# NAVEGANDO CONSULTAS SEMANTICAMENTE ANOTADAS PARA ENTENDIMENTO DE TAREFAS

ARTHUR BARBOSA CÂMARA

# NAVEGANDO CONSULTAS SEMANTICAMENTE ANOTADAS PARA ENTENDIMENTO DE TAREFAS

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação do Instituto de Ciências Exatas da Universidade Federal de Minas Gerais como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação.

Orientador: Rodrygo Luis Teodoro Santos

Belo Horizonte

Outubro de 2018

ARTHUR BARBOSA CÂMARA

# NAVIGATING SEMANTICALLY ANNOTATED QUERIES FOR TASK UNDERSTANDING

Dissertation presented to the Graduate Program in Computer Science of the Federal University of Minas Gerais in partial fulfillment of the requirements for the degree of Master in Computer Science.

Advisor: Rodrygo Luis Teodoro Santos

Belo Horizonte

October 2018

**Ficha catalográfica elaborada pela Biblioteca do ICEx - UFMG**

Câmara, Arthur Barbosa

C172n Navigating semantically annotated queries for task understanding / Arthur Barbosa Câmara. — Belo Horizonte, 2018.
xxii, 54 f.: il.; 29 cm.

Dissertação (mestrado) - Universidade Federal de Minas Gerais – Departamento de Ciência da Computação.

Orientador: Rodrygo Luis Teodoro Santos

1. Computação – Teses. 2. Recuperação da Informação. 3. Banco de dados – busca. 4. Redes Neurais. I. Orientador. II. Título.

CDU 519.6*73(043)

UNIVERSIDADE FEDERAL DE MINAS GERAIS
INSTITUTO DE CIÊNCIAS EXATAS
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

# FOLHA DE APROVAÇÃO

Navigating Semantically Annotated Queries for Task Understanding

**ARTHUR BARBOSA CÂMARA**

Dissertação defendida e aprovada pela banca examinadora constituída pelos Senhores:

PROF. RODRYGO LUIS TEODORO SANTOS - Orientador
Departamento de Ciência da Computação - UFMG

PROF. EDLENO SILVA DE MOURA
Departamento de Ciência da Computação - UFAM

PROF. MARCOS ANDRÉ GONÇALVES
Departamento de Ciência da Computação - UFMG

PROF. ANISIO MENDES LACERDA
Departamento de Computação - CEFET-MG

Belo Horizonte, 5 de novembro de 2018.

*To my wife, Laís, for her never-ending support and love.*

# Acknowledgments

I would like to express my gratitude for everyone that made an impact in my life during the years that lead to this dissertation.

First and foremost, I am grateful to God. To Him all the glory and praise, for providing me with everything (and everyone) that led me to my current academic path.

I would like to thank my advisor, Rodrygo, for his infinite patience over many hours for many days (and even a couple of nights!) of discussions, ideas and so much more. I'm grateful for all the times he pointed at my mistakes (and, of course, helped me fixing them). Ultimately, he is the responsible for my willingness to move forward with my studies and becoming a real researcher.

My wife, Laís, that embarked with me on this (crazy) journey, was my pillar during the whole time. Knowing that I am able to rely on someone during hard and easy times, rant and discuss ideas (even if she has no idea what I am talking about) is one of the greatest feelings that someone can have. For these (and many other) reasons, I love her, and thank her immensely for bearing with me during these years, and many more to come.

To my family, Kleber, Neila and Ivan, for their patience, for pushing me through every step of the process, for every time they asked me how my work was going, for their love and support, even when far away, I am extremely thankful.

Last but not least, to every one of my friends, that were always there for me, for cheering with me and supporting me. Namely, Filipe Arcanjo, Caroline Scholles, Guilherme Cordeiro, Deborah Morais, Guilherme Gomes, Jacqueline Torres, Paula Gomes, Pedro Lopes, Lílian Dornelas, Guilherme Paiva, Mariana Santos, Alberto Ueda, Bruno Laporais, Felipe Moraes, Gustavo Penha, Jordan Silva, Rafael Glater, and Sabir Ribas.

# Resumo

Enquanto sistemas de busca gradualmente se transformam em assistentes pessoais, usuários cada vez mais se voltam a máquinas de busca para completar tarefas complexas, como planejar uma viagem, alugar um apartamento ou investir em ações. Um desafio-chave para uma máquina de busca é o de entender a tarefa de um usuário por trás de uma consulta de exemplo, como "passagens para o panamá", "estúdios em los angeles" ou "ações do spotify", e recomendar outras consultas que ajudariam o usuário a completar sua tarefa. Nesta dissertação, propomos três estratégias para entendimento de tarefas, navegando um histórico de consultas semanticamente anotadas e usando uma mistura de representações explícitas e latentes de consultas inteiras e partes de consultas. Avaliamos minuciosamente as estratégias propostas no contexto da TREC 2016 Tasks track e via crowdsourcing. Nossos resultados demonstram a efetividade das estratégias propostas em termos de diversidade e novidade, além de sua complementaridade, com melhoras significativas em relação a várias abordagens de recomendação de consultas do estado-da-arte adaptadas para essa tarefa. Além disso, mostramos que nossa proposta é particularmente efetiva para consultas na cauda-longa e consultas difíceis, que englobam um grande número de sub-tarefas.

# Abstract

As search systems gradually turn into intelligent personal assistants, users increasingly resort to a search engine to accomplish a complex task, such as planning a trip, renting an apartment, or investing in stocks. A key challenge for the search engine is to understand the user's underlying task given a sample query like "tickets to panama", "studios in los angeles", or "spotify stocks", and to recommend other queries to help the user complete the task. In this dissertation, we propose three strategies for task understanding by navigating a semantically annotated query log using a mixture of explicit and latent representations of entire queries and of query parts. We thoroughly evaluate our proposed strategies in the context of the TREC 2016 Tasks track and via crowdsourcing. Our results demonstrate the effectiveness of the proposed strategies in terms of diversity and novelty, as well as their complementarity, with significant improvements compared to multiple state-of-the-art query suggestion baselines adapted for this task. Moreover, we show that our proposal is particularly effective for long-tail queries as well as for hard queries, which encompass a large number of subtasks.

# List of Figures

# List of Tables

# Contents

# Chapter 1

# Introduction

Nowadays, people rely more and more on Information Retrieval systems, on its various forms and shapes, for daily tasks. From traditional search engines to recommender systems and smart assistants embedded on watches and smart speakers, these systems play a key role on how we interact with and consume data and information. A classic interaction with a search engine, for instance, consists on a user submitting a series of queries, usually in short, textual form, to such system, expressing their information needs. Generally speaking, a user may be expecting in return a document, image, video, or any other artifact that may satisfy their information need [9].

However, a query usually represents a complex task that the user might be trying to complete. For instance, a user looking to invest in stocks will need to search for multiple companies, valuation history, profitability, etc. In these cases, a task may contain a number of subtasks that will require multiple interactions with the system, until the task is fulfilled. For instance, a user trying to accomplish a task like *"learn how to program"* may need to accomplish multiple subtasks related to the problem, like *"Find what are good websites to learn about programming"*, *"What are good programming languages for beginners"* and so on. Understanding this task and uncovering its underlying subtasks is key to supporting the user toward the task completion.

From this scenario, task understanding appears as a natural next step for Information Retrieval. The goal of such a system is to, given some piece information of the user's task (say, a query related to a part of the task), try and generate outputs that will help the user on completing such task, including all relevant subtasks that could be related to the original task. Of particular interest of this dissertation is the scenario where a user submits a query to a search engine and the system tries to generate a set of new queries that would help the user on their task completion.

Since the main output of task understanding is a set of key phrases[1] for a given input query, it can be modeled as a query suggestion problem. Some of the most successful approaches to query suggestion leverage different neural network architectures to generate suggestions. For example, Sordoni et al. [44] tackled the query suggestion problem by using a hierarchical recurrent encoder-decoder neural network that learns representations for both queries and sessions. Chen et al. [16] proposed attention-based networks for capturing session structure and generating new suggestions. Nonetheless, these methods may fail to address the complex nature of task understanding, which often involves producing a diverse set of suggestions given a previously unseen query as input.

To address this gap, the Text REtrieval Conference (TREC) included a Tasks track in their 2016 edition, where one of the goals was to evaluate task understanding systems [45]. The most successful solutions presented during the 2016 and 2017 editions [25] sought to achieve task understanding by aggregating and re-ranking results from multiple external sources, notably, query suggestion systems from commercial search engines [23]. However, these approaches rely heavily on external sources, dealing with most of them as black boxes, with little to no control of how their outputs are generated, which limits not only reproducibility, but also potential applications to other domains and extensions.

In this dissertation, we seek to study a more principled approach to task understanding. To promote the understanding of tasks underlying rare or even unseen queries, we model semantically annotated query segments into a bipartite graph connecting named entities and the contexts where these entities appear in a query log. To produce suggestions that cover a wide variety of subtasks, we propose three strategies for navigating the entity-context graph given an input query (i.e., an edge on the graph): *direct expansion*, in which we look for other contexts associated with the same entity present in the original query; *syntagmatic expansion*, where we explore contexts associated with other, similar entities to the original query entity; and *analogical expansion*, where we leverage topically similar entities for expanding our set of entities with other, non-trivial entities.

Through a comprehensive analysis using the experimentation paradigm provided by the TREC Tasks track [25, 45] enriched via crowdsourcing, we contrast these strategies in terms of the utility of their produced suggestions, as well as their diversity and novelty. Moreover, we show that these strategies are complementary and that their combination consistently outperforms state-of-the-art query suggestion baselines

[1] In this dissertation, we use the terms queries and key phrases indistinctly.

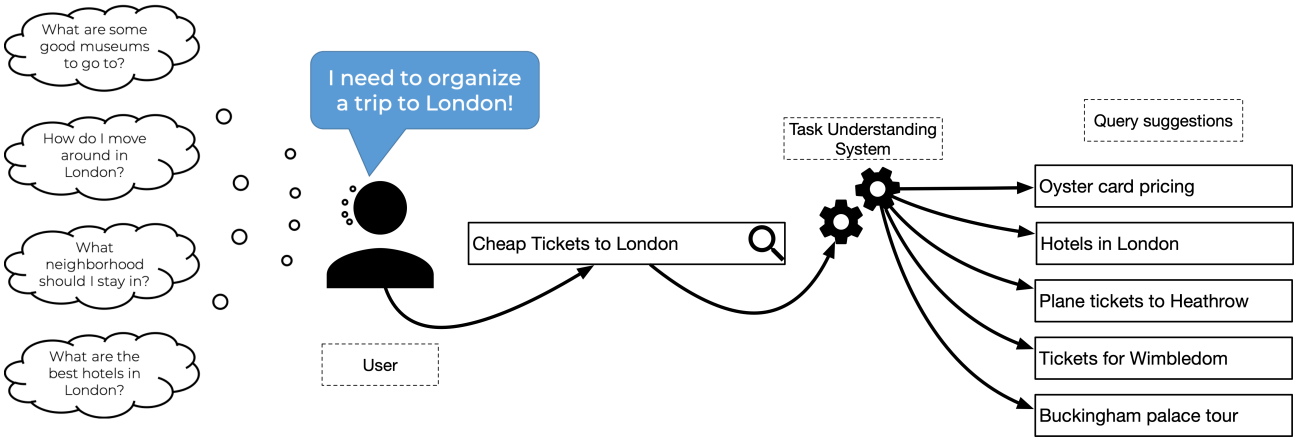


**Figure 1.1.** User in a complex task environment.

from the literature. Further breakdown analyses reveal that our combined approach is particularly effective for long-tail queries as well as hard queries, which demand the identification of a large number of subtasks.

## 1.1 Motivation

The omnipresence of the World Wide Web in our daily lives is not up for discussion, and major increases are expected in the next few years, as we expand to more and more areas previously unreachable with standard connection. No longer restricted to a desktop, we can easily search for anything using desktop computers, laptops, tablets, smart phones, watches and smart speakers. Finding exactly what a user is looking for is always a challenge, and trying to understand the underlying tasks they aim to complete is crucial to increase the efficiency of these systems.

Given the ease of access to to IR systems, almost every daily task we want to accomplish eventually encompass at least a couple of interactions with such system. Be it planning a trip to a foreign country, learning something new, looking for information on a book or even adopting a new cat, it is very likely that we will interact with an IR system in some point in time. More specifically, with a search engine.

Given the rise of smart assistants, like Microsoft Cortana, Apple Siri, Google Assistant, Samsung Bixby and Amazon Alexa, coupled with the aforementioned ubiquitous presence of IR, it is even more critical to understand how users interact on search engines, in order to better understand what tasks they may be looking to complete, making this search process faster and more accurate.

In Figure 1.1, we can see an example of a user exploring such system. Say that user, looking to complete the task *"Plan a trip to London"* submits a query such as

*"Cheap tickets to London"* to a search engine. A useful interaction with such system would yield him a list of next queries like *"Hotels in London"*, *"Tickets for Wimbledon"*, *"Buckingham palace tour"* and so on. This system could also, without explicit command, perform one of the suggested queries and suggest actions within a smart assistant interface, like *"Do you want to book this hotel for* $100.00 *a night?"*, essentially completing a subtask for the user.

## 1.2 Dissertation Statement

In this dissertation, we state that current query suggestion systems and other approaches for task understanding are not efficient enough for tasks where a user may need to cover multiple subtasks. We claim that, by modeling a query log as a bipartite graph and by using multiple strategies when navigating on that graph, we can accomplish a better understanding of the task at hand to the user, and further produce better query suggestions on these tasks. As such, the three main research questions to be answered by this dissertation, given our three proposed approaches to the problem, are the following:

Q1. How effective are our proposed strategies?

Q2. How complementary are our proposed strategies?

Q3. How do our strategies perform for different types of query?

## 1.3 Contributions

We believe that the main contributions of this work are the following:

1. Three new strategies for task understanding.

2. A new method for combining the proposed strategies, which outperforms current state-of-the-art approaches.

3. A thorough examination on how these strategies behave under different scenarios.

## 1.4 Dissertation Outline

This dissertation is structured as follows:

- **Chapter 2: Related Work** discusses some of the related work to this dissertation, like query suggestions, word embeddings and query diversification.
- **Chapter 3: Navigating Semantically Annotated Queries** shows in details each of our three strategies for task understand, how they work and their intuitions.
- **Chapter 4: Experimental Setup** presents details on how we ran our experiments in order to answer our research questions, our assumptions and evaluation methodologies.
- **Chapter 5: Experimental Evaluation** shows the results of each of our methods, how they are complementary and how they behave under multiple situations.
- **Chapter 6: Conclusions** concludes the dissertation, pointing to possible future research to be made on the area.

# Chapter 2

# Background and Related Work

In this chapter, we discuss some background themes that are of great importance to our approaches on task understanding, in Section 2.1. We also discuss some of the works related to this dissertation in Section 2.2.

## 2.1 Background

In this section, we discuss some themes that are of great importance on our approaches for task understanding. Specifically, we discuss works on word embeddings in Section 2.1.1, entities in queries in Section 2.1.2 and general diversification techniques in Section 2.1.3.

### 2.1.1 Word Embeddings

Another area that has seen a significant growth in the last few years is the area of word embeddings or distributed representations. The goal of such approaches is to generate dense and compact representations of words or topics. Instead of classic, one-hot approaches [38], these techniques try to capture more meaningful semantic representations of words.

In what can be considered the seminal paper on using neural networks for generating dense representations of words, Bengio et al. [4] proposed a simple neural network architecture for learning distributed representations of words, based on their neighborhood.

