Learning to Rank for Query Auto-Complete with Language Modelling in Enterprise Search

Colin Daly[1,2][a] and 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, Query Auto-Complete, Language Modelling.

Abstract: Query Auto-Completion (QAC) is of particular importance to the field of Enterprise Search, where query suggestions can steer searchers to use the appropriate organisational jargon/terminology and avoid submitting queries that produce no results. The order in which QAC candidates are presented to users (for a given prefix) can be influenced by signals, such as how often the prefix appears in the corpus, most popular completions, most frequently queried, anchor text and other of a document, or what queries are currently trending in the organisation. We measure the individual contribution of each of these heuristic signals and supplement them with a feature based on Large Language Modelling (LLM) to detect jargon/terminology. We use Learning To Rank (LTR) to combine the weighted features to create a QAC ranking model for a live Enterprise Search service. In an online A/B test over a 12-week period processing 100,000 queries, our results show that the addition of our jargon/terminology detection LLM feature to the heuristic LTR model results in a Mean Reciprocal Rank score increase of 3.8%.

1 INTRODUCTION

Query Auto-Completion (QAC) presents users with a ranked list of suggested queries in a drop-down box 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.

The user's query prefix can be reformulated inline to use specific 'wording' that ensures relevance or semantic matching. This reformulation is particularly useful for Enterprise Search (ES), where organisations have their own jargon and terminology. QAC can also help avoid users submitting queries that produce no results (a potentially common occurrence, as an ES corpus is small compared to WS) (Kruschwitz and Hull, 2017).

A consequence of the ubiquity of commercial/Internet Web Search (WS) is that users have high expectations when it comes to interacting with search engines (Davis et al., 2011). Cleverley and Burnett refer to this as 'Google Habitus' (Cleverley and Burnett, 2019). Ranked query suggestions are an expected characteristic of every interactive search service (White, 2018).

The simple definition of ES is finding the information needed within an organisation (Bentley, 2011). Many employees or members of an organisation may not be proficient in using their organisation's jargon and terminology. 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. 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) (Scott, 2022). Where an ES service presents a restricted number of suggestions, their ranking is more important.

Learning to Rank (LTR) is the application of su-

[a] https://orcid.org/0000-0001-7218-7765
[b] https://orcid.org/0000-0001-6073-4063

Daly, C. and Hederman, L.
Learning to Rank for Query Auto-Complete with Language Modelling in Enterprise Search.
DOI: 10.5220/0012888100003838
Paper published under CC license (CC BY-NC-ND 4.0)
In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 15-26
ISBN: 978-989-758-716-0; ISSN: 2184-3228

pervised machine learning techniques to train a model to list the best ranking order (Li, 2011; Xu et al., 2020). In the context of QAC, this involves combining signals to present the best order of query candidates for a given query prefix. LTR computes the optimum 'weight' (importance) of signals, which are extracted from the ES corpus, and query log files.

The challenge of deciphering 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. While LLMs and embeddings are regularly used in e-commerce (Singh et al., 2023a) and commercial search engines (Li et al., 2017), their application for ES has not been sufficiently explored.

In this paper, we introduce a ranking feature explicitly designed for ES. We call this 'QACES' (Query Auto-Complete for Enterprise Search). This feature is centred on the relative unusualness of words, such as those used in organisational jargon/terminology. QACES is scalable and can be applied to any ES service. Our hypothesis is that adding the QACES feature to a heuristic LTR ranking model will significantly increase the QAC ranking performance, as measured by Mean Reciprocal Rank (MRR). The major contribution of this research is the introduction of our new feature designed to detect, understand and suggest jargon/terminology specific to organisations. For performance context, we undertake an offline ablation study to measure the individual improvement of each ranking feature. Subsequently, we perform an A/B test on a live ES service of a large third-level academic institution to confirm that the addition of the QACES feature to our heuristic model can significantly improve QAC ranking performance.

2 RELATED WORK

2.1 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 (Croft, 2010), 'search as you type' (Turnbull and Berryman, 2016), predictive search, auto-suggest, real-time query expansion (RTQE) (White and Marchionini, 2007), query modification suggestions (Kruschwitz and Hull, 2017) and subword completion (Kim, 2019). 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.'[1]. 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 (Yadav et al., 2021; Li et al., 2017), as is the case in this study.

2.2 Approaches to QAC Ranking

There are two principal approaches to ranking QAC candidates (Cai and De Rijke, 2016). The first, more traditional approach is heuristic and combines domain-specific signals from a corpus and query logs. This approach, where ranking signals are handcrafted 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. The second approach employs NLP or Language Modelling to produce context-aware suggestions (Singh et al., 2023a; Kim, 2019). Learning to Rank can combine or 'fuse' the two approaches with any number of features (Rahangdale and Raut, 2019; Guo et al., 2016) as outlined in §3.

