Neural Embedding-based Metrics for Pre-Retrieval Query Performance Prediction

by

Negar Arabzadehghahyazi

Bachelor of Engineering, Amirkabir University of Technology, Tehran, 2017

A thesis

presented to Ryerson University

in partial fulfillment of the

requirements for the degree of

Master of Applied Science

in the Program of

Electrical, Computer and Biomedical Engineering

Toronto, Ontario, Canada, , 2019

AUTHOR'S DECLARATION FOR ELECTRONIC SUBMISSION OF A THESIS

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I authorize Ryerson University to lend this thesis to other institutions or individuals for the purpose of scholarly research.

I further authorize Ryerson University to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

I understand that my dissertation may be made electronically available to the public.

Neural Embedding-based Metrics for Pre-Retrieval Query Performance Prediction

Master of Applied Science , 2019

Negar Arabzadehghahyazi

Electrical, Computer and Biomedical Engineering

Ryerson University

Abstract

Pre-retrieval Query Performance Prediction (QPP) methods are oblivious to the performance of the retrieval model as they predict query difficulty prior to observing the set of documents retrieved for the query. Among pre-retrieval query performance predictors, *speci city*-based metrics investigate how corpus, query and corpus-query level statistics can be used to predict the performance of the query. In this thesis, we explore how neural embeddings can be utilized to define corpus-independent and semantics-aware specificity metrics. Our metrics are based on the intuition that a term that is closely surrounded by other terms in the embedding space is more likely to be specific while a term surrounded by less closely related terms is more likely to be generic. On this basis, we leverage geometric properties between embedded terms to define four groups of metrics: (1) neighborhood-based, (2) graph-based, (3) cluster-based and (4) vector-based metrics. Moreover, we employ learning-to-rank techniques to analyze the importance of individual specificity metrics. To evaluate the proposed metrics, we have curated and publicly share a test collection of term specificity measurements defined based on Wikipedia category hierarchy and DMOZ taxonomy. We report on our extensive experiments on the effectiveness of our metrics through metric comparison, ablation study and comparison

against the state-of-the-art baselines. We have shown that our proposed set of pre-retrieval QPP metrics based on the properties of pre-trained neural embeddings are more effective for performance prediction compared to the state-of-the-art methods. We report our findings based on Robust04, ClueWeb09 and Gov2 corpora and their associated TREC topics.

Acknowledgements

First and foremost I want to express my deepest gratitude to my incredible supervisor, Dr. Ebrahim Bagheri, from whom I have benefited immensely. Dr. Bagheri supported me throughout my tenure at Ryerson, and not only he was potent in composition and development of my research interest and projects but also in my goals and career. Above all, he ignites within me, the desire to pursue my goals and dreams. It is evident that without him none of this would be possible and I would not be where I am today. I cannot imagine having a better supervisor, he is truly one of a kind. I would also like to thank my fellow lab member, Dr. Fattane Zarrinkalam. She too was crucial in helping me finish my thesis and other projects. But more importantly, I have to thank her for being a friend who was always there for me and helped more ways than I can count. I am as well appreciative of my mother, Fatemeh Zolfaghari and all her sacrifices. It was because of her that I am the person that I am, both professionally and personally. Finally, I would like to thank my best friends, Parisa Mahvelati and Paria Samani whose support provided me with much needed tranquility to keep me sane through this endeavour.

Negar Arabzadehghahyazi

Contents

List of Tables

List of Figures

List of Symbols

E	Set of all edges in a graph
μ	Predictor
$\mu(x)$	Degree of similarity of the most similar term to x
υ	Vertex
$\grave{AP}(q)$	Average precision of a query q
$\xi(t_i)$	Ego network for term t_i
ζ	Graph
CS	Coherency Score
D	Collection or Corpus
D_q	Set of retrieved document for query q
e	Edge in ego network
I_q	Information need behind the query q

K	Number of clusters in a centroid network
M_{t_i}	Most similar neighbor to term t_i
N	Number of documents in the collection
$N\varepsilon(t)$	ε-Neighborhood of term t
N_t	Number of documents containing the query term t in the corpus
q	Query
t	Term
tf	Term frequency
V_t	Pre-trained embedding vector of a term t

Chapter 1

Introduction

1.1 Background and Problem Statement

Information Retrieval (IR) is obtaining material relevant to an information need which is represented by a query within comprehensive collections [1]. In terms of text analytics, information retrieval can be reworded to the science of searching for relevant text documents in a collection of documents. In an information retrieval system, documents are usually unstructured, and queries are formed by simple keywords. An information retrieval system performance is not consistent across all the queries and all the corpora. A potential reason for explaining the discrepancy in retrieval systems performance is that several variables are involved in an information retrieval system, including the corpus, the query, and the retrieval system [2]. While a retrieval system can perform well for query A on corpus C_1, the same query might have an extremely poor performance on corpus C_2. Further, on corpus C_1, query A might have unsatisfactory performance whereas query B might be very easy to satisfy on the same cor-

pus. Therefore, the need to predict information retrieval performance motivated. Predicting an information retrieval system performance can be beneficial from several aspects, e.g., we can find the proper retrieval system for a given query, or we can expand the query in order to represent the information need in a more effectively. Consequently, we can enhance the overall performance of the information retrieval system.