Extending this work and, arguably, bringing the topic to the spotlight for Natural Language Processing, Mikolov et al. [33] proposed two different shallow network architectures for training word embeddings, SkipGram and Continuous Bag of Words

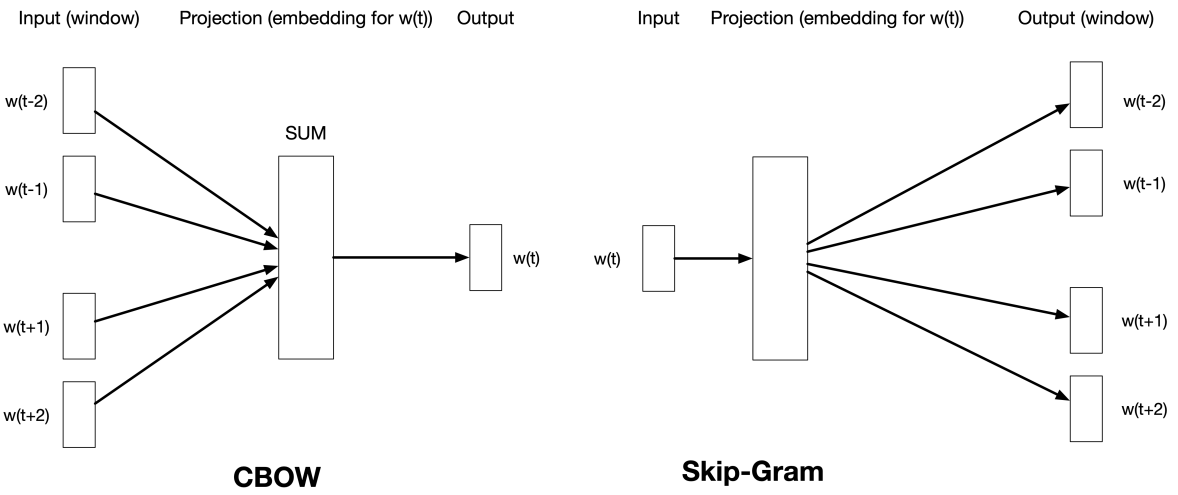


**Figure 2.1.** Word2Vec architectures. Note that CBOW predicts the current word given the context, and Skip-Gram, the opposite.

(CBOW). Both of these architectures are shown in Figure 2.1. As seen in the figure, CBOW tries to predict the target word given a window of context around it. Skip-Gram, on the other hand, tries to predict the context window around the target word. In that same work, they also show that these representations are capable of performing analogies over the embedding space. As a classical example, $\vec{e}(king) - \vec{e}(man) + \vec{e}(woman) \approx \vec{e}(queen)$, where $\vec{e}(\cdot)$ is the embedding operation.

Finally, other approaches, like the one demonstrated by Pennington et al. [36], use matrix decomposition techniques over a matrix of term co-occurrence to generate such representations. In fact, according to Levy and Goldberg [29], the approaches proposed by Pennington et al. [36] and Mikolov et al. [33] are equivalent. Another interesting word embeddings strategy is Facebook's FastText [21], that employees a faster architecture, based on bags of n-grams.

One of the key advantages of word embeddings is their ability to hold semantic relationships between words. Naturally, one would like to measure the semantic distance between two words, in a similar fashion to regular string distance techniques, like edit distance. One common approach when dealing with this is to calculate the distance between two word embeddings in the same space using the cosine distance. If two words are similar, it is expected that they are close to each other in the embedding space. However, calculating the distance between two documents (i.e., two bags of word embeddings), a problem that will be central to this dissertation, is not straightforward. A classical approach to solving this problem is to calculate the distance between centroids of the embeddings of every word on both documents. Kusner et al. [27] however, introduced the Word Movers Distance technique, an adaptation

of the transport problem of Earth's Movers Distance. Essentially, the WMD metric can be seen as the minimum amount of distance the words of one document need to "travel" in the embedding space to reach the other document words. Their approach shows great performance in tasks like nearest neighbor document classification, demonstrating their effectiveness in mapping distances between documents. Since our work relies heavily on embedding distances, a more robust approach for calculating the semantic distance between queries (represented as a bag of words) is needed, and WMD provides this metric throughout this work.

### 2.1.2 Entities in Queries

Finally, the study of entities in queries is not new. However, only a handful of these explicitly use them for query suggestions.

Guo et al. [22] demonstrated a method for extracting named entities from queries using a weakly-supervised version of LDA. While the scope of their work is limited to a few categories (like books, games and movies), their WS-LDA approach is highly effective on also classifying these entities.

Blanco et al. [6] introduced a probabilistic model coupled with dynamic programming for fast entity extraction on queries. Their framework, named FEL (Fast Entity Linker), is shown to perform fast enough to be used on large scale implementations of search engines, retrieving entities in under 1ms on most modern laptops.

Meij et al. [32] proposed a machine learning framework, coupled with a language model for entity retrieval, with over 30 hand-crafted features in order to rank these entities for further suggestions.

Another work in the topic comes from Glater et al. [20]. In their work, they proposed a learning framework for annotating queries with entities and entities attributes, focused on the user intent of the query.

### 2.1.3 Search Result Diversification

Diversification is a key topic on information retrieval, and of particular interest for our work. As stated before, the goal of the task understanding problem is to be able to retrieve suggestions that are relevant for multiple subtasks of a main task. Therefore, strategies that prioritize diversity are of paramount importance for task understanding. The natural tendency to ambiguity of queries, coupled with a focus on standard, relevance-based retrieval may lead to poor results, or even a user completely abandoning her search session [15].

As discussed in Santos et al. [41], the search result diversification problem is an instantiation of the maximum coverage problem, therefore, a NP-Hard problem. Thus, a number of different approaches can be used for retrieving diverse documents, most of these using greedy heuristics for re-ranking an initial ranking, varying on their definition of a scoring function for an item $d$ in the original ranking, $f(q, d, \mathcal{D}_q)$, where $q$ is the input query and $\mathcal{D}_q$ the set of re-ranked documents [41].

According to Radlinski et al. [37], these approaches can be classified as either *extrinsic*, where the system tries to explicitly address the ambiguity of the user's information need, or *intrinsic*, where the system tries to avoid ambiguity within the search result. Of specific interest of this work, the latter approach presents more opportunities for strategies using latent or explicit representations of the queries on a query log. For instance, the Maximum Marginal Relevance approach, proposed by Carbonell and Goldstein [12] in the context of IR systems, uses the following formulation for the function $f(q, d, D_q)$, as described by Santos et al. [41]:

$$f_{MMR}(q, d, \mathcal{D}_q) = \lambda f_1(q, d) - (1 - \lambda) \max_{d_j \in \mathcal{D}_q} f_2(d, d_j), \tag{2.1}$$

where $f_1(q, d)$ is a relevance function between the query $q$ and a document $d$ and $f_2$ estimates the similarity of $q$ to the documents already re-ranked in $\mathcal{D}_q$.

## 2.2 Related Work

In this section, we present some works that are relevant to this dissertation. Namely, we discuss results from Query Suggestion in Section 2.2.1, Query Suggestion Diversification in Section 2.2.2 and the TREC Task Understanding track from 2015 to 2017 in Section 2.2.3.

### 2.2.1 Query Suggestion

Given a query conveying a user's complex task, which may encompass multiple subtasks, the goal of a task understanding system is to generate a set of key phrases that are useful toward completing as many of these subtasks as possible. This problem can be seen as a specialization of the query suggestion problem, which has been extensively investigated over the past decade. In particular, query suggestion approaches leverage a plethora of ranking signals from a query log in order to infer the relevance of a suggestion given an input query [42]. Given this proximity, the most natural approach to

solving the task understanding problem is to model it as a classical query suggestion problem.

It is also worth noting the difference between the task understanding problem, as stated in Yilmaz et al. [46], and a regular query suggestion problem. Given the nature of task understanding, the definition of relevance changes when comparing with regular query suggestion. For task understanding, the coverage of multiple subtasks is more important than regular relevance traditionally measured. It can be seen, for instance, when the TREC Tasks Track chose $\alpha$-nDCG and ERR-IA as the most important metrics on their evaluation [25, 45, 46], enforcing the focus on diversification and novelty metrics. However, given the similarity in the approaches, in this work, we use both task understanding and query suggestions interchangeably.

One of the seminal papers in the area, Beeferman and Berger [3] discuss how to cluster queries submitted to a search engine based on clickthrough data. Effectively, they pioneered the idea of generating a bipartite graph over the query log, connecting queries that produced a click on a document. This first set of strategies, over the same bipartite graph, has been used, for years, as a successful approach for query suggestions.

Further exploring this path, Baeza-Yates et al. [1] state that these connections between URLs and queries can be treated as semantic connections that can be useful for query suggestions. On Baeza-Yates and Tiberi [2] they show that such graphs is sparse, and follows a number of power-law distributions, like that there are a few URLs with a lot of clicks and a lot of URLs with a few clicks, few queries with high frequency and many queries with a low frequency and so on. In the same vein, Cao et al. [11] extended that same concept by clustering similar queries. Their work shows that these clusters can be defined as concepts, and a path within these clusters can be used to generate query suggestions.

Also following the same bipartite graph strategy, Mei et al. [31] introduced the concept of hitting time, the time it took for the query to be visited on a random walk on the bipartite graph, starting on the input query. Similarly, Boldi et al. [7] used a variety of syntax and time signals to build a machine learning model that connects queries by similar intents, building a graph that can be used to generate query suggestions via biased random walks.

Another common approach is to cluster queries based on user sessions. Jones et al. [24] proposed to generate query suggestions based on other sessions with similar substrings to the original entity. Their work leverages a number of signals from these sessions, like edit distance, for generating these suggestions. Also leveraging user sessions, Fonseca et al. [17] proposed a query suggestion system based on association rules for retrieving query pairs that are highly related across sessions.

A problem with such approaches is the lack of diversity that comes from the focus on queries that actually generated a click on a document. To solve this problem, as well as to tackle long-tail queries, Broccolo et al. [8] proposed to represent queries at the term level, by indexing each query as a "virtual document" comprising the terms of other queries that preceded it in a successful search session (i.e., a session that led to a click). Therefore, a user session could be abbreviated by suggesting the last query of previous similar sessions, casting the problem as a standard search over the index of virtual documents.

Given the advance of machine learning in natural language processing, in particular deep learning techniques, it was natural to apply these techniques for the query suggestion problem. In the last few years, approaches using deep neural networks for query suggestion have also been proposed, with major improvements over the previous state-of-the-art. For instance, Sordoni et al. [44] used a recurrent encoder-decoder network which captures the notion of queries and user sessions (meaning, a set of queries submitted by the same user in a short time period). They show that their approach is capable of generate query suggestions that are highly adherent to the search intent in the original query. More recently, Chen et al. [16] employed an attention-based network for capturing semantic structures on user preferences, on top of query and sessions.

### 2.2.2 Query Suggestion Diversification

When dealing with task understanding, one of the most important aspects is the coverage of subtasks. Given a query, there could be multiple related subtasks, all unknown to the user. Therefore, diversifying the results of such a system is extremely important.

A complex task may comprise multiple subtasks, each of these may need a different query to be completed. Therefore, a task understanding system, in order to be successful, need to be able to generate a set of query suggestions that is diverse, in order to encompass a potentially large number of subtasks. Thus, query suggestion diversification is also of great importance for this work.

Some works focused on diversifying document rankings [41] have also been adapted for diversifying query suggestions. Song et al. [43] proposed a learning approach to produce diverse suggestions in response to a query, by measuring the deviation of the document rankings produced for each suggestion and the original query. By retrieving a set of candidates using multiple techniques, these are re-ranked using a function explicitly designed for diversification, including multiple features, like category of the queries, URLs similarity and so on.

Extending the work of Broccolo et al. [8], Santos et al. [40] built a new learning

to rank framework for promoting suggestions that are useful for a document ranking diversification approach, such as xQuAD [39].

### 2.2.3 TREC Task Understanding Track

The Text REtrieval Conference (TREC), in their 2015, 2016 and 2017 editions [25, 45, 46] proposed a new track entitled Task Understanding. In this conference, participants are asked to submit, given a query related to a task, a set of key phrases that could be used as queries to complete a number of unknown subtasks. These submissions are then submitted for human judgment, where their efficacy is evaluated on each subtask, on a scale from 0 (non relevant) to 2 (highly relevant). Participants are ranked by the diversity and novelty of their suggestions when covering the subtopics.

The most successful and published approaches to the track are usually centered around two strategies: (1) retrieving a set of suggestions from a number of commercial search engines and re-ranking those [23] or (2) retrieving key phrases from web documents [5].

As an example of strategy (1), Hagen et al. [23] submitted the original query and entity to nine different systems, from commercial engines query suggestions to Wikipedia, and re-rank the results using manually defined weights.

As for (2), Bennett and White [5] extracted anchor texts from the ClueWeb12 dataset, and retrieving these extracted anchor texts when a candidate is a superset of the original query.

Finally, Garigliotti and Balog [19] tried to mix both strategies (1) and (2), by submitting the original query to a commercial search engine, retrieving their query suggestions and extracting a set of keywords from the retrieved documents.

## 2.3 Summary

In this chapter, we discussed some works that are related to this dissertation. In Section 2.2.1, we outlined some of the key works that deals with suggesting queries to a user in a search engine. In Section 2.2.2 we discussed about strategies that specifically try to maximize diversity of suggestions in query suggestion environments, a key component for effective task understanding. Finally, in Section 2.2.3, we described the TREC Task Understanding track that initially inspired this work. We also presented the strategies participants on the competition adopted in the last few years.

Section 2.1 outlined some works that are of paramount importance to this work, providing background for the strategies described further Chapter 3. In Section 2.1.1 we

showed how word embedding techniques can be used for representing terms as compact and dense vectors. In Section 2.1.2 we described some strategies that focus on entities on search engines. Finally, Section 2.1.3 discussed some techniques for diversification of results in information retrieval systems.

In this dissertation, we propose three different strategies for query suggestions for task understanding. In contrast to the approaches described in this chapter, our proposed strategies represent a more principled approach to query understanding. In particular, we show that one can leverage a bipartite graph, diverse from Beeferman and Berger [3] to generate effective queries suggestions focused on task understanding. By focusing on named entities, our methods prioritize queries that are related to the original target of the task. We also discuss how word embedding techniques, such as the one presented by [33] can be used to increase suggestion diversity and coverage of subtopics. Our proposed strategies are also highly complementary, each covering a different aspect of task understanding, as we describe in the next chapter.

# Chapter 3

# Navigating Semantically Annotated Queries

As discussed before, a user task can cover a variety of subjects, from planning a trip to another country ("tickets to panama"), to building an investment portfolio ("spotify stocks"), to learning something new ("python tutorial"), with each task encompassing multiple subtasks.

Formally, one can define the task understanding problem as follows: Given a query $q$ as a sample of the user's underlying task $\mathcal{T}$, which comprises $n$ unknown subtasks $\{t_1, t_2, \cdots, t_n\}$, our goal is to produce an ordered set of $k$ key phrases $\mathcal{S} = \{s_1, s_2, \cdots, s_k\}$, that can be used as queries to a search engine, with maximum coverage of the subtasks in $\mathcal{T}$ and with minimum redundancy.

Given this definition, the diversification factor appears as a key component of any task understanding approach. Since the goal is to cover the largest number of subtasks, and these are unknown beforehand, this factor is a key distinction between regular query suggestion and a system that aims to solve the task understanding problem.

Given this formulation, in particular the fact that the subtasks $\{s_1, s_2, \cdots, s_k\}$ are not known beforehand, employing a "one-size-fits-all" method may not be ideal. This is due to the fact that any strategy would need to cover varying task complexity, number of entities and query rarity. Therefore, we state that, by using multiple, complementary techniques and further combining these is a more promising direction for solving these issues.