2.3 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 (SERP) (Joachims, 2002; Daly, 2023). 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 (Chang and Demg, 2020). Previously recorded behaviour and queries provide useful information for any user's intent and can be leveraged to suggest completions that are more

[1]https://blog.google/products/search/how-google-autocomplete-works-search/, accessed 29th June 2023

relevant while adhering to the user's prefix (Yadav et al., 2021).

Most Popular Completions (MPC) is a ranking feature that proposes completion candidates based on user preferences recorded in historical search data. For this reason, and because of a reluctance of users to provide explicit feedback (Kruschwitz and Hull, 2017), MPC is typically considered the main indicator of historical relevance (Li et al., 2017) 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) (Yadav et al., 2021). 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.

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 (Pass et al., 2006), which includes over 10 million queries. ES differs from WS insofar as the content may be indexed from multiple databases (e.g. corporate directories) and intranet document repositories. ES may also include searches for explicitly indexed usernames, course codes, tracking numbers, purchasing codes or any datum specific to the organisation (Craswell et al., 2005). In §3.3, we demonstrate that the AOL query history is sweeping, centred around popular culture and often archaic. While the AOL QAC dataset is not ideal, we could not find a more relevant ES benchmark. 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).

Personalisation of an individual user's session or recent history enables 'contextual suggestions', which has proved very effective for completing a user's query prefix in Web Search (Fiorini and Lu, 2018). For example, if a particular user submits a Google search for "Past American Presidents", then if his/her 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 members of the organisation may not wish to be 'profiled'.

2.4 Trending Queries

Queries that have been popular in a recent time period merit a ranking feature to capture temporal behavioural trends. In 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 quite static, and emergent topics were volumetrically drowned out. This has sometimes been referred to as 'The Britney Spears Problem' (Hayes, 2008). 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 and is based on the relative spike in the volume of clustered topic searches in relation to the absolute volume of searches. Unlike the popularity list, the trend list would exclude the constantly popular 'Britney Spears'.

2.5 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 like Apache Solr (The Apache Software Foundation., 2004) are designed for the explicit retrieval of Term Frequency (TF) within a particular field of a schema, such as title, content body, anchors, footers, etc. A candidate indexed from anchor text within a corpus is likely to be more important than a candidate from the content field.

2.6 Jargon/Terminology

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. The same term can have a different meaning outside of the organisation. A search for 'timetable' in WS will probably return bus or train times. Krushwitz 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) (Kruschwitz et al., 2013).

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.

As we will see in §3, the task of detecting enterprise jargon/terminology terms within a corpus lends itself to the fields of NLP and LLM. Once the terms have been detected and scored, they can be used as another feature in our model.

2.7 Learning to Rank for QAC

Users' search history is generally the most likely indicator of current search intent (Chang and Demg, 2020). The use of MPC and MFQ to extract 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 therefore 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 can retrieve and dynamically score suggestions from multiple sources using a 'weightExpression' in real-time.

2.8 Metrics for QAC

The evaluation of QAC performance has two general approaches (Chang and Demg, 2020), each of which has its own metric:

1. The MRR metric focuses on the quality of ranking.
2. The Minimum Keystroke Length (MKS) metric that focuses on savings of a user's keystroke effort (Duan and Hsu, 2011).

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 (Li et al., 2017; Cai and De Rijke, 2016). The MRR metric is appropriate whenever evaluating a list of possible responses to a sample of queries, ordered by the probability of correctness. The Reciprocal Rank (RR) of a query response 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 (Singh et al., 2023a). 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).

3 METHODS

This section describes how logging is configured to capture users' autocomplete session behaviour. This historical behaviour is converted into a QAC dataset, which is used to train a baseline ranking model using LTR-weighted features. We describe each feature and analyse its contribution. We describe how our QACES concept can supplement the baseline heuristic features. The subsequent ranking models are then evaluated using the MRR metric.

3.1 Log Collection for QAC

To enable detailed capture of users' session behaviour, the log4j module[2] of the Apache Solr Enterprise Search platform (The Apache Software Foundation., 2004) is modified to record the suggestion candidates for a given query prefix, the selected candidate (if any), and finally, the submitted query. This session data enables the calculation of an RR score (§2.8). Table 1 lists the recorded parameters for each query session.

Table 1: QAC session logging parameters to capture suggestion candidates, record the user's selection, and the submitted query.