The problem of predicting an information retrieval system performance for a given query is called *Query Performance Prediction (QPP)* [2]. Query performance prediction metrics are categorized into two groups: (1) *Pre-retrieval Query Performance Prediction* and (2) *Post-retrieval Query Performance Prediction*. The former predicts a retrieval performance for a given query q, without knowing the retrieved documents while the latter predict the retrieval performance after the retrieval is conducted by using the query and the retrieved documents. In general, post-retrieval predictors are more expensive in terms of computation and time complexity. However, they outperform pre-retrieval predictors. The post-retrieval QPP methods are out of the scope of this thesis, and we only focus on pre-retrieval QPP methods.

According to the intuition behind the pre-retrieval query performance predictors, they can be classified into (1) Similarity-based (2) Coherency-based (3) term relatedness-based and (4) Specificity-based methods [3]. Similarity-based pre-retrieval QPP methods are based on the similarity between the query and the corpus. The higher similarity among a query and a corpus implies a higher probability for relevant documents existence. Coherency-based pre-retrieval QPP methods measure the similarity among documents containing the query terms. Term relatedness-based pre-retrieval QPP methods assess the co-occurrence of query terms in the collection based on the idea that frequent co-occurrence of query terms can be interpreted

as the ability of the query to be easily satisfied. The last group of pre-retrieval methods, i.e., specificity-based query performance predictors, are the focus of this thesis. Specificity-based QPP methods have been proposed based on the idea that more specific queries are easier to answer. Therefore, there is a need to estimate the specificity of query terms.

Estimating term specificity not only is useful to solve the QPP problem but also is one of the has several applications, especially in the web domain, e.g., personalized recommendation and user interest modeling on social media. Specificity, often defined as inversely related to ambiguity, has traditionally been estimated using corpus-specific frequency statistics [4]. For example, the inverse document frequency (IDF) of a term in a given corpus can be considered as an indicator of the term specificity [5]. Frequency-based specificity metrics suffer from losing the semantic aspects of the terms and the dependency among them. On the other hand, neural embeddings representation of terms, allow us to identify and measure inter-term associations based on how the vector representations of terms are distributed within the embedding space. However, corpus-level term frequency information is not explicitly maintained in the vector representation of terms in neural embeddings. The objective of this thesis is to investigate the possibility of estimating term specificity by utilizing neural embedding-based representation of terms. Such specificity metrics can be utilized to predict query performance based on the idea that the more a query is specific, the higher performance it has.

While existing work in pre-retrieval query performance has been predominantly focused on defining various statistical measures based on term and corpus-level frequency, the information retrieval community has embarked on exploring the impact and importance of neural information retrieval techniques. More recently, the information retrieval community has embarked

on exploring the impact and importance of neural embeddings in different applications such as query expansion, query classification, and ranking, just to name a few [6, 7]. By using pre-trained neural embeddings, unlike existing pre-retrieval QPP methods, our work is not dependent on collection-specific frequency statistics. Therefore, performance prediction can be computed even for terms that are not frequent in the document collection. For instance, while predicting the performance of queries containing less frequently observed terms in the document collection is a major challenge for frequency-based metrics, such terms can be handled using their relative position to other terms within the embedding space. In this thesis, we explore how the geometric properties of embeddings can be exploited to define metrics to be used for QPP. The benefit of this is that while traditional QPP metrics focus on various forms of query term frequency, the consideration of the associations between term vectors in the neural embedding space will allow us to derive other forms of query term metrics beyond frequency.

To the best of our knowledge, while there has been recent work that uses neural methods to train query performance prediction models based on a host of signals [7], there is yet to be work that explores the characteristics of pre-trained neural embeddings for effective query performance prediction. Moreover, our proposed QPP metrics in this thesis distinguishes itself from very recent related work [8], which proposes to cluster neural embeddings for difficulty prediction, by suggesting an ego-network representation of neural embeddings for performing QPP. More specifically, we propose to measure term specificity based on the distribution of terms that are observed around the term of interest in the embedding space. In addition, while Roy et al. [8] proposed metric, i.e, $P_{clarity}$ has outstanding performance among post-retrieval QPP metrics, it shows poor performance among pre-retrieval state-of-the-art baselines.

Our work is driven by the intuition that more specific terms have a higher likelihood of being surrounded by a higher number of specific terms compared to generic terms. In other words, given the fact that highly specific terms express precise semantics, they are more likely to be highly similar to other specific terms (and hence be surrounded by many specific terms) whereas generic terms tend to have weaker relationships with other terms.

We employ the geometric properties of each term's neural embeddings to define its degree of specificity. More concretely, the more a term is surrounded by specific terms, the more specific it will be. We benefit from term vector associations in the neural embedding space for formalizing term specificity, which has shown to be correlated with query difficulty [9]. We base our work on the intuition that a term that has been closely surrounded by other terms in the embedding space is more likely to be specific while a term with less closely surrounded terms is more likely to be generic.

