how each research question is linked to the original research problems, objectives, thesis chapters and experiments.

Research Question	Research Problem	Research Objective	Thesis Chapter	Experiment
RQ-1	PS-1	OBJ-1,2	3	1,2,6,7
RQ-2	PS-2	OBJ-3,4	3,4	5,8
RQ-3	PS-3	OBJ-4	4	6,7,8

6.3 Research Contributions Revisited

In Section §1.5, it was stated that the overarching contribution of this research would be a bridge of the gap between state-of-the-art AI ranking models and real-world Enterprise Search implementations, while introducing novel features to detect and decode organisational jargon.

- **Contribution RC-1**

An identification of the features influencing a ranking model's performance for ES search results and QAC and how these features differ (or are weighted differently) from those used in Web Search.

This contribution has been fully realised and achieved via EXP-2 and EXP-7 (ablation studies for ranking of search results and QAC respectively). The experiment identified ranking features specific to enterprise content, and how these differ from those commonly adopted for WS. Feature weights (e.g., URL length, LinkRank) were determined. BM25 was the most significant contributor. Field boosting and document recency proved less critical due to the variety of document types in the enterprise corpus.

- **Contribution RC-2**

  A comparison of the effectiveness of traditional ES ranking methods with ranking performance achieved using LTR and LM methods.

  This contribution has been fully realised and demonstrated through EXP-1 (nDCG improvements) and EXP-6 (MRR gains). Marginal gains for certain nDCG cut-offs for search results ranking highlight the balance of performance and implementation effort for smaller organisation with limited resources. Although the QAC ranking model performance gain is substantial using the AI approach, it too must be weighed against the additional skills, resources and effort required for an organisation to implement (and periodically re-train) it.

- **Contribution RC-3**

  The creation of an Enterprise Search dataset based on document characteristics and search log data from a 'real world' ES service. The dataset is in the LTR format and is publicly available.

  This contribution has been fully realised and the completed and anonymised dataset (`ENTRP-SRCH`) has been published on GitHub[17], enabling reproducibility and future ES research. Query clustering and the Search Demand Curve were used as an innovative approach for selection of suitable topics for annotation. The auxiliary IAA experiment (EXP-4) confirmed that while human relevance judgments exhibited moderate agreement levels on a Q-D pair, variability among annotators highlighted subjective differences in document assessment.

- **Contribution RC-4**

  A demonstration that click-through rate can be used in place of human relevance judgements in training data for ES search results.

  This contribution has been fully realised and validated in EXP-3, showing CTR's viability for training. While analysis of the CTR data detected position bias (as expected), it also uncovered a new kind of 'organisational bias' specific to ES. A novel approach to mitigation of both kinds of bias was also discussed, via the use a hybrid annotation scheme, where Q-D scores are formed by a

[17] https://github.com/ColinDaly75/ES-Ranking/blob/master/ENTRP-SRCH-v14.txt

mixture of CTR and human judgements. In the field of WS, click models have long been regarded as an effective method for inferring document relevance to a given query. It can now be said to hold true for ES as well.

- **Contribution RC-5**

  A Language Model algorithm that improves ranking and recall for jargon/terminology specific to an organisation (applied to both search results and QAC).

  This contribution has been partially realised. The novel QACES feature (EXP-8) leverages the 'divergence' between the meaning of a query as understood using a local corpus and that of the GPT LLM. This divergence, as measured by the Jaccard Distance, is used not just to detect jargon, but also to measure how 'jargony' a term is. Similarly, the 'JARGES' feature was used to detect and decode jargon (EXP-5), but quantitative testing using the `ENTRP-SRCH` dataset failed to demonstrate a statistically significant increase in ranking performance. This result is disappointing, but not entirely unexpected, as the dataset is small, with limited query diversity and a scarcity of jargon rich Q-D pairs. In spite of this, the qualitative analysis of the generated synonyms revealed promising results for recall as a foundation for semantic search.

6.4 Publications

During the process of realising the research contributions, three papers were accepted for publication[18], and an additional (Paper IV) has been submitted, and currently awaiting acceptance for publication. These publications ensure that the findings and insights from this thesis reach the broader scientific audience.

- **Paper-I** Daly C. "Learning to Rank: Performance and Practical Barriers to Deployment in Enterprise Search." 3rd Asia Conference on Information Engineering (ACIE), Chongqing, China, 2023 (long paper) [136]. This paper compares the implementation and performance of traditional, moderately uncomplicated methodologies such as BM25 with Learning to Rank and examines the common assumption that whatever method is best for WS must also be best for ES.

- **Paper-II** Daly C and Hederman L. "Enterprise Search: Learning to Rank with Click-Through Data as a Surrogate for Human Relevance Judgements." In Proceedings of the 15th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (KDIR), Porto, Portugal, 2023 (long paper) [231]. This paper explores whether click-through data relevance feedback correlates with that of the human relevance judgements, and whether small relevance judgement training data can be dispensed with while relying entirely on abundant quantities of click-through data instead.

[18] https://scholar.google.com/citations?user=j7zs9-YAAAAJ

- **Paper-III** Daly C and Hederman L. "Learning to Rank for Query Auto-Complete with Language Modelling in Enterprise Search." in Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (KDIR), Porto, Portugal, 2024 (long paper) [247]. In this paper, traditional QAC ranking functions, such as how often the prefix appears in the corpus, most popular completions, most frequently queried, anchor text frequency, and trending queries are implemented. The individual contribution of each of these signals is computed and then supplemented with a novel LLM feature (QACES) to detect jargon/terminology. LTR is used to combine the weighted features to create a QAC ranking model and an A/B test is used to compare ranking performance.

- **Paper-IV** Submitted for publication (awaiting acceptance decision). Daly C and Hederman L. "JARGES: Detecting and Decoding Jargon for Enterprise Search." submitted to (KDIR), Marbella, Spain 2025 (short paper) attached as Appendix B. This paper introduces 'JARGES', a novel algorithm and feature for detecting and decoding jargon for ES. It is designed to enhance a ranking model combining LTR and transformer-based synonym expansion.

I am the primary author of each of the publications. Notably, Paper-III received the **KDIR 2024 Best Overall Paper** award. Table 6.2 presents a mapping of each contribution to its corresponding research problem, question, objective, and associated paper. In contrast to the previous table, this version is structured around Research Contributions as the organising principle.

Table 6.2: A mapping of how each research contribution is linked to the original research problems, questions, objectives, thesis chapters, experiments and associated papers.

Research Contribution	Research Problem	Research Question	Research Objective	Thesis Chapter	Experiment	Paper Number
RC-1	PS-1	RQ-1	OBJ-1,4	3,4	2,7	I,III
RC-2	PS-1,2	RQ-1,2	OBJ-2,3	3,4	1,6	I,III
RC-3	PS-1	RQ-1	OBJ-4	3	3,4	I,II
RC-4	PS-1	RQ-1,3	OBJ-2	3	3	II
RC-5	PS-2,3	RQ-2,3	OBJ-3,4	3,4	5,8	III,IV

6.5 Limitations

Limitations are documented as an integral part of each experiment described in Chapters 3 and 4. Similarly, scope limitations are detailed in Section §5.8.

This research journey has revealed how methodological choices profoundly shape outcomes. While these limitations represent valuable learning opportunities, addressing them earlier would have strengthened the study:

- It was foolhardy to assume that a single judge, even a subject matter expert, could be used to award annotations for a topic or subject area. A better

approach would have been to seek out two judges, each willing to participate and annotate for their topic. More thought should have been put into *planning the dataset* rather than prematurely jumping into creating it. The IAA test in auxiliary EXP-3 was introduced latterly to confirm the integrity of the `ENTRP-SRCH` dataset in spite of this limitation.

- Similarly, for EXP-5, the relatively small `ENTRP-SRCH` had a limited query diversity and scarcity of jargon-rich Q-D pairs. The dataset used human annotations and was designed primarily for ranking the most frequent topic queries. Had the dataset been generated using CTR data as a surrogate for human annotations (EXP-4 proves that this is a valid approach), then the evaluation would have been able to capture the nuances of enterprise specific terminology better (in line with the qualitative success of synonym generation).

These limitations were avoided in the QAC experiments, as Most Popular Completions (MPC) data was used as the 'ground truth' for annotations of the Prefix-Candidate (P-C) pairs. These pairs combined to generate a QAC dataset that contained 902,412 P-C pairs, and EXP-8 proved that this was sufficiently diverse to capture an improvement in ranking performance for jargon terms.

6.6 Future Directions

While this study has demonstrated the effectiveness of the AI approach for ES ranking, several areas merit further exploration. Future research could include using a combination of human and CTR data to create a larger, more diverse dataset with de-biased annotations, investigation of the causes of QAC abandonment, and an exploration of the use of AI summaries for ES. These opportunities are detailed in the discussion chapter (Section §5.9).

6.7 Final Comment

Six years is a considerable span in the rapidly evolving field of artificial intelligence. When this study commenced in 2019, BERT and GPT-1 had only just emerged. Since then, the pace of advancement has been remarkable, and continued transformation is inevitable. By leveraging LTR and LM methods in search results and QAC ranking today, organisations will be better positioned for the implementation of potentially more disruptive future AI developments. If this work contributes to that ongoing progress and helps organisations make informed decisions about adopting the AI approach, it will have served its purpose.

This research demonstrates that the AI approach (specifically LTR and LM) can be effectively adapted and applied to ES, a domain traditionally underserved by the academic community. By designing, implementing, and evaluating intelligent ranking models in a live, real-world ES system, this work closes the gap between theoretical AI advancements and practical organisational search needs.

The creation of a jargon-aware Language Model and its integration into LTR pipelines marks a novel contribution to enterprise information retrieval. The public release

of a LETOR-formatted ES dataset also addresses a known barrier in the field: the lack of open training data tailored to enterprise contexts.

Beyond performance improvements in both search results and query auto-completion, this thesis offers a replicable methodology, reusable tools, and a clear demonstration of how AI ranking systems can enhance search effectiveness for knowledge workers. It provides a solid foundation for future ES research and serves as a template for organisations seeking to modernise their internal search capabilities using AI-driven methods.

References

[1] P. H. Cleverley and S. Burnett, "Enterprise search and discovery capability: The factors and generative mechanisms for user satisfaction:," *Journal of Information Science*, vol. 45, no. 1, pp. 29–52, 5 2019. [Online]. Available: https://journals.sagepub.com/doi/full/10.1177/0165551518770969

[2] S. Feldman and C. Sherman, "The High Cost of Not Finding Information An IDC White Paper," *IDC Technical Report*, no. 29127, 2001.

[3] J. Bentley, "Mind the Enterprise Search Gap: Smartlogic Sponsor MindMetre Research Report," 2011. [Online]. Available: https://www.prweb.com/releases/2011/8/prweb8692712.htm(accessed5thJune2022)

[4] T. Paulman, "Gartner Survey Reveals 47% of Digital Workers Struggle to Find the Information Needed to Effectively Perform Their Jobs." [Online]. Available: https://www.gartner.com/en/newsroom/press-releases/

[5] Microsoft, "Will AI Fix Work?" *Work Trend Index: Annual Report*, pp. 1–29, 2023.

[6] U. Kruschwitz and C. Hull, "Searching the Enterprise," *Foundations and Trends® in Information Retrieval*, vol. 11, no. 1, pp. 1–142, 2017. [Online]. Available: http://dx.doi.org/10.1561/1500000053

[7] I. Ruthven, "Re-examining the potential effectiveness of interactive query expansion," *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pp. 213–220, 7 2003. [Online]. Available: https://dl.acm.org/doi/10.1145/860435.860475

[8] R. Mukherjee and J. Mao, "Enterprise Search: Tough Stuff," *Queue*, vol. 2, no. 2, p. 36, 4 2004.

[9] A. Molnar, "Google Search Appliance Retirement Explained," Search Explained, Diosd, Tech. Rep., 2016. [Online]. Available: http://searchexplained.com.

[10] N. Craswell, M. Cambridge, and I. Soboroff, "Overview of the TREC-2005 Enterprise Track," in *TREC 2005 conference notebook*, 2005, pp. 199–205. [Online]. Available: http://www.cs.cmu.edu/

[11] T. Sullivan, "Webinar: Simpler Semantic Search with Solr - YouTube," 2017. [Online]. Available: https://www.youtube.com/watch?v=mHtU1KIUP2U

[12] "Google Click-Through Rates (CTRs) by Ranking Position in 2025 – First Page Sage," 2025. [Online]. Available: https://firstpagesage.com/reports/google-click-through-rates-ctrs-by-ranking-position/

[13] S. E. Robertson, S. Walker, S. Jones, M. M. Hancock-Beaulieu, and M. Gatford, "Okapi at TREC-3," *Nist Special Publication Sp*, vol. 109, p. 109, 1995.

[14] The Apache Software Foundation., "Apache Solr," 2004. [Online]. Available: https://solr.apache.org/

[15] "similarity — Reference." [Online]. Available: https://www.elastic.co/docs/reference/elasticsearch/mapping-reference/similarity

[16] L. Page and S. Brin, "The anatomy of a large-scale hypertextual Web search engine," *Computer Networks*, vol. 30, no. 1-7, pp. 107–117, 1 1998.

[17] A. N. Langville and C. D. Meyer, *Google's PageRank and beyond: The science of search engine rankings*. USA: Princeton University Press, 2011.

[18] K. Järvelin and J. Kekäläinen, "Cumulated gain-based evaluation of IR techniques," *ACM Transactions on Information Systems (TOIS)*, vol. 20, no. 4, pp. 422–446, 10 2002. [Online]. Available: https://dl.acm.org/doi/pdf/10.1145/582415.582418

[19] D. Wang, W. Chen, G. Wang, Y. Zhang, and B. Hu, "Explore click models for search ranking," *International Conference on Information and Knowledge Management, Proceedings*, pp. 1417–1420, 2010.

[20] F. Cai and M. De Rijke, "A survey of query auto completion in information retrieval," *Foundations and Trends in Information Retrieval*, vol. 10, no. 4, pp. 273–363, 2016.

[21] F. Cai, S. Liang, and M. De Rijke, "Time-sensitive personalized query auto-completion," *CIKM 2014 - Proceedings of the 2014 ACM International Conference on Information and Knowledge Management*, pp. 1599–1608, 11 2014. [Online]. Available: https://dl.acm.org/doi/10.1145/2661829.2661921

[22] D. Sullivan, "How Google autocomplete works in Search," p. 1, 2018. [Online]. Available: https://blog.google/products/search/how-google-autocomplete-works-search/

[23] M. Korayem, C. Ortiz, K. Aljadda, and T. Grainger, "Query sense disambiguation leveraging large scale user behavioral data," *Proceedings - 2015 IEEE International Conference on Big Data, IEEE Big Data 2015*, pp. 1230–1237, 12 2015.

[24] J. Jiang and C. Ni, "What affects word changes in query reformulation during a task-based search session?" *CHIIR 2016 - Proceedings of the 2016 ACM Conference on Human Information Interaction and Retrieval*, pp. 111–120, 2016.

[25] L. Li, H. Deng, A. Dong, Y. Chang, R. Baeza-Yates, and H. Zha, "Exploring query auto-completion and click logs for contextual-aware web search and query suggestion," *26th International World Wide Web Conference, WWW 2017*, pp. 539–548, 2017. [Online]. Available: https://dl.acm.org/doi/10.1145/3038912.3052593

[26] G. Pass, A. Chowdhury, and C. Torgeson, "A picture of search," *ACM International Conference Proceeding Series*, vol. 152, 2006. [Online]. Available: https://dl.acm.org/doi/10.1145/1146847.1146848

[27] Q. Wang, H. Li, H. Xiong, W. Wang, J. Bian, Y. Lu, S. Wang, Z. Cheng, D. Dou, and D. Yin, "A Simple yet Effective Framework for Active Learning to Rank," *Machine Intelligence Research*, vol. 21, no. 1, pp. 169–183, 2 2024. [Online]. Available: https://link.springer.com/article/10.1007/s11633-023-1422-z

[28] P. D. Turney and P. Pantel, "From frequency to meaning," *Journal of Artificial Intelligence Research*, vol. 37, pp. 141–188, 1 2010. [Online]. Available: https://dl.acm.org/doi/10.5555/1861751.1861756

[29] T.-Y. Liu, J. Xu, T. Qin, W. Xiong, and H. Li, "LETOR: Benchmark Datasets for Learning to Rank," *SIGIR 2007 Workshop on Learning to Rank for Information Retrieval*, vol. 1, no. Lr4ir, pp. 3–10, 2007. [Online]. Available: http://research.microsoft.com/en-us/um/beijing/events/LR4IR-2007/

[30] O. Chapelle, "Yahoo! Learning to Rank Challenge Overview," pp. 1–24, 1 2011. [Online]. Available: http://proceedings.mlr.press/v14/chapelle11a/chapelle11a.pdfhttp://proceedings.mlr.press/v14/chapelle11a.html

[31] T. Qin, T. Y. Liu, J. Xu, and H. Li, "LETOR: A benchmark collection for research on learning to rank for information retrieval," *Information Retrieval*, vol. 13, no. 4, pp. 346–374, 2010.