Parameter	ES QAC Dataset example
user id (anon)	qtp1209411469-21
session id	33418(rid)
time stamp	2024-01-14 16:25:37.697
prefix	"aca"
top suggestion candidates	academic registry, academic calendar, academic year structure, academic registry fees, academic year, academic practice, academic resources
submitted query	"academic registry"

The QAC session id commences as soon as the first letter is typed into the search box and ends when the user selects a suggestion candidate or hits submit with the fully typed query (Li et al., 2017).
This enhanced logging has no discernible impact on the responsiveness or latency of our live ES service.

[2]https://cwiki.apache.org/confluence/display/solr/SolrLogging, accessed 16th Jan 2024

Storage was added to the backend linux servers to host both the enhanced logging file size and file retention for 180 days.

3.2 QAC Dataset Construction

3.2.1 Ground Truth

Yossef et al. claim that MPC carries 'the wisdom of the crowds' (Bar-Yossef and Kraus, 2011). It can be regarded as an approximate maximum likelihood estimator (Li et al., 2017; Yadav et al., 2021). In our study, we use MPC as a surrogate for judgements of prefix-candidate pairs. For a given prefix, enumerated judgements {2-5} are allocated to the candidates, representing {irrelevant, moderately relevant, relevant, highly relevant}. The judgement scores are computed as a percentile of the MPC score. Figure 1 shows an extract of our dataset, which has been formatted for use with an LTR framework (§3.5).

3.2.2 Sensical Suggestions

An important pre-processing step applied to any QAC dataset for ES 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. This step is sometimes referred to as producing 'plausible completions' or filtering out of 'nonsensical' suggestions (Yadav et al., 2021). For example, many suggestions extracted from the 2006 AOL WS dataset (Table 2) cannot be used. Even the top AOL queries, such as 'mapquest' and 'myspace' do not produce search results within our ES corpus.

3.2.3 Pre-Processing

Further pre-processing steps involved converting all queries, prefixes, and completions to lowercase. Full stops, other punctuation, and diacritics were removed. Queries and suggestions with less than three characters, more than eight words, or 50 characters (whichever was the bigger) were removed or truncated. All non-English queries were removed.

3.3 Heuristic Ranking Features

We describe the list of ranking functions below as 'heuristic' because they are carefully hand-crafted based on domain knowledge and information retrieval principles.

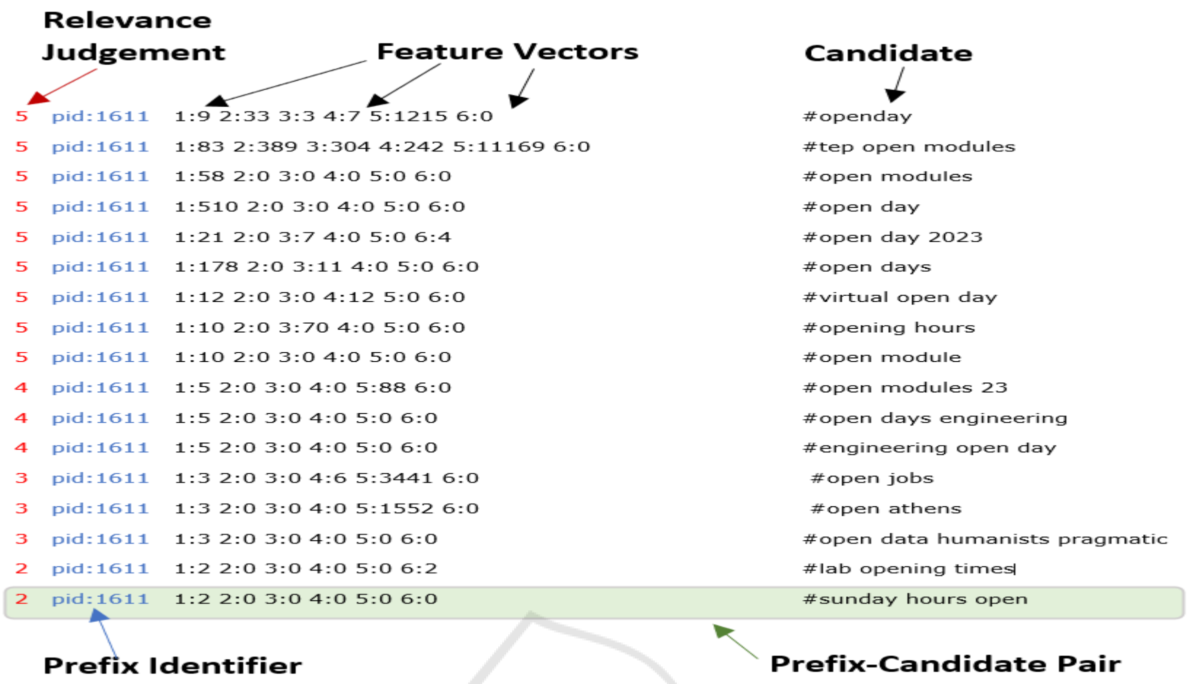
- *MFQ*. The 'Most Frequent Queries' feature consists of a table of queries with their frequency of occurrence extracted from the query logs. An example is "data science 410", which tells us that the query 'data science' has been submitted to the search engine 410 times. The Search Demand Curve (Figure 2) shows the outsize impact that a small number of popular 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 SEO community in 2011, the 18.5% of searchers with the highest occurrence is known as the Fat Head. The next 11.5% is termed the Chunky Middle (Kritzinger and Weideman, 2013). The Long Tail (x-axis) in Figure 2 has been limited for presentation purposes but actually represents 70% of the search volume. In our ES query logs, we see that 65 queries account for 18.5% of all search volume[3].
- *aolFeature* This feature extracts pertinent queries from the AOL WS dataset (20 million search queries from about 650,000 users collected between May and July 2006). Since our ES service must only offer 'sensical' candidates, our extracted list contains just 1740 candidates, which is 0.009% of the total. The feature consists of 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. The AOL queries differ markedly from the types of query we expect for ES, which has a much narrower focus, as shown in Table 2.