Specific terms are expected to have a denser neighborhood in the embedding space because they tend to have more semantically related terms than generic terms. Given a term t and its pre-trained embedding vector V_t, $\varepsilon - neighborhood$ of V_t consists of terms in the embedding space that have higher similarity than a specific threshold to t. On this basis, we conceptualize the space surrounding a term by using an ego network where the term of interest forms the ego and is contextualized within a set of alter nodes, which are other terms that are closely positioned around it in the embedding space. In the context of this ego network, we apply various measures of node connectedness on the ego node to determine the specificity of the term that is being represented by the ego, which would then indicate query difficulty.

We define four groups of unsupervised specificity metrics based on their characteristics

including (1) *Neighborhood-based metrics*, (2) *Graph-based metrics*, (3) *Cluster-based metrics* and (4) *Vector-based metrics*. Neighborhood-based metrics only utilize the ε-neighborhood, and they are based on the relationship among the neighbors to the ego term t. At the same time, graph-based metrics consider the relationship among the neighbors to the ego term and relationship among the neighbors simultaneously. The other group of our proposed metrics, i.e., cluster-based metrics are founded on the number of potential clusters in the ε-neighborhood. Finally, vector-based metrics analyze the characteristic of an embedding vector alone and disregard other vectors in the embedding space. In total, we propose 25 different unsupervised specificity metrics. Besides the four groups of unsupervised specificity metrics, we proposed an unsupervised specificity metrics by applying learning to rank, which is fed by all the 25 proposed metrics as input features.

Before evaluating our proposed specificity metrics for the QPP task, we directly assessed specificity metrics, neglecting the application. We introduce and publicly share two test collections, which are structured collections of terms with associated human-defined specificity values. The test collections have been derived from the Wikipedia category hierarchy and DMOZ taxonomy with the understanding that categories higher up in the hierarchy are more generic, while those further down in the hierarchy are more specific. Specifically, we have extracted sequences of categories from the Wikipedia's category hierarchy, where each sequence consists of semantically related categories with a progressively higher degree of specificity, reflecting the change in specificity from the higher towards the lower levels in the hierarchy. These sequences are used to measure the performance of the proposed neural embedding-based specificity metrics.

We examine the robustness of our proposed metrics to the gold standard hierarchy and

the pre-trained embedding by conducting an experiment on two different test collections and three different pre-trained embeddings. Our metrics show almost consistent performance among different varieties of the experiments. This fact implies that our proposed metrics are robust to the pre-trained embeddings. In other words, the characteristics of the corpus that the embeddings were trained on, and also the word embeddings training methodology, do not affect our proposed metrics.

In order to evaluate the final goal of this thesis, which is proposing novel QPP metrics, we have performed experiments based on three widely used TREC corpora, namely Robust04, ClueWeb09 and Gov2, and their corresponding topic sets [10, 11, 12]. Compared to existing pre-retrieval query performance prediction metrics, our experiments show that the proposed neural embedding-based specificity metrics are effective in predicting query performance when using pre-trained neural embeddings.

1.2 Contributions

This thesis makes the following contributions:

1. We formally introduce the task of predicting corpus-independent term specificity based on a collection of embeddings.

2. We propose a host of specificity metrics based on supervised and unsupervised methods for predicting specificity based on neural embedding. We show that these specificity metrics have outstanding performance in pre-retrieval QPP task.

3. We present two test collections consisting of term specificity measurements defined in

relation to Wikipedia and DMOZ categories. These collections are made publicly available for replication studies.

1.3 Structure of the thesis

This thesis is structured as follow:

- Chapter 2 - Related Work: In this chapter, we elaborate more on the importance of QPP problem, and we review the related works in this domain. We explain state-of-the-art pre-retrieval methods in detail and especially specificity-based ones. We also cover details about the pre-trained neural embedding that we have used in this thesis.

- Chapter 3 - The Proposed Approach: This chapter describes the proposed approaches for the problem of proposing novel pre-retrieval query performance predictor using neural embeddings. We have introduced several specificity metrics in four different categories that have the potential to estimate term specificity. They can be used in order to predict query performance in an information retrieval system.

- Chapter 4 - Experiments and Evaluation: This chapter focuses on the evaluations performed on the proposed approaches. We have evaluated our proposed specificity metrics directly and indirectly via QPP application. We have reported the results of our pre-retrieval QPP metrics on TREC collections, including Robust04, ClueWeb09, and GOV2 topics.

- Chapter 5 - Conclusion: In this chapter, we summarize the proposed approaches and discuss potential future work.

Chapter 2

Related Work

In this chapter, we investigate the crucial role of query performance prediction and review the most prestigious works in this area. Query performance can be predicted before the retrieval happens known as *pre-retrieval query performance prediction* or after the retrieval happens known as *post-retrieval query performance prediction*. The ultimate goal of this thesis is to propose a novel pre-retrieval query performance prediction which unlike other existing works, not only it is not based on the frequency of terms in the corpus, but also it can utilize the semantic aspects of terms. In order to preserve the semantic aspects of terms while solving query performance prediction problem, based on term specificity, we used *pre-trained word embeddings*.