[32] D. Ganguly, D. Roy, M. Mitra, and G. J. Jones, "A word embedding based generalized language model for information retrieval," *SIGIR 2015 - Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 795–798, 8 2015. [Online]. Available: https://dl.acm.org/doi/10.1145/2766462.2767780

[33] A. Singhal, "Introducing the Knowledge Graph: things, not strings," p. 1, 2012. [Online]. Available: https://blog.google/products/search/introducing-knowledge-graph-things-not/

[34] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed Representations of Words and Phrases and their Compositionality," *Advances in Neural Information Processing Systems*, vol. 26, 2013.

[35] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "RoBERTa: A Robustly Optimized BERT Pretraining Approach," *arXiv*, vol. 1, no. abs/1907.11692, 7 2019. [Online]. Available: http://arxiv.org/abs/1907.11692

[36] OpenAI 2023, "OpenAI GPT-4," 2023. [Online]. Available: https://openai.com/gpt-4

[37] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," in *The 22nd ACM SIGKDD International Conference*, 8 2016, pp. 785–794.

[38] C. Lucchese, C. I. Muntean, F. M. Nardini, R. Perego, and S. Trani, "RankEval: An evaluation and analysis framework for learning-To-rank solutions," *SIGIR 2017 - Proceedings of the 40th International ACM SIGIR Conference on Research and Development*

in Information Retrieval, pp. 1281–1284, 8 2017. [Online]. Available: https://www.researchgate.net/publication/318763909_RankEval_An_Evaluation_and_Analysis_Framework_for_Learning-to-Rank_Solutions

[39] C. Lucchese, C. Muntean, F. Nardini, R. Perego, and S. Trani, "RankEval: Evaluation and investigation of ranking models," *SoftwareX*, vol. 12, p. 100614, 7 2020.

[40] W. B. Croft, "Search engines : information retrieval in practice / by Bruce Croft, Donald Metzler, Trevor Strohman." pp. 0–0, 2010.

[41] L. Freund, E. G. Toms, and C. L. Clarke, "Modeling task-genre relationships for IR in the workplace," *SIGIR 2005 - Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 441–448, 2005. [Online]. Available: https://dl.acm.org/doi/10.1145/1076034.1076110

[42] G. Schymik, K. Corral, D. Schuff, and R. St. Louis, "The Benefits and Costs of Using Metadata to Improve Enterprise Document Search," *Decision Sciences*, vol. 46, no. 6, pp. 1049–1075, 12 2015. [Online]. Available: https://onlinelibrary.wiley.com/doi/full/10.1111/deci.12154

[43] R. Fagin, R. Kumar, K. S. McCurley, J. Novak, D. Sivakumar, J. A. Tomlin, and D. P. Williamson, "Searching the workplace web," *Proceedings of the 12th International Conference on World Wide Web, WWW 2003*, pp. 366–375, 2003. [Online]. Available: https://dl.acm.org/doi/10.1145/775152.775204

[44] M. Abrol, N. Latarche, U. Mahadevan, J. Mao, R. Mukherjee, P. Raghavan, M. Tourn, J. Wang, and G. Zhang, "Navigating Large-Scale Semi-Structured Data in Business Portals," in *Proceedings of the 27th International Conference on Very Large Data Bases*, ser. VLDB '01. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2001, p. 663–666.

[45] S. Robertson and H. Zaragoza, *The probabilistic relevance framework: BM25 and beyond*. Now Publishers, Inc., 2009, vol. 3, no. 4.

[46] B. Insight, "BAInsight." [Online]. Available: https://www.bainsight.com/application-integration,

[47] D. Lewandowski, "Understanding Search Engines," *Understanding Search Engines*, 2023.

[48] S. Allard, K. J. Levine, and C. Tenopir, "Design engineers and technical professionals at work: Observing information usage in the workplace," *Journal of the American Society for Information Science and Technology*, vol. 60, no. 3, pp. 443–454, 3 2009. [Online]. Available: https://onlinelibrary.wiley.com/doi/full/10.1002/asi.21004

[49] A. Stocker, A. Richter, C. Kaiser, and S. Softic, "Exploring barriers of enterprise search implementation: a qualitative user study," *Aslib Journal of Information Management*, vol. 67, no. 5, pp. 470–491, 9 2015.

[50] M. Pelletier, "What is the Average Website Lifespan? 10 Factors In Website Longevity."

[51] S. E. Lindley and D. J. Wilkins, "Building Knowledge through Action: Considerations for Machine Learning in the Workplace," *ACM Transactions on Computer-Human Interaction*, 7 2021. [Online]. Available: https://dl.acm.org/doi/10.1145/3584947

[52] M. Lykke, A. Bygholm, L. B. Søndergaard, and K. Byström, "The role of historical and contextual knowledge in enterprise search," *Journal of Documentation*, vol. 78, no. 5, pp. 1053–1074, 12 2021. [Online]. Available: https://vbn.aau.dk/en/publications/the-role-of-historical-and-contextual-knowledge-in-enterprise-sea

[53] B. Maiworm, "Why information management is becoming increasingly important for employees," *IQ: The RIMPA Quarterly Magazine*, vol. 39, no. 1, pp. 35–37, 3 2023. [Online]. Available: https://search.informit.org/doi/full/10.3316/informit.907289601446189

[54] M. White, *Enterprise search*. Sebastopol, CA: O'Reilly Media, 2018.

[55] D. Hawking, "Challenges in Enterprise Search," in *Proceedings of the 15th Australasian Database Conference - Volume 27*, ser. ADC '04. AUS: Australian Computer Society, Inc., 2004, p. 15–24.

[56] K. Balog, *Entity-Oriented Search*. Springer Open, 2018, vol. 39. [Online]. Available: https://eos-book.org

[57] BroderAndrei, "A taxonomy of web search," *ACM SIGIR Forum*, vol. 36, no. 2, pp. 3–10, 9 2002. [Online]. Available: https://dl.acm.org/doi/10.1145/792550.792552

[58] R. W. White, M. Richardson, and W.-T. Yih, "Questions vs. Queries in Informational Search Tasks," *Microsoft Research*, no. Technical Report MSR-TR-2014-96, 2014. [Online]. Available: http://dx.doi.org/10.1145/2740908.2742769

[59] M. Kutlu, T. McDonnell, Y. Barkallah, T. Elsayed, and M. Lease, "Crowd vs. Expert: What can relevance judgment rationales teach us about assessor disagreement?" *41st International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2018*, pp. 805–814, 6 2018. [Online]. Available: https://dl.acm.org/doi/pdf/10.1145/3209978.3210033

[60] M. White, "Critical success factors for enterprise search," pp. 110–118, 6 2015.

[61] D. Turnbull and J. Berryman, *Relevant Search*. New York: Manning Publications Co., 2016.

[62] G. Marchionini, "Exploratory search," *Communications of the ACM*, vol. 49, no. 4, pp. 41–46, 4 2006. [Online]. Available: https://dl.acm.org/doi/10.1145/1121949.1121979

[63] M. Oberstein, "The State of Search 2022," Semrush, Global Report, Tech. Rep., 2022.

[64] U. Kruschwitz, D. Lungley, M. D. Albakour, and D. Song, "Deriving query suggestions for site search," *Journal of the American Society for Information Science and Technology*, vol. 64, no. 10, pp. 1975–1994, 10 2013.

[65] T. Liu, *Learning to Rank for IR*. Springer Berlin Heidelberg, 2004. [Online]. Available: http://onlinelibrary.wiley.com/doi/10.1002/cbdv.200490137/abstract

[66] W. T. Kritzinger and M. Weideman, "Search Engine Optimization and Pay-per-Click Marketing Strategies," *Journal of Organizational Computing and Electronic Commerce*, vol. 23, no. 3, pp. 273–286, 2013.

[67] "Organizing Information - How Google Search Works." [Online]. Available: https://www.google.com/intl/en_us/search/howsearchworks/how-search-works/organizing-information/?utm_source=chatgpt.com

[68] A. P. B. Veyseh, F. Dernoncourt, W. Chang, and T. H. Nguyen, "MadDog: A web-based system for acronym identification and disambiguation," *EACL 2021 - 16th Conference of the European Chapter of the Association for Computational Linguistics, Proceedings of the System Demonstrations*, pp. 160–167, 2021.

[69] C. Wang, P. Nulty, and D. Lillis, "A Comparative Study on Word Embeddings in Deep Learning for Text Classification," *ACM International Conference Proceeding Series*, pp. 37–46, 12 2020. [Online]. Available: https://dl.acm.org/doi/10.1145/3443279.3443304

[70] S. Singh, S. Farfade, and P. M. Comar, "Multi-Objective Ranking to Boost Navigational Suggestions in eCommerce AutoComplete," *ACM Web Conference 2023 - Companion of the World Wide Web Conference, WWW 2023*, pp. 469–474, 4 2023. [Online]. Available: https://dl.acm.org/doi/10.1145/3543873.3584649

[71] Bossgalaga, "Explain Like I'm Five — Don't Panic!" 2011. [Online]. Available: https://www.reddit.com/r/explainlikeimfive/

[72] M. White, *A history of enterprise search*. Sheffield University PressBooks OA, 2022. [Online]. Available: https://sheffield.pressbooks.pub/eshistory1

[73] A. Stocker, M. Zoier, S. Softic, S. Paschke, H. Bischofter, and R. Kern, "Is enterprise search useful at all? lessons learned from studying user behavior," *ACM International Conference Proceeding Series*, vol. 16-19-Sept, 9 2014. [Online]. Available: http://dx.doi.org/10.1145/2637748.2638425

[74] "BM25Similarity (Lucene 9.0.0 core API)." [Online]. Available: https://lucene.apache.org/core/9_0_0/core/org/apache/lucene/search/similarities/BM25Similarity.html

[75] N. Craswell, B. Mitra, E. Yilmaz, and D. Campos, "Overview of the TREC 2020 deep learning track," *TREC*, 2021. [Online]. Available: https://microsoft.github.io/TREC-2020-Deep-Learninghttp://arxiv.org/abs/2102.07662

[76] V. Bush, "As We May Think," *The Atlantic*, vol. 176, no. 1, pp. 101–108, 1945.

[77] T.-Y. Liu, *Learning to Rank for Information Retrieval*, 2nd ed. Springer Berlin Heidelberg, 2010, vol. 3.

[78] Y. Fan, J. Guo, Y. Lan, J. Xu, L. Pangy, and X. Cheng, "Learning visual features from snapshots for web search," in *International Conference on Information and Knowledge Management, Proceedings*, vol. Part F1318. Association for Computing Machinery, 11 2017, pp. 247–256. [Online]. Available: www.nngroup.com/articles/let-users-control-font-size/

[79] J. M. Kleinberg, "Authoritative sources in a hyperlinked environment," *Journal of the ACM*, vol. 46, no. 5, pp. 604–632, 9 1999. [Online]. Available: https://dl.acm.org/doi/abs/10.1145/324133.324140

[80] N. Grover and R. Wason, "Page Rank And HITS Algorithms," *International Journal of Engineering Research & Technology (IJERT)*, vol. 1, no. 8, pp. 1–15, 2012. [Online]. Available: http://www.ijert.org/browse/october-2012-edition?download=1344%3Acomparative-analysis-of-pagerank-and-hits-algorithms&start=120

[81] Y. Kim, S. W. Son, and H. Jeong, "LinkRank: Finding communities in directed networks," *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, vol. 81, no. 1, 2010.

[82] "Website Structure A to Z (With Examples)." [Online]. Available: https://slickplan.com/blog/types-of-website-structure

[83] A. Arzani, T. Vogl, M. Handte, and P. Marrón, "A Hybrid Approach for Mining the Organizational Structure from University Websites," *Proceedings of the 17th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management*, pp. 188–199, 2025. [Online]. Available: https://www.scitepress.org/DigitalLibrary/Link.aspx?doi=10.5220/0013658600004000

[84] "The Query Elevation Component — Apache Solr Reference Guide 7.7." [Online]. Available: https://solr.apache.org/guide/7_7/the-query-elevation-component.html

[85] B. Gomes, R. Com], and S. Thakur, "Trial Exhibit-UPX2044: U.S.v. Plaintiff States v. Google LLC." US Department of Justice, Google, Tech. Rep., 2019.

[86] S. K. K. Santu, P. Sondhi, and C. Zhai, "On application of learning to rank for e-commerce search," *SIGIR 2017 - Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, vol. 10, pp. 475–484, 8 2017.

[87] "The birth of the Web — CERN." [Online]. Available: https://www.home.cern/science/computing/birth-web

[88] "Top Websites in the World - November 2024 Most Visited & Popular Rankings - Semrush — Semrush." [Online]. Available: https://www.semrush.com/website/top/?utm_campaign=how-many-websites-on-the-internet&utm_source=explodingtopics.com&utm_medium=referral

[89] P. C. Yeung, S. Büttcher, C. L. Clarke, and M. Kolla, "A Bayesian approach for learning document type relevance," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 4425 LNCS. Springer Verlag, 2007, pp. 753–756.

[90] N. Nielson, "PDF: Still Unfit for Human Consumption, 20 Years Later," 8 2020. [Online]. Available: https://www.nngroup.com/articles/pdf-unfit-for-human-consumption/

[91] J. Rabin, C. McCathieNevile, and W3C Mobile Web Best Practices Working Group, "Mobile Web Best Practices 1.0," W3C Recommendation, 7 2008. [Online]. Available: https://www.w3.org/TR/mobile-bp/

[92] Microsoft Support, "View OpenDocument Format (ODF) files in Office for iPad and Office for iPhone," 2024. [Online]. Available: https://support.microsoft.com/en-gb/office/view-opendocument-format-odf-files-in-office-for-ipad-and-office-for-iphone

[93] L. Vassio, I. Drago, M. Mellia, Z. Ben Houidi, and M. L. Lamali, "You, the web, and your device: Longitudinal characterization of browsing habits," *ACM Transactions on the Web*, vol. 12, no. 4, p. 30, 5 2018. [Online]. Available: http://arxiv.org/abs/1806.07158http://dx.doi.org/10.1145/3231466

[94] "Desktop vs Mobile Market Share Worldwide — Statcounter Global Stats." [Online]. Available: https://gs.statcounter.com/platform-market-share/desktop-mobile/worldwide/

[95] R. M. Chory, L. E. Vela, and T. A. Avtgis, "Organizational Surveillance of Computer-Mediated Workplace Communication: Employee Privacy Concerns and Responses," *Employee Responsibilities and Rights Journal*, vol. 28, no. 1, pp. 23–43, 3 2016. [Online]. Available: https://link.springer.com/article/10.1007/s10672-015-9267-4

[96] P. Inman, "If you let Google have your data, why not the NHS? — Phillip Inman — The Guardian," 2024. [Online]. Available: https://www.theguardian.com/business/2024/oct/19/if-you-let-google-have-your-data-why-not-the-nhs

[97] I. Lopez de Vallejo, S. Hailes, R. Conroy-Dalton, and A. Penn, *Location Tracking in an Office Environment*. University College London, 3 2008.

[98] T. Joachims, "Optimizing search engines using clickthrough data," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2002.

[99] A. Paulsson, "USING CLICKTHROUGH DATA TO OPTIMIZE SEARCH RESULT RANKING," Ph.D. dissertation, University of Skovde, 2017.

[100] D. X. Zhou, L. Liu, A. Anubhai, M. Shandilya, S. Sigalas, W. Y. Wang, and Z. Huang, "Beyond Accurate Answers: Evaluating Open-Domain Question Answering in Enterprise Search," *Proceedings of the 2023 Conference on Human Information Interaction and Retrieval*, pp. 308–312, 3 2023. [Online]. Available: https://dl.acm.org/doi/10.1145/3576840.3578314

[101] G. Dupret and C. Liao, "A model to estimate intrinsic document relevance from the clickthrough logs of a web search engine," *WSDM 2010 - Proceedings of the 3rd ACM International Conference on Web Search and Data Mining*, no. January, pp. 181–190, 2010.

[102] S. Ceri, A. Bozzon, M. Brambilla, E. Della Valle, P. Fraternali, and S. Quarteroni, *An Introduction to Information Retrieval*. Cambridge University Press, 2013.

[103] C. D. Gull, "Seven years of work on the organization of materials in the special library," *American Documentation*, vol. 7, no. 4, pp. 320–329, 10 1956. [Online]. Available: https://onlinelibrary.wiley.com/doi/10.1002/asi.5090070408

[104] D. R. Swanson, "Historical Note: Information Retrieval and the Future of an Illusion," *Journal of the American Society for Information Science*, vol. v39, no. n2, pp. 92–98, 1988.

[105] D. Harman, *Information Retrieval Evaluation*, G. Marchionini, Ed. Springer, 2011, vol. 3, no. 2.