In this chapter, we begin by describing our method for semantically annotating the query log, in Section 3.1. We then move to discuss our three proposed strategies for query understanding: *Direct Expansion* in Section 3.2, *Syntagmatic Expansion* in

Section 3.3 and *Analogical Expansion* in Section 3.4.[1]

## 3.1 Entity-Context Graph

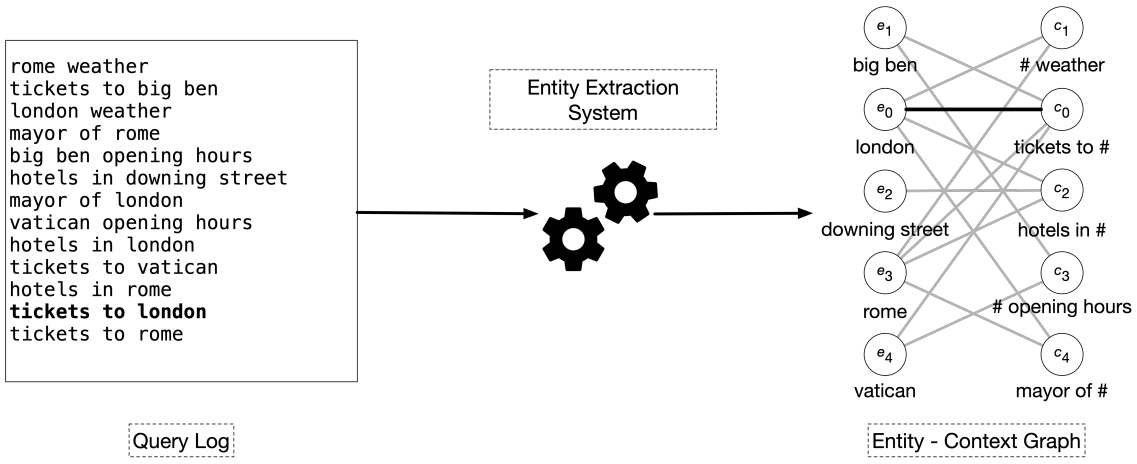


**Figure 3.1.** Extracting entities from the query log and building a bipartite graph of entities and contexts.

Most previous works on query suggestion deal with entire queries, as discussed in Section 2.2.1. However, this can be problematic, specially when dealing with rare or even unseen queries. In these situations, such approaches may not have enough background information on the input query to make good recommendations of key phrases related to the subtasks from $\mathcal{T}$.

Instead of working with entire queries, we propose to represent queries at the segment level, by focusing on tasks centered around queries with a named entity, which amount to over 70% of the web search traffic [22]. In particular, inspired by Guo et al. [22], we segment each query into two parts: a named entity $e$ and its associated context $c$. Hence, we can represent a query such as "tickets to london" as a tuple of entity and context $(\text{london}, \text{tickets to } \#)$, where $\#$ is a placeholder for the entity [2]. We assume that a named entity is any concept that can be matched to an unique URL in Wikipedia.

The problem of extracting entities from queries can be tackled using multiple approaches, as shown in Section 2.1.2. In this work, we focus on the framework proposed by Blanco et al. [6]. Given its high accuracy and speed, we are able to extract every entity in the query log, with over 2M queries, in a couple of hours.

[1] The names syntagmatic and paradigmatic are derived from linguistics. Syntagmatic similarity can also be described as "topical" similarity (meaning, from the same topic) and paradigmatic as "typical" similarity (meaning, from the same type).

[2] Also note that we allow for empty contexts.

Therefore, we can represent a query log as a weighted bipartite graph $\mathcal{G} = (\mathcal{E}, \mathcal{C}, \mathcal{L})$ where $\mathcal{E}$ denotes the set of entities present in the query log, $\mathcal{C}$ the set of contexts in the log, and $\mathcal{L} = \{(e, c) \mid e \in \mathcal{E}, c \in \mathcal{C}\}$ denotes the set of edges corresponding to queries in the log, weighted by their lift $\ell(e, c)$, according to:

$$\ell(e, c) = \frac{\Pr(e, c)}{\Pr(e)\Pr(c)}, \tag{3.1}$$

where $\Pr(e, c)$ is the probability that both $e$ and $c$ appear in the same query in the query log, $\Pr(e)$ the prior of the entity $e$ in the log, and $\Pr(c)$ the prior of the context $c$. This formulation is used so entities or contexts that are too popular in the log, but not locally relevant, will receive a lower weight. For instance, entities like "Google", "AOL" and the empty context "#" are extremely popular on the query log. However, it is highly unlikely that they would be helpful to a user accomplishing any specific task centered on an entity. Figure 3.1 illustrates the graph $\mathcal{G}$ (right) produced for a query log sample (left), with the input query highlighted in bold.

With this representation, the task understanding problem then becomes a problem of recommending edges on $\mathcal{G}$, even if the recommended edge (or any part of it, namely $e$ or $c$) is not present in the original query log. In the remainder of this section, we describe three complementary strategies for navigating the graph $\mathcal{G}$ in order to recommend useful and diverse suggestions for task understanding. These strategies are illustrated in Figures 3.2. In these examples, for the Direct expansion strategy, the recommended edges (or queries) would be ("london", "hotels in #"), ("london", "mayor of #") and ("london", "# weather"). For Syntagmatic expansion, the two generated suggestions are ("big ben", "# opening hours") and ("big ben", "tickets to #"). Finally, for the Analogical expansion, we use the entity "rome" as an intermediate similar entity. As "vatican" is a frequent next entity, following "rome", we map this movement back to the original entity ("london"), resulting in a suggested entity "downing street". This suggested entity is then matched with relevant contexts, resulting in the suggestion ("downing street", "hotels near #") and ("downing street", "how to get to #").

## 3.2 Direct Expansion

Given an input query $q = (e_0, c_0)$, a simple strategy to identify related subtasks is to look for additional contexts related to $e_0$. The intuition behind such strategy is to search for new actions (contexts) than can be applied to the same object (entity) $e_0$ that the user initially searched for. Following the idea of task understanding as an

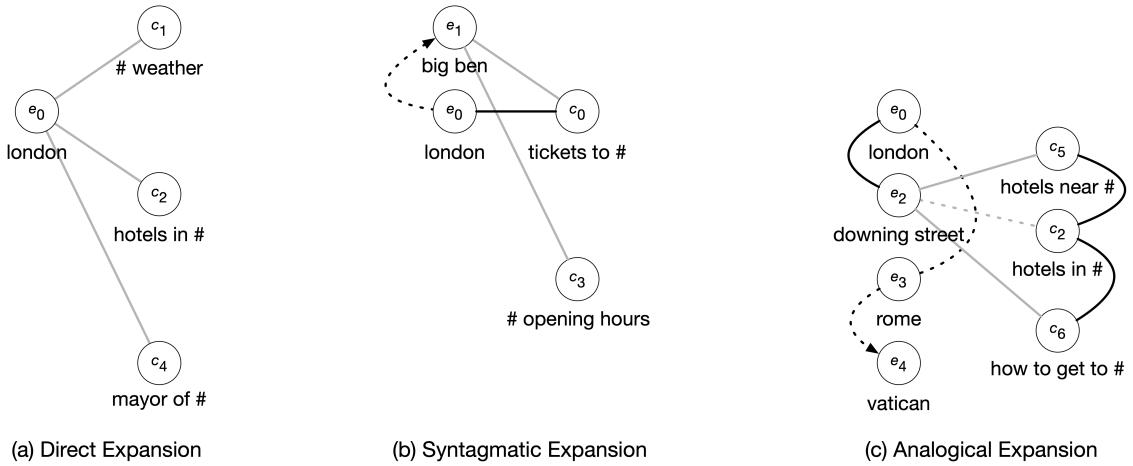


**Figure 3.2.** Different navigation strategies (a-c) over $\mathcal{G}$.

edge recommendation problem, this step can be seen as a ranking the set of edges $\{(e_0, c) \mid c \in (C), (e_0, c) \in \mathcal{L}\}$ by their weight $\ell(e_0, c)$

This basic strategy, called *direct expansion*, is illustrated in Figure 3.2(a). In the example, for the input query "tickets to london", this strategy could return useful queries such as "london weather" and "hotels in london". However, not so useful queries such as "mayor of london" could also be retrieved.

This problem is alleviated by the weighting applied to the edges in $\mathcal{G}$, proportional to the frequencies of $e$, $c$ and $(e, c)$ in the query log (see Equation (3.1)). We hypothesize that, given this lift formulation, informational queries such as "mayor of london" will be demoted in favor of navigational and transactional queries [10], which are not only more frequent, but also more likely to convey subtasks [25, 45].

## 3.3 Syntagmatic Expansion

While the direct expansion strategy is targeted at retrieving alternative contexts related to the input entity $e_0$, it can lead to a lack of diversity on situations where other entities are relevant for some subtask. For instance, a user can be interested in some other points of interest when planning her trip, or other companies when building her investment portfolio.

Due to the focus on the original entity, the direct expansion strategy is not able to retrieve such suggestions. One simple strategy that could solve this problem is to look for an entity $e_i$ that co-occurs frequently with $e_0$ across sessions in the query log, and to recommend queries that combine $e_i$ with its best matching contexts according


**Figure 3.3.** t-SNE projection using (a) Google News and (b) Wiki2Vec.

to Equation (3.1). While this strategy could work on scenarios with popular entities, it could lead to poor suggestions when dealing with rare input entities. Indeed, if $e_0$ is rare, the set of its co-session entities will be small, if not empty.

To overcome this limitation, we propose another strategy for uncovering suggestions associated with entities other than $e_0$. As illustrated in Figure 3.2(b), this strategy, denoted *syntagmatic expansion*, promotes entities $e_i$ (e.g., "big ben") that are topically related to the input entity $e_0$ (e.g., "london"). To attenuate the aforementioned sparsity problem, instead of exploiting topical relationships explicitly stated in the query log (e.g., entities that share many contexts), we resort to matching entities in a semantic space.

In particular, we leverage entity embeddings trained on Wikipedia using Word2Vec [33] [3]. The choice of using Wiki2Vec embeddings, instead of general-purpose embeddings is justified by the specialization of an embedding space for entities. In such space, we can discard words that cannot be matched to entities.

We show, in Figure 3.3 two low-dimension projection of the embeddings, using t-SNE, near the entity "london" using a general purpose pre-trained embeddings (a) and the Wikipedia trained embeddings (b). While both spaces display useful entities, like "Battersea" and "Westminster", there are several low-quality suggestions, like "Pylas_contributed" and "EURASIAN_NATURAL_RESOURCES_CORP". In our initial experiments, we noted that these noisy terms could contribute negatively to the results of our method. As a future work, we also plan on training entities embeddings

[3] Pre-trained embeddings available at `https://github.com/idio/wiki2vec`.

using the query log itself. We believe that this could further improve our results.

Given this embedding space, we them identify the 50 nearest neighbor entities to $e_0$, given by the cosine distance between their embeddings. Candidate suggestions (or edges) are then produced by pairing each neighbor entity $e_i$ with its 50 most salient contexts $c_i \in \mathcal{C}$ as given by Equation (3.1).

In our example in Figure 3.2(b), we could pair "big ben" with salient contexts such as "tickets to #" and "# opening hours." Finally, to rank the set of up to $50 \times 50 = 2{,}500$ suggestions, we score each pair entity-context $(e_i, c_i)$ according to:

$$s(e_i, c_i) = (\vec{e}_0 \bullet \vec{e}_i) + \ell(e_i, c_i), \tag{3.2}$$

where $(\vec{e}_0 \bullet \vec{e}_i)$ denotes the dot product between dense vector representations of $e_0$ and $e_i$ in the Wiki2Vec embedding space and $\ell(e_i, c_i)$ is given by Equation (3.1). This formulation demotes entities that are too far away, as well "bad" matches between entity and contexts. In our experiments, we normalized both the embeddings and the graph weights.

## 3.4 Analogical Expansion

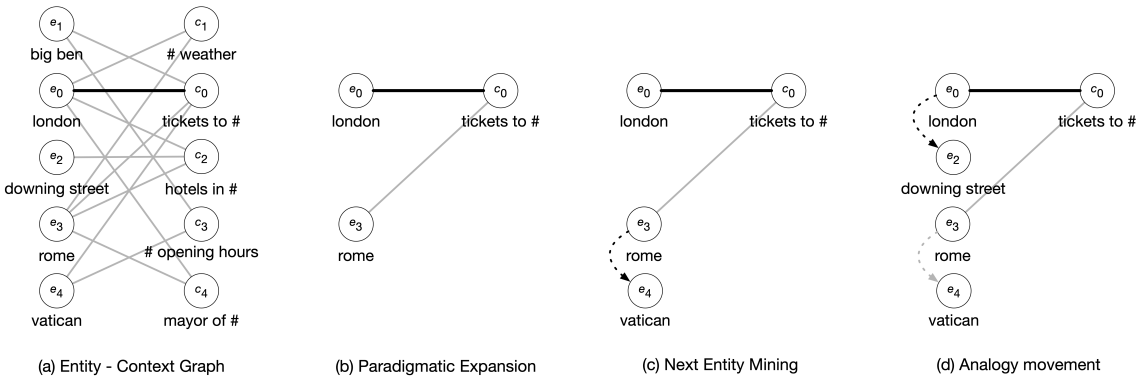


**Figure 3.4.** Analogical expansion strategy.

While direct expansion covers the original entity and syntagmatic expansion covers a set of typically similar entities, we state that some related entities, especially non-trivial ones, are not reached through any of these strategies.

As a further strategy for finding related entities, particularly non-trivial ones, we propose an *analogical expansion* strategy. The intuition behind this strategy is that

topical relationships to $e_0$ can be derived indirectly, by looking at analogous relationships to entities of the same type as $e_0$. For instance, as illustrated in Figure 3.2(c), "downing street" is to "london" as "vatican" is to "rome", which in turn is of the same type as "london." We hypothesize that this indirect relationship will likely uncover unusual (yet related) alternatives to the original entity (like some neighborhood or point of interest mostly known by locals, as in this example), and hence increase the diversity of the suggested queries.

As an added bonus, this strategy is able to retrieve entities that are not present on the original query log, greatly increasing the set of retrievable candidates and diversity.

### 3.4.1 Paradigmatic Expansion

The first step of our proposed analogical expansion is to find a set of similar entities to $e_0$. While in principle this step could be performed using the same pre-trained embeddings used for finding related entities to $e_0$ in Section 3.3, those embeddings modeled syntagmatic (topical) relationships, as seen on Figure 3.3, as opposed to the paradigmatic (or typical) relationships that are needed here [28, 34]. As a simple alternative, we select the top 50 entities $e_s$ that share the same context $c_0$ with the input entity $e_0$ in $\mathcal{G}$, ranked by $\ell(e_s, c_0)$.

In Figure 3.4(b) we see an example of such expansion. In this example, "rome" is an entity that frequently co-occurs with the "tickets to #" context.

### 3.4.2 Next Entity Mining

After retrieving a set of paradigmatically similar entities $e_s$ (e.g., "rome" in Figure 3.4(b)) to $e_0$, we search $\mathcal{G}$ for entities $e_n$ that frequently follow $e_s$ across sessions.[4] The intuition behind this step is that we can further project the movement $e_s \rightarrow e_n$ back to $e_0$. In our example, in Figure 3.4(c), $e_n$ is "vatican."

For instance, if someone searches for a museum after searching for a city similar to the one the user is looking for, we want to move in the same general direction in the embedding space of entities, so we can find another museum (or, more generally, any other entity) related to $e_0$. In order to decide which movements $e_s \rightarrow e_n$ are worth projecting back to $e_0$ we score each movement according to:

$$s(e_s, e_n) = \ell(e_s, c_0) + \ell(e_s, e_n), \tag{3.3}$$

[4]Note that we do not require that $e_n$ immediately follow $e_s$ in a session.

where both $\ell(e_s, c_0)$ and $\ell(e_s, e_n)$ are given by Equation (3.1), except that the latter is overloaded to weight the co-occurrence of two entities within sessions rather than co-occurrences of an entity and a context within queries. In total, we keep the top 5 entities $e_n$ that follow each entity $e_s$. In our example, some retrieved movements are the following:

"new york" $\rightarrow$ "brooklyn"
"rome" $\rightarrow$ "vatican"
"paris" $\rightarrow$ "louvre"

### 3.4.3 Analogies

Finally, to project the identified movements back to the original entity $e_0$, we perform the same embedding analogy operation proposed by Mikolov et al. [33], illustrated on Figure3.4(d). Inspired by their analogy example, where $\vec{e}_{king} - \vec{e}_{man} + \vec{e}_{woman} \approx \vec{e}_{queen}$, we want to map the movements discovered in the previous steps back to the original entity $e_0$. Essentially, following our example on Figure 3.4, we want to ask *What is the "vatican" of "london"*? By performing these movements, we believe that rare entities can be discovered. The analogical expansion can be seen as a high-risk, high-reward step, promoting serendipity and discovery of rare entities.