In this chapter, first, to have a deeper understanding of specificity, we cover diverse definitions of specificity that were derived from different perspectives, including linguistic and computational views. We also take a look at specificity from the biomedical aspect since the ability of the human brain to distinguish specific and generic terms draw researchers' atten-

tion in recent years. Next, we discuss how specificity metrics have been evaluated directly, as we want to propose a specificity metric. Further, we emphasize the applications of specificity, especially in web applications such as user interest modeling. Finally, we cover the literature using neural embeddings in the context of specificity and its applications.

2.1 Query Performance Prediction

In this section, we explore the query performance prediction problem in information retrieval. First, we define the query performance problem and describe the predicted performance evaluation method. In the performance evaluation process, we set which queries are defined as *hard queries* and which are *easy queries*. Moreover, we investigate how predicting query performance could be beneficial, and we explain why this problem is essential to solve. Later, we categorize different methods for QPP. Finally, as the focus of this thesis is only on the pre-retrieval query performance, we will explain almost all the different pre-retrieval methods in the literature.

Information retrieval (IR) is finding content (in our case documents) of an unstructured nature (in our case text) that satisfies an information need which is represented by a query within extensive collections known as corpora. Information retrieval systems always suffer from performance discrepancy. No information retrieval system exists whose performance is homogeneous for all the queries among all the corpora. There are several reasons to explain this issue in the context of text mining;

1. *Query term ambiguity:* A query term ambiguity sometimes may cause failure in retrieving the desired information need. In simple words, one may look for the query "subway"

without any context. A search engine cannot identify whether the desired information need is *subway the restaurant* or *subway the transportation*.

2. *Query and document language inconsistency:* Another possible reason for information retrieval systems performance discrepancy could be the inconsistencies between the query and the document languages.

3. *Lack of relevant document:* If there is no relevant document to the content of the query, the system performance might be fairly poor.

2.1.1 Problem Definition

Different performance of retrieval systems for different queries and corpora leads to the definition of a problem known as *Query Performance Prediction (QPP)* or *Query Di culty Estimation (QDE)*. QPP is defined as follows:

Definition 1. ***Query Performance Prediction Problem*** *Predicting the quality of retrieved documents D_q in satisfying the information needs I_q behind the query q, the information needs behind q. Given a collection D, a list of retrieved documents D_q and a query q, predictor μ has to estimate $\hat{AP}(q)$, i.e., the average precision of query q is as follows:*

$$\hat{AP}(q) \leftarrow \mu(q, D_q, D) \tag{2.1}$$

After estimating the quality of the retrieved document, the question of *"How good is this predicted quality?"* may arise. Eq. 2.2 shows how the quality of the prediction with the correla-

tion between the actual precision and the predicted values can be measured.

$$Quality(\mu) = correlation([AP(q_1...AP(q_n)], [AP\acute{}(q_1)...AP\acute{}(q_n)]) \tag{2.2}$$

This correlation can be either a linear correlation or a ranked correlation. In this study, we leverage Pearson correlation and Kendall's Tau correlation [13].

A successful prediction of the query performance can lead to the effective selection of the most suitable retrieval method and consequently to improve the systems' effectiveness.

2.1.2 Hard Queries vs Easy Queries

Queries which have low performance, i.e., their average precision score is less than a threshold, are called as *di cult queries*. On the other hand, *easy queries* are the ones which are easy to satisfy. Deciding whether a query is a difficult one for a retrieval system is not an easy task even for a human expert. Because a retrieval system performance depends on several attributes such as the collection, query, and documents' features such as length and structure.

2.1.3 Advantages of QPP

Query Difficulty estimation can be beneficial from the following perspectives:

- *Feedback to users*: The user can rephrase the query.
- *Feedback to a search engines*: The search engine can use different strategies for a different query. Further, it can help parameter tuning when there is no training data.
- *Feedback to system administrator*: When there is no relevant documents for the query,

especially in commercial search engines, the need to expand the collection for difficult queries is sensed.

- *Information retrieval applications*: It can help to merge result of a query over different data sets.

2.1.4 Categorization of QPP

There is already a well-established body of work in the literature that explores query performance prediction through either a post-retrieval or a pre-retrieval strategy (1) *Pre-retrieval query performance prediction methods* and (2) *Post-retrieval query performance prediction methods*. The former methods can be applied before the search is conducted; therefore, they are less expensive compared to the latter one. On the other hand, post-retrieval methods have a better performance compared to pre-retrieval methods; however, they have a heavy process, and they need more input data [2]. Methods in the post-retrieval strategy are focused on measuring retrieval success and hence query difficulty, by analyzing the result set obtained from the retrieval system as a response to the query. In contrast, pre-retrieval methods are interested in the linguistic and statistical features of the query and document spaces. Our work in this thesis falls within the scope of the latter.

2.1.5 Post-retrieval QPP Methods

While Post-retrieval methods analyze the query terms, corpus statistics, and the search result in order to predict the performance of the query in a retrieval process, pre-retrieval methods only utilize query terms. This can explain why pre-retrieval QPP methods are not time consuming,

and they are computationally cheaper, but it should be noted that they have lower performance compared to post-retrieval methods. Post-retrieval methods depend on a retrieval process; for the same query, a different result may happen for different retrieval methods. Using only pre-retrieval methods may lead to misjudging a query as an easy one. Post-retrieval methods are categorized into clarity-based, robustness-based and score distribution-based[2]. Among all the post-retrieval methods, the WIG predictor [14] and the NQC predictor [15] from the score distribution-based category have shown outstanding performance. In this thesis, we focus on pre-retrieval QPP methods, and the post-retrieval methods are out of the scope of this thesis.