Table 2: The top 10 most popular query terms for the 2006 AOL WS compared with our ES query history.

Top 10	AOL WS query terms	ES query terms
1	google	scholarship
2	ebay	fees
3	yahoo	library
4	mapquest	phd
5	yahoo.com	medicine
6	google.com	pshychology
7	myspace.com	erasmus
8	internet	courses esc
9	myspace	vacancies
10	www.google.com	law

- *trendingFeature*. The new or nearly new terms that have been queried in the past 24 hours. To measure an abnormal spike, we must first deter-

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



```
5  pid:1611  1:9 2:33 3:3 4:7 5:1215 6:0          #openday
5  pid:1611  1:83 2:389 3:304 4:242 5:11169 6:0   #tep open modules
5  pid:1611  1:58 2:0 3:0 4:0 5:0 6:0             #open modules
5  pid:1611  1:510 2:0 3:0 4:0 5:0 6:0            #open day
5  pid:1611  1:21 2:0 3:7 4:0 5:0 6:4             #open day 2023
5  pid:1611  1:178 2:0 3:11 4:0 5:0 6:0           #open days
5  pid:1611  1:12 2:0 3:0 4:12 5:0 6:0            #virtual open day
5  pid:1611  1:10 2:0 3:70 4:0 5:0 6:0            #opening hours
5  pid:1611  1:10 2:0 3:0 4:0 5:0 6:0             #open module
4  pid:1611  1:5 2:0 3:0 4:0 5:88 6:0             #open modules 23
4  pid:1611  1:5 2:0 3:0 4:0 5:0 6:0              #open days engineering
4  pid:1611  1:5 2:0 3:0 4:0 5:0 6:0              #engineering open day
3  pid:1611  1:3 2:0 3:0 4:6 5:3441 6:0           #open jobs
3  pid:1611  1:3 2:0 3:0 4:0 5:1552 6:0           #open athens
3  pid:1611  1:3 2:0 3:0 4:0 5:0 6:0              #open data humanists pragmatic
2  pid:1611  1:2 2:0 3:0 4:0 5:0 6:2              #lab opening times
2  pid:1611  1:2 2:0 3:0 4:0 5:0 6:0              #sunday hours open
```

Figure 1: An extract of our LTR formatted dataset including a sample of the prefix-candidate pairs for the “open” prefix. Each candidate has an associated judgement for a particular prefix (generated using MPC). The candidates also have an prefix identifier (pid) and a series of feature vectors.

mine what would be a normal baseline score. This type of calculation lends itself to z-scores, which consider the burst of popularity against the backdrop of the historical average (including its standard deviation). This feature consists of a table of queries with their computed z-score (e.g. ”graduate studies 22.81”).[4] A higher z-score indicates that the query is more ‘trending’.

- *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). In the field of Web Search, this feature can be expected to have a high weighting co-efficient as it both tags meaningful descriptive text and also 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 our corpus. Considerable filtering was required to remove non-descriptive terms and repetitive labels such as ’Next page’, ’Previous Page’, ’Home’, etc.
- *titleFeature*. This TF feature is a list of candidates with their corresponding frequency retrieved from the field of our enterprise corpus. An example is ”communication 333”, which tells us that the term ‘communication’ occurs 333 times in the title field of our corpus.
- *contentFeature*. This TF feature is a list of candidates with their corresponding frequency retrieved from the content field of our enterprise corpus.

Although MPC is typically considered the main indicator of historical relevance (Li et al., 2017), it cannot be included as a feature here (since we already use it as a proxy for ground truth (i.e. the target variable) in our LTR dataset). Similarly, while Personalisation has proved very effective for completing a user’s query prefix in WS, we exclude it as part of this ES research as our organisation does not permit the profiling of individual user data.