Given embeddings of the same shape for the original entity ($e_0$), typically similar entity ($e_s$), and next entity ($e_n$), discovered using the strategies described above, we resort to Equation (3.4) below to identify a set of recommended analogy entities $e_i$:

$$e_i = \sigma(e_0, e_s, e_n) = \vec{e}_0 - \vec{e}_s + \vec{e}_n, \tag{3.4}$$

where $\vec{e}_\bullet$ denotes the dense vector representation of entity $e_\bullet$ in the embedded space. Following our example, we end up with these suggested entities, illustrated on Figure 3.5:

$\sigma$("london", "new york", "brooklyn") = "battersea"
$\sigma$("london", "rome", "vatican") = "downing street"
$\sigma$("london", "paris", "louvre") = "sloane square"

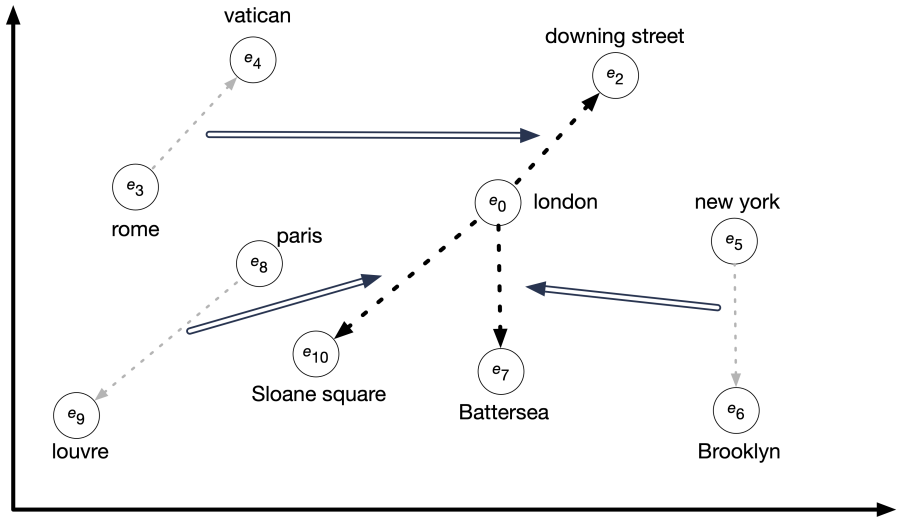


**Figure 3.5.** Analogical movements.

### 3.4.4 Context Generation

The analogical movement is agnostic when dealing with contexts. Meaning, any technique for generating contexts can be coupled with the suggested entities to produce a tuple $(e, c)$.

While a naive strategy like simply choosing the context $c_i$ that most frequently co-occur with the suggested entity $e_i = \sigma(e_0, e_s, e_n)$ could work, we argue that, different from our syntagmatic expansion strategy in Section 3.3, our analogical expansion strategy tends to select entities that are much farther away from the input entity $e_0$. As a result, this naive approach may cause an undesired topic drift. For instance, our suggested entity "battersea" has a highly salient edge with the context "# power station". While they are, indeed, very related, a suggestion like this could cause a severe deviation from the original task.

An alternative could be to look for contexts $c_i$ that co-occur with the input entity $e_0$, which in turn could lead to poor diversity, as well as some edges that are not related at all. For instance, this could yield results like "mayor of sloane square", which, besides showing a major deviation from $\mathcal{T}$, is also a nonsensical suggestion.

As a compromise, we train a state-of-the-art neural query suggestion approach [44] on the underlying query log to produce session-sensitive contexts $c_i$ given the input query $q$, to be matched with the identified entities $e_i$.

We feed the network with the input query $q_0$, sampling, one character at a time, a set of complete query suggestions. We then send those suggestions to the same entity

extractor described in 3.1, collecting a set of contexts.

We hypothesize that this approach yields contexts that are highly related to the original task $\mathcal{T}$, and that, coupled with the suggested entities, we are able to generate complete queries that cover rare subtasks.

### 3.4.5 Entity-Context Matching

Because entities and contexts are generated independently in the analogical expansion strategy, we risk recommending semantically mistaken tuples like ("big ben", "mayor of #"). To make sure only meaningful tuples are recommended, we further score each tuple $(e_i, c_i)$ according to:

$$s(e_i, c_i) = \ell(e_s, e_n) + \frac{1}{|\mathcal{C}_{e_i}|} \sum_{c_j \in \mathcal{C}_{e_i}} (1 - d(\vec{c_i}, \vec{c_j})), \tag{3.5}$$

where $\ell(e_s, e_n)$ denotes the cross-session compatibility (see Equation (3.3)) of the movement $e_s \rightarrow e_n$ that led to the analogous $e_0 \rightarrow e_i$, $d(\vec{c_i}, \vec{c_j})$ is the word mover's distance [27] between the word embeddings contained in $c_i$ and $c_j$, and $\mathcal{C}_{e_i}$ is the set of all contexts linked to $e_i$ in $\mathcal{G}$.

The word movers distance, proposed by Kusner et al. [27] is a distance metric between phrases. Distinct from the traditional approach of taking the average of the embeddings on each phrase and calculating the cosine distance, the WMD is shown to be more sensitive to semantical similarities, as well as indifferent to the length of each phrase.

The intuition behind this step is that we only match entities with contexts that are somewhat similar to contexts that are already present with that entity. In the case of entities that are not present in the query log, we take the risk of suggesting some nonsensical queries, over the possibility of retrieving previously unknown and rare candidates, that could be highly relevant.

## 3.5 Summary

This chapter introduced our three proposed strategies for task understanding. In Section 3.1 we showed how to build our basic data structure, the graph $\mathcal{G}$, that is used as a base for every one of our approaches, by modelling a query log as a bipartite graph of contexts and entities, where a query is represented as an edge on this graph. In Section 3.2 we showed our first approach to the task understanding problem, the *direct expansion* strategy. By focusing on the original entity, this strategy looks for other

actions that could be performed over the same entity, by mining contexts on the graph $\mathcal{G}$. Section 3.3 described our second strategy, the *syntagmatic expansion*. We leverage an embedding space built over entities to search for topically similar entities, increasing our coverage of subtasks. Finally, in Section 3.4 we describe the *analogical expansion*, an arguably riskier approach, but one that can bring highly valuable entities, by using analogies over entity embeddings that might not be retrieved by more conservative strategies.

In the next chapter, we discuss our experimental setup, describing our dataset, evaluation methodology and baselines. We also show the results of these strategies, discussing how to fuse their rankings to produce state-of-the-art query suggestions for task understanding, as well as a breakdown analysis on how our methods fare on multiple scenarios.

# Chapter 4

# Experimental Evaluation

In this chapter, we describe the general setup for evaluation of our proposed strategies for task understanding discussed on Chapter 3. In Section 4.1 we describe our test collection, evaluation methodology and discuss our choice of baselines. Section 4.2 covers the evaluation results of our methods. In particular, we outline strategies for taking advantage of the complementarity of our methods, show that our combined strategies outperform current state-of-the-art approaches and show how they fare in multiple scenarios of task complexity and query rarity, demonstrating that our approaches can outperform current state-of-the art approaches, when properly combined.

## 4.1 Experimental Setup

In this section, we outline the experimental setup that supports our experiments in order to assess the effectiveness of our three proposed strategies for task understanding. In particular, we aim to answer the following research questions:

Q1. How effective are our proposed strategies?

Q2. How complementary are our proposed strategies?

Q3. How do our strategies perform for different types of query?

In the remainder of this section, we describe the setup that supports these investigations. In Section 4.1.1 we describe our query log dataset and the queries used as a test set for our methods. Section 4.1.2 describes how we evaluate our results. In particular, we discuss how the reuse of "golden" results can be problematic in a scenario like ours, and how to overcome these limitations. Finally, in Section 4.1.3 we discuss the baselines chosen for our evaluation.

| # of queries | # of unique queries | # of unique entities | # of unique contexts |
|---|---|---|---|
| 36389567 | 10154742 | 512926 | 3447330 |

**Table 4.1.** AOL query log numbers.

### 4.1.1 Test Collections

As discussed before, our methods rely heavily on a bipartite graph $\mathcal{G}$ built on top of a query log. In this work, we mine candidate suggestions on the graph $\mathcal{G}$ built over the AOL 2006 query log. This log, as shown in Table 4.1.1 is a large dataset, with over 35 million queries, 10 million of these unique, collected over a period of a month.

After extracting the entities form this log, using the method described in Section 3.1, we end up with about 512 thousand unique entities and 3 million contexts, constituting a rather large graph $\mathcal{G}$.

Following [1], we plot the query, entity and context frequency distributions in Figure 4.1. As expected, these follow a clear power law distribution, with a few highly popular queries, entities and contexts, and a long tail of low-frequency elements.

Instead of extracting a set of user sessions from the same query log, we consider the 50 queries provided by the TREC 2016 Tasks track [45].[1] This set of queries are representative of multiple types of tasks that users can be trying to achieve, with varying degrees of complexity.

Most queries include an explicit reference to the at least one entity $e_0$ that is the target of the underlying task. For the few cases of queries without explicitly mentioned entities, we extract entities using the same process described above. For a few other cases, multiple entities may occur. In these scenarios, we also process these with FEL [6] and only consider the entity with higher probability. Figure 4.2(a) exemplifies one input query from this dataset. There we can see that the input query "cure indigestion" is paired with the entity "indigestion". It also illustrates the fact that the system does not have access to any subtask (or the main task) beforehand. Figure 4.2(b) also exemplifies the underlying task and subtasks that should be addressed, defined by human judges after the suggested queries were submitted. In this example, we see that the original query represents the task "You wish to find remedies for curing indigestion", and that it contains 5 distinct subtasks.

For each query, the TREC 2016 Tasks track provides relevance assessments for suggested key phrases on a scale from 0 (non-relevant) to 2 (highly relevant) for each subtask and an extra "others" subtask. The number of defined subtasks ranges from

[1] The ground truth produced by TREC 2017 Tasks track are of lower quality when compared with 2016 [25].

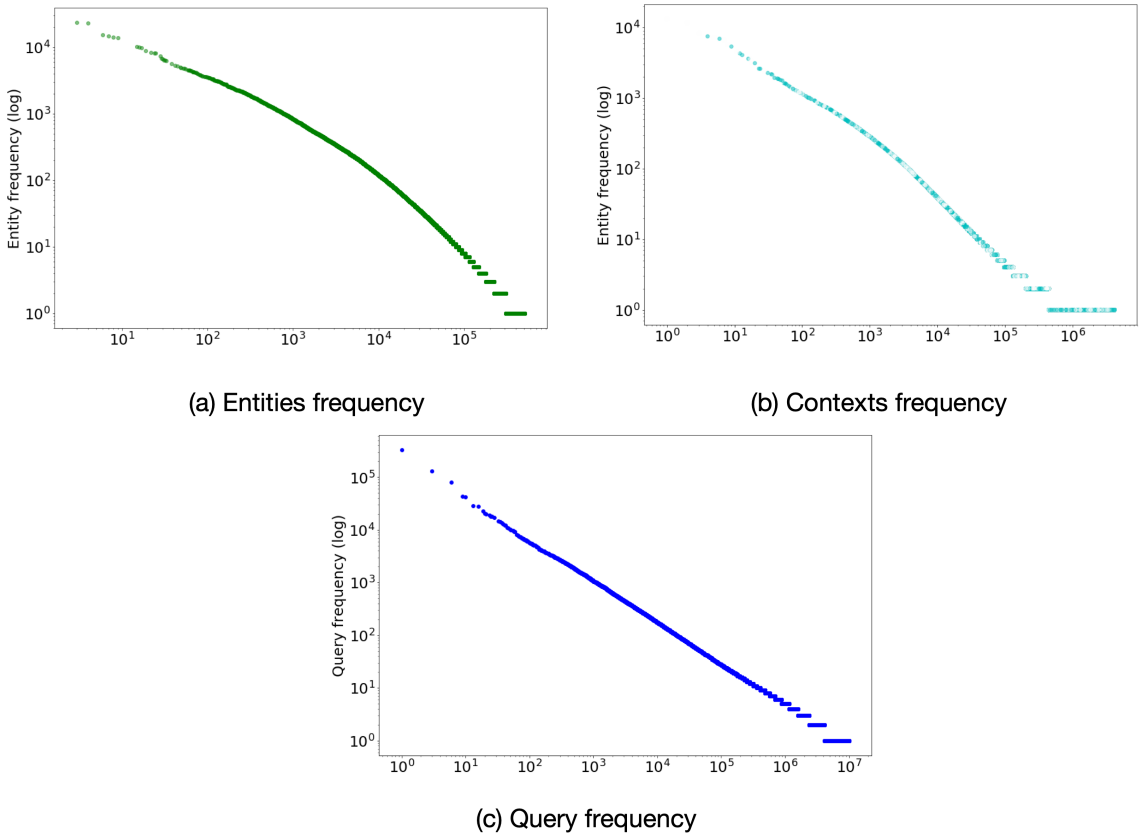


**Figure 4.1.** Frequency distribution of entities (a), contexts (b) and queries (c) in the query log.

| Strategy | Limit | Value |
|---|---|---|
| Direct Expansion | Retrieved contexts | 100 |
| Syntagmatic Expansion | Retrieved entities | 50 |
| | Retrieved contexts per entity | 50 |
| Analogical Expansion | Similar entities retrieved | 50 |
| | Next entity per similar entity | 5 |
| | Maximum candidate contexts | 100 |

**Table 4.2.** Size limits for our methods.

4 to 14 across the 50 queries. On average, there are 80 unique key phrases that are relevant to at least one subtask per task, as well as 17 highly relevant key phrases.

Given time and computational constraints, we limited our methods in the number of possible candidates retrieved in each step. Our choices are described in Table 4.1.1.

```
<task id = "7">
    <query>cure indigestion</query>
    <freebase entity = "1">
        <id>/m/04kl78</id>
        <text>Indigestion</text>
    </freebase>
</task>
```

(a) TREC task input with entity

```
Task id: 7
cure indigestion
[You wish to find remedies for curing indigestion.]
*   Home remedy for indigestion
*   Food to prevent indigestion
*   Medicines that cure indigestion
*   Doctor to consult for chronic indigestion
*   What causes indigestion?
```

(b) TREC task description with subtasks

**Figure 4.2.** TREC 2016 Tasks track query #7 (a) and target task description (b).

### 4.1.2 Evaluation Methodology

The ground truth provided by the TREC 2016 Tasks track comprises a relatively small, non-exhaustive set of judged key phrases, which limits the reusability of this test collection for evaluating new task understanding systems [14]. Meaning that, if any of our methods (or any baseline) generate a suggestion that was not also generated by any of the original competitors, a naive approach would always consider these as not relevant, regardless of their actual usefulness for the task.

To overcome this limitation, we propose a parametrized relaxation scheme to match suggested key phrases to those in the ground-truth. Our evaluation methodology consists in relaxing these matches, by varying threshold limits between 0 (the suggestion is identical to the ground-truth) and 1 (the suggestion is completely different from the ground-truth), increasing in steps of 0.1. By doing so, we create a curve of similarity, like the ones shown in Figure 4.3, where the x axis is the maximum distance allowed before considering a suggestion the same as one in the ground truth and the y axis the measured metric on that threshold.

This evaluation procedure, while not perfect, can capture how our methods perform under multiple scenarios of relaxation. At the same time, this relaxation can cause some issues. Since it is not supervised, some queries that are deemed similar could have completely different meaning, rendering the results problematic. To mitigate this, we also submit the results to a crowdsourcing system.