2.1.6 Pre-retrieval QPP Methods

The two principal pre-retrieval prediction methods are linguistic and statistical predictors. While linguistic predictors use natural language processing methods to analyze the queries such as the number of morphs per query or the average number of synonyms per query, they did not offer an acceptable performance in predicting the retrieval system performance. On the other hand, statistical approaches which are based on the distribution of query terms in the corpus, perform in a superior way. The only accessible data as an input of a pre-retrieval QPP method is the query terms and its pre-computed statistics in the corpus. While it may seem that query length could be effective in predicting the retrieval performance in a sense that the longer the queries are the easier they are, however, He et al. [9] have shown that there is almost no correlation between the unique number of query terms and the retrieval performance due to the noise drawback of long queries.

Statistical pre-retrieval query performance can be predicted based on a variety of term

features such as specificity, similarity, coherency, or term relatedness. We cover the first one in Section 2.2 and the others in the rest of this section. Among all, *speci city based query performance prediction* stands out; accordingly, we try to employ word embeddings in order to come up with a novel metric to estimate the terms' specificity to predict query performance.

Similarity-based pre-retrieval QPP methods

Similarity-based pre-retrieval QPP metrics measure the quality of retrieved documents based on the similarity between the query and the corpus. If a query and a corpus are highly similar, then most probably multiple documents exist in the corpus that corresponds to the query. In order to determine the similarity score, the collection query similarity known as SCQ of each query term is calculated by measuring the similarity between the query and collection as presented in Eq 2.3 [16].

$$SCQ(t) = (1 + log(tf(t, D))).idf(t) \tag{2.3}$$

where IDF of a query term t is defined as:

$$idf(t) = log(\frac{N}{N_t}) \tag{2.4}$$

In Equation 2.4 N refers to the number of documents in the collection and N_t is the number of documents which contain the query term t. Further, *tf(t,D)* refers to term frequency of term t in the collection D.

Three variations of SCQ exist, namely, *Sum*SCQ, *Max*SCQ, and *Avg*SCQ. Here, sumSCQ measures the sum of all the SCQ values for all the query terms, maxSCQ determines the

maximum collection query similarity over all the query terms, and avgSCQ determines the average over all the query terms as are given in Equation 2.5, 2.6 ,and 2.7 respectively.

$$\mathit{sum}SCQ(q) = \sum_{t \in q} SCQ(t) \tag{2.5}$$

$$\mathit{max}SCQ(q) = max_{t \in q} SCQ(t) \tag{2.6}$$

$$\mathit{avg}SCQ(q) = \frac{1}{|q|} \sum_{t \in q} SCQ(t) \tag{2.7}$$

In general, SCQ metrics show competitive performance among pre-retrieval query performance prediction methods over the TREC benchmarks [16].

Coherency-based pre-retrieval QPP methods

Coherency-based group of pre-retrieval metrics calculate the inter-similarity of documents D_t, which contain the query terms. They are computationally expensive and time demanding due to heavy analysis during index time. Coherency score of term t is calculated by the equation 2.8. [5].

$$CS(t) = \frac{\sum_{(d_i,d_j) \in D_t} sim(d_i, d_j)}{|D_t|(|D_t|-1)} \tag{2.8}$$

where $sim(d_i, d_j)$ is the cosine similarity between the vector-space representation of the documents.

In simple terms, CS calculates similarity among all the document pairs containing query term t. Although this computation can be done at index time, a large amount of computation is required to develop a point-wise similarity matrix for all the pairs of indexed documents. Zhang et al.[16] reduce the computational load of this metric by proposing ***VAR(t)*** in order to calculate the variance of term weights over D_t in the collection. Term weights could be calculated in several ways such as ***tf-idf*** based methods as presented in equation 2.9 where $|d|$ is the number of terms in a document.

$$w(t,d) = \frac{log(1 + tf(t,d)).idf(t)}{|d|} \tag{2.9}$$

VAR(t) is proposed based on the idea that if term weights over the documents are close to each other, it is hard to differentiate among the documents. Whereas high variance among term weights in D_t means distinguishing relevant and non-relevant documents is much easier since their term weights are different from each other.

Term relatedness-based pre-retrieval QPP methods

Term relatedness predictors evaluate the co-occurrence of query terms in the collection. When the query terms co-occur frequently, it indicates that the query is less difficult to answer. The most common predictor for term relatedness is the Point-wise Mutual Information known as PMI predictor. It is used to determine the co-occurrence statistics of two terms in the collection, and is calculated by Eq 2.10 where $Pr(t_1, t_2|D)$ denotes the probability of two terms co-occurring

in the collection.

$$PMI(t_1, t_2) = log \frac{P_r(t_1, t_2|D)}{P_r(t_1|D)P_r(t_2|D)} \tag{2.10}$$

The avgPMI(q) and maxPMI(q) calculate the average and maximum PMI for all pairs of query terms. Queries that are best evaluated through retrieval methods which have taken into account term, proximity will have higher average PMI values. It should be mentioned that PMI does not work when the query consists of only one term.