3.4 QACES LLM Feature for ES

The task of detecting enterprise jargon/terminology terms within a corpus lends itself to the fields of NLP and LLM. We 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

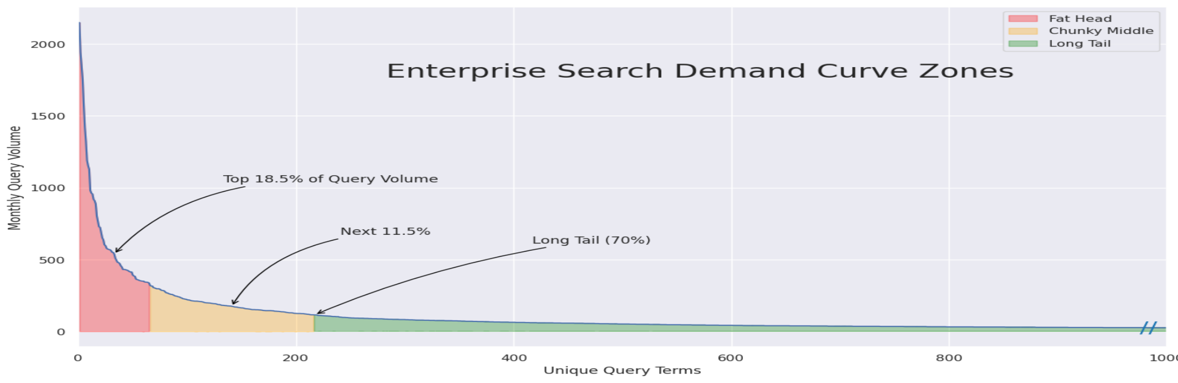


Figure 2: Search Demand Curve for our Enterprise Search query history, showing the so-called 'Fat Head', 'Chunky Middle' and 'Long Tail' zones. For our ES service, the most popular 65 queries represent 18.5% of all search volume.

the jargon terms have been detected, they can be used as a feature in our ranking model.

3.4.1 LLM Synonyms

LLMs are trained on enormous datasets containing vast amounts of text from diverse sources. We use the 'client.chat.completions.create' API response of OpenAI's GPT-4 model (OpenAI 2023, 2023) to produce a list of ten English language 'LLM nearest' terms for each query in the 'fat head' zone of our Search Curve (this is the second column of Table 3). An alternative tool to GPT-4 would have been WordNet (Princeton University, 2010). We opted not to use WordNet, however, as frequency data are not independently available, making it impossible to determine the *nearest* terms.

Use of the GPT-4 API prevents data leaking (Balloccu et al., 2024). We tested the temperature parameter at both 0.5 and 1.0 and observed no obvious changes in the retrieved 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". We struggled to achieve repeatable lists of near-synonyms on each run. The detailed wording of the prompt was necessary to achieve repeatable results. The array included all of the query terms in the 'fat head' of our Search Demand Curve.

3.4.2 ES Corpus Vectorisation

Word embeddings represent terms as dense vectors where similar words are closer together in vector space. We use Word2Vec (Mikolov et al., 2013) to learn representations based on their contextual usage in our ES corpus. This produces word and phrase vectors where vectors close together in vector space have similar meanings based on context. We produce a list of 'Corpus Nearest' terms for each query in the 'fat head' zone of our Search Curve (this is the third column of Table 3).

3.4.3 Detecting Jargon/Terminology

Jargon/terminology consists of enterprise-specific words, phrases, expressions, and idioms that diverge from those universally familiar or understood outside of the 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 3 lists the divergent terms.

3.4.4 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 3. 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 3). Note that the jargony score is applied to the query rather than to the divergent terms.

3.5 Learning to Rank Methodology

The Apache Solr 'weightExpression' parameter of the 'DocumentExpressionDictionaryFactory' dictionary implementation is used to score the suggestions.

Table 3: 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.

'Fat Head' Query	LLM Nearest (X)	Corpus Nearest (Y)	Divergent Terms ($\widetilde{X} \cap Y$)	Jaccard Distance
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

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 RankEval framework (Lucchese et al., 2020). Figure 3 depicts how each feature weighting contributes to ranked suggestions in our ES search box. The solrconfig.xml file is published on github[4]. The RankEval Python open-source tool (Lucchese et al., 2017; Lucchese et al., 2020), based on ensembles of decision trees, is then employed to determine the optimal relative feature weighting and calculate each feature's contribution to the overall ranking efficiency.

3.5.1 Model

The ranking model is generated using the XGBoost implementation of the LambdaMART list-wise ranking algorithm[4]. Table 4 lists the hyper-parameters used.