In order to summarize these curves in one, comparable number, we define a summarization function with exponential decay over the threshold:

$$\sigma_\mu = \sum_{\theta=0.0}^{1.0} \mu_\theta \times \exp(1-\theta), \tag{4.1}$$

where $\sigma_\mu$ is the summarized score for a given metric $\mu$, $\theta$ is the dissimilarity threshold, and $\mu_\theta$ is the measured value of the metric $\mu$ given the threshold $\theta$. Following the standard practice at the TREC 2016 Tasks track [45], we use ERR-IA@20 as our primary evaluation metric, while also reporting the $\alpha$-nDCG@20. As for the underlying similarity metric, we consider two distinct definitions of distance:

- **Syntactic Distance** Given by the normalized Levenshtein Distance [35], we measure the edit distance between a suggestion and a ground-truth key phrase.
- **Semantic Distance** Given by the complement of the word mover's distance [27] between a suggestion and a ground-truth key phrase, with pre-trained word embeddings.[2]

The second approach is of specific interest for us. It allows us to abstract differences between phrases with similar meaning, despite a different wording. We hypothesize that this is metric is more closely related to what a user would be looking for, abstracting queries that are dissimilar syntactically, but semantically similar. (i.e.: "subway tickets" vs "metro tickets").

To make sure our reported findings are not a mere artifact of these relaxed evaluation schemes, we conducted a further evaluation round with human judges via crowdsourcing. To keep our costs manageable, we restrict this additional evaluation to our best method as well as the baselines described in Section 4.1.3.

### 4.1.3 Baselines

As baselines in our investigations, we consider two query suggestion approaches, considered current state-of-the-art, as representatives of two different strategies, one focused on classical IR techniques and another on newer, Deep Learning based: Search Shortcuts (SS) [8] and Hierarchical Recurrent Encoder-Decoders for Query Suggestions (HRED) [44]. SS was shown to perform on par with state-of-the-art session-based query suggestion approaches for head queries, with substantial improvements for tail queries. For SS, we indexed the AOL query log into virtual documents using Terrier [30], with BM25 used for retrieval. HRED deploys coupled recurrent neural network architectures for modelling both word transitions within queries as well as query transitions within sessions. We used the implementation provided by the authors and trained it on the AOL query log using the same training-test partitioning described in the original paper [44].

[2] https://code.google.com/archive/p/word2vec/

## 4.2 Experimental Results

In this section, we assess the complementarity of our proposed strategies for task understanding and their effectiveness in light of the state-of-the-art. We structure this section in response to our research questions, posed in Section 4.1. We also discuss different methods for fusing the results of our methods and show their respective effectiveness. Finally, we also compare our method against Search Shortcuts (SS) [8] and Hierarchical Recurrent Encoder-Decoders for Query Suggestions (HRED) [44].

As stated in section 4.1, our main evaluation methodology revolves around managing a threshold for how different our suggestion is, related to a ground-truth on a limited set of golden results. In the end, we also submitted our best method, together with the two baselines, to a crowdsourcing platform, again using the same evaluation methodology proposed by TREC.

### 4.2.1 Strategies Effectiveness

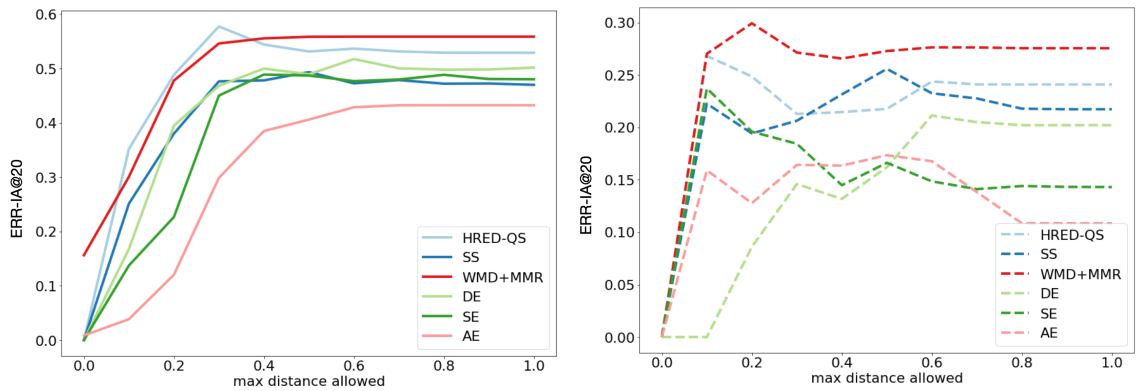


**Figure 4.3.** ERR-IA@20 for various dissimilarity thresholds.

We begin our analysis by addressing our first research question. In this section we explore, individually, how each of our methods for task understanding, namely direct expansion (DE), syntagmatic expansion (SE) and analogical expansion (AE) fare, in comparison to other approaches.

We also investigate how their combination compares to our two state-of-the-art query suggestion baselines from the literature: SS [8] and HRED [44]. To combine our strategies, we experimented with a simple approach based on negative word mover's distance (WMD) [27] to rescore the union of the rankings produced by the three individ-

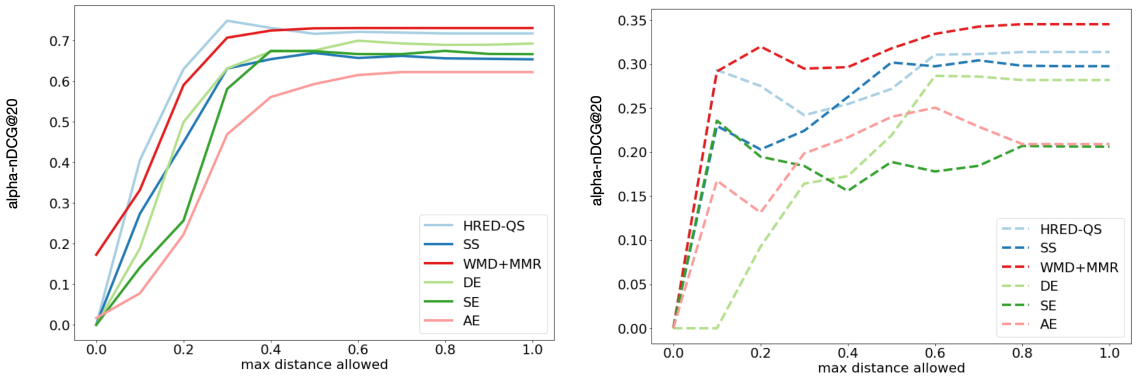


**Figure 4.4.** $\alpha$-nDCG@20 for various dissimilarity thresholds.

ual strategies with respect to the input query, further reranked via maximal marginal relevance (MMR) [13] to penalize redundant suggestions in the ranking. The function for scoring the similarity between suggestions is also given by the negative WMD. In Section 4.2.2 we better describe two other possible approaches.

Figure 4.3 shows the result of this investigation in terms of ERR-IA@20 for semantic and syntactical similarities, as we vary the maximum dissimilarity threshold on the x-axis. Figure 4.4 shows the same results but measuring $\alpha$-nDCG@20.

From Figures 4.3(a) and 4.4(a), we first note that, when considering the semantic distance, among our three individual strategies, DE performs the best, followed by SE and AE, particularly for stricter scenarios (i.e., toward lower dissimilarity). This result is somewhat expected, given that DE is the most conservative of the three strategies, whereas AE is the most aggressive.

However, when considering the Syntactic distance, in Figures 4.3(a) and 4.4(a), SE performs better on stricter scenarios, followed by AE. When increasing the maximum allowed similarity, however, the situation is very similar to what happens on Semantic Distance.

The explanation for this perhaps counter-intuitive result is due to the fact that, usually, the context is the largest portion of the complete query, and, since the normalized Levenshtein distance takes into consideration the complete length of the query, it will tend to give higher similarity scores to queries with very similar contexts. Therefore, DE, by changing the whole context, will change most of the original query, thus, a higher divergence from the ground-truth queries when compared to SE, that, given that it will look for entities similar to $e_0$, the contexts are more likely to be similar to

$c_0$ [3], and AE, that, given the use of a version of HRED-QS for contexts, will keep the context in a not too distant scenario from $c_0$.

It is also worth noting the magnitude of the scores. Semantic distance, as shown in Figure 4.3(a) ranges from 0.0 up to 0.6, while syntactic distance, in Figure 4.3(b), ranges from 0.0 to only 0.3. This also shows that our proposed semantic distance is perhaps more permissive when considering similarities. A very similar phenomenon can be seen in Figure 4.4.

Figures 4.3 and 4.4 also show that, individually, our strategies do not outperform the baselines. However, when combined via WMD+MMR, they perform on par with the strongest baseline up until a dissimilarity threshold around $\theta = 0.3$, when considering semantic distances, and $\theta = 0.1$ when considering syntactic distance, with consistent improvements afterwards.

Recalling Q1, these results demonstrate the effectiveness of our strategies on their own, and particularly in combination, with improvements over state-of-the-art query suggestion baselines from the literature. These results also point towards the complementary nature of our proposed strategies, which will be explored in the next section.

### 4.2.2 Strategies Complementarity

As discussed on Chapter 3, a user task can cover multiple subtasks, spanning over multiple entities and contexts. Therefore, our proposed strategies for task understanding aim to promote a diverse coverage of these possible subtasks underlying the user's task, represented by a single input query.

To address question Q2, we investigate the extent to which the suggestions produced by these strategies are actually complementary to one another.

To this end, Figure 4.5 shows Venn diagrams illustrating the number of suggestions that are unique to each strategy, as well as the size of their intersections. The reported numbers are aggregated for all 50 queries,[4] with each strategy contributing up to 20 suggestions per query. It is interesting to note that the number of relevant suggestions generated by AE is considerably large, compared to the relatively low performance reported in Figure 4.3. The reasoning behind this is that the queries that are considered equal for the given threshold, on AE, are usually queries with lower scores overall. Therefore, despite a large number of relevant suggestions, these are ranked lower on the ground truth, yielding a lower score.

[3] On a deeper analysis of the competitors from TREC, their results tend to focus on the original

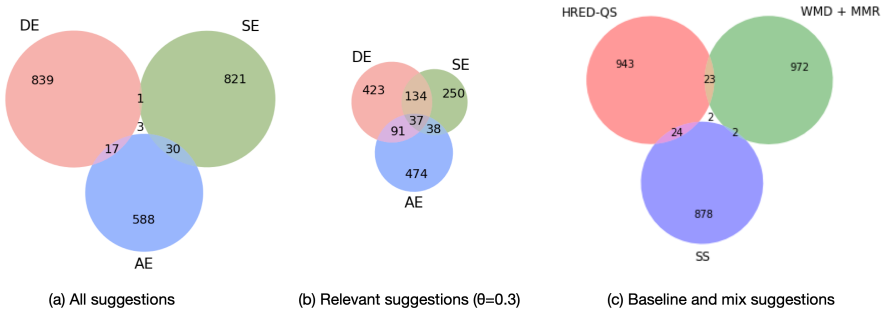


**Figure 4.5.** Venn diagram of suggestions returned by our three proposed strategies.

From Figure 4.5(a), we observe a very small intersection of the output produced by our three proposed strategies, with a substantial number of unique suggestions contributed by each one of them. When only relevant suggestions are considered (using a semantic dissimilarity threshold $\theta = 0.3$ between the suggestions and the golden results) in Figure 4.5(b), we note larger intersections, particularly among direct expansions (DE) and syntagmatic expansions (SE), with analogical expansions (AE) providing the largest number of unique relevant suggestions, giving its wildly different strategy.

Finally, on Figure 4.5(c), we show that our final mixing method also generates disjoint suggestions when compared to the baselines. This also raises the question on what would be the results if we also mixed the baselines (especially HRED-QS) with our methods. While we leave this as a possible future work, it could be a somewhat complex task to achieve, given that our methods are unsupervised, the addition of HRED-QS may make us lose the benefit of the speed of our method, given the long training time needed on HRED-QS.

These results answer Q2, by demonstrating that each strategy is able to produce a sizeable amount of relevant suggestions which complement the suggestions produced by the other two strategies.

We also explore how different data fusion techniques are able to combine our methods. in particular, we explored how BordaCount, CombMNZ [18] and ranking the candidates by WMD can be used. Finally, we also explored how MMR re-ranking can be beneficial for each of the techniques.

BordaCount is a classical counting technique for voting, where each voter (in this case, a method that generates candidates), ranks its preferences (or suggestions). Each

context or small variations of it.

[4]Per query intersections are near empty.

document is scored according to the inverse of its ranking position, and these scores are summed. The score for a single suggestion $q$, over $n$ rankings with $k$ results for each is described on Equation 4.2:

$$BordaCount_q = \sum_{i=0}^{n} \sum_{j=0}^{k} \begin{cases} k-j, & \text{if } q \in \mathcal{I} \\ 0, & \text{otherwise} \end{cases} \tag{4.2}$$

where $\mathcal{I}$ is the set of suggestions ranked in the i-th ranking.

CombMNZ is another strategy for combining multiple rankings, proposed initially by Fox et al. [18], where the final score of each document is given by the sum of the scores in each of the rankings, multiplied by the number of times the result appear on each ranking, as shown in Equation 4.3:

$$CombMNZ_q = m \times \sum_{i=0}^{n} \sum_{q \in \mathcal{I}} \mu_{i,q}, , \tag{4.3}$$

where $m$ is the number of methods that generated the query $q$ and $\mu_{i,q}$ is the score of the query $q$ in the i-th method.

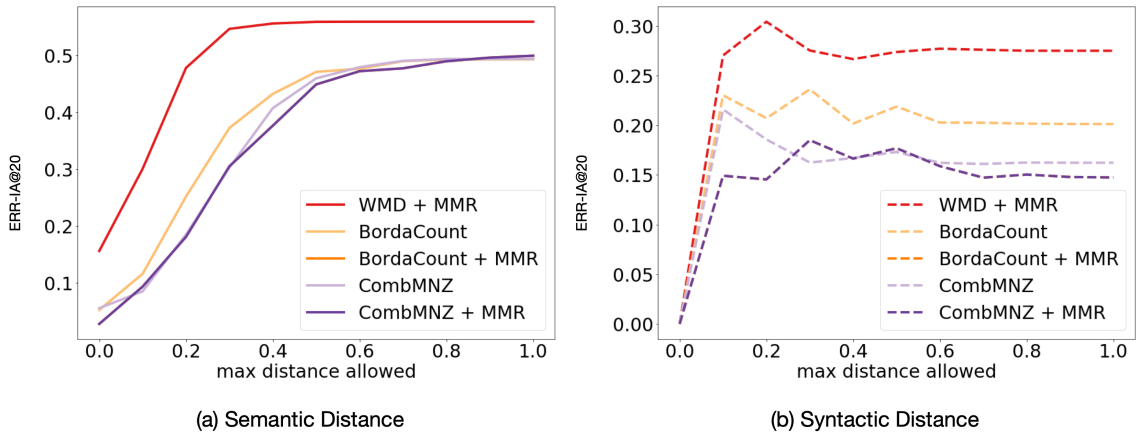


**Figure 4.6.** ERR-IA@20 curves for Data Fusion.

The curves of these results can be seen in Figure 4.6 and 4.7, where our WMD+MMR method clearly outperformed other data fusion techniques, on all breakdowns. It is also interesting to note that MMR on both CombMNZ and BordaCount re-ranked the documents in exactly the same way. In Figures 4.12 and 4.13 we see these methods compared by each of the breakdowns that are further evaluated, where, again,

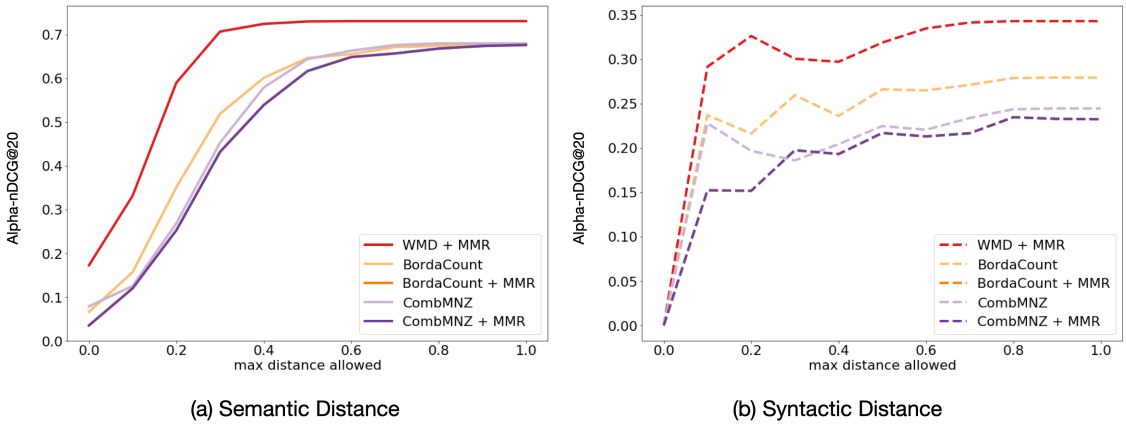


**Figure 4.7.** $\alpha$-nDCG@20 for Data Fusion.

it is clear that our mixing method outperforms other ranking fusion methodologies by a good margin.