Term specificity-based pre-retrieval QPP methods

One of the state-of-the-art statistical pre-retrieval query performance prediction methods relies on the specificity of the query. Specificity is determined through the distribution of the query terms over the corpus; the more specific the terms are in the query, the easier it will be to answer. The focus of this thesis is on this category of pre-retrieval query performance prediction methods. Therefore, we will elaborate more on the concept of specificity. Since our goal is basically to introduce a new specificity estimator, we explain more details about specificity metrics in section 2.2

2.2 Specificity and QPP

In this section, we cover specificity and its recent applications in Sections 2.2.1 and 2.2.4, respectively. Moreover, we have conducted a study on other available evaluation methods on specificity in Section 2.2.3. In the end, we present all specificity-based pre-retrieval query

performance prediction metrics in Section 2.2.2. Recently, Kangassalo et al. [17] utilized ElectroEncephaloGraphy (EEG) to explore the correlation between the specificity of terms and human brain activities. They tried to investigate why humans are able to select specific terms when they want to form their query with respect to their information need. Kangassalo et al. [17] monitored 15 different people's brain activities while they were reading specific and non-specific terms that are chosen randomly from Wikipedia. Their experiments in [17] showed that term specificity has a neural basis since the evoked brain activity is significantly different while reading specific or non-specific terms.

2.2.1 The Definition of Specificity

Over the last 100 years, many researchers have attempted to define *speci city* from a variety of aspects. While most of the studies determined a concept's specificity based on their related application, a few of the works focused on linguistic aspects. In 1958, Brown [18] defined the specificity or abstractness of a word or a concept by the number of known subordinate words it embraces. Later, Zhang [19] focused on defining the opposite concept, which is generality. Based on his work, a concept is general or unspecified if it does not specify certain details. Zhang exemplified friend as a general concept since details of what exactly constitutes a friend is not known. He also distinguished between the vagueness, generality, fuzziness, and ambiguity from a linguistic perspective, which are out of the scope of this thesis. We can consider all of those definitions as different generality measures because an expression that has more than one semantically unrelated meaning can also be counted as an unspecific one. Likewise, Allen et al. [20] described a document to be general when it addresses general things or concepts. Moreover,

in [4] specificity is interpreted as the level of detail in which a given term is represented. In this thesis, we agree to accept this definition since it matches our application. We interpret specificity as a statistical measure rather than a semantic property of terms, i.e., a term specificity is highly related to the number of documents to which it pertains in a collection. Orlandi et.al [21] defined an entities' specificity as the level of abstraction that an entity has in a common conceptual schema shared withf humans.

2.2.2 Specificity Metrics

There are multiple predictors to measure specificity; They include the Query Scope predictor (QS), predictors based on query term statistics, and the Simplified Clarity score (SCS(q)) [9].

Query Scope (QS). The Query Scope predictor determines the percentage of documents in the collection that include at least one of the query terms. A higher query scope signifies that there are multiple documents pertaining to the query, however it may be difficult to distinguish between the relevant and non-relevant results.

Term Statistics (IDF and ICTF). Term statistics is believed to be an intrinsic feature that reflects retrieval performance. Query terms metrics such as Inverse Document Frequency (IDF) and Inverse Collection Term Frequency (ICTF) [22] are used to show the correlation to the system performance. The higher the average value of these metrics over the query terms, the easier it will be to satisfy the query. By higher average we mean the query is composed of infrequent terms.

ICTF, the other query term measure, is measured by equation 2.11 where the total number

of all terms in the collection D is represented by $|D|$.

$$ictf(t) = log(\frac{|D|}{tf(t.D)}) \tag{2.11}$$

Simplified Clarity Score (SCS). The simplified clarity score determines the divergence between the simplified query language model and the collection language model. The calculation of SCS is given in equation 2.12 for query $q$with the length of $|q|$ when considering a query with each term appearing once in its entirety.

$$SCS(q) = log(\frac{1}{|q|}) + avgICTF(q) \tag{2.12}$$

Thus the SCS predictor considers the length of the query while determining the SCS score. All of the mentioned pre-retrieval methods are based on specificity, and they showed promising performance over several TREC collections of documents including robust04, ClueWeb09, ClueWeb12 and GOV2 [2]. We have compared some of the mentioned frequency-based specificity metrics as baselines in our work in Section 4.2.