3.6 MRR Metric Calculation

In our offline study, we break down the MRR scores for the three zones of our ES Search Demand Curve (Figure 2). We compute the MRR scores after three keystrokes. This breakdown helps us distinguish the QAC ranking performance for the most popular queries versus more obscure queries in the long tail zone.

Table 4: Hyper-parameter 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

For the online study, MRR is computed for the users' last keystroke (sometimes referred to as MRR@last (Chang and Demg, 2020)) since this is where the user selects a suggested candidate for submission to the search engine. 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. We calculate MRR@k, where k is the number of offered completion candidates. We present results for k=10 (as is the norm) and k =7 because our live ES service presents a maximum of seven candidates.

The calculated MRR scores are initially computed for heuristic features, which serve as our baseline system (i.e., the 'A' in our online A/B test). The online test will also give us an overall MRR score based on 'real' data (i.e. the combined score across all of the Search Demand Curve zones).

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

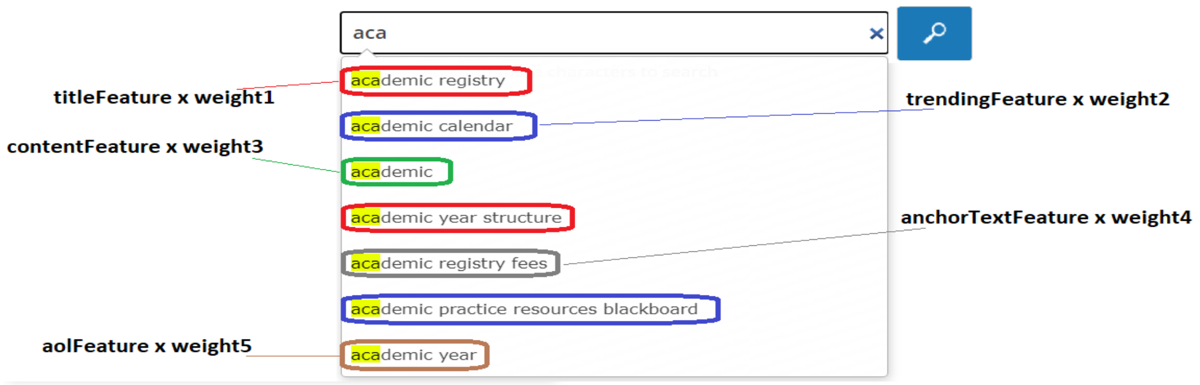


Figure 3: The typed ”aca” prefix presents a list of completion choices via a ranked list of prefix-candidate pairs. Candidates are generated from various sources/features, each of which is ‘weighted’ using LTR.

3.7 Ablation Study Methodology

Not all of the described ranking features are equally important. We perform an ablation study to better understand the contribution of each of the different features toward our QAC ranking model learning capability. We remove one feature from each iteration and perform again the LTR training and testing steps as before. If we observe a large decrease in QAC MRR scores, this indicates that the ablated feature is very important for the model. Our ablation study is carried out for MRR@{1,3,7 and 10}.

3.8 Online A/B Test Methodology

An online test will give us the overall MRR score (across all of the Search Demand Curve zones). Before commencing the A/B test, we perform an A/A test. Also known as a null test, the A/A test is used to establish trust in our experimental platform. This 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, we expect a t-test to prove that any difference in MRR results is not statistically significant (Kohavi et al., 2020).

Having established the integrity of our experimental platform, we subsequently undertake the A/B test to compare the ranking performance of QAC candidates using two pools: -

- A: This is the Control pool and encompasses all of the heuristic features.
- B: This Treatment pool includes A’s heuristic features *and* the additional QACES feature.

We use a load balancer with two servers in an active/active configuration, with a 50% traffic allocation to both the Control and Treatment groups. 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. This ensures that each log file has a complete record.

Since the data collected from group A is independent of the data collected from group B, we use an *unpaired two-sample t-test* to validate statistical significance with α set to 0.05.

4 EVALUATION

In this section, we present the computed LTR weights for each feature, the results of the ablation study and the MRR scores for our offline study. Finally, we compare the MRR scores for our online evaluation (A/B test) of our ranking models, with and without the QACES feature.

4.1 LTR Weights

Table 5 shows the RankEval computed weighting associated with each of our ranking features.

Table 5: LTR weightings for features calculated using the RankEval framework.