[106] C. Buckley and E. Voorhees, "Evaluating Evaluation Measure Stability," *SIGIR Forum (ACM Special Interest Group on Information Retrieval)*, 10 2000.

[107] S. Robertson, "On the history of evaluation in IR," *Journal of Information Science*, vol. 34, no. 4, pp. 439–456, 8 2008. [Online]. Available: /doi/pdf/10.1177/0165551507086989?download=true

[108] X. Dai, J. Hou, Q. Liu, Y. Xi, R. Tang, W. Zhang, X. He, J. Wang, and Y. Yu, "U-rank: Utility-oriented Learning to Rank with Implicit Feedback," in *International Conference on Information and Knowledge Management, Proceedings*, vol. 8, no. 20. Association for Computing Machinery, 10 2020, pp. 2373–2380. [Online]. Available: https://doi.org/10.1145/3340531.3412756

[109] P. Ingwersen and K. Jarvelin, *The turn : integration of information seeking and retrieval in context*. Berlin, Heidelberg: Springer, 2005.

[110] S. E. Robertson, "THE PROBABILITY RANKING PRINCIPLE IN IR," *Journal of Documentation*, vol. 33, no. 4, pp. 294–304, 1977.

[111] D. Harman, "Evaluation issues in information retrieval," *Information Processing and Management*, vol. 28, no. 4, pp. 439–440, 1992.

[112] V. Murdock, "Ellen Voorhees and Donna Harman (eds): TREC Experiment and Evaluation in Information Retrieval," *Information Retrieval*, vol. 11, no. 5, pp. 473–475, 10 2008. [Online]. Available: https://link.springer.com/article/10.1007/s10791-008-9064-x

[113] K. K. Karmaker, P. Sondhi, and C. Zhai, "Empirical Analysis of Impact of Query-Specific Customization of nDCG: A Case-Study with Learning-to-Rank Methods," in *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. New York, NY, USA: ACM, 2020. [Online]. Available: https://doi.org/10.1145/3340531.3417454

[114] N. Tax, S. Bockting, and D. Hiemstra, "A Cross-Benchmark Comparison of 87 Learning to Rank Methods," in *Information Processing and Management*, 2015. [Online]. Available: http://research.microsoft.com/en-us/um/beijing/projects/letor/letor4baseline.aspx

[115] T. Qin and T.-Y. Liu, "Introducing LETOR 4.0 Datasets," *Microsoft Research Asia*, 6 2013. [Online]. Available: http://arxiv.org/abs/1306.2597

[116] K. A. Hallgren, "Computing Inter-Rater Reliability for Observational Data: An Overview and Tutorial," Inversity of New Mexico, Tech. Rep. 1, 2012.

[117] K. Krippendorff, *Content analysis: An introduction to its methodology*, 2nd ed. Thousand Oaks: SAGE Publications Ltd, 2018.

[118] J. Cohen, "A Coefficient of Agreement for Nominal Scales," *Educational and Psychological Measurement*, vol. 20, no. 1, pp. 37–46, 1960.

[119] ——, "Weighted kappa: nominal scale agreement provision for scaled disagreement or partial credit." *Psychological Bulletin*, vol. 70, no. 4, pp. 213–220, 1968. [Online]. Available: https://psycnet.apa.org/record/1969-00069-001

[120] Fleiss. Joseph L, "Measuring nominal scale agreement among many raters." *Psychological bulletin*, vol. 76, no. 5, 1971. [Online]. Available: https://psycnet.apa.org/record/1972-05083-001

[121] A. J. Viera and J. M. Garrett, "Understanding interobserver agreement: The kappa statistic," *Family Medicine*, vol. 37, no. 5, pp. 360–363, 2005. [Online]. Available: http://www1.cs.columbia.edu/~julia/courses/CS6998/Interrater_agreement.Kappa_statistic.pdf

[122] M. L. McHugh, "Interrater reliability: the kappa statistic," *Biochemia Medica*, vol. 22, no. 3, p. 276, 2012. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3900052/

[123] H. Bast and I. Weber, "Type Less, Find More: Fast Autocompletion Search with a Succinct Index," *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, 2006. [Online]. Available: http://search.mpi-inf.mpg.de/wikipedia.

[124] European Union, *Study supporting the review of the application of the Web Accessibility Directive (WAD) VIGIE 2020-0656*. eu-accessibility-directive, 2022, no. October.

[125] A. Zhang, A. Goyal, W. Kong, H. Deng, A. Dong, Y. Chang, C. A. Gunter, and J. Han, "AdaQAC: Adaptive query auto-completion via implicit negative feedback," *SIGIR 2015 - Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 143–152, 8 2015. [Online]. Available: https://dl.acm.org/doi/10.1145/2766462.2767697

[126] "Text input – GOV.UK Design System." [Online]. Available: https://design-system.service.gov.uk/components/text-input/#use-the-autocomplete-attribute

[127] E. Scott, "9 UX Best Practice Design Patterns for Autocomplete Suggestions (Only 19% Get Everything Right) – Articles – Baymard Institute," 2022. [Online]. Available: https://baymard.com/blog/autocomplete-design

[128] R. W. White and G. Marchionini, "Examining the effectiveness of real-time query expansion," *Information Processing & Management*, vol. 43, no. 3, pp. 685–704, 5 2007.

[129] G. Kim, "Subword Language Model for Query Auto-Completion," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, 11 2019, pp. 5022–5032. [Online]. Available: https://aclanthology.org/D19-1507

[130] N. Yadav, R. Sen, D. N. Hill, A. Mazumdar, and I. S. Dhillon, "Session-Aware Query Auto-completion using Extreme Multi-Label Ranking," *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 3835–3844, 8 2021. [Online]. Available: https://dl.acm.org/doi/10.1145/3447548.3467087

[131] Y. Chang and H. Demg, *Query Understanding for Search Engines*, ser. The Information Retrieval Series. Jilin: Springer International Publishing, 2020, vol. 46. [Online]. Available: http://link.springer.com/10.1007/978-3-030-58334-7

[132] S. Whiting and J. M. Jose, "Recent and robust query auto-completion," *WWW 2014 - Proceedings of the 23rd International Conference on World Wide Web*, pp. 971–981, 4 2014. [Online]. Available: https://dl.acm.org/doi/pdf/10.1145/2566486.2568009

[133] S. Wang, W. Guo, H. Gao, and B. Long, "Efficient Neural Query Auto Completion," *International Conference on Information and Knowledge Management, Proceedings*, pp. 2797–2804, 10 2020. [Online]. Available: https://dl.acm.org/doi/10.1145/3340531.3412701

[134] A. Rahangdale and S. Raut, "Deep Neural Network Regularization for Feature Selection in Learning-to-Rank," *IEEE Access*, vol. 7, pp. 53 988–54 006, 2019.

[135] J. Guo, Y. Fan, Q. Ai, and W. B. Croft, "A deep relevance matching model for Ad-hoc retrieval," in *International Conference on Information and Knowledge Management, Proceedings*, vol. 24-28-Octo. Association for Computing Machinery, 10 2016, pp. 55–64. [Online]. Available: http://dx.doi.org/10.1145/2983323.2983769

[136] C. Daly, "Learning to Rank: Performance and Practical Barriers to Deployment in Enterprise Search," in *3rd Asia Conference on Information Engineering (ACIE)*. IEEE, 2023, pp. 21–26.

[137] N. Fiorini and Z. Lu, "Personalized neural language models for real-world query auto completion," *NAACL HLT 2018 - 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, vol. 3, pp. 208–215, 2018.

[138] B. Hayes, "The Britney Spears Problem," *American Scientist*, vol. 96, no. 4, p. 274, 2008.

[139] H. Duan and B. J. Hsu, "Online spelling correction for query completion," *Proceedings of the 20th International Conference on World Wide Web, WWW 2011*, pp. 117–126, 2011. [Online]. Available: https://dl.acm.org/doi/10.1145/1963405.1963425

[140] T. Grainger, D. Turnbull, and M. Irwin, *AI Powered Search*, 2025th ed., M. Michaels and J. Guthrie, Eds. Shelter Island: Manning Publications Co., 2025. [Online]. Available: https://livebook.manning.com/#!/book/ai-powered-search/discussion

[141] A. Trotman, A. Puurula, and B. Burgess, "Improvements to BM25 and Language Models Examined," *Proceedings of the 2014 Australasian Document Computing Symposium*, 2014. [Online]. Available: http://dx.doi.org/10.1145/2682862.2682863

[142] M. Li, X. Zhang, J. Xin, H. Zhang, J. Lin, and D. R. Cheriton, "Certified Error Control of Candidate Set Pruning for Two-Stage Relevance Ranking," in *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2022, pp. 333–345. [Online]. Available: https://aclanthology.org/2022.emnlp-main.23

[143] C. Macdonald, R. L. Santos, and I. Ounis, "The whens and hows of learning to rank for web search," *Information Retrieval*, vol. 16, no. 5, pp. 584–628, 10 2013. [Online]. Available: https://www.academia.edu/2686818/The_whens_and_hows_of_learning_to_rank_for_web_search

[144] D. Lillis, R. Collier, F. Toolan, and J. Dunnion, "ProbFuse: A probabilistic approach to data fusion," *Proceedings of the Twenty-Ninth Annual International ACM SIGIR Conference on Research and Development*

in Information Retrieval, vol. 2006, pp. 139–146, 2006. [Online]. Available: https://dl.acm.org/doi/10.1145/1148170.1148197

[145] J. Zilles, G. Lucca, and E. N. Borges, "A Literature Review on Methods for Learning to Rank," *International Conference on Enterprise Information Systems, ICEIS - Proceedings*, vol. 1, no. Iceis, pp. 545–552, 2022.

[146] X. Wang and M. Bendersky, "Google AI Blog: TF-Ranking: A Scalable TensorFlow Library for Learning-to-Rank," 2018. [Online]. Available: https://ai.googleblog.com/2018/12/tf-ranking-scalable-tensorflow-library.html

[147] J. Segalovich, "Machine learning in search quality at yandex," in *33rd Annual ACM SIGIR Conference*, 2010.

[148] Google, "Search and SEO Blog — Google Search Central — Google Search Central Blog — Google for Developers," 2025. [Online]. Available: https://developers.google.com/search/blog

[149] H. Li, "A Short Introduction to Learning to Rank," *IEICE Transactions*, vol. 94-D, pp. 1854–1862, 10 2011.

[150] S. Singh, S. Farfade, and P. M. Comar, "Multi-Objective Ranking to Boost Navigational Suggestions in eCommerce AutoComplete," *ACM Web Conference 2023 - Companion of the World Wide Web Conference, WWW 2023*, pp. 469–474, 2023.

[151] J. Xu, Z. Wei, L. Xia, Y. Lan, D. Yin, X. Cheng, and J.-R. Wen, "Reinforcement Learning to Rank with Pairwise Policy Gradient," in *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, vol. 10. New York, NY, USA: ACM, 2020, p. 10. [Online]. Available: https://doi.org/10.1145/3397271.3401148

[152] C. C. Vogt and G. W. Cottrell, "Fusion Via a Linear Combination of Scores," *Information Retrieval*, vol. 1, no. 3, pp. 151–173, 1999. [Online]. Available: https://dl.acm.org/doi/10.1023/A%3A1009980820262

[153] D. R. Cox, "The Regression Analysis of Binary Sequences," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 21, no. 1, pp. 238–238, 1 1959.

[154] Q. Ai, T. Yang, H. Wang, and J. Mao, "Unbiased Learning to Rank: Online or Offline?" *arXiv*, vol. 9, p. 29, 4 2020. [Online]. Available: http://arxiv.org/abs/2004.13574

[155] "Learning To Rank :: Apache Solr Reference Guide," 2025. [Online]. Available: https://solr.apache.org/guide/solr/latest/query-guide/learning-to-rank.html

[156] A. Baheti, A. Ritter, J. Li, and B. Dolan, "Generating More Interesting Responses in Neural Conversation Models with Distributional Constraints," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Brussels: Association for Computational Linguistics, 2018, pp. 3970–3980. [Online]. Available: https://www.ranks.nl/stopwordshttps://www.aclweb.org/anthology/D18-1431

[157] L. Zhang, S. Mille, Y. Hou, D. Deutsch, E. Clark, Y. Liu, S. Mahamood, S. Gehrmann, M. Clinciu, K. Chandu, and J. Sedoc, "A Needle in a Haystack: An Analysis of High-Agreement Workers on MTurk for Summarization," *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*, vol. 1, no. Long Papers, pp. 14 944–14 982, 2023. [Online]. Available: https://www.mturk.com/

[158] E. M. Voorhees, "The TREC robust retrieval track," *ACM SIGIR Forum*, vol. 39, no. 1, pp. 11–20, 2005.

[159] F. Radlinski and T. Joachims, "Query chains: Learning to rank from implicit feedback," *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 239–248, 5 2005. [Online]. Available: http://arxiv.org/abs/cs/0605035

[160] H. Wang, R. Langley, S. Kim, E. McCord-Snook, and H. Wang, "Efficient exploration of gradient space for online learning to rank," in *41st International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2018*, 2018.

[161] Q. Ai, J. Mao, Y. Liu, and W. B. Croft, "Unbiased Learning to Rank: Theory and Practice," in *Proceedings of the 2018 ACM SIGIR International Conference on Theory of Information Retrieval*, ser. ICTIR '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 1–2. [Online]. Available: https://doi.org/10.1145/3234944.3234980

[162] I. H. Woodhouse, "On 'ground' truth and why we should abandon the term," *https://doi.org/10.1117/1.JRS.15.041501*, vol. 15, no. 4, p. 041501, 11 2021. [Online]. Available: https://www.spiedigitallibrary.org/journals/journal-of-applied-remote-sensing/volume-15/issue-4/041501/On-ground-truth-and-why-we-should-abandon-the-term/10.1117/1.JRS.15.041501.full

[163] R. Savolainen, "Cognitive barriers to information seeking: A conceptual analysis:," *Journal of Information Science*, vol. 41, no. 5, pp. 613–623, 5 2015. [Online]. Available: https://journals.sagepub.com/doi/10.1177/0165551515587850

[164] C. Cleverdon, "The cranfield tests on index language devices," *Aslib Proceedings*, vol. 19, no. 6, pp. 173–194, 1 1967. [Online]. Available: https://doi.org/10.1108/eb050097

[165] P. Röttger, B. Vidgen, D. Hovy, and J. B. Pierrehumbert, "Two Contrasting Data Annotation Paradigms for Subjective NLP Tasks," in *North Americal Association of Computational Linguistics*, Seattle, 2022.

[166] F. Diaz, "Learning to rank with labeled features," in *ICTIR 2016 - Proceedings of the 2016 ACM International Conference on the Theory of Information Retrieval*. Association for Computing Machinery, Inc, 9 2016, pp. 41–44.

[167] E. Yilmaz and S. Robertson, "Deep versus shallow judgments in learning to rank," *Proceedings - 32nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2009*, no. 10, pp. 662–663, 2009.

[168] C. J. Burges, R. Ragno, and Q. Viet Le, "Learning to Rank with Nonsmooth Cost Functions," in *Proceedings of the 20th International Conference on Neural Information Processing Systems*. Cambridge: MIT Press, 2006, pp. 193–200.

[169] L. Pang, Y. Lan, J. Guo, J. Xu, J. Xu, and X. Cheng, "DeepRank: A New Deep Architecture for Relevance Ranking in Information Retrieval," in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management.*, New York, 2017.

[170] S. Siebert, S. Dinesh, and S. Feyer, "Extending a research-paper recommendation system with scientometric measures," in *CEUR Workshop Proceedings*, vol. 1823. CEUR-WS, 2017, pp. 112–121.

[171] H. Li, J. Chen, W. Su, Q. Ai, and Y. Liu, "Towards Better Web Search Performance: Pre-training, Fine-tuning and Learning to Rank," *arXiv*, 2 2023. [Online]. Available: http://arxiv.org/abs/2303.04710

[172] T. Russell-Rose, J. Lamantia, and M. Burrell, "A Taxonomy of Enterprise Search," in *European Workshop on Human-Computer Interaction and Information Retrieval*, 2011. [Online]. Available: https://api.semanticscholar.org/CorpusID:957977

[173] X. Geng, T. Y. Liu, T. Qin, and H. Li, "Feature selection for ranking," *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR'07*, pp. 407–414, 2007.

[174] J.-Y. Yeh and C.-J. Tsai, "Graph-based Feature Selection Method for Learning to Rank; Graph-based Feature Selection Method for Learning to Rank," in *2020 the 6th International Conference on Communication and Information Processing*. New York, NY, USA: ACM, 2020. [Online]. Available: https://doi.org/10.1145/3442555.3442567

[175] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik, "Gene Selection for Cancer Classification using Support Vector Machines," *Machine Learning 2002 46:1*, vol. 46, no. 1, pp. 389–422, 2002. [Online]. Available: https://link.springer.com/article/10.1023/A:1012487302797

[176] B. Dimanov and M. Jamnik, "Step-wise Sensitivity Analysis: Identifying Partially Distributed Representations for Interpretable Deep Learning," *ICLR 2019 Debugging Machine Learning Models Workshop*, 2018.