2.2.3 Evaluation of Specificity

Specificity can be subjective and biased by personal experiences [21]. Therefore, we can not have an exact value for specificity, and there is always an estimation measure of specificity, which is accompanied by a certain percentage of error. Consequently, no absolute ground truth exists for evaluating specificity. However, hierarchies have been used as one of the decent gold standards in order to evaluate different levels of concepts specificity in both linguistic [23] and other

applications [24, 21] because the level of each concept in a hierarchy can be a suitable indicator of the concepts specificity. We will elaborate more on this intuition in Chapter 3. For example, Benz et al. [24] studied different levels of generality by identifying the hierarchical relationships between concepts. They leveraged several taxonomies and ontologies such as WordNet [25], Yago [26], DMOZ [27], and WikiTaxonomy [28] as the ground truth level of specificity. They aimed at using terms specificity to measure tag relatedness in social metadata and generality of terms using their four proposed categories of metrics, including frequency-based measures, entropy-based measures, centrality measures, and statistical subsumption. In addition, Orlandi et al. [21] utilized DMOZ taxonomy as one of the baselines to assess their proposed method of specificity estimation. Based on the same idea, we utilize the level of concepts in Wikipedia Category Hierarchy [29] and DMOZ as the ground truth value for concepts specificity.

While hierarchies always show a fair agreement with human judgments [21, 24], some works still prefer to employ human judgment as of the gold standard [21]. One of the common ways of making the gold standard for evaluating specificity metrics is to set the ground truth as human perception and ask different people to rate different concepts as a numeric value for example from 1 (the most specific one) to 10 (the most generic one) or to classify the concepts into specific and generic classes. However, generating the gold standard like this can be inefficient, expensive, and impractical in some cases.

2.2.4 Applications of Specificity

Term specificity can be used in several applications, especially on the web, including user interest modeling, social media personalization, query performance prediction, among others. Orlandi

et al. [21] utilized specificity in order to recommend more relevant entities to users on the social web. They claimed that while their proposed specificity methods can be calculated and updated in real-time, it is also knowledge and domain independent. Benz et al. [24] employed a Linked Data graph and analyzed the predicated connecting entities the graph. They considered the relation between incoming and outgoing predicates as a gauge for specificity.

2.3 Neural Embeddings

The goal of this thesis is to propose a set of specificity metrics that preserve the semantic aspect of the terms in order to enhance the quality of the query performance prediction task. These are sufficient reasons to exploit embeddings in our proposed approach. In this section, we elaborate on the pre-trained embeddings that are used in this thesis, including Hierarchy Category Embedding (HCE) [29], FastText Pre-trained embedding [30], and Word2Vec [31].

Neural embeddings maintain interesting geometric properties between embedded terms [32] where the direction and magnitude of the relationship between the vector representation of terms derived from embeddings are meaningful. Such geometric relations are manifested by how term vectors are distributed in the embedding space. It is demonstrated that semantic relationships are commonly preserved in vector operations on word vectors [31]. For example, applying these operations on the vector representation of terms Berlin, Germany, and France will result in a vector very close to *Paris*' vector as in equation 2.13.

$$[\overrightarrow{Berlin}] - [\overrightarrow{Germany}] + [\overrightarrow{France}] \approx [\overrightarrow{Paris}] \tag{2.13}$$

This suggests that relations among embedded word vectors are almost semantically meaningful [33].

A document can be represented in many different ways according to the applications. Among all the document representations, word embedding is one of the most prevalent ones. Word embeddings are valuable because this vector-based distributed representation of words can conserve the semantic side of words in its representation unlike the vocabulary indexing of a document. Also, word embedding distinguishes itself from other document representations by its ability to preserve the dependencies among the words.

2.3.1 Neural Embeddings in Information Retrieval

The geometric relations among neural embeddings which leads to semantic of words vectors, can also describe a fair correlation with term specificity . The concentration of this thesis is to utilize the correlation among term specificity and embedded words in order to predict query performance prediction.

Neural methods have been used in retrieval for purposes such as query expansion, query classification, ranking, and text classification [6]. It has been empirically shown that while neural embedding-based methods cannot solely outperform existing retrieval methods, their systematic interpolation can lead to improved retrieval performance specifically on hard queries [34].

Whereas some recent work proposed training query performance prediction models using neural methods on a host of signals [7]. To the best of our knowledge, there is only one recent work that utilizes the neural embedding-based representation of query terms for performing

query performance prediction [8]. Their approach works the best for post-retrieval QPP, and as it is shown in Chapter 4, however, it does not perform well in the pre-retrieval QPP task. We define various specificity metrics based on pre-trained neural embeddings and systematically explore their effectiveness on query performance prediction.

2.3.2 Pre-trained Embeddings

In order to compute word vectors, a larger text corpus is needed. Depending on the corpus, the word vectors will capture different information. Every sequence of terms in the one-hot vector representation will be fed in a neural network as input and the output of them will be the next following term. The main idea behind it is that you train a model on the context of each word, so similar words will have similar numerical representations. The architecture of the network (hidden layers and softmax) is out of scope of this thesis since we are working with pre-trained word embeddings.

In this thesis, we have used three different pre-trained embeddings in order to compare our proposed metrics on them and also to investigate the robustness of our method to different embeddings. In the following, we will explain all the pre-trained embeddings that we have used in this study, including Hierarchy Category Embedding (HCE) [29], Fast Text embeddings [35], and Word2Vec embeddings [36, 37].

Pre-trained Hierarchy Category Embedding (HCE)

The data set, which is collected from Wikipedia in 2015, has been used as training data for the model in [29]. The advantage of this model compared to other available pre-trained embedding is

that while other pre-trained embeddings are trained on the unstructured data, they incorporate the hierarchical relationship between Wikipedia entities and categories in order to train their 400-dimension embedding model. The authors kindly provided their pre-trained embedding vectors to us.