### 4.2.3 Effectiveness Breakdown

To further assess our proposed strategies, we address Q3, *"How do our strategies perform for different types of query?"*, by breaking down our evaluation for different groups of queries, according to the following criteria:

- **Occurrences in the query log**: we divided the test questions into head queries (10+ occurrences in the query log), torso queries (1-10 occurrences) and tail queries (0 occurrences)
- **Number of entities in the ground-truth results**: to measure the retrieval quality of entities, we also divided the query log by the number of relevant unique entities in the ground-truth.
- **Task complexity**: we believe that the complexity of the task can be measured by how many subtasks it encompasses. Therefore, we also divided the test queries by number of target subtasks.

These breakdowns describe key areas where task understanding systems play the most important roles. Rare queries (or long-tail) are always considered a challenge for information retrieval systems. Given their lack of information on the query log, being able to efficiently retrieve suggestions in these cases is a complex task on itself, addressed in works like Broccolo et al. [8]. We consider head queries as queries with at

least 10 mentions in the query log, long-tail queries that are not present in the query log and torso queries any query with at least one mention in the query log, but no more than 9.

Given our focus on entities, and the fact that the TREC dataset explicitly includes the entities from the original query in their dataset (see Figure 4.2(a)), we measure the number of entities in the reported ground truth. Our intuition is that this would measure how diverse the expected results for a given task is.

Finally, we state that a given task $\mathcal{T}$ is more complex according to the number of subtasks it encompasses. We believe that this breakdown will be able to distinguish between easy tasks (few subtasks needed to complete $\mathcal{T}$) and hard tasks (lots of subtasks needed), making this breakdown extremely important for task understanding.

Figures 4.8 and 4.9 shows the results of this breakdown evaluation in terms of the relaxed ERR-IA@20 (see Equation (4.1)). From the figure, we further confirm the consistent superiority of DE compared to SE and AE for queries in all groups, when considering semantic distance, and the opposite when considering syntactic distance.

Moreover, the combination of the three strategies into the WMD+MMR rescoring approach yield consistently improved results compared to both SS and HRED in most scenarios. Interestingly, our combined approach performs the best for hard queries (tail queries, and those with a large variety of underlying entities and subtasks). Two notable exceptions are for head queries (5/50 queries with 10+ occurrences in the log) and for queries with 17-31 entities in the log (17/50 queries), where HRED performs the best.

A similar behaviour can be seen when measuring their $\alpha$-nDCG. For the sake of completeness, we also include these analyses in Figures 4.10 and 4.11

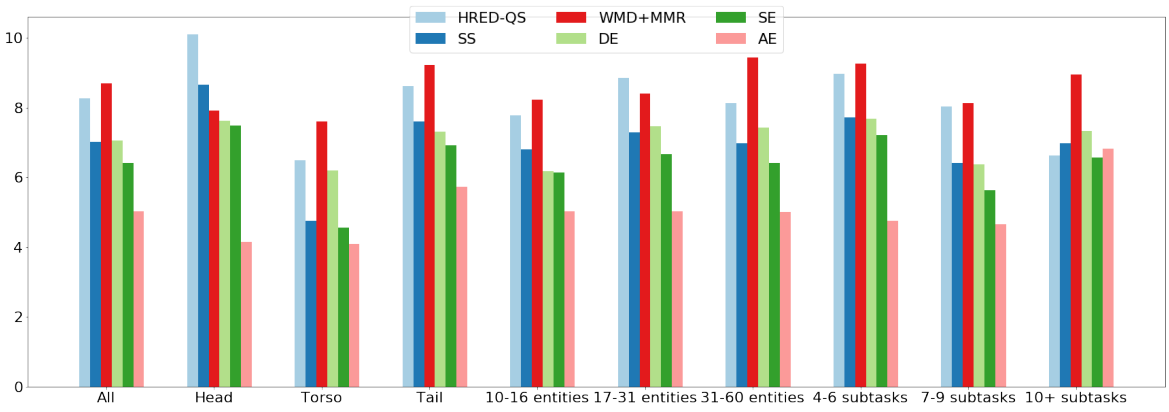


**Figure 4.8.** Relaxed Semantic ERR-IA@20 for various query groups.

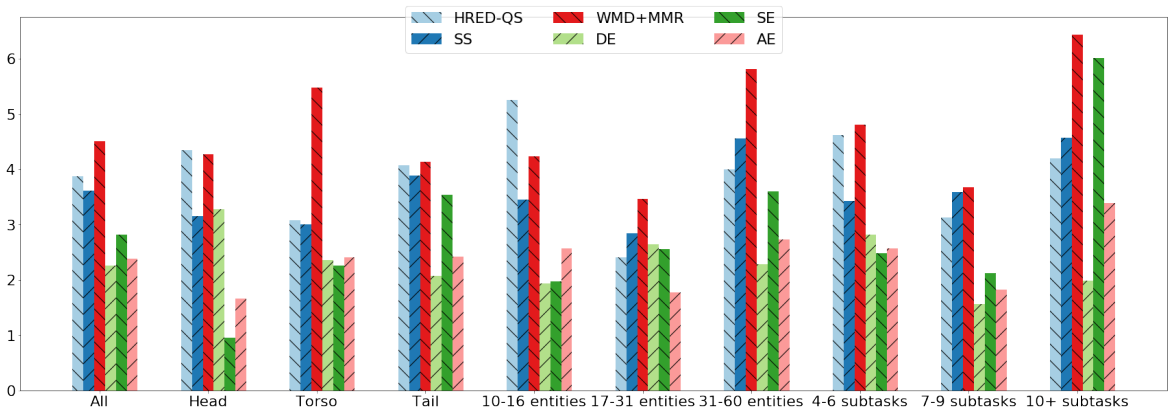


**Figure 4.9.** Relaxed Syntactic ERR-IA@20 for various query groups.

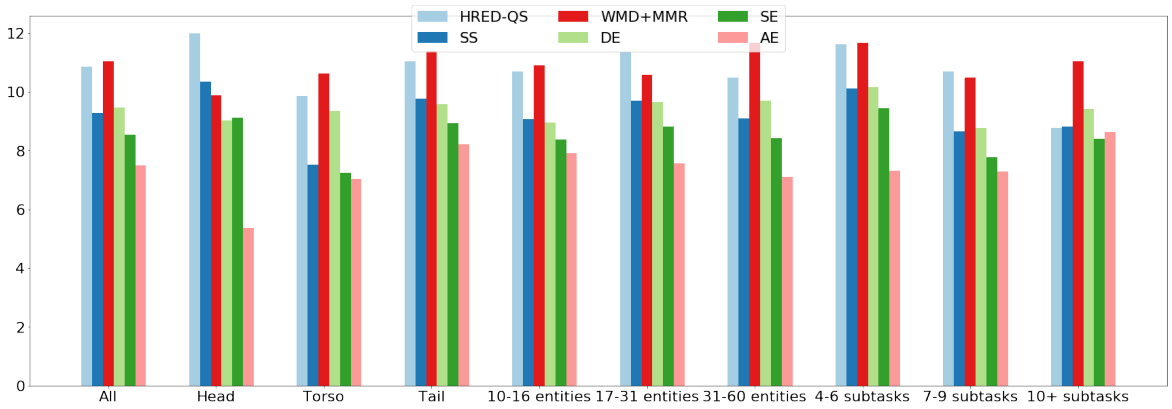


**Figure 4.10.** Relaxed Semantic $\alpha$-nDCG@20 for various query groups.

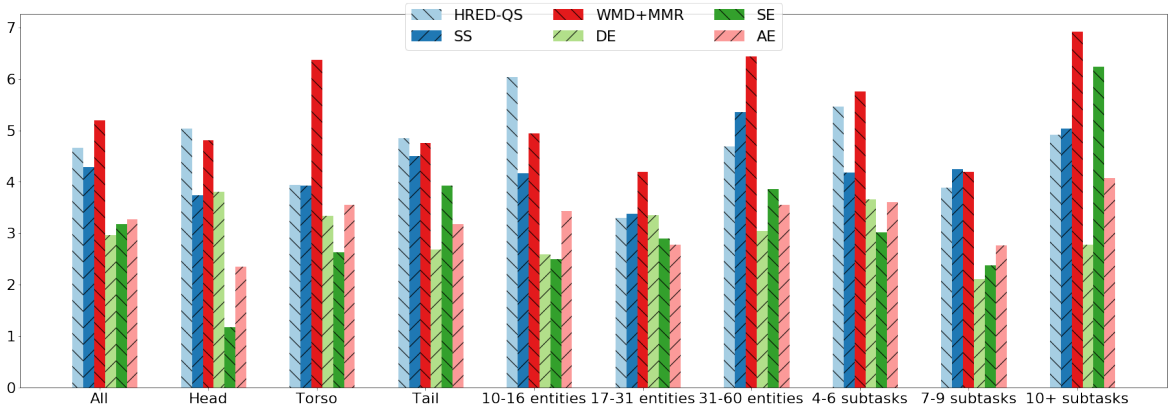


**Figure 4.11.** Relaxed Syntactic $\alpha$-nDCG@20 for various query groups.

Following on our investigation from Section 4.2.2, we also show the same breakdowns for our mixing strategies in Figures 4.12 to 4.15.

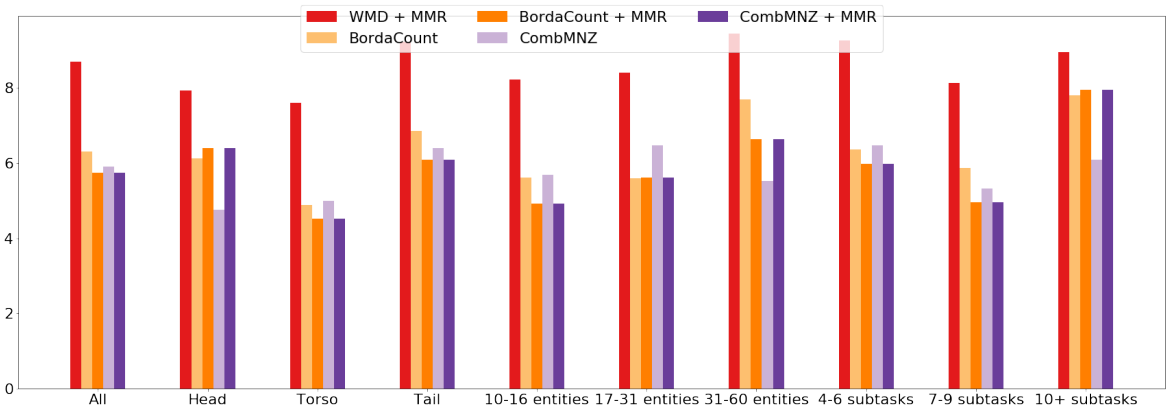


**Figure 4.12.** Relaxed Semantic ERR-IA@20 for various query groups.

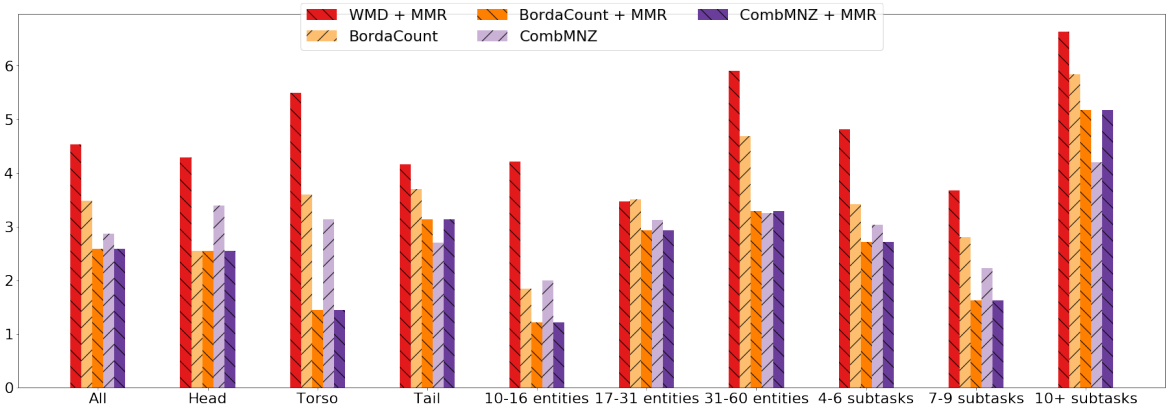


**Figure 4.13.** Relaxed Syntactic ERR-IA@20 for various query groups.

In Tables 4.3, 4.4 and 4.5 we present our numerical results. Specifically, we show the values for our summarization metric (see Equation 4.1), measuring the results by ERR-IA@20. ($\alpha$-nDCG@20 values and significances are very similar and will not be presented for the sake of brevity.)

These results show that our mixing method is robust across every subdivision of the test queries. Overall, our method outperforms the baselines in almost every scenario, especially for rare queries and in complex scenarios, with multiple relevant entities and multiple subtasks.

Our significance analysis was done with the null-hypothesis ($H_0$) stated as: *Our method is similar to our strongest baseline, HRED-QS.* We report values with $p < 0.05$ marking them with $\triangle$ and $\triangledown$, and $p < 0.01$ using ▲ and ▼.