[177] H. Yang and T. Gonçalves, "Field features: The impact in learning to rank approaches," *Applied Soft Computing*, vol. 138, p. 110183, 5 2023.

[178] A. Collins, D. Tkaczyk, A. Aizawa, and J. Beel, "A Study of Position Bias in Digital Library Recommender Systems," *arXiv*, 2 2018.

[179] T. Joachims, H. Hembrooke, F. Radlinski, and G. Gay, "Evaluating the Accuracy of Implicit Feedback from Clicks and Query Reformulations in Web Search," in *ACM Transactions on Information Systems*, vol. 25, 2007.

[180] J. R. Griffiths, F. Johnson, and R. J. Hartley, "User satisfaction as a measure of system performance:," *http://dx.doi.org/10.1177/0961000607080417*, vol. 39, no. 3, pp. 142–152, 6 2016. [Online]. Available: https://journals.sagepub.com/doi/10.1177/0961000607080417

[181] A. Agarwal, I. Zaitsev, K. Takatsu, and T. Joachims, "A general framework for counterfactual learning-to-rank," in *SIGIR 2019 - Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, vol. 10, no. 19. Association for Computing Machinery, Inc, 7 2019, pp. 5–14. [Online]. Available: https://doi.org/10.1145/3331184.3331202

[182] L. Yu, Y. Wang, X. Sun, K. Bi, and J. Guo, "Feature-Enhanced Network with Hybrid Debiasing Strategies for Unbiased Learning to Rank," *arXiv*, 2 2023. [Online]. Available: https://arxiv.org/abs/2302.07530v1

[183] P. Petrescu, "Google Organic Click-Through Rates in 2014," 2015. [Online]. Available: https://moz.com/blog/google-organic-click-through-rates-in-2014

[184] H. Oosterhuis and M. De Rijke, "Unifying Online and Counterfactual Learning to Rank A Novel Counterfactual Estimator that Effectively Utilizes Online Interventions," in *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*. New York, NY, USA: ACM, 2021. [Online]. Available: https://doi.org/10.1145/3437963.3441794

[185] Z. Hu, Q. Peng, Y. Wang, and H. Li, "Unbiased LambdaMart: An unbiased pairwise learning-to-rank algorithm," in *The Web Conference 2019 - Proceedings of the World Wide Web Conference, WWW 2019*. Association for Computing Machinery, Inc, 5 2019, pp. 2830–2836.

[186] F. Xia, T.-Y. Liu, J. Wang, W. Zhang, and H. Li, "Listwise approach to learning to rank," *ICML '08: Proceedings of the 25th international conference on Machine learning*, pp. 1192–1199, 2008. [Online]. Available: https://dl.acm.org/doi/10.1145/1390156.1390306

[187] J. Peng, C. MacDonald, and I. Ounis, "Learning to select a ranking function," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 5993 LNCS, 2010.

[188] T. Teofili, "Deep Learning for Search," pp. 1–34, 2018. [Online]. Available: https://www.manning.com/books/deep-learning-for-search

[189] O. Khattab and M. Zaharia, "ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT," *SIGIR 2020 - Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 39–48, 2020.

[190] B. Mitra and N. Craswell, "An Introduction to Neural Information Retrieval," *Foundations and Trends® in Information Retrieval*, vol. 13, no. 1, pp. 1–126, 12 2018. [Online]. Available: https://www.microsoft.com/en-us/research/publication/introduction-neural-information-retrieval/

[191] L. Dostert and P. Garvin, "Machine Translation of Languages." *Technology REview*, no. 56(7), pp. 21–24, 1954.

[192] J. Devlin, M.-W. Chang, K. Lee, K. T. Google, and A. I. Language, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," *Proceedings of the 2019 Conference of the North*, pp. 4171–4186, 2019. [Online]. Available: https://aclanthology.org/N19-1423/

[193] I. Beltagy, K. Lo, and A. Cohan, "SciBERT: A Pretrained Language Model for Scientific Text," *EMNLP-IJCNLP 2019 - 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing, Proceedings of the Conference*, pp. 3615–3620, 3 2019. [Online]. Available: https://arxiv.org/pdf/1903.10676

[194] J. Lee, W. Yoon, S. Kim, D. Kim, S. Kim, C. H. So, and J. Kang, "BioBERT: a pre-trained biomedical language representation model for biomedical text mining," *Bioinformatics*, vol. 36, no. 4, pp. 1234–1240, 2 2020. [Online]. Available: https://dx.doi.org/10.1093/bioinformatics/btz682

[195] E. Reid, "How Google is improving Search with Generative AI." [Online]. Available: https://blog.google/products/search/generative-ai-search/

[196] H. Fang, T. Tao, and C. Zhai, "Diagnostic evaluation of information retrieval models," *ACM Transactions on Information Systems*, vol. 29, no. 2, 4 2011. [Online]. Available: http://doi.acm.org/10.1145/1961209.1961210

[197] A. Severyn and A. Moschittiy, "Learning to rank short text pairs with convolutional deep neural networks," in *SIGIR 2015 - Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*. Association for Computing Machinery, Inc, 8 2015, pp. 373–382. [Online]. Available: http://dx.doi.org/10.1145/2766462.2767738.

[198] P. S. Huang, X. He, J. Gao, L. Deng, A. Acero, and L. Heck, "Learning deep structured semantic models for web search using clickthrough data," in *International Conference on Information and Knowledge Management, Proceedings*, 2013, pp. 2333–2338.

[199] Y. Shen, X. He, J. Gao, L. Deng, and G. Mesnil, "Learning semantic representations using convolutional neural networks for web search," in *WWW 2014 Companion - Proceedings of the 23rd International Conference on World Wide Web*. Association for Computing Machinery, Inc, 4 2014, pp. 373–374. [Online]. Available: http://dx.doi.org/10.1145/2567948.2577348

[200] S. Pudasaini, L. Miralles-Pechuán, D. Lillis, and M. L. Salvador, "Survey on Plagiarism Detection in Large Language Models: The Impact of ChatGPT and Gemini on Academic Integrity," *Journal of Academic Ethics*, vol. 11, no. 04, 6 2024. [Online]. Available: https://arxiv.org/pdf/2407.13105

[201] A. Belfathi, Y. Gallina, N. Hernandez, R. Dufour, and L. Monceaux, "Language Model Adaptation to Specialized Domains through Selective Masking based on Genre and Topical Characteristics," *arXiv*, vol. 2402.12036, 2 2024. [Online]. Available: http://arxiv.org/abs/2402.12036

[202] B. Wang and D. Klabjan, "An Attention-Based Deep Net for Learning to Rank," *unpublished*, 2 2017. [Online]. Available: http://arxiv.org/abs/1702.06106

[203] R. K. Pasumarthi, H. Zhuang, X. Wang, M. Bendersky, and M. Najork, "Permutation Equivariant Document Interaction Network for Neural Learning to Rank," in *ICTIR 2020 - Proceedings of the 2020 ACM SIGIR International Conference on Theory of Information Retrieval*. New York, NY, USA: ACM, 2020, pp. 145–148. [Online]. Available: https://doi.org/10.1145/3409256.3409819

[204] Y. Mehdi, "Reinventing search with a new AI-powered Microsoft Bing and Edge, your copilot for the web - The Official Microsoft Blog," 2023. [Online]. Available: https://blogs.microsoft.com/blog/2023/02/07/

[205] European Union, "Regulation 2024/1689," *Official Journal of the European Union*, vol. 1689, no. 3, pp. 1–144, 2024.

[206] "Organic CTR - AWR SEO Guide," 2021. [Online]. Available: https://www.advancedwebranking.com/seo/organic-ctr

[207] "Apache Lucene, Version 10.0.0," 2024. [Online]. Available: https://lucene.apache.org/

[208] "Elasticsearch version 8.16.1," 2024. [Online]. Available: https://www.elastic.co/elasticsearch

[209] S. Robertson, S. Walker, M. Jones, and M. Hancock-Beaulieu Gatford, "Okapi at TREC-3," in *Proceedings of the Third Text REtrieval Conference*, 1994.

[210] T. G. Potter and Timothy, *Solr in Action MEAP v1*. Manning, 2012. [Online]. Available: https://beta.manning.com/dashboard%5Cnpapers2://publication/uuid/17DA7ECA-A9EC-4DE8-924B-5016C8859C40

[211] L. Page and S. Brin, "The anatomy of a large-scale hypertextual Web search engine," *Computer Networks*, vol. 30, no. 1-7, pp. 107–117, 1 1998.

[212] "The DisMax Query Parser — Apache Solr Reference Guide 7.7." [Online]. Available: https://solr.apache.org/guide/7_7/the-dismax-query-parser.html

[213] B. Dean, "We Analyzed 11.8 Million Google Search Results. Here's What We Learned About SEO," Semrush, Tech. Rep., 2024. [Online]. Available: https://backlinko.com/search-engine-ranking

[214] A. Paulsson, A. Márki, and H. Gustavsson, "Using Clickthrough Data To Optimize Search Result Ranking Användning Av Clickthrough Data För Att Optimera Rankning Av Sökresultat," Ph.D. dissertation, Universiy of Skövde, Sweden, 2017.

[215] A. Białecki, "Implementing Click-through Relevance Ranking in Solr and LucidWorks Enterprise," LucidWorks, Tech. Rep., 2011.

[216] T. Moon, L. Li, W. Chu, C. Liao, Z. Zheng, and Y. Chang, "Online learning for recency search ranking using real-time user feedback," *International Conference on Information and Knowledge Management, Proceedings*, pp. 1501–1504, 2010.

[217] C. van der Lee, A. Gatt, E. van Miltenburg, S. Wubben, and E. Krahmer, "Best practices for the human evaluation of automatically generated text," *INLG 2019 - 12th International Conference on Natural Language Generation, Proceedings of the Conference*, pp. 355–368, 2019. [Online]. Available: https://aclanthology.org/W19-8643

[218] "Standard Query Parser :: Apache Solr Reference Guide." [Online]. Available: https://solr.apache.org/guide/solr/latest/query-guide/standard-query-parser.html#boosting-a-term-with

[219] "SolrLogging - Solr - Apache Software Foundation." [Online]. Available: https://cwiki.apache.org/confluence/display/solr/SolrLogging

[220] "Learning To Rank :: Apache Solr Reference Guide." [Online]. Available: https://solr.apache.org/guide/solr/latest/query-guide/learning-to-rank.html

[221] D. Kelly and J. Teevan, "Implicit feedback for inferring user preference," *ACM SIGIR Forum*, vol. 37, no. 2, pp. 18–28, 9 2003.

[222] G. Jawaheer, M. Szomszor, and P. Kostkova, "Comparison of implicit and explicit feedback from an online music recommendation service," *Proceedings of the 1st International Workshop on Information Heterogeneity and Fusion in Recommender Systems, HetRec 2010*, pp. 47–51, 2010.

[223] H. Akoglu, "User's guide to correlation coefficients," *Turkish Journal of Emergency Medicine*, vol. 18, no. 3, p. 91, 9 2018. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6107969/

[224] J. C. Klie, B. Webber, and I. Gurevych, "Annotation Error Detection: Analyzing the Past and Present for a More Coherent Future," *Computational Linguistics*, vol. 49, no. 1, pp. 157–198, 3 2023. [Online]. Available: https://dx.doi.org/10.1162/coli_a_00464

[225] J. Frost, "Introduction to Bootstrapping in Statistics with an Example - Statistics By Jim." [Online]. Available: https://statisticsbyjim.com/hypothesis-testing/bootstrapping/

[226] P. H. Westfall, "On using the bootstrap for multiple comparisons," *Journal of Biopharmaceutical Statistics*, vol. 21, no. 6, pp. 1187–1205, 2011. [Online]. Available: http://statisticsbyjim.com/hypothesis-testing/bootstrapping/

[227] C. Daly and L. Hederman, "JARGES : Detecting and Decoding Jargon for Enterprise Search," *KDIR 2025 : 17th International Conference on Knowledge Discovery and Information Retrieval*, vol. 1, no. Ic3k, pp. 357–363, 2025.

[228] R. Kohavi, D. Tankg, and Y. Xu, *Trustworthy Online Controlled Experiments: A Practical Guide to A/B Testing - Ron Kohavi, Diane Tang, Ya Xu - Google Books*. Cambridge: Cambridge University Press, 2020.

[229] K. S. Jones, "A statistical interpretation of term specificity and its application in retrieval," *Journal of Documentation*, vol. 28, no. 1, pp. 11–21, 1972.

[230] F. Pedregosa, V. Michel, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, J. Vanderplas, D. Cournapeau, F. Pedregosa, G. Varoquaux, A. Gramfort, B. Thirion, O. Grisel, V. Dubourg, A. Passos, M. Brucher, M. Perrot andÉdouardand, A. Duchesnay, and F. Duchesnay EDOUARD-DUCHESNAY, "Scikit-learn: Machine Learning in Python," *The Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 11 2011. [Online]. Available: https://dl.acm.org/doi/pdf/10.5555/1953048.2078195

[231] C. Daly and L. Hederman, "Enterprise Search: Learning to Rank with Click-Through Data as a Surrogate for Human Relevance Judgements," in *15th International Conference on Knowledge Discovery and Information Retrieval (KDIR)*, 2023, pp. 240–247.

[232] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is All You Need," in *Advances in Neural Information Processing Systems*, 2017, p. 5998–6008. [Online]. Available: https://arxiv.org/pdf/1706.03762.pdf

[233] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Reference Format*, vol. 41, no. 15, 2009. [Online]. Available: http://doi.acm.org/10.1145/1541880.1541882

[234] C. Lucchese, C. I. Muntean, F. M. Nardini, R. Perego, and S. Trani, "RankEval: An evaluation and analysis framework for learning-To-rank solutions," *SIGIR 2017 - Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, no. August, pp. 1281–1284, 2017.

[235] Y. Li, A. Dong, H. Wang, H. Deng, Y. Chang, and C. X. Zhai, "A two-dimensional click model for query auto-completion," *SIGIR 2014 - Proceedings of the 37th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 455–464, 2014. [Online]. Available: https://dl.acm.org/doi/10.1145/2600428.2609571

[236] Z. Bar-Yossef and N. Kraus, "Context-sensitive query auto-completion," *Proceedings of the 20th International Conference on World Wide Web, WWW 2011*, pp. 107–116, 2011. [Online]. Available: https://dl.acm.org/doi/10.1145/1963405.1963424

[237] Y. Li, A. Dong, H. Wang, H. Deng, Y. Chang, and C. X. Zhai, "A two-dimensional click model for query auto-completion," *SIGIR 2014 - Proceedings of the 37th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 455–464, 2014.

[238] L. Project, "RankLib: A Library for Learning to Rank," \url{https://sourceforge.net/p/lemur/wiki/RankLib/}, 2017.

[239] Princeton University, "About WordNet." 2010. [Online]. Available: https://wordnet.princeton.edu/

[240] S. Balloccu, P. Schmidtová, M. Lango, and O. Dušek, "Leak, Cheat, Repeat: Data Contamination and Evaluation Malpractices in Closed-Source LLMs," *EACL 2024 - 18th Conference of the European Chapter of the Association for Computational Linguistics, Proceedings of the Conference*, vol. 1, pp. 67–93, 2 2024. [Online]. Available: https://arxiv.org/abs/2402.03927v2

[241] M. Dalcastagne and G. D. Fabbrizio, "Hyperparameter Optimization for Search Relevance in E-Commerce," in *KDIR 2024 : 16th International Conference on Knowledge Discovery and Information Retrieval*, Porto, 2024.

[242] S. Singh, S. Farfade, and P. M. Comar, "Multi-Objective Ranking to Boost Navigational Suggestions in eCommerce AutoComplete," *ACM Web Conference 2023 - Companion of the World Wide Web Conference, WWW 2023*, pp. 469–474, 2023.

[243] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, vol. 1, pp. 4171–4186, 2019. [Online]. Available: https://aclanthology.org/N19-1423/

[244] M. Hoffmann, S. Boysel, F. Nagle, S. Peng, and K. Xu, "Generative AI and the Nature of Work," *Harvard Business Review*, 10 2024. [Online]. Available: https://papers.ssrn.com/abstract=5007084

[245] B. Liu and B. Wang, "Pairwise learning for personalized ranking with noisy comparisons," *Information Sciences*, vol. 623, pp. 242–257, 4 2023.

[246] S. Tahery and S. Farzi, "Customized query auto-completion and suggestion — A review," *Information Systems*, vol. 87, p. 101415, 1 2020.

[247] C. Daly and L. Hederman, "Learning to Rank for Query Auto-Complete with Language Modelling in Enterprise Search," in *16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (KDIR)*, 2024, p. 12.