Feature	Weight
MFQ	69
titleFeature	22
anchortextFeature	25
contentFeature	20
trendingFeature	9
aolFeature	6
QACES	4

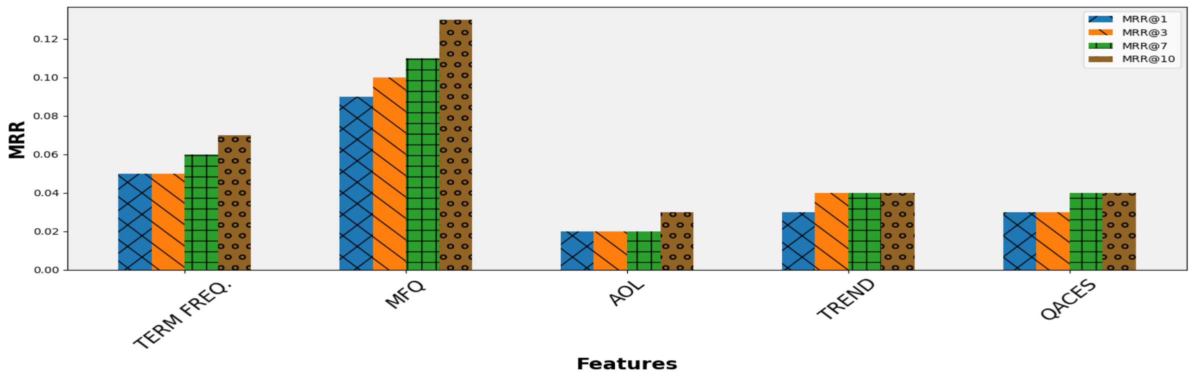


Figure 4: Ablation/leave-one-out analysis showing the contribution of individual features to the MRR performance across the QAC ranking model.

4.2 Offline Evaluation

The MRR scores for selected k-value cutoffs for the three Search Demand Curve zones are listed in Table 6. For comparison, we include the scores for the AOL WS dataset as well as our ES dataset. Across all zones, the AOL scores are consistently higher; we speculate this may be due to the use of a Personalization feature in the AOL WS dataset.

Singh et al. have achieved QAC MRR scores in the region of 0.485 for e-Commerce site search (Singh et al., 2023b), but any comparison is complicated as they omitted to break down scores into Search Demand Curve zones.

Table 6: Offline MRR scores for WS and ES data, with a breakdown for the sections of the Search Demand Curve.

Dataset / Zone	**MRR @1**	**MRR @7**	**MRR @10**
AOL WS Data			
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
ES Data			
Fat Head	0.60	0.68	**0.68**
Chunky Mid	0.32	0.34	0.36
Long Tail	0.19	0.24	0.24
All Zones	0.29	0.33	0.34

4.3 Ablation Study Results

We performed an ablation analysis to better understand the contribution of each of the different features of our QAC ranking model (for MRR@{1,3,7 and 10}). Figure 4 shows their relative contribution totals.

We see that 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. The AOL feature makes the smallest contribution to our model; this may be because many of the suggestions in the feature are deprecated or simply because the feature was not generated from our ES index. The contribution of our QACES jargon/terminology feature is comparable to that of the 'trend' feature.

4.4 Online Evaluation (A/B Test)

We evaluated two ranking models. The first (A) included heuristic features only. The second (B) also included our QACES feature. The ranking score for each ranking model is presented in Table 7. The A/B test was undertaken on the live Enterprise Search service of a large third-level education institution. The model was tested for 16 weeks on 140,000 queries, which resulted in a statistically significant increase in MRR score of +3.8% with a p-value < 0.05.

Table 7: 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 our online evaluation (0.227) is substantially lower than that calculated in our offline study (0.34). 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. We 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).

5 CONCLUSIONS AND FUTURE WORK

In this paper, we discuss the unique requirements for QAC in Enterprise Search and demonstrate the implementation of a QAC ranking model using features whose weightings are computed using Learning to Rank.

We hypothesise and prove that adding our QACES LLM-based jargon/terminology ranking feature to our baseline heuristic LTR model results in a statistically significant improvement to QAC ranking performance (MRR score) on a live Enterprise Search service.

A limitation of the GPT-4 language processing model is that, unlike WordNet, it does not always produce repeatable results. For example, we may get a slightly different list of synonyms for the same query each time it is run. While this does not influence the general reproducibility of results, it may affect the consistency of the generated list of jargon/terminology terms. Our initial investigation of the GPT-4o LM also seems to produce more consistent results and this will be detailed in future work.

Another idea for future work is to explore how our QACES innovation could be adapted for use with query expansion. Finally, an investigation of the causes and factors that affect 'QAC abandon' would be an interesting new direction for the field of auto-completion.

ACKNOWLEDGEMENTS

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. For the purpose of Open Access, the author has applied a CC BY public copyright licence to any Author Accepted Manuscript version arising from this submission.

REFERENCES

Balloccu, S., Schmidtová, P., Lango, M., and Dušek, O. (2024). 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*, 1:67–93.