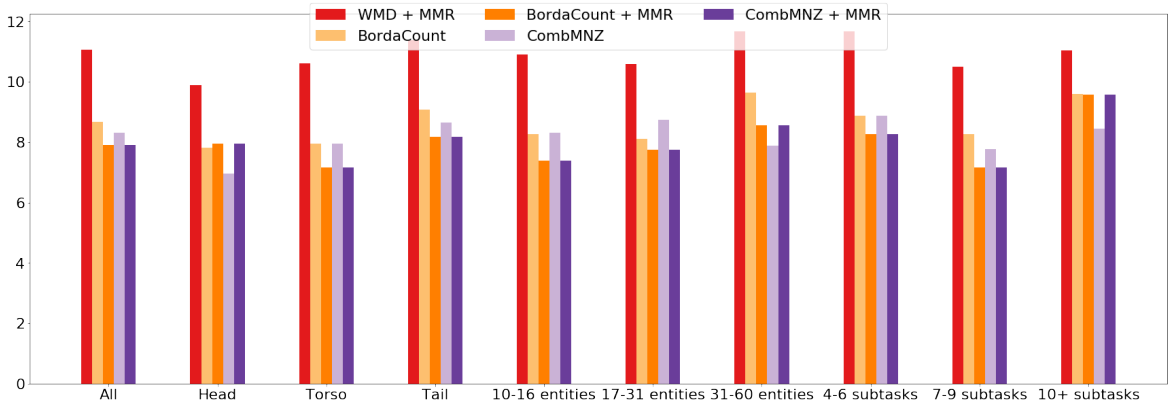


**Figure 4.14.** Relaxed Semantic $\alpha$-nDCG@20 for various query groups.

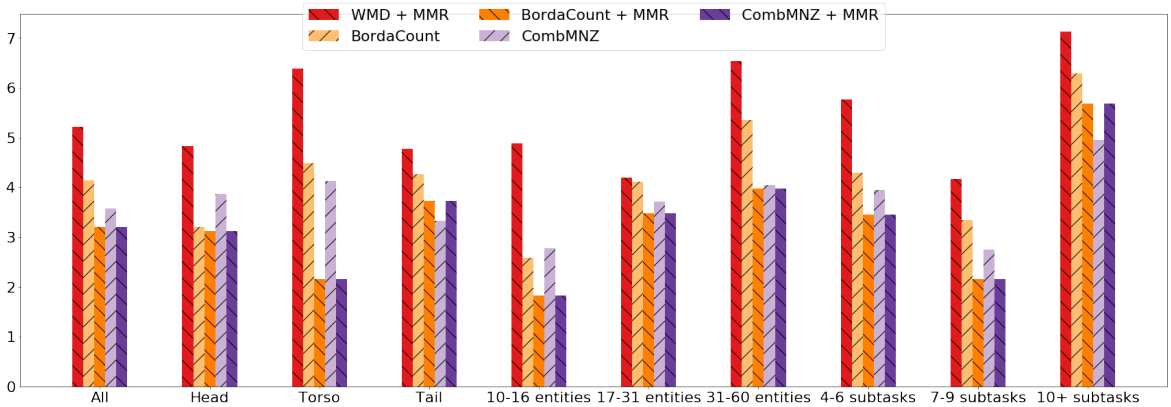


**Figure 4.15.** Relaxed Syntactic $\alpha$-nDCG@20 for various query groups.

| | All ($n = 50$) | | Head ($n = 5$) | | Torso ($n = 12$) | | Tail ($n = 33$) | |
|---|---|---|---|---|---|---|---|---|
| | Syn | Sem | Syn | Sem | Syn | Sem | Syn | Sem |
| HRED-QS | 3.84 | 8.26 | **4.39** | **10.10** | 2.90 | 6.50 | 4.07 | 8.62 |
| SS | 3.57 | 7.03▼ | 3.15 | 8.65 | 3.04 | 4.77▽ | 3.81 | 7.60▼ |
| DE | 2.16▼ | 7.06▼ | 3.28 | 7.62 | 2.24 | 6.21 | 1.97▼ | 7.32▼ |
| SE | 2.80▼ | 6.41▼ | 0.96 | 7.49 | 2.22 | 4.57▽ | 3.52 | 6.92▼ |
| AE | 2.40▼ | 5.02▼ | 1.70 | 4.16 | 2.48 | 4.09▽ | 2.39▼ | 5.74▼ |
| WMD+MMR | **4.53** △ | **8.71** | 4.29 | 7.92 | **5.50**△ | **7.61** | **4.16** | **9.22** |

**Table 4.3.** Results by query frequency.

| | All ($n = 50$) | | 10-16 ($n = 16$) | | 17-31 ($n = 17$) | | 31-60 ($n = 17$) | |
|---|---|---|---|---|---|---|---|---|
| | Syn | Sem | Syn | Sem | Syn | Sem | Syn | Sem |
| HRED-QS | 3.84 | 8.26 | 5.12 | 7.78 | 2.40 | **8.84** | 3.99 | 8.13 |
| SS | 3.57 | 7.03 ▼ | 3.47▼ | 6.80▼ | 2.76 ▲ | 7.30▼ | 4.50 ▲ | 6.97▼ |
| DE | 2.16 ▼ | 7.06▼ | 1.82▼ | 6.19▼ | 2.52 ▲ | 7.46 ▼ | 2.23▼ | 7.42▼ |
| SE | 2.80 ▼ | 6.41▽ | 1.98▼ | 6.15▼ | 2.50 ▲ | 6.66▼ | 3.59▼ | 6.41▼ |
| AE | 2.40 ▼ | 5.02▼ | 2.37▼ | 5.02▼ | 2.00▼ | 5.03▼ | 2.76▼ | 5.02▼ |
| WMD+MMR | **4.53** △ | **8.71** | 4.21▲ | **8.23▲** | **3.47▲** | 8.41▼ | **5.90▲** | **9.44▲** |

**Table 4.4.** Results by number of entities.

| | All ($n = 50$) | | 4-6 ($n = 21$) | | 7-9 ($n = 23$) | | 10+ ($n = 6$) | |
|---|---|---|---|---|---|---|---|---|
| | Syn | Sem | Syn | Sem | Syn | Sem | Syn | Sem |
| HRED-QS | 3.84 | 8.26 | 4.52 | 8.98 | 3.13 | 8.03 | 4.27 | 6.64 |
| SS | 3.57 | 7.03▼ | 3.34 | 7.72▼ | 3.55 | 6.41▼ | 4.65 | 6.82 |
| DE | 2.16▼ | 7.06▼ | 2.74 | 7.67 ▼ | 1.40 ▽ | 6.37▼ | 2.00 | 7.32 |
| SE | 2.80 ▽ | 6.41 ▼ | 2.47 | 7.21▼ | 2.09 ▽ | 5.64 ▼ | 6.01 | 6.57 |
| AE | 2.40 ▼ | 5.02▼ | 2.54▽ | 4.76▼ | 1.84 ▽ | 4.67▼ | 3.48 | 6.82 |
| WMD+MMR | **4.53**△ | **8.71** | **4.82**△ | **9.26** | **3.67** | **8.13** | **6.63**△ | **8.96** |

**Table 4.5.** Results by number of subtasks.

| | All($n = 50$) | | Head($n = 5$) | | Torso($n = 12$) | | Tail($n = 33$) | |
|---|---|---|---|---|---|---|---|---|
| | Syn | Sem | Syn | Sem | Syn | Sem | Syn | Sem |
| HRED-QS | 3.87 | 8.26 | **4.39** | **10.10** | 3.07 | 6.50 | 4.05 | 8.62 |
| DE | 2.14 ▼ | 7.06 ▼ | 2.85 | 7.62 | 2.25 | 6.21 | 2.01 ▼ | 7.32 ▼ |
| DE+SE | 2.56 ▼ | 7.80 | 1.55 | 9.15 | 1.65 | 6.34 | 3.20△ 8.13 | 8.13 |
| DE+AE | 3.33 | 7.50 ▼ | 2.56 | 8.78 | 2.28 | 6.38 | 3.80 | 7.72 ▼ |
| SE | 2.83 ▽ | 6.41 ▼ | 0.95 | 7.49 | 2.21 | 4.57 ▽ | 3.58 | 6.92 ▼ |
| SE+AE | 2.52 ▼ | 7.39 ▼ | 1.71 | 8.36 | 0.91 ▼ | 5.81 | 3.27 | 7.82 △ |
| AE | 2.40▼ | 6.91 ▼ | 1.72 | 9.23 | 2.44 | 5.39 | 2.40 ▼ | 7.31 ▼ |
| DE+SE+AE | **4.50** △ | **8.35** | 4.29 | 9.68 | **5.48**△ | **6.80** | **4.12** | **8.71** |

**Table 4.6.** Ablation analysis by query frequency.

Finally, we also considered every possible combination between our methods. These results can be seen in Tables 4.6 to 4.8. From these analyses, we see that our intuition is correct. Direct Expansion usually performs best when dealing with low diversity scenarios, while the Analogical Expansion is able to further improve for complex tasks.

| | All($n = 50$) | | 10-16($n = 16$) | | 17-31($n = 17$) | | 31-10($n = 17$) | |
|---|---|---|---|---|---|---|---|---|
| | Syn | Sem | Syn | Sem | Syn | Sem | Syn | Sem |
| HRED-QS | 3.87 | 8.26 | **5.20** | 7.78 | 2.39 | **8.84** | 4.02 | 8.13 |
| DE | 2.14 ▼ | 7.06 ▼ | 1.88 ▼ | 6.19 ▼ | 2.52 ▲ | 7.46 ▼ | 2.12 ▼ | 7.42 ▼ |
| DE+SE | 2.56 ▼ | 7.80 | 1.14 ▼ | 7.24 ▼ | 2.49 ▲ | 8.04 ▼ | 3.59 ▲ | 8.09 ▼ |
| DE+AE | 3.33 | 7.50 ▼ | 2.61 ▼ | 6.58 ▼ | 3.15 ▲ | 7.86 ▼ | 4.13 ▲ | 8.02 ▼ |
| SE | 2.83 ▽ | 6.41 ▼ | 1.99 ▼ | 6.15 ▼ | 2.51 ▲ | 6.66 ▼ | 3.67 ▲ | 6.41 ▼ |
| SE+AE | 2.52 ▼ | 7.39 ▼ | 1.15 ▼ | 6.90 ▼ | 2.76 ▲ | 7.53 ▼ | 3.20 ▼ | 7.71 ▼ |
| AE | 2.40▼ | 6.91 ▼ | 2.62 ▼ | 6.49 ▼ | 1.76 ▼ | 7.17 ▼ | 2.72 ▼ | 7.07 ▼ |
| DE+SE+AE | **4.50** △ | **8.35** | 4.20 ▼ | **7.85** ▲ | **3.46** ▲ | 8.68 ▼ | **5.84** ▲ | **8.47** ▲ |

**Table 4.7.** Ablation analysis by number of entities.

| | All($n = 50$) | | 4-6($n = 21$) | | 7-9($n = 23$) | | 10+($n = 6$) | |
|---|---|---|---|---|---|---|---|---|
| | Syn | Sem | Syn | Sem | Syn | Sem | Syn | Sem |
| HRED-QS | 3.87 | 8.26 | **4.59** | **8.98** | 3.13 | **8.03** | 4.24 | 6.64 |
| DE | 2.14 ▼ | 7.06 ▼ | 2.64 ▽ 7.67 ▼ | 6.19 ▼ | 1.49 ▽ | 6.37 ▼ | 1.95 | 7.32 |
| DE+SE | 2.56 ▼ | 7.80 | 2.31 ▽ 8.71 | 7.24 ▼ | 1.79 | 6.96 ▼ | 5.08 | 7.86 |
| DE+AE | 3.33 | 7.50 ▼ | 3.09 ▽ 8.41 △ | 6.58 ▼ | 3.25 | 6.61 ▼ | 4.31 | 7.77 |
| SE | 2.83 ▽ | 6.41 ▼ | 2.50 ▽ 7.21 ▼ | 6.15 ▼ | 2.14 | 5.64 ▼ | 6.04 | 6.57 |
| SE+AE | 2.52 ▼ | 7.39 ▼ | 1.15 ▼ | 6.90 ▼ | 1.79 ▽ | 6.60 ▼ | 4.92 | 6.96 |
| AE | 2.40▼ | 6.91 ▼ | 2.29 ▽ 8.37 | 6.49 ▼ | 1.81 ▽ | 5.89 ▼ | 3.46 | 7.01 |
| DE+SE+AE | **4.50** △ | **8.35** | 2.58 ▼ | 7.83 ▼ | **3.66** | 7.67 ▽ | **6.46** △ | **8.37** |

**Table 4.8.** Ablation analysis by number of subtasks.

### 4.2.4 Crowdsourcing Results

To further validate these observations, we submitted the results produced by our combined approach using WMD+MMR, as well as those produced by the SS and HRED baselines to additional assessment by human judges via crowdsourcing, using the Figure Eight platform [5]. In particular, the results produced by these methods were aggregated and shown to three judges each, with similar instructions as those used by TREC judges, and no further information about which approach produced which result.

The platform used also introduces a honey-pot mechanism, to avoid malicious users. For this, we used ground-truth results, asking users to assess the quality of those golden queries. The results of this investigation are shown in Figure 4.16, once again broken down according to the aforementioned groups of queries. When judges disagree on any topic, we take the rounded average score attributed by all three judges.

From Figure 4.16, we first note a generally consistent trend with the approximated results reported in Figures 4.8 and 4.9. However, the results attained by our approach

[5] https://figure-eight.com

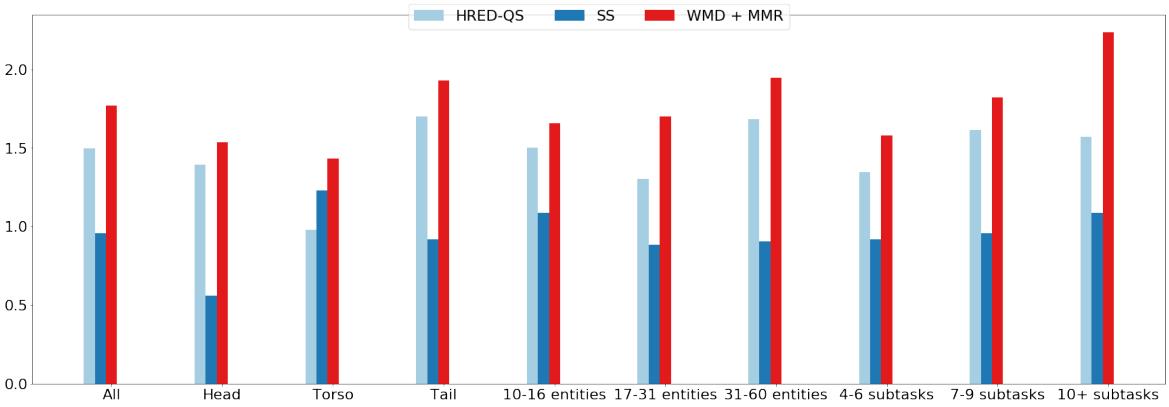


**Figure 4.16.** ERR-IA@20 with exhaustive (crowdsourced) judgments.

are even stronger in this exhaustive judging scenario, with improvements compared to both SS and HRED in all tested query groups. Recalling Q3, these observations corroborate the effectiveness of our approach for task understanding, with improvements over the state-of-the-art for queries with various popularity levels, number of expected relevant entities, and number of underlying subtasks.

Finally, in Table 4.9, we present the numerical results for human judges. Once again, it's clear that our combined result consistently improves over the current state-of-the-art, particularly hard tasks and long-tail queries.

| | HRED-QS | SS | WMD+MMR |
|---|---|---|---|
| All | 1.50 | 0.96▼ | 1.77 |
| Head | 1.40 | 0.56 | **1.54** |
| Torso | 0.98 | 1.23 | **1.43** |
| Tail | 1.70 | 0.92▼ | **1.93** |
| 10-16 entities | 1.50 | 1.09▼ | **1.66▲** |
| 17-31 entities | 1.30 | 0.88▼ | **1.70 ▲** |
| 32-60 entities | 1.68 | 0.91▼ | **1.95 ▲** |
| 4-6 subtasks | 1.35 | 0.92 | **1.58** |
| 7-9 subtasks | 1.61 | 0.96▽ | **1.82** |
| 10+subtasks | 1.57 | 1.09 | **2.24** |

**Table 4.9.** Human judges results.

In order to understand how reliable the human judges are, we also computed the Cohen's Kappa for each of the tasks. It is computed by the average kappa of each of the subtasks. The histogram of the average kappa for each subtask can be seen in Figure 4.17. The reported average value of 0.34 is in line with what is expected in human judgment scenarios, as reported by [26], given that relevance judgment is not a trivial task, specially when the judge is not an expert.