[248] D. Lewandowski, J. Drechsler, and S. Von MacH, "Deriving query intents from web search engine queries," *Journal of the American Society for Information Science and Technology*, vol. 63, no. 9, pp. 1773–1788, 9 2012.

[249] T. Russell-Rose, J. Lamantia, and M. Burrell, "A taxonomy of enterprise search," *CEUR Workshop Proceedings*, vol. 763, pp. 15–18, 2011.

Appendices

A

ES and WS DIFFERENCES

This appendix highlights the main variations between Enterprise Search (ES) and Web Search (WS). It functions as a detailed compilation of differences noted during the course of this study. Given the absence of a similar compilation elsewhere, it is included here as an appendix. The list below is in no particular order.

DIFF-1 ES: Data components may be both internally and externally accessible. WS: Data components are exclusively externally crawlable (discussed in Section §2.1.2 and 2.4.4).

DIFF-2 ES content may be indexed from multiple sources including corporate directories and intranet document repositories. WS data has been typically crawled from web pages (discussed in Section §2.2.10).

DIFF-3 ES documents may be stored without search optimisation (no tags, hrefs, or even titles). Document may have been placed on an intranet drive purely for storage, with no intention of being made searchable. Different ranking functions will therefore be required (discussed in Section §3.3).

DIFF-4 ES frequently handles alphanumeric searches for organisation, including specific data like usernames, course codes, tracking numbers, and purchasing codes (discussed in Section §2.1.3).

DIFF-5 The typical number of tokens in a query is generally smaller for ES compared to WS. Average query length: 2.187 words in The University's ES service (measured 20-Dec-2023). In WS, users have discovered that search engines can successfully handle full length sentences / questions as a query. Similarly, in The University's ES system, only 0.84% of queries are in the form of a question. This compares to 14.1% for WS, based on analysis of question-word frequency (discussed in Section §3.4.2).

DIFF-6 An effective ES service helps users avoid navigating hierarchical menus by providing a single search interface for organisational assets. The web offers no such top level portal (at least not since 'Yahoo! Directory' was retired in 2014). White defines ES as a service that "enables employees to find all company information without needing to know where it's stored" [54]. ES exists on a continuum between web search and desktop search [6] (discussed in Section §2.1.1).

DIFF-7 Query Auto-Completion (QAC) helps users formulate queries when they understand their intent but struggle with the organisation's specific terminology. This is especially useful for newcomers (prior to orientation and onboarding, etc and discussed in Section §2.1.8 and §4.5).

DIFF-8 ES research uses various terms including 'workplace search systems', 'enterprise document search', and 'Enterprise AI search'. Lewandowski notes the lack of consistent terminology between information scientists and WS search engine optimisers [248](discussed in Section §2.1.1).

DIFF-9 Crowdsourcing platforms like Amazon Mechanical Turk (AMT) can be employed for WS annotation, but they cannot access intranet content. ES relevance judgments require internal domain experts to achieve the equivalent (discussed in Section §3.4.3).

DIFF-10 While WS relevance judgments show low inter-annotator agreement, ES benefits from higher consensus as the annotators are internal domain experts or at least have an approximately shared understanding of the organisation (discussed in Section §3.4.3 and in Section §2.2.20).

DIFF-11 Scale difference: ES indices are typically of size measured in hundreds of MB. This is tiny compared to WS's petabyte scale systems. The Google search index is estimated to exceed 100 PB in size (discussed in Section §2.1.7).

DIFF-12 Resource difference: Google employs 130,000 staff (20,000 engineers dedicated to search). White [72] notes that most organisations have a significant shortage of people with skills to support and manage ES (discussed in Section §2.5.7).

DIFF-13 ES query intent often involves navigational queries ('lookup searches') with typically one perfect result, as classified by Broder and Russell-Rose et al [57, 249]. Hence, ranking scores (in terms of the nDCG metric) will be higher for ES. For example, a search for the name of a department (or school etc) is expected to return the homepage of that department at top billing (discussed in Section §2.1.4).

DIFF-14 WS QAC may use session based queries and collaborative filtering, considering previous queries even unrelated to the current prefix. Some ES organisations shy away from profiling, even at the session level (discussed in Section §2.2.11).

DIFF-15 ES users are typically employees accessing via workstations (desktops/laptops), with fewer mobile users than WS, affecting the weighting of features like `userFromMobile` (discussed in Section §2.2.10).

DIFF-16 ES QAC helps prevent zero-result queries, which is a potentially common occurrence in smaller ES corpora. It has become increasing rare to submit a search to a commercial WS provider and receive 'no results found'. An ES QAC service must produce 'sensical candidates' (discussed in Section §1.1.2 and §4.7.1).

DIFF-17 Molnar notes ES challenges: users demand perfect results but don't tag documents, misuse collaboration tools as file systems, and create content silos [9]. This is the opposite of what SEO engineers strive for in WS (discussed in Section §2.1.2).

DIFF-18 Some organisations guide publishers to use document tags for search and QAC, while others have untagged legacy files never intended for search (discussed in Section §4.3).

DIFF-19 For The University's ES service, 91.8% of search terms are long-tail keywords. The top 500 WS terms account for 8.4% of search volume, versus 38.5% in The University's ES dataset (discussed in Section §2.1.7).

DIFF-20 Location based ranking is generally inappropriate (or not permitted) for ES, except perhaps in multinational organisations or on exceptionally large campuses (discussed in Section §2.2.12 and §5.8.2.)

DIFF-21 Personalisation and Location based ranking both raise GDPR concerns in ES, requiring transparency and consent to prevent employee profiling (discussed in Section §5.8).

DIFF-22 Effective QAC in ES reduces unnecessary queries, improving server load and retrieval performance. This is not a problem for WS, where there are more resources available to deal with server load issues (discussed in Section §2.3.2).

DIFF-23 WS requires strict autocomplete content filtering; ES doesn't need such policies as employees are accountable for content (discussed in Section §3.2.3).

DIFF-24 WS faces legal challenges over content use (e.g., the infamous case of Perplexity's news aggregation), unlike ES where crawling and indexed content is applied within the organisation (discussed in Section §2.2.5 and §2.2.7).

DIFF-25 Unlike WS, ES has no SEO industry, and document publishers/content-providers do not generally compete for rankings. View count manipulation (possible via scripts) is common in WS but rare in ES organisational contexts. ES content publishers are rarely compete for top-billing (discussed in Section §3.2.3 and §3.2.2).

DIFF-26 ES document paths include network drive letters (e.g., `W:\path\to\document`) alongside URLs, unlike WS which uses forward-slash-only paths (discussed in Section §4.3).

B

JARGES: Detecting and Decoding Jargon for Enterprise Search.

Paper IV

Colin Daly[1,2][a], Lucy Hederman[1,2][b]
[1]*The ADAPT SFI Research Centre, Ireland*
[2]*School of Computer Science and Statistics, Trinity College Dublin, Ireland*
{*dalyc24, hederman*}*@tcd.ie,*

Keywords: Enterprise Search, Learning to Rank, Jargon, Language Modelling

Abstract: Newcomers to an organisation often struggle with unfamiliar internal vocabulary, which can affect their ability to retrieve relevant information. Enterprise Search (ES) systems frequently underperform when queries contain jargon or terminology that is specific to the organisation. This paper introduces 'JARGES', a novel feature for detecting and decoding jargon for ES. It is designed to enhance a ranking model combining Learning to Rank (LTR) and transformer-based synonym expansion. The ranking model is evaluated using the `ENTRP-SRCH` dataset. Our experiments showed, however, that the JARGES feature yielded no significant improvement over the baseline (nDCG@10 = 0.964, $\Delta = 0.001$, $p > 0.05$). These failures are likely due to the dataset's lack of jargon-rich pairs. This highlights the need for larger ES datasets derived from diverse, real world click-through data or other implicit feedback to detect subtle ranking signals.

1 INTRODUCTION

Enterprise Search (ES) plays a key role in enabling organisations to efficiently access internal knowledge. But ES systems are often perceived to be 'relevance-blind' [Turnbull and Berryman, 2016] as users "cannot find the information they seek within an acceptable amount of time, using their own enterprise search applications" [Bentley, 2011]. For newcomers, this challenge is compounded by the prevalence of specialised jargon and terminology unique to an organisation.

Such jargon, while familiar to long-term employees, can impede effective searches by new staff or external stakeholders. This is sometimes referred to as 'vocabulary mismatch' [Ganguly et al., 2015]. Query formulation using incorrect terms or phraseology can lead to poor ranking and recall of query results, with a potential reduction in staff productivity.

To address this problem, we propose a ranking model that integrates Learning to Rank (LTR) and Language Modelling (LM). The LM component is called 'JARGES' (JARGon for Enterprise Search), and is designed to detect and decode jargon in ES queries and corpora. The detection and decoding stages of JARGES are demonstrated in Figure 1. This study aims to evaluate the effectiveness of this approach in improving search result rankings, particularly for content rich in organisational terminology. To test this hypothesis, we perform a quantitative evaluation of the performance of an LTR ranking model with and without JARGES using the LTR-formatted `ENTRP-SRCH` dataset (2,544 human-annotated Q-D pairs) [Daly, 2023]. We subsequently perform a qualitative analysis of the decoded jargon terms via their contextual synonyms.

While the quantitative experiments did not result in an improved ranking performance, the nuances of enterprise specific terminology may have been better captured had a larger and more diverse ES dataset been available. Moreover, JARGES offers a promising direction for decoding organisational language and semantic search.

2 RELATED WORK

Enterprise Search (ES) can be simply defined as finding the information needed from within an organisation [Bentley, 2011] or as a service that "enables employees to find all the information the company possesses without knowing where the information is stored" [White, 2015].

Jargon is enterprise specific vocabulary that employees/members can understand. It encompasses words, phrases, expressions, and idioms that are not

[a] https://orcid.org/0000-0001-7218-7765
[b] https://orcid.org/0000-0001-6073-4063

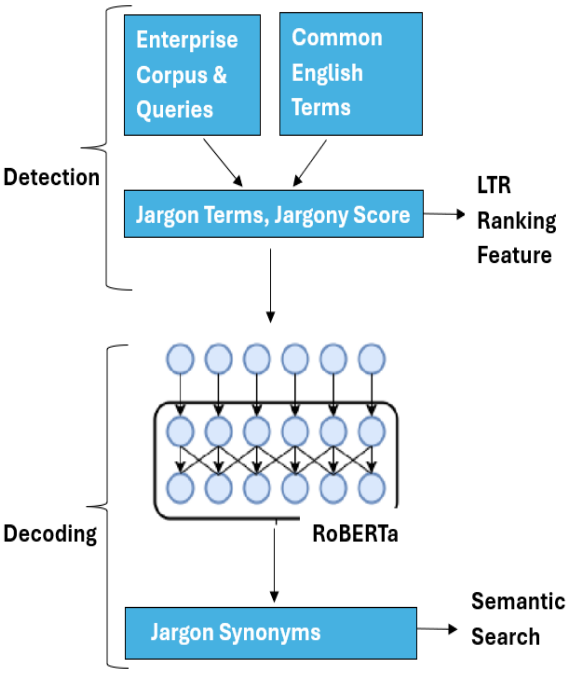


Figure 1: Architecture of the JARGES algorithm, including the detection (classification) and decoding (synonym generation) stages.

universally familiar or properly understood.

Although excessive use of jargon and terminology in organisations is often perceived as exclusionary, we use the terms here in a positive context for conveying complex ideas, processes, or services among employees/members who share common knowledge of the enterprise. In this context, jargon and terminology facilitate efficient communication.

The challenge of detecting and decoding enterprise jargon/terminology within a corpus lends itself to the fields of natural language processing (NLP) and LM. Comparative TF-IDF scoring and sparse matrix vectorisation can be used for tasks like plagiarism classification [Pudasaini et al., 2024] and also to distinguish word or phrase salience patterns between corpora [Belfathi et al., 2024]. The TF-IDF divergence calculation is computationally lightweight and can be executed using the standard IT hardware commonly available in most organisations. Word embeddings are another NLP tool that can capture semantic relationships between words in text data via dense vector matrices. LMs, such as Google's Bidirectional Encoder Representations from Transformers (BERT) [Devlin et al., 2019], are trained on large datasets containing vast amounts of text from diverse sources. While embeddings and transformers are regularly used in e-commerce [Singh et al., 2023] and commercial search engines [Li et al., 2017], their application for ES has not been sufficiently explored. A 2024 study by Belfathi to classify document genres used BERT to distil 'linguistic attributes' for legal terms and demonstrated elevated ranking performance for specific tasks in the LegalGLUE dataset [Belfathi et al., 2024].

In 2019, Facebook developed an improved version of BERT called RoBERTa (Robustly Optimised BERT Approach) [Liu et al., 2019]. Studies have shown that RoBERTa's larger training data leads to superior generation of synonyms. RoBERTa is especially effective for jargon by leveraging its ability to discern subtle contextual nuances that smaller models like BERT might miss.

Learning to Rank (LTR) is the application of supervised machine learning techniques to train a model to list the best ranking order [Li, 2011, Xu et al., 2020]. In the context of search results, LTR involves combining ranking signals to present the best order of documents for a given query. LTR computes the optimum 'weight' (importance) of signals, which can be extracted from an ES corpus and associated query log data. While LTR has been in use since 2004, major commercial Web Search (WS) providers like Baidu and Google still use LTR in 2024 [Wang et al., 2024, Google, 2025], with Baidu referring to it as the "standard workhorse" [Wang et al., 2024] for ranking search results. LTR methods, such as LambdaMART [Burges et al., 2005], optimise ranking by learning from relevance-labelled data, often evaluated using metrics like normalised discounted cumulative gain (nDCG), as seen in [Liu, 2010]. For ES, LTR has been adapted to handle domain specific challenges, including the presence of jargon, though with mixed success due to dataset limitations [Hawking, 2004]. A test collection or dataset based on Enterprise Search is hard to come by, as organisations are not inclined to open their intranet to public distribution, even for research purposes [Craswell et al., 2005, Cleverley and Burnett, 2019].

3 METHODS

This section outlines the methodology employed to create the JARGES feature, integration with an Apache Solr ES service, and the subsequent evaluation of the LTR ranking model. The ranking model is trained and evaluated using the `ENTRP-SRCH` dataset, and incorporates offline A/B testing to assess ranking performance.

3.1 JARGES

The JARGES algorithm is designed to detect and decode jargon. It processes three input files and generates two output lists, as depicted in Figure 2. These outputs are tailored to enhance distinct components of Enterprise Search (ES):

- A ranked list of the organisation's **detected** jargon terms, ordered by their 'jargony' score. This list serves as a ranking function for integration into an LTR model.
- A list of synonyms for each **decoded** jargon term from within the organisation. This output enables query expansion in the search engine, thereby improving recall.

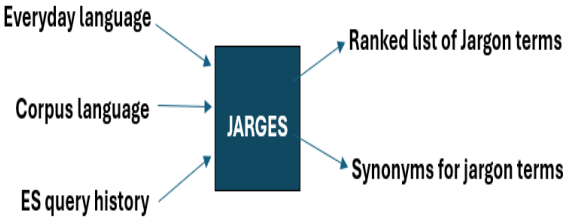


Figure 2: The three inputs for the JARGES algorithm, which outputs two lists (for ranking and recall respectively).

In this study, we use the `ENTRP-SRCH` LTR dataset, which includes 2544 human-annotated Q-D pairs of twenty most frequent queries of a large third-level education institution. One popular PDF document in the corpus is entitled 'jargon buster' and describes 126 commonly used jargon terms within the institution. The full code for the JARGES ranking and recall feature has been published to GitHub[1].

3.1.1 Detection

The JARGES feature is centred on the relative unusualness of words, such as those used in organisational jargon/terminology. Figure 3 is a demonstration of how JARGES works when applied to a real sentence in the organisation's corpus. Detections works by harnessing TF-IDF (Term Frequency, Inverse Document Frequency). TF-IDF is a core NLP statistical technique and was chosen as it is better at detecting semantic importance than raw word counts [Jones, 1972]. TF-IDF is used identify important (single and multi-word) terms in an enterprise corpus and compare their scores against general English frequencies. A substantial difference between the two scores indicates that the term may be jargon.

[1]https://github.com/ColinDaly75/JARGES

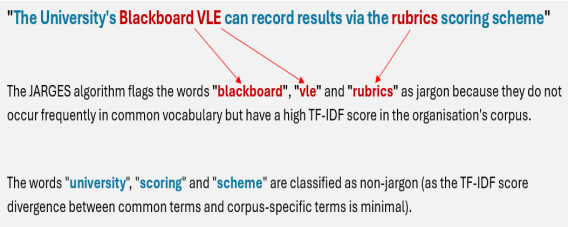


Figure 3: An example of jargon / non-jargon classification by JARGES when applied to a sentence in the corpus.