Bar-Yossef, Z. and Kraus, N. (2011). Context-sensitive query auto-completion. *Proceedings of the 20th International Conference on World Wide Web, WWW 2011*, pages 107–116.

Bentley, J. (2011). Mind the Enterprise Search Gap: Smartlogic Sponsor MindMetre Research Report.

Cai, F. and De Rijke, M. (2016). A survey of query auto completion in information retrieval. *Foundations and Trends in Information Retrieval*, 10(4):273–363.

Chang, Y. and Demg, H. (2020). *Query Understanding for Search Engines*, volume 46 of *The Information Retrieval Series*. Springer International Publishing, Jilin.

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, N., Cambridge, M., and Soboroff, I. (2005). Overview of the TREC-2005 Enterprise Track. In *TREC 2005 conference notebook*, pages 199–205.

Croft, W. B. (2010). Search engines : information retrieval in practice / by Bruce Croft, Donald Metzler, Trevor Strohman.

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.

Davis, M., Eager, A., Edwards, R., Ganesh, B., Holt, M., and Mukherjee, S. (2011). Enterprise Search and Retrieval 2011/2012. In James, M., editor, *Technology Evaluation and Comparison Report*, page 277. OVUM.

Duan, H. and Hsu, B. J. (2011). Online spelling correction for query completion. *Proceedings of the 20th International Conference on World Wide Web, WWW 2011*, pages 117–126.

Fiorini, N. and Lu, Z. (2018). 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*, 3:208–215.

Guo, J., Fan, Y., Ai, Q., and Croft, W. B. (2016). A deep relevance matching model for Ad-hoc retrieval. In *International Conference on Information and Knowledge Management, Proceedings*, volume 24-28-Octo, pages 55–64. Association for Computing Machinery.

Hayes, B. (2008). The Britney Spears Problem. *American Scientist*, 96(4):274.

Joachims, T. (2002). Optimizing search engines using clickthrough data. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Kim, G. (2019). 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)*, pages 5022–5032, Hong Kong, China. Association for Computational Linguistics.

Kohavi, R., Tankg, D., and Xu, Y. (2020). *Trustworthy Online Controlled Experiments: A Practical Guide to A/B Testing - Ron Kohavi, Diane Tang, Ya Xu - Google Books*. Cambridge University Press, Cambridge.

Kritzinger, W. T. and Weideman, M. (2013). Search Engine Optimization and Pay-per-Click Marketing Strategies. *Journal of Organizational Computing and Electronic Commerce*, 23(3):273–286.

Kruschwitz, U. and Hull, C. (2017). Searching the Enterprise. *Foundations and Trends® in Information Retrieval*, 11(1):1–142.

Kruschwitz, U., Lungley, D., Albakour, M. D., and Song, D. (2013). Deriving query suggestions for site search. *Journal of the American Society for Information Science and Technology*, 64(10):1975–1994.

Li, H. (2011). A Short Introduction to Learning to Rank. *IEICE Transactions*, 94-D:1854–1862.

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.

Lucchese, C., Muntean, C., Nardini, F., Perego, R., and Trani, S. (2020). RankEval: Evaluation and investigation of ranking models. *SoftwareX*, 12:100614.

Lucchese, C., Muntean, C. I., Nardini, F. M., Perego, R., and Trani, S. (2017). 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*, pages 1281–1284.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013). Distributed Representations of Words and Phrases and their Compositionality. *Advances in Neural Information Processing Systems*, 26.

OpenAI 2023 (2023). OpenAI GPT-4.

Pass, G., Chowdhury, A., and Torgeson, C. (2006). A picture of search. *ACM International Conference Proceeding Series*, 152.

Princeton University (2010). About WordNet.

Rahangdale, A. and Raut, S. (2019). Deep Neural Network Regularization for Feature Selection in Learning-to-Rank. *IEEE Access*, 7:53988–54006.

Scott, E. (2022). 9 UX Best Practice Design Patterns for Autocomplete Suggestions (Only 19% Get Everything Right) – Articles – Baymard Institute.

Singh, S., Farfade, S., and Comar, P. M. (2023a). 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.

Singh, S., Farfade, S., and Comar, P. M. (2023b). 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). Apache Solr.

Turnbull, D. and Berryman, J. (2016). *Relevant Search*. Manning Publications Co., New York.

White, M. (2018). *Enterprise search*. O'Reilly Media, Sebastopol, CA.

White, R. W. and Marchionini, G. (2007). Examining the effectiveness of real-time query expansion. *Information Processing & Management*, 43(3):685–704.

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.

Yadav, N., Sen, R., Hill, D. N., Mazumdar, A., and Dhillon, I. S. (2021). Session-Aware Query Auto-completion using Extreme Multi-Label Ranking. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 3835–3844.