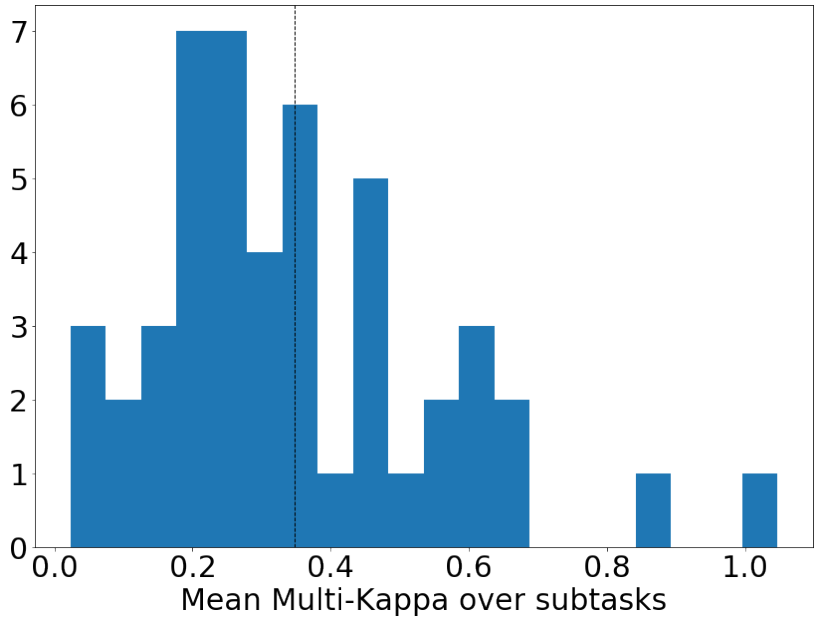


**Figure 4.17.** Cohen Kappa histogram for each task.

## 4.3 Summary

In this chapter, we described in detail our setup for experimentation and shown the results for these experiments. In particular, we focused our experiments in trying to answer the following research questions:

Q1. How effective are our proposed strategies?

Q2. How complementary are our proposed strategies?

Q3. How do our strategies perform for different types of query?

We began by exploring our test collections, in Section 4.1.1, where we discussed how we built our bipartite graph $\mathcal{G}$ over the AOL query log, and some statistics extracted from $\mathcal{G}$. We also discussed briefly the test collection provided by the TREC 2006 Task Understanding.

In Section 4.1.2 we described the difficulty of reusing the provided ground-truth, and how to address this issue with a scheme of relaxation using two dissimilarity thresholds. Section 4.1.3 described our choices of baselines, and how they were implemented.

Section 4.2 explored the results of our experiments. Specifically, how they address each of our previously mentioned research questions. Section 4.2.1 answered Q1 by describing how each of our methods fare against the baselines. In particular, we demonstrated that, by mixing our methods, we achieve new state-of-the-art on query

suggestion for task understand. For addressing Q2, Section 4.2.2 discussed how distinct are the rankings produced by each of our methods. We also discussed how different data fusion techniques could be used for fusing the results. Finally, Section 4.2.3 showed, in details, how each of our methods and their combination fares in multiple scenarios. We showed that our fusing method is especially effective for long tail queries and hard tasks.

In order to validate that our results are not just an artefact of the evaluation scheme, we also submitted them to a crowdsourcing platform. As shown in Section 4.2.4, our results were validated by a number of human judges, where we found results consistent with our proposed methodology.

# Chapter 5

# Conclusions and Future Work

Information retrieval systems are a key part of our daily lives. Their presence in our desktops, mobile devices and even integrated in our home appliances make such systems an easy to access point of entrance to the Web, helping us to learn, find facts and make decisions. When trying to complete any task, on any domain, it is increasingly common for users to resort to one of such systems, from the simplest factoid tasks, like searching for who is the president of New Zealand all the way to complex tasks, like planning a trip to another country or learning a complex new topic. Therefore, understanding the task underlying a user query can be a great way to optimize the user experience in these scenarios.

In this dissertation, we proposed three novel strategies for navigating a semantically annotated query log for task understanding. Our proposed strategies vary in complexity and reach, providing alternative mechanisms for producing suggestions with a diverse coverage of the subtasks underlying the user's task. As proposed by the TREC Task track, in these situations, given a sample input query by a user, a task understanding system need to discover the underlying task the user is trying to accomplish and suggest a set of ranked key phrases. These suggested key phrases can be used as queries to a search engine, hopefully helping the user solve the largest number of subtasks related to the original task as possible.

We presented three novel approaches for task understanding. Namely, we introduced the *Direct Expansion*, *Syntagmatic Expansion* and *Analogical Expansion* strategies. We have shown that they are complementary, in the sense that they produce rankings that are almost completely disjoint, and that each of these expands the set of candidates in a different direction. Thus, we also showed that, by combining these strategies, we can produce a new state-of-the-art approach, useful for long tail queries and hard tasks.

In the remainder of this chapter, we summarize the conclusions drawn from our experiments, the main contributions of this work, possible directions for future work and our final remarks.

## 5.1 Summary of Contributions

We believe that this dissertation has the following main contributions:

- **A new approach for representing and navigating a query log.** We presented, in Section 3.1, a new approach of representing a query log, by building a bipartite graph using an annotated query log and framing the query suggestion problem as an edge suggestion problem. We also described three different strategies to navigate this query log, with varying complexity and effectiveness in multiple scenarios. In particular, we described the Direct Expansion approach in Section 3.2, which works by changing the original context of the input query, varying the actions to be taken on the same action. In Section 3.3 we introduced the Syntagmatic Expansion, which expands over the original entity by using a space of latent variables, generated using word embedding techniques. Finally, in Section 3.4, we described the Analogical Expansion technique, a high-risk-high-reward strategy, that generates candidates by performing analogies over entities typically similar to the input entity.

- **A new state-of-the-art task understanding approach.** We showed, in Chapter 4, how our strategies fare against current state-of-the-art approaches to query suggestion. We demonstrated that, by combining our strategies, we surpassed current state-of-the-art approaches. We showed that our fusing strategy is highly effective when dealing with the most difficult scenarios, long-tail queries and complex tasks. We also presented the result of an evaluation made on a crowdsourcing environment, where human judges validated our results.

## 5.2 Summary of Conclusions

In this dissertation, we presented three new approaches for task-understanding, framing the problem as an edge suggestion problem over a bipartite graph built over a semantically annotated query log. In order to understand the impact of each of these strategies, we built an evaluation scheme based on relaxation of similarity thresholds,

both semantic and syntactic. In addition, we also employed human judges in order to access the quality of our suggestion.

We found that our strategies are highly complementary, and, while not able to perform better than our baselines individually, a combination of them can generate a new state-of-the-art approach to the problem. This finding was further corroborated by our human judges' evaluation, which effectively showed that our methods are especially useful for long tail and complex tasks.

## 5.3 Directions for Future Research

Based on our findings, we believe that this work could have further impact in the following research directions:

- We believe that approaches similar to the ones presented here could be highly useful in systems where complex tasks are explicitly being completed. For instance, a flight tickets website could deploy a similar system for recommending activities for shoppers in the area. Other area that could see improvements from this work is in search-as-learning, where a learning task is modeled as a searching problem. Therefore, we look forward for other instantiations of our approaches, applied to scenarios similar to these.

- Another interesting direction of research is in the application of the strategies presented in smart assistant scenarios. Given the task-oriented nature of such systems, we hypothesize that approaches similar to the ones we described in this dissertation could be highly useful on these scenarios.

- Other approaches to navigating the proposed bipartite graph could also be implemented. For instance, using neural networks for extracting edge and node embeddings could generate interesting results, leveraging intrinsic graph relationships that are perhaps not so obvious.

- In the evaluation front, we think that other evaluation schemes could be useful. The problem of reusing small ground-truth datasets is complex, and strategies for addressing this are needed, even after showing that our approaches were highly correlated with human judges. We also believe that other evaluation metrics, for different goals, diverse from task understanding could be studied, in order to better understand if our findings are replicable in other domains, like classic query suggestions.

- Another interesting future direction of work would be to look into different methods for merging navigation strategies on $\mathcal{G}$. While we demonstrated that using Word Mover's Distance for ranking, followed by MMR re-ranking works well, it is possible that other approaches could yield better results.

- One of the shortcomings of our proposed methods is the lack of an intent-aware approach to distinguish between task-oriented and non-task-oriented suggestions. We hypothesize that an approach toward this could yield relevant results on task understanding.

- Another interesting approach that can be further examined is to use different kinds of word embeddings. We believe that, instead of using general purpose embeddings, one could train these using the query log itself, with possible improvements on syntagmatic and analogical expansions.

- Due to the inherent dependency between our methods and the data it derives from (specifically, the Wiki2Vec embeddings), it is a complex task to actually measure which part of our method produces the largest improvements on the results. Therefore, measuring these gains and identifying possible pathways for further improvements based on these findings is another good direction for future works.

## 5.4 Final Remarks

This dissertation contributed three new, complementary approaches to task understanding, beating current state-of-the-art techniques. In the research perspective, it also helped build a deeper understanding of how queries, entities and contexts are related in a task understanding scenario.

# Bibliography

[1] Baeza-Yates, R., Hurtado, C., and Mendoza, M. (2004). Query recommendation using query logs in search engines. In *International Conference on Extending Database Technology*, pages 588--596. Springer.

[2] Baeza-Yates, R. A. and Tiberi, A. (2007). Extracting semantic relations from query logs. In *Proc. of KDD*, pages 76--85.

[3] Beeferman, D. and Berger, A. (2000). Agglomerative clustering of a search engine query log. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 407--416. ACM.

[4] Bengio, Y., Ducharme, R., Vincent, P., and Jauvin, C. (2003). A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137--1155.

[5] Bennett, P. N. and White, R. W. (2015). Mining tasks from the web anchor text graph: MSR notebook paper for the TREC 2015 tasks track. In *TREC*, volume Special Publication 500-319. National Institute of Standards and Technology (NIST).

[6] Blanco, R., Ottaviano, G., and Meij, E. (2015). Fast and space-efficient entity linking for queries. In *Proc. of WSDM*, pages 179--188.

[7] Boldi, P., Bonchi, F., Castillo, C., Donato, D., and Vigna, S. (2009). Query suggestions using query-flow graphs. In *Proc. of WSCD*, pages 56--63.

[8] Broccolo, D., Marcon, L., Nardini, F. M., Perego, R., and Silvestri, F. (2012). Generating suggestions for queries in the long tail with an inverted index. *Inf. Process. Manage.*, 48(2):326--339.

[9] Broder, A. (2002a). A taxonomy of web search. *SIGIR Forum*, pages 3--10.

[10] Broder, A. (2002b). A taxonomy of web search. In *ACM Sigir forum*, pages 3--10.

[11] Cao, H., Jiang, D., Pei, J., He, Q., Liao, Z., Chen, E., and Li, H. (2008). Context-aware query suggestion by mining click-through and session data. In *Proc. of KDD*, pages 875--883.

[12] Carbonell, J. and Goldstein, J. (1998a). The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 335--336. ACM.

[13] Carbonell, J. and Goldstein, J. (1998b). The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Proc. of SIGIR*, pages 335--336.

[14] Carterette, B., Kanoulas, E., Pavlu, V., and Fang, H. (2010). Reusable test collections through experimental design. In *Proc. of SIGIR*, pages 547--554.

[15] Chen, H. and Karger, D. R. (2006). Less is more: probabilistic models for retrieving fewer relevant documents. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 429--436. ACM.

[16] Chen, W., Cai, F., Chen, H., and de Rijke, M. (2018). Attention-based hierarchical neural query suggestion. In *Proc. of SIGIR*, pages 1093--1096.

[17] Fonseca, B. M., Golgher, P. B., De Moura, E. S., Pôssas, B., and Ziviani, N. (2003). Discovering search engine related queries using association rules. *Journal of Web Engineering*, 2(4):215--227.

[18] Fox, E. A., Koushik, M. P., Shaw, J., Modlin, R., Rao, D., et al. (1993). Combining evidence from multiple searches. In *The first text retrieval conference (TREC-1)*, pages 319--328.

[19] Garigliotti, D. and Balog, K. (2016). The university of stavanger at the TREC 2016 tasks track. In *TREC*, volume Special Publication 500-321. National Institute of Standards and Technology (NIST).

[20] Glater, R., Santos, R. L., and Ziviani, N. (2017). Intent-aware semantic query annotation. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 485--494. ACM.

[21] Grave, E., Mikolov, T., Joulin, A., and Bojanowski, P. (2017). Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL*, pages 3--7.

[22] Guo, J., Xu, G., Cheng, X., and Li, H. (2009). Named entity recognition in query. In *Proc. of SIGIR*, pages 267--274.

[23] Hagen, M., Kiesel, J., Adineh, P., Alahyari, M., Fatehifar, E., Bahrami, A., Fichtl, P., and Stein, B. (2016). Webis at TREC 2016: Tasks, Total Recall, and Open Search tracks. In *Proc. of TREC*.

[24] Jones, R., Rey, B., Madani, O., and Greiner, W. (2006). Generating query substitutions. In *Proceedings of the 15th international conference on World Wide Web*, pages 387--396. ACM.

[25] Kanoulas, E., Yilmaz, E., Mehrotra, R., Carterette, B., Craswell, N., and Bailey, P. (2017). TREC 2017 Tasks track overview. In *Proc. of TREC*.

[26] Kazai, G., Craswell, N., Yilmaz, E., and Tahaghoghi, S. (2012). An analysis of systematic judging errors in information retrieval. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, pages 105--114.

[27] Kusner, M. J., Sun, Y., Kolkin, N. I., and Weinberger, K. Q. (2015). From word embeddings to document distances. In *Proc. of ICML*, pages 957--966.

[28] Levy, O. and Goldberg, Y. (2014a). Dependency-based word embeddings. In *Proc. of ACL*, pages 302--308.

[29] Levy, O. and Goldberg, Y. (2014b). Neural word embedding as implicit matrix factorization. In *Proc. of NIPS*, pages 2177--2185.

[30] Macdonald, C., McCreadie, R., Santos, R. L., and Ounis, I. (2012). From puppy to maturity: experiences in developing Terrier. *Proc. of OSIR at SIGIR*, pages 60--63.

[31] Mei, Q., Zhou, D., and Church, K. (2008). Query suggestion using hitting time. In *Proceedings of the 17th ACM conference on Information and knowledge management*, pages 469--478. ACM.

[32] Meij, E., Bron, M., Hollink, L., Huurnink, B., and Rijke, M. (2009). Learning semantic query suggestions. In *Proc. of ISWC*, pages 424--440.

[33] Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781.

[34] Mitra, B. (2015). Exploring session context using distributed representations of queries and reformulations. In *Proc. of SIGIR*, pages 3--12. ACM.

[35] Navarro, G. (2001). A guided tour to approximate string matching. *ACM Comput. Surv.*, pages 31--88.

[36] Pennington, J., Socher, R., and Manning, C. D. (2014). Glove: Global vectors for word representation. In *Proc. of EMNLP*, pages 1532--1543.

[37] Radlinski, F., Bennett, P. N., Carterette, B., and Joachims, T. (2009). Redundancy, diversity and interdependent document relevance. In *ACM SIGIR Forum*, volume 43, pages 46--52. ACM.

[38] Salton, G. and Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. *Information processing & management*, 24(5):513--523.

[39] Santos, R. L. T., Macdonald, C., and Ounis, I. (2010). Exploiting query reformulations for web search result diversification. In *Proc. of WWW*, pages 881--890.

[40] Santos, R. L. T., Macdonald, C., and Ounis, I. (2013). Learning to rank query suggestions for adhoc and diversity search. *Inf. Retr.*, 16(4):429--451.

[41] Santos, R. L. T., Macdonald, C., and Ounis, I. (2015). Search result diversification. *Found. Trends. Inf. Retr.*, 9(1):1--90.

[42] Silvestri, F. (2010). Mining query logs: turning search usage data into knowledge. *Found. Trends Inf. Retr.*, 4(1–2):1--174.

[43] Song, Y., Zhou, D., and wei He, L. (2011). Post-ranking query suggestion by diversifying search results. In *Proc. of SIGIR*, pages 815--824.

[44] Sordoni, A., Bengio, Y., Vahabi, H., Lioma, C., Simonsen, J. G., and Nie, J.-Y. (2015). A hierarchical recurrent encoder-decoder for generative context-aware query suggestion. In *Proc. of CIKM*, pages 553--562.

[45] Verma, M., Yilmaz, E., Mehrotra, R., Kanoulas, E., Carterette, B., Craswell, N., and Bailey, P. (2016). Overview of the TREC Tasks track 2016. In *Proc. of TREC*.

[46] Yilmaz, E., Verma, M., Mehrotra, R., Kanoulas, E., Carterette, B., and Craswell, N. (2015). Overview of the TREC 2015 tasks track. In *TREC*, volume Special Publication 500-319. National Institute of Standards and Technology (NIST).