ES Corpus Vectorisation. The TfidfVectorizer python library [Pedregosa et al., 2011] is used to convert a collection of raw text document into a matrix of TF-IDF features. It splits and tokenises the document's words into n-grams and then computes the TF-IDF score for each n-gram in a document. The text is vectorised into a numerical matrix where each row represents a document and each column represents a word's TF-IDF score.

Detecting Divergence. The challenge is to detect enterprise specific words, phrases, expressions, and idioms that diverge from those universally familiar or understood outside of the organisation. This divergence is measured by computing the absolute difference in TF-IDF scores for the terms.

SpaCy is an open source Python NLP library for fast text processing that includes a small, pre-trained statistical model for English called 'en_core_web_sm'. The model is trained on web and news text data. SpaCy has been shown to work well for standardised English (e.g., formal writing, news, technical documents). For this reason, we use it as our reference source of TF-IDF for common English.

Jargony. Moreover, the magnitude of the divergence between the common score and the corpus score represents a measure of 'jargony' (i.e. how much jargon is likely imbued in the term). For example, TfidfVectorizer computes the TF-IDF score of the word 'blackboard' in common English (i.e. en_core_web_sm) is X, whereas the score recorded from the organisation corpus is Y. In the event where a term does not occur in common English, but occurs frequently in the organisation's corpus, this too is treated as jargon (e.g. the 'SCSS' term shown in Table 1).

Refining the List. The size of the sorted list, which includes both the jargon term and its corresponding jargony score, is determined by the min_divergence parameter. In the case of our corpus, this was set to be 15%, meaning that any jargon term with a TF-IDF divergence smaller that this is discarded. This is the same as the threshold adopted in Belfathi's study [Belfathi et al., 2024]. Moreover,

Table 1
JARGES classification. The terms in red font were successfully classified as jargon.

Jargon Term	Wikipedia-api Definition	Meaning within Organisation
Blackboard	A reusable writing surface on which text or drawings are made with sticks of calcium sulphate or calcium carbonate, known, when used for this purpose, as chalk. Blackboards were originally made of smooth, thin sheets of black or dark grey slate stone.	Blackboard is an abbreviation of 'Blackboard Learn', which is the organisation's Virtual Learning Environment (VLE). Students can use Blackboard to access lecture notes, online assignments and other activities.
Atrium	One of the two upper chambers in the heart that receives blood from the circulatory system. The blood in the atria is pumped into the heart ventricles through the atrioventricular mitral and tricuspid heart valves.	Event space and meeting place in the Dining Hall building which is often used by student societies.
SCSS	No definition.	Initialism for 'School of Computer Science and Statiscics'.
Botany Bay	An open oceanic embayment, located in Sydney, New South Wales, Australia	A residential square located behind the Graduates' Memorial Building.
Hilary	Hillary Diane Rodham Clinton (née Rodham;[a] born October 26, 1947) is an American politician, lawyer and diplomat.	Hilary is the second of The University's three annual semesters, running from January to April.

on visual inspection, the terms below this threshold did not intuitively appear jargon-like. This list is further refined by filtering the jargon terms with those query terms previously submitted to the search engine (this is the ES query history input shown in Figure 2). Two years of ES log data included 62,044 unique query terms. The resultant list (named refined-jargon-list.txt) consists of overlapping terms that are a) likely to be jargon and b) have a history of being queried. For our corpus, the refined list consisted of 547 records.

Blind Spot. A key limitation of the JARGES algorithm is its reliance on TF-IDF score divergence between common terms and corpus specific terms. A situation can occur when the divergence is too small for detection, even where a term is clearly jargon. For instance, as illustrated in the last row of Table 1, the jargon term 'Hilary'. The top Wikipedia-api definition presented changes the spelling from Hilary to Hiliary (i.e. Clinton). The term Hilary exhibits comparable prominence in both sources, thereby preventing its identification as jargon. Resolving this issue would require an alternative algorithm that differentiates terms based on semantic meaning instead of TF-IDF scoring.

3.1.2 Decoding

In the decoding stage, the RoBERTa language model is adapted for synonym generation by fine tuning it on the organisation's corpus. RoBERTa is used to predict plausible alternatives for the previously detected jargon terms. Synonyms can enhance semantic search by enabling the system to recognise and retrieve conceptually related terms beyond exact keyword matches.

The synonyms are output to a text file in a format that can be understood by the search engine. A practical consideration for Apache Solr is that the the more specific terms should be listed in the file before general ones [The Apache Software Foundation., 2004]. The synonyms are therefore ordered by their jargony score (computed in the detection phase). To further prioritise specific over more general synonyms, the Solr SynonymGraphFilterFactory is used as it has better handling of multi-term and large synonym sets than the default SynonymFilterFactory . This allows for query-time synonym expansion, where unprioritised synonyms are added to the query, while the original query term is still gets top billing. The synonyms of the jargon query terms should align with the description in the organisation's 'jargon buster' PDF document.

3.2 LTR Ranking Model

Our ES LTR ranking model is generated using eight features as part of the ENTRP-SRCH dataset. These features include BM25, recency, document hits, linkrank and click-though rate and are described fully in [Daly and Hederman, 2023]. A ninth feature, representing JARGES is then added to the dataset to test its impact. Table 2 describes the hyper-parameters used for the LTR calculation.

Table 2
Hyperparameters used in the Learning to Rank experiment.

Parameter	Value
Dataset	ENTRP-SRCH
Algorithm	GradientBoosting
n_estimators	100
max_depth	3
learning_rate	0.1
rank	nDCG
random_state	42
n_splits	5
num_features	8 (9 incl. JARGES)

4 EVALUATION

We conducted an A/B test comparing the ranking performance for two models generated and evaluated using the ENTRP-SRCH dataset. The first model incorporated eight features from our base model, while the second additionally included the JARGES feature. The ranking scores for both models are displayed in Table 3. The results indicate no significant percentage change between the two.

Table 3
A/B test results for ranking models with the percentage change in nDCG score after implementation of the JARGES feature.

Feature	nDCG@10
Base LTR model	0.9646 ± 0.001
With JARGES	0.9639 ± 0.001
Percentage change	+0.0%

We also conducted a 'leave-one-out' ablation study to evaluate the contribution of individual features. This method systematically removes one feature at a time to measure its impact on the overall model performance. As shown in Figure 4, removing the JARGES feature from the baseline model has a negligible effect on ranking performance.

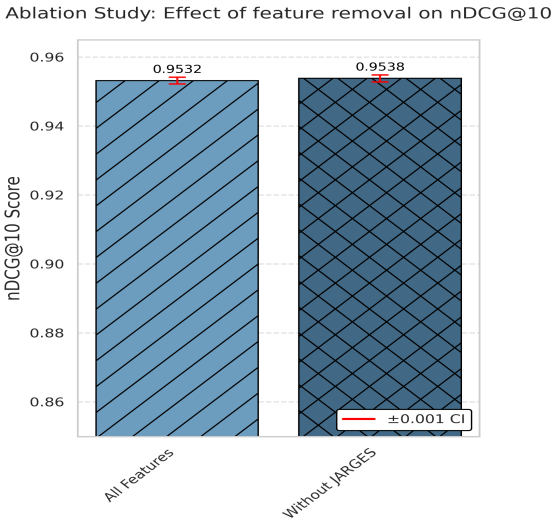


Figure 4: LTR ranking model performance with and without JARGES.

Both experiments showed that there is no statistically significant improvement for nDCG@10 at the 5% level ($p > 0.05$). This is attributed to the ENTRP-SRCH dataset's limited query diversity and scarcity of jargon rich Query-Document pairs, which deflated the performance score of the JARGES feature.

The 'Jargon Buster' PDF file acts as a independent reference for assessing both the precision and the comprehensiveness of the decoded jargon terms (i.e. synonyms). Of the 126 jargon terms defined in the PDF, 53 also appear in our generated list. Furthermore, a cursory inspection of the synonyms generated for these terms appears to reflect the correct semantic context. Some examples are shown in Table 4.

5 CONCLUSIONS AND FUTURE WORK

This study proposed the innovative 'JARGES' LM based feature for detecting and decoding jargon, and investigated the subsequent performance of ES ranking models calibrated with LTR weightings. Quantitative testing using the ENTRP-SRCH dataset failed to demonstrate a statistically significant increase in ranking performance, as evidenced by an nDCG@10 change of less than 0.001 (where $p > 0.05$). This result is disappointing, but not entirely unexpected, as the ENTRP-SRCH is small, with limited query diversity and a scarcity of jargon rich Q-D pairs. In spite of this, the qualitative analysis of the generated synonyms revealed promising results for recall as a foundation for

Table 4
Some examples of correctly decoded jargon terms via their contextual synonyms generated by RoBERTa.

Jargon Term	Synonyms	Meaning per 'Jargon Buster'
pav	pavilion, pavilion bar	Formally, the Pavilion Bar. The University's student bar, managed by the Sport Union, located at the eastern end of College Park.
tangent	climate entrepreneurship, social data science	The ideas workspace, providing courses and events centred on business and innovation.
forum	cafe, restaurant	College-operated eatery located in the Business School. Here you'll find hot and cold lunch offerings, barista coffee, and - often - pop-ups run by local businesses.

semantic search.

Future work plans will address the limitations of our `ENTRP-SRCH` dataset, which centres on just twenty of the most frequently submitted queries. The use of click-through data in place of human judgements for Q-D pair annotation would facilitate larger and more diverse ES datasets that are better able to capture the nuances of enterprise specific terminology. Finally, it would be interesting to perform a longitudinal study to gauge the JARGES impact on semantic and exploratory search, based on query expansion and recall on a real-world ES system.

REFERENCES

[Belfathi et al., 2024] Belfathi, A., Gallina, Y., Hernandez, N., Dufour, R., and Monceaux, L. (2024). Language Model Adaptation to Specialized Domains through Selective Masking based on Genre and Topical Characteristics. *arXiv*, 2402.12036.

[Bentley, 2011] Bentley, J. (2011). Mind the Enterprise Search Gap: Smartlogic Sponsor MindMetre Research Report.

[Burges et al., 2005] Burges, C., Shaked, T., Renshaw, E., Lazier, A., Deeds, M., Hamilton, N., and Hullender, G. (2005). Learning to rank using gradient descent. In *ICML 2005 - Proceedings of the 22nd International Conference on Machine Learning*, pages 89–96.

[Cleverley and Burnett, 2019] Cleverley, P. H. and Burnett, S. (2019). Enterprise search and discovery capability: The factors and generative mechanisms for user satisfaction:. *Journal of Information Science*, 45(1):29–52.

[Craswell et al., 2005] Craswell, N., Cambridge, M., and Soboroff, I. (2005). Overview of the TREC-2005 Enterprise Track. In *TREC 2005 conference notebook*, pages 199–205.

[Daly, 2023] Daly, C. (2023). Learning to Rank: Performance and Practical Barriers to Deployment in Enterprise Search. In *3rd Asia Conference on Information Engineering (ACIE)*, pages 21–26. IEEE.

[Daly and Hederman, 2023] Daly, C. and Hederman, L. (2023). Enterprise Search: Learning to Rank with Click-Through Data as a Surrogate for Human Relevance Judgements. In *15th International Conference on Knowledge Discovery and Information Retrieval (KDIR)*, pages 240–247.

[Devlin et al., 2019] Devlin, J., Chang, M.-W., Lee, K., Google, K. T., and Language, A. I. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *Proceedings of the 2019 Conference of the North*, pages 4171–4186.

[Ganguly et al., 2015] Ganguly, D., Roy, D., Mitra, M., and Jones, G. J. (2015). A word embedding based generalized language model for information retrieval. *SIGIR 2015 - Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 795–798.

[Google, 2025] Google (2025). Search and SEO Blog — Google Search Central — Google Search Central Blog — Google for Developers.

[Hawking, 2004] Hawking, D. (2004). Challenges in Enterprise Search. In *Proceedings of the 15th Australasian Database Conference - Volume 27*, ADC '04, page 15–24, AUS. Australian Computer Society, Inc.

[Jones, 1972] Jones, K. S. (1972). A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, 28(1):11–21.

[Li, 2011] Li, H. (2011). A Short Introduction to Learning to Rank. *IEICE Transactions*, 94-D:1854–1862.

[Li et al., 2017] Li, L., Deng, H., Dong, A., Chang, Y., Baeza-Yates, R., and Zha, H. (2017). Exploring query auto-completion and click logs for contextual-aware web search and query suggestion. *26th International World Wide Web Conference, WWW 2017*, pages 539–548.

[Liu, 2010] Liu, T.-Y. (2010). *Learning to Rank for Information Retrieval*, volume 3. Springer Berlin Heidelberg, 2nd edition.

[Liu et al., 2019] Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019). RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv*, 1(abs/1907.11692).

[Pedregosa et al., 2011] Pedregosa, F., Michel, V., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Vanderplas, J., Cournapeau, D., Pedregosa, F., Varoquaux, G., Gramfort, A., Thirion, B., Grisel, O., Dubourg, V., Passos, A., Brucher, M., Perrot andÉdouardand, M., Duchesnay, A., and Duchesnay EDOUARDDUCHESNAY, F. (2011). Scikit-learn: Machine Learning in Python. *The Journal of Machine Learning Research*, 12:2825–2830.

[Pudasaini et al., 2024] Pudasaini, S., Miralles-Pechuán, L., Lillis, D., and Salvador, M. L. (2024). Survey on Plagiarism Detection in Large Language Models: The Impact of ChatGPT and Gemini on Academic Integrity. *Journal of Academic Ethics*, 11(04).

[Singh et al., 2023] Singh, S., Farfade, S., and Comar, P. M. (2023). Multi-Objective Ranking to Boost Navigational Suggestions in eCommerce AutoComplete. *ACM Web Conference 2023 - Companion of the World Wide Web Conference, WWW 2023*, pages 469–474.

[The Apache Software Foundation., 2004] The Apache Software Foundation. (2004). Apache Solr.

[Turnbull and Berryman, 2016] Turnbull, D. and Berryman, J. (2016). *Relevant Search*. Manning Publications Co., New York.

[Wang et al., 2024] Wang, Q., Li, H., Xiong, H., Wang, W., Bian, J., Lu, Y., Wang, S., Cheng, Z., Dou, D., and Yin, D. (2024). A Simple yet Effective Framework for Active Learning to Rank. *Machine Intelligence Research*, 21(1):169–183.

[White, 2015] White, M. (2015). Critical success factors for enterprise search.

[Xu et al., 2020] Xu, J., Wei, Z., Xia, L., Lan, Y., Yin, D., Cheng, X., and Wen, J.-R. (2020). Reinforcement Learning to Rank with Pairwise Policy Gradient. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, volume 10, page 10, New York, NY, USA. ACM.

C

Research Ethics Approval

SCSS Research Proposal and Approval

CHECKLIST

1.	• SCSS Ethical **Application Form**	✓
2.	• **Participant's Information Sheet** must include the following: *a) Declarations from Part A of the application form.* b) Details provided to participants about how they were selected to participate. c) Declaration of all conflicts of interest.	N/A
3.	• **Participant's Consent Form** must include the following: *a) Declarations from Part A of the application form.* **b)** Researchers contact details provided for countersignature (your participant will keep one copy of the signed consent form and return a copy to you).	N/A
4.	• **Research Project Proposal** must include the following: *a) You must inform the Ethics Committee **who** your intended participants are.* i.e. are they your work colleagues, classmates etc. b) How will you recruit the participants i.e. **how** do you intend asking people to take part in your research? For example, will you stand on Pearse Street asking passers-by? c) If your participants are under the age of 18, you must seek both parental/guardian AND child consent.	✓
5.	• Intended **questionnaire**/survey/interview protocol/screen shots/representative materials (as appropriate)	N/A
6.	• **URL** to intended on-line survey (as appropriate)	N/A

List of Amendments following *Final Comments of REC* meeting (Feb 7th, 2022)

Amendment 5. Sections 3.1 & 3.2: Since the answers are "No", there is no need for the content in the tables included in those questions. Input has been removed from the form.

No other changes have been made to the document.

Research Ethics Application

Part A

Project Title: **Learning to Rank: A longitudinal study of the effectiveness of various algorithms as applied to** [redacted]

Name of Lead Researcher (student in case of project work): Colin Daly

Name of Supervisor: Prof. Yvette Graham

TCD E-mail: dalyc24@tcd.ie Contact Tel No.: [redacted]

Course Name and Code (if applicable): ……

Estimated start date of survey/research: ……Q2/2021……

I confirm that I will (where relevant):

- Familiarize myself with the Data Protection Act and the College Good Research Practice guidelines http://www.tcd.ie/info_compliance/dp/legislation.php;
- Tell participants that any recordings, e.g. audio/video/photographs, will not be identifiable unless prior written permission has been given. I will obtain permission for specific reuse (in papers, talks, etc.)
- Provide participants with an information sheet (or webpage for web-based experiments) that describes the main procedures (a copy of the information sheet must be included with this application)
- Obtain informed consent for participation (a copy of the informed consent form must be included with this application)
- Should the research be observational, ask participants for their consent to be observed
- Tell participants that their participation is voluntary
- Tell participants that they may withdraw at any time and for any reason without penalty
- Give participants the option of omitting questions they do not wish to answer if a questionnaire is used
- Tell participants that their data will be treated with full confidentiality and that, if published, it will not be identified as theirs
- On request, debrief participants at the end of their participation (i.e. give them a brief explanation of the study)
- Verify that participants are 18 years or older and competent to supply consent.
- If the study involves participants viewing video displays, then I will verify that they understand that if they or anyone in their family has a history of epilepsy then the participant is proceeding at their own risk
- Declare any potential conflict of interest to participants.
- Inform participants that in the extremely unlikely event that illicit activity is reported to me during the study I will be obliged to report it to appropriate authorities.
- Act in accordance with the information provided (i.e. if I tell participants I will not do something, then I will not do it).

Signed: ..... Date: 22-MARCH-2021

Lead Researcher/student in case of project work

Part B

Please answer the following questions.		***Yes/No***
Has this research application or any application of a similar nature connected to this research project been? refused ethical approval by another review committee of the College (or at the institutions of any collaborators)?		***No***
Will your project involve photographing participants or electronic audio or video recordings?		***No***
Will your project deliberately involve misleading participants in any way?		***No***
Does this study contain commercially sensitive material?		***YES***
Is there a risk of participants experiencing either physical or psychological distress or discomfort? If yes, give details on a separate sheet and state what you will tell them to do if they should experience any such problems (e.g. who they can contact for help).		***No***
Does your study involve any of the following?	Children (under 18 years of age)	***No***
	People with intellectual or communication difficulties	***No***
	Patients	***No***

School of Computer Science and Statistics
Research Ethical Application Form

Details of the Research Project Proposal must be submitted as a separate document to include the following information:

1. Title of project
2. Purpose of project including academic rationale
3. Brief description of methods and measurements to be used
4. Participants - recruitment methods, number, age, gender, exclusion/inclusion criteria, including statistical justification for numbers of participants
5. Debriefing arrangements
6. A clear concise statement of the ethical considerations raised by the project and how you intend to deal with them
7. Cite any relevant legislation relevant to the project with the method of compliance e.g. Data Protection Act etc.

Part C

I confirm that the materials I have submitted provided a complete and accurate account of the research I propose to conduct in this context, including my assessment of the ethical ramifications.

Signed: ..... Date: 22-MARCH-2021
Lead Researcher/student in case of project work

There is an obligation on the lead researcher to bring to the attention of the SCSS Research Ethics Committee any issues with ethical implications not clearly covered above.

Part D

If external or other TCD Ethics Committee approval has been received, please complete below.

External/TCD ethical approval has been received, and no further ethical approval is required from the School's Research Ethical Committee. I have attached a copy of the external ethical approval for the School's Research Unit.

Signed: ..... Date: .....

Lead Researcher/student in case of project work

Part E

If the research is proposed by an undergraduate or postgraduate student, please have the below section completed.

I confirm, as an academic supervisor of this proposed research that the documents at hand are complete (i.e. each item on the submission checklist is accounted for) and are in a form that is suitable for review by the SCSS Research Ethics Committee.

Signed: .... Y. Graham ..... Date: .....8/4/21.....

Supervisor

Completed application forms together with supporting documentation should be submitted electronically to the online ethics system - https://webhost.tchpc.tcd.ie/research_ethics/ When your application has been reviewed and approved by the Ethics committee, hardcopies with original signatures should be submitted to the School of Computer Science & Statistics, Room 104, Lloyd Building, Trinity College, Dublin 2.

SCSS Research Project Proposal

Colin Daly (Research Student)
ADAPT Centre,
School of Computer Science and Statistics,
Trinity College Dublin
dalyc24@tcd.ie

Yvette Graham (Supervisor)
ADAPT Centre,
School of Computer Science and Statistics,
Trinity College Dublin
ygraham@tcd.ie

Carl Vogel (Co-Supervisor)
ADAPT Centre,
School of Computer Science and Statistics,
Trinity College Dublin
carl.vogel@tcd.ie

SECTION 1 – Project Description

1. Title of study

Learning to Rank: A longitudinal study of the effectiveness of various algorithms as applied to an [redacted]

2. Dates and duration of study

Pending research ethics approval, the study utilizing the dataset will commence in Q2 2021 and complete no later than the end of Q4 2025. This period coincides with duration of my part-time studies towards a PhD.

3. Purpose of the project including academic rationale.

In the field of Information Retrieval, Learning to Rank has become the machine learning method of choice for researchers interested in ranking. As with commercial Web Search engines (Google, Baidu, etc), ranking is also the primary challenge for engineers when deploying an Enterprise Web Search service in their organizations. We have chosen a university website [redacted] to build our Enterprise corpus and dataset. There are several differentiating factors between commercial Web Search and Enterprise Web Search (rigorous site hierarchy, multiple home pages per domain etc) and their optimal ranking algorithms will not be the same.
The purpose of the project is to compare the performance of alternate search algorithms. This requires examination of user behaviour as they search [redacted]. Given a query entered by the user, the project will examine what search results were returned and which of the results the user chose to click on. This data will be anonymised in a first step that replaces the user IP address with an anonymous key.

4. Describe how participants will be recruited (inclusion and exclusion criteria).

Participants will be recruited when they first use the [redacted]. All users, who have accepted the [redacted] cookie, create data that may be included in this study. The exception being

that if there is a query that uniquely identifies a source IP address, this will be excluded from analysis.
No means will be provided to inform end-users that their use of the website will be used beyond the standard "performance" analysis implied by "performance cookies", the use "beyond the standard" being that which forms the topic of the PhD project addressed by this research.

The remainder of the text in this section outlines the wording and mechanism of the cookies and consent management to justify this approach.

The first time that any user hits the [redacted] website, he or she is presented with a cookie before proceeding to use the site. The Consent Management box is prominently positioned. The data required for this study falls under both the 'Strictly Necessary Cookies' (i.e. the collected data is essential for the necessary operation of the website) and 'Performance Cookies'.

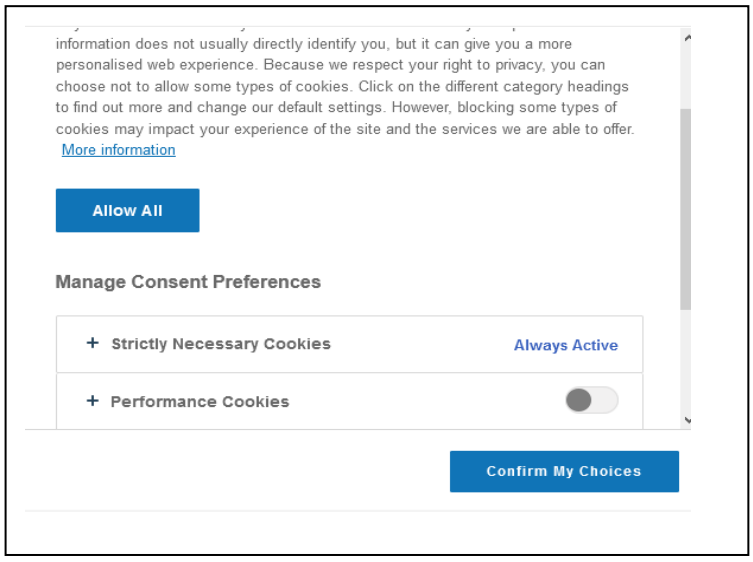


Figure 1: The default setting for the Performance Cookie is set to 'active'. It also allows the end-user a one-click opt-out.

The cookie's 'More information' link leads any inquiring user to the [redacted] Usage policies (reproduced below in Figure 2): -

[redacted] uses cookies to enable our website to operate, improve your browsing experience and provide personalised advertising which we believe may be of interest to you. By clicking "Accept All Cookies" you consent to the use of all cookies including social media and advertising cookies. You can choose to not accept certain cookies however this may impact on your browsing experience. Select "Cookies Settings" to find out more and customise your preferences. For further information please see our **Cookie Policy**

Figure 2: This is the cookie banner that points inquiring users to the [redacted] Cookie Policy site.

The Cookie Policy URL points to [redacted][1] and cookies apply to "any website within the [redacted]". Figure 3 highlights the pertinent section of the page which describes why and how IT Services use traffic data to 'analyse how users use the site' and to 'compile statistical reports on website activity'.

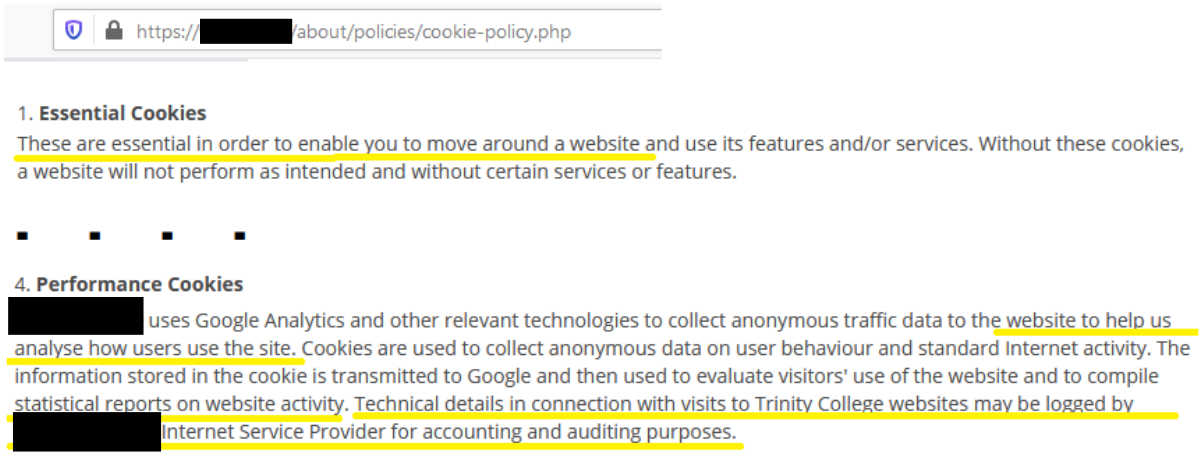


Figure 3: The Cookie Policy for usage of the [redacted]. The highlighted sections of the 'Strictly Necessary' (Essential) and 'Performance' cookies, when combined, provide the basis for Log Data collection and end-user consent.

While dataset size is known to have a significant impact on Machine Learning in the field of Learning to Rank, the training, testing and validation datasets are expected to number in tens of thousands rather than millions. A tiny portion of the Log Data will be used in the dataset creation. Data Quality is fundamental to project outcomes and strong filtering rules are required to mitigate prevalence of bots and spammers etc.

Credibility of the 'opt out'.

As illustrated in Figure 1, the end-user can opt out of the Performance Cookie with just a single click. As per article 7 of the GDPR legislation, 'it shall be as easy to withdraw as to give consent'. If consent is not granted, the IP address is temporarily recorded (separately from Log Data) and all traces of this user's session will be deleted automatically. Furthermore, the IP address will be deleted from the session cookie and excluded from Log Data when it is converted into the feature vector array dataset. No trace of the user's IP address will remain in the Log Data or any derived logs.

The [redacted] website receives over one million hits per year (i.e. one million different IP addresses are recorded in the Log Data).

[1] Accessed on 27th January 2021

A negligible proportion of these end-users choose to later withdraw consent (i.e. having previously accepted the cookie). This is rare, but not unprecedented. There have been several test-cases where the deletion procedure has been successfully carried out. The deletion procedures include 'deletion propagation' (i.e. where the individual user's IP address is deleted from any derived log files or dataset). This is in accordance with Article 17 of GDPR legislation ('right to be forgotten').

5. **Procedures of the study**.

The researcher will provide a script named "anonymize.py" to IT Services; this will be used to strip the IP address from Log Data to create a working snapshot, which will be taken offline and further refined to generate the Metadata.

Approval must come from two parties: -

- SCSS Research Ethics Committee
- IT Services Digital Director (in her role as Data Controller, i.e. exclusive owner of Log Data)

On receipt of the dataset (a feature vector array) from IT Services, the research will be conducted using various Machine Learning algorithms and techniques to produce a new ranking (as measured by nDCG and MAP, etc).

A practical outcome of the study is that the various algorithms and techniques will, subject to organisational change control and approvals, be applied to the Enterprise Search facility on the live [redacted]e.

The researcher has no direct contact with the end-user. In GDPR terms, the researcher is neither a data controller nor a data processor.

6. **Debriefing arrangements following the study**.

There will be no targeted debriefing. Users will simply leave the website, close the browser etc and there is no way to communicate after session termination. A targeted debriefing of each of the millions of end-users is not practicable, not least because all we have is the users' IP Address (no usernames or email addresses).

However, for any of the end-users that may be interested, all outputs of this research will be made openly accessible (both via IT Services pages and academic publications).

7. **Intended questionnaire/survey/interview protocol (w/screen shots etc)**

The Dataset is metadata derived from an anonymized snapshot of Log Data. The snapshot will be generated by IT Services – and subsequently made available to the researcher. This research will not be engaged directly in any surveys/questionnaires or interviews. Note that it is the role of [redacted] to give feedback and engage with the web authoring community (who will constitute the judges and annotators). The researcher will not

communicate with annotators or end-users and the SCSS Informed Consent template form will therefore not be required.

SECTION 2 – Ethical Concerns

Describe and discuss all ethical issues relating to the study (including any relating to participant autonomy [2]and personal data). As part of this you should:

1. **Potential benefits and potential harms to participants and how they are addressed**.

The Participants are the end-users of the [redacted] search service. They will benefit from, over the long-term, gradual improvement in ranking results when engaging with the Enterprise Web Search box.

During the period of the study, astute end-users may note that they receive different ranking of their search results on a day-to-day basis. This is potentially harmful, but users are accustomed to this from their interactions with commercial web search services (google, etc). Moreover, participant autonomy is reasonably shielded insofar that ranking does not exclude any search results ('recall' is unchanged); it merely re-orders the presentation order.

Since no personal data or sensitive data is included in the dataset (feature vector), it is inconceivable that any potential harm would be incurred to participants by way of exposure of personal data.

2. **Conflicts of interest and how they will be addressed.**

This study has a potential conflict of interest. The PhD student is funded by two sources (IT Services and self-funded). [redacted] IT Services and is therefore exposed to privileged data.

The key challenge, and mitigating technique, for the researcher will be to inquire 'in which capacity am I carrying out this function?'.

SECTION 3 – CONFIDENTIALITY AND DATA PROTECTION

At this point, it is important to clarify and distinguish between 'Log Data' and the 'Metadata' associated with this research project (Figure 4).
This distinction is essential to resolving the potential conflict of interested inherent to the dual role played by the researcher, who is also an [redacted]

2

- **Log Data**

This is privileged data recorded in Apache and Solr log files and is collected by IT Services as an essential part of providing a web and enterprise web search service. This data contains both personally identifiable information (IP address) and commercially sensitive information (number of hits, date of last modification of a particular URL). The [redacted] search service is publicly accessible; no user login is required. Hence, no user profiling (collaborative-based filtering) is used. IT services have long-standing and strict protocols for the protection from unauthorised access of electronic locations where data are stored. The Log Data is retained by IT Services and will never be made accessible, in its raw form, for research purposes. Data is backed up using the CommVault Enterprise Grade backup technology on dedicated [redacted] encryption. Backup data is retained no longer than 21 days.

A snapshot of the above Log Data will be taken before it is anonymized and converted into Metadata. This snapshot has two purposes: -

- The snapshot will have personal data removed (i.e. the end-users' IP address), thereby introducing a security layer between the researcher and IT Services.
- The snapshot servers as a 'working copy' and enables the researcher to investigate any inconsistencies in metadata without having to revert to the original Log Data stored on the production servers managed by IT Services.

The snapshot is therefore an intermediate layer (Log Data will be converted to the anonymized snapshot, which in turn will be converted into Metadata). The snapshot will be made available exclusively to the researcher.

Whilst processing Log Data, the applicant will function in his role of [redacted] as a researcher).

- **Metadata**

This is the feature vector array dataset derived from the snapshot of the Log Data. This metadata cannot be re-constituted to recreate any privileged or sensitive information. The Metadata (i.e. feature vector array dataset) contains pseudonymised data (DOC_ID is substituted for the URL). No other field of the array contains sensitive or personally identified or identifiable data. The Learning to Rank methodology consequently has an element of 'privacy by design' (Article 25 of the GDPR legislation). It is proposed that this dataset is made available for research purposes. IT Services will process the snapshot of Log Data to generate the metadata array and present it to the researcher. An explanation of what each row and column represent is illustrated in Figure 5.

Neither the Log Data nor the Metadata contain usernames or access credentials.

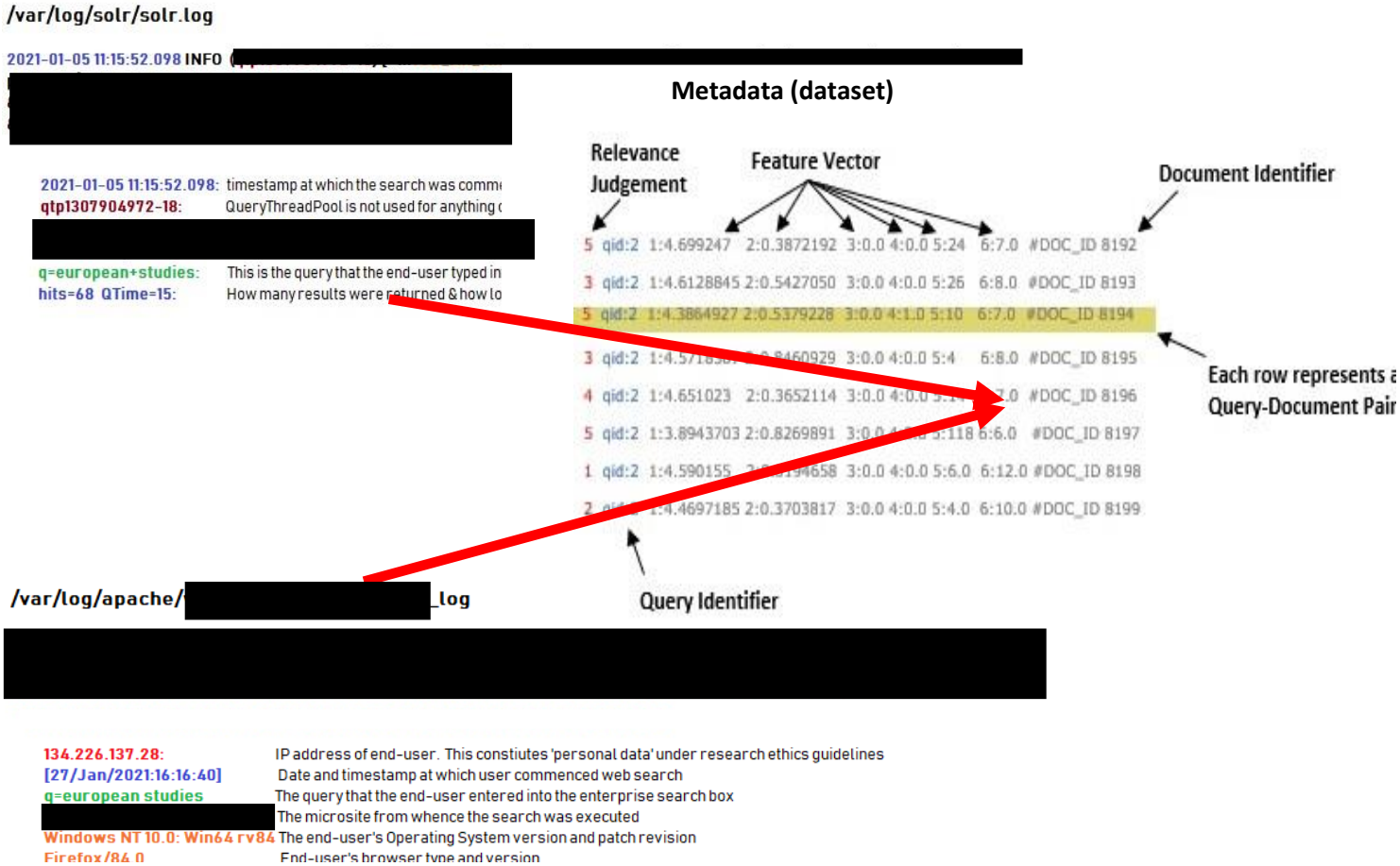


Figure 4: Log Data (Apache and Solr log files) is collected [redacted] part of essential operational requirements). The two files are then combined, stripped and pseudonymized into a snapshot to produce the Learning to Rank Feature Vector array (dataset). Note that controls are applied on all elements of the data chain and that there is no personal or sensitive data in the generated dataset.

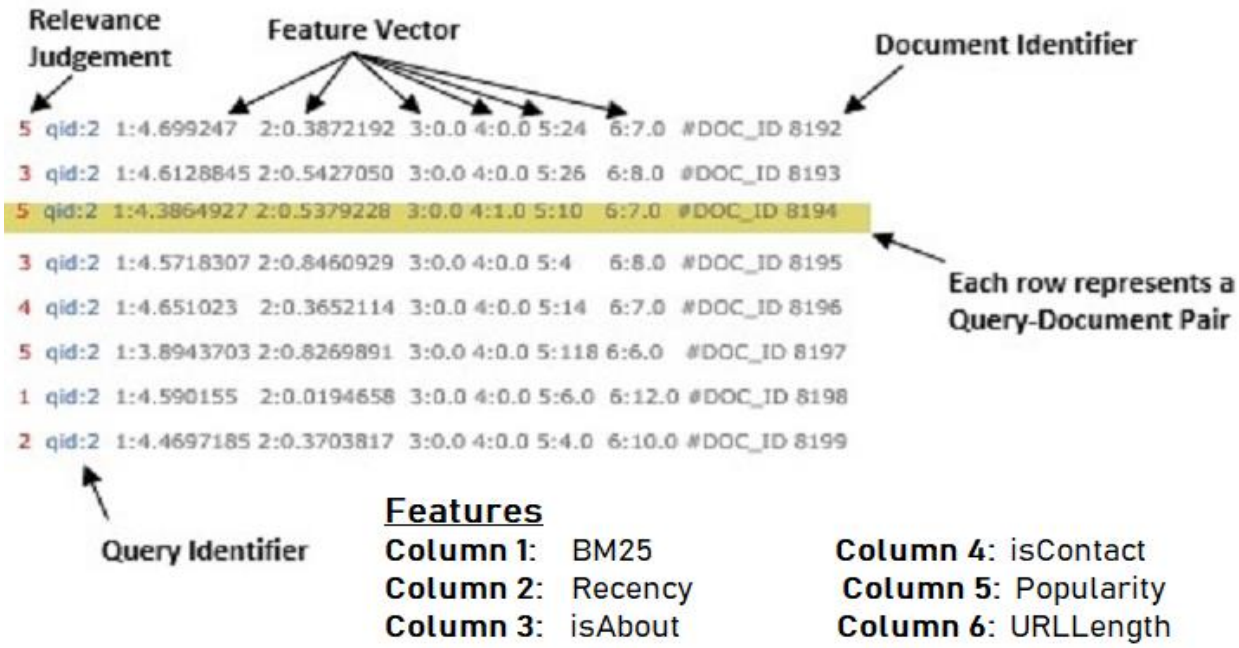


Figure 5: Metadata. This is the feature vector array dataset. Each row represents a Query-Document pair. The columns represent a feature and its associated value. Recency is the last modification date of the file, while Popularity represents the number of hits. The DOC_ID column is a pseudorandomized value for the URL and cannot be tied back to the URL without access to a key.

Whilst processing the Metadata, the applicant will function in his role of [redacted]

3.1 Does this study involve collecting, using, accessing or sharing personal data[3]?

Yes [ ] No [X]

If **NO**, please go to section 3.4.
If **YES,** please list[4] all categories of personal data.
Please see checklist on secure storage available here.

[3] Personal data is information which can identify a person. In particular: a name, address, email, telephone number, an identification number, location data, an online identifier, an IP address, a code key linking back to identifiable data etc. Please note that pseudonymised data is personal data under GDPR. Pseudonymised data means data which cannot be attributed to an individual without the use of additional information which is kept separately. (i.e. a key) . It is sometimes referred to as 'coded data'. Please note that in order to be considered personal data in our hands, XXXXX must hold the key.

[4] If using Personal data. Records must be kept pursuant to Article 30, GDPR; https://gdpr-info.eu/art-30-gdpr/

Type of Data	Justification: Why do you need the data?	Data Format	Technical and Organisational Controls	Identifiable coded, or anonymised

3.2 Does the study involve collecting, using, accessing or sharing sensitive data[5]?

Yes [ ] No [X]

If **NO,** please go to question 3. 3.

If **YES**, please list all categories of the sensitive data collected.
Please see checklist on secure storage available here.

Type of Data	Justification: Why you need the data?	Data Format	Technical and Organisational Controls	Identifiable coded, or anonymised

[5] Sensitive personal data means personal data, which poses a higher risk to the individual. It includes personal data revealing racial and ethnic origin, political opinions, religious or ideological convictions, trade union membership, criminal convictions and offences, genetic, biometric (photos, videos, audio etc.) data concerning physical or mental well-being , information relating to education, professional training employment and career history, questionnaires, Information relating to the family of the individual and the individual's lifestyle and social circumstances.

3.3 Who determines how and why the personal and/or sensitive data is used?

(Data Controller[6] or Joint Data Controllers**)**

There is no Personal or Sensitive information in the Dataset (feature vector array) derived from the Log Data. The data controller of the Log Data (apache and solr log files) is [redacted]

3.4 Will the personal and/or sensitive personal data be shared with any third parties[7]?

No. The only personal data is the end-user's IP address, and this is stored in the Log Data. It is removed in the generated dataset. The only commercially sensitive data is the URL, and this is stored in the Log Data. This field is pseudonymised (DOC_ID is substituted for the URL) in the generated dataset.

3.5 How long will you retain the personal data?

N/A. No personal data is contained in the dataset used for this research.

3. 6 Will the personal data be fully anonymised or deleted after it is no longer necessary?

N/A. No personal data is contained in the dataset used for this research.

3.7 How will you inform participants of their rights under GDPR[8]:

N/A. IT services manage communication with end-users. IT Security have created a page at [redacted]/ITSecurity/gdpr/. This page includes the[redacted]. The dataset used in this study contain no personally identifiable information about participants.

---

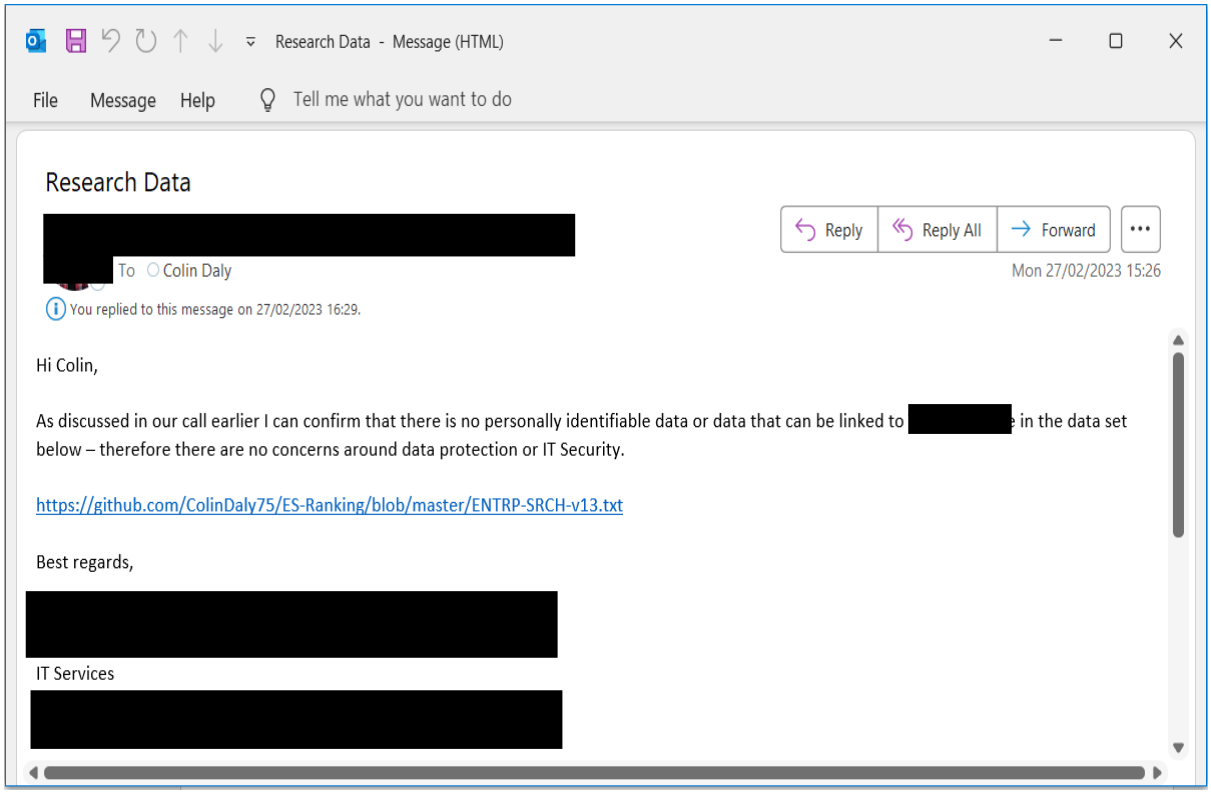
6 *Employees and students of XXX are not data controllers. XXX is the data controller for the institution. However, if other institutes jointly decide how and why the data will be used, they should also be noted as controllers here.*

7 Third parties could be collaborators (institutes/industry) or service providers (transcribers, cloud storage etc.)

8 Under GDPR, these include:

- right of access;
- right to rectification;
- right to erasure;
- right to object to processing based on legitimate or public interest;
- right to data portability;
- right to object to profiling or making decisions about individuals by automated means?

Consent from IT Services / IT Security Department



Research Data - Message (HTML)

File Message Help Tell me what you want to do

Research Data

[redacted]

Reply Reply All Forward ...

To Colin Daly

Mon 27/02/2023 15:26

You replied to this message on 27/02/2023 16:29.

Hi Colin,

As discussed in our call earlier I can confirm that there is no personally identifiable data or data that can be linked to [redacted] in the data set below – therefore there are no concerns around data protection or IT Security.

https://github.com/ColinDaly75/ES-Ranking/blob/master/ENTRP-SRCH-v13.txt

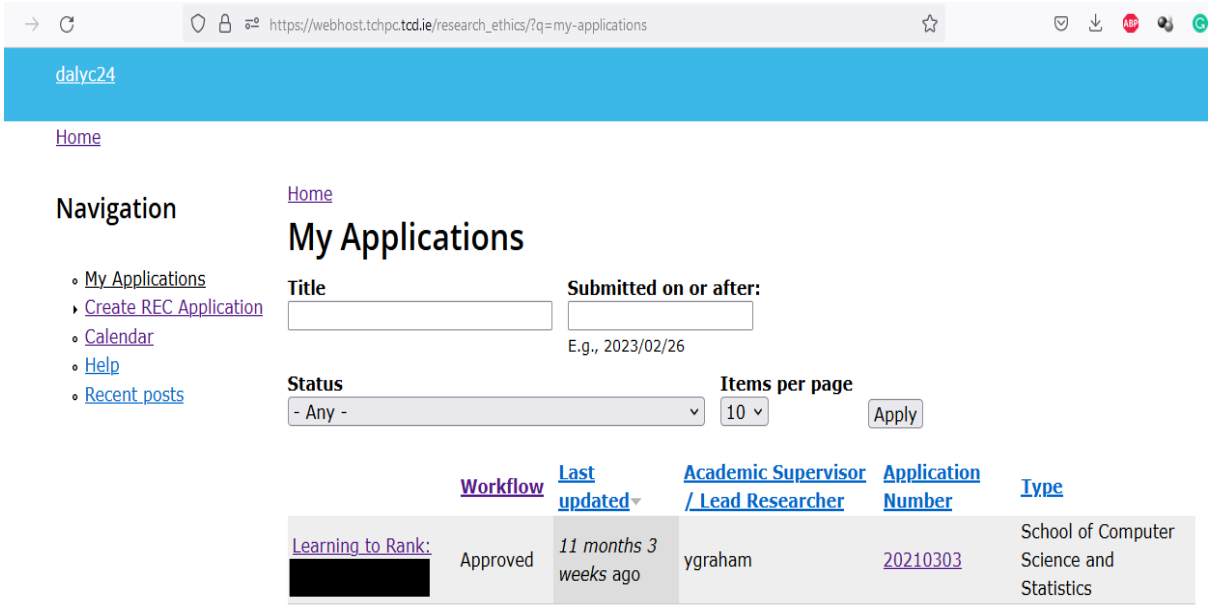
Best regards,

[redacted]

IT Services

[redacted]

REC Approval



https://webhost.tchpc.tcd.ie/research_ethics/?q=my-applications

dalyc24

Home

Navigation

- My Applications
- Create REC Application
- Calendar
- Help
- Recent posts

Home

My Applications

Title

Submitted on or after:

E.g., 2023/02/26

Status

- Any -

Items per page

10

Apply

	Workflow	Last updated	Academic Supervisor / Lead Researcher	Application Number	Type
Learning to Rank: [redacted]	Approved	*11 months 3 weeks ago*	ygraham	20210303	School of Computer Science and Statistics

This page is intentionally blank.