FastText Pre-trained Word Embedding

In [35], the authors represent words as a bag of n-gram characters in a large unlabeled corpus to train vector representation of terms. While other pre-trained embedding models suffer from lack of word representation for unknown words in the training data, fastText is the first model which can compute word representation for words that did not appear in the training data. Facebook has made 300-dimension word embedding pre-trained models available for 294 languages. Fast text has different versions that been trained on either Wikipedia or Common Crawl [30]. In this work, we used the 1 million word vectors trained on the Wikipedia 2017 dump [1].

Google News Pre-trained Word Embedding

This Pre-trained embedding, also known as Word2Vec[36, 37], has been trained on Google News corpus which consists of roughly 100 billion words. More than 3 million words have been embedded into vectors with the size of 300.

[1] `https://dl.fbaipublicfiles.com/fasttext/vectors-english/wiki-news-300d-1M.vec.zip`

2.4 Summary

In this chapter, we reviewed the state-of-the-art works to predict query performance prediction without having access to the retrieval result. Also, we over-viewed specificity term definitions and its application in the web domain. Besides, we elaborated on word embeddings and its applications in information retrieval. Further, we explained all the three pre-trained embeddings that we utilized in the implementation step of this thesis.

In the following chapters, we are going to propose new specificity metrics based on the geometric properties of neural embeddings. Later on, we are going to evaluate the proposed specificity metrics for ebility to support the query performance prediction task.

Chapter 3

Proposed Approach

It is now well understood that the performance of retrieval models is not always consistent across different queries and corpora and there are often some queries that have lower performance, often referred to as hard or difficult queries [38]. As such, the area of query performance prediction (also known as, difficulty estimation) is concerned with estimating the performance of a retrieval system for a given query. This is specifically useful because it will allow the adoption of alternate retrieval strategies when such hard queries are identified. As previously stated in Section 2.1, query performance can be predicted based on different attributes of queries. In this thesis, our focus is on specificity-based pre-retrieval query performance prediction methods.

The objective of our work in this thesis is to investigate the possibility of estimating term specificity by utilizing neural embedding-based representation of terms. We explore how the geometric properties of neural embeddings can be exploited to define individual and collective *speci city* metrics. This suggests that by considering the associations between term vectors in the neural embedding space, we can go beyond frequency-based metrics and derive other

measures of specificity which are not corpus dependent. More specifically, we propose to measure *term speci city* based on the distribution of terms that form the neighborhood of a term of interest in the embedding space. Our metrics are based on the intuition that a term that is closely surrounded by other terms in the embedding space is more likely to be specific while a term with less nearby terms is more likely to be generic. In the context of our intuition, we define four groups of specificity metrics based on their characteristics including (1) *Neighborhood-based metrics*, (2) *Graph-based metrics*, (3) *Cluster-based metrics* and (4)*Vector-based metrics*.

We propose and gather several metrics in each of the four class of specificity metrics. The following contains the description of each of the proposed metrics with respect to the idea behind them. In total, we have inspected 25 different metrics that we will assess and compare them in Chapter 4. In addition, we will evaluate them based on their performance in pre-retrieval query performance prediction on TREC collections [11, 10, 12, 39].

3.1 Preliminaries

This thesis focuses on how vector representations of terms within a given embedding space, i.e., geometric properties of neural embeddings, could be used to define appropriate metrics for estimating term specificity. Our intuition is that the local neighborhood of a term in a given embedding space can be used to derive indicators of the terms specificity. Based on the fact that, in an embedding space, two semantically related terms have similar embedding vectors (as measured, e.g., by cosine similarity of their vectors). In order to organize the similarities among embedding vectors, we define a local neighborhood for an embedding called ε-Neighborhood in section 3.1.1 and a more complex neighborhood known as ego network in section 3.1.2.

3.1.1 ε-Neighborhood

We select the local neighborhood surrounding an embedding vector of term t_i, by retrieving a set of highly similar terms to t_i. Formally, let $\mu(t_i)$ be the degree of similarity of the most similar term to t_i in the embedding space. We select the ε-Neighborhood of t_i, denoted as $N\varepsilon(t_i)$, as follows:

$$N_\varepsilon(t_i) = \{t_j : \frac{\upsilon_{t_i}.\upsilon_{t_j}}{\|\upsilon_{t_i}\|\|\upsilon_{t_j}\|} \geqslant \varepsilon \times \mu(t_i)\} \tag{3.1}$$

Simply put, we first calculate the cosine similarity between the embedding vector of t_i and other terms embedding vectors in the embedding space and then the terms with a similarity higher than $\varepsilon \times \mu(t_i)$, are selected as the ε-Neighborhood of term t_i.

3.1.2 Ego Network

In order to examine the inter-term associations in the neighborhood of t_i, we conceptualize the embedding space surrounding a term by defining an *ego network* where the term of interest forms the *ego* and is contextualized within a set of *alter* nodes, which are other terms that are closely positioned around the term of interest in the embedding space. We formalize the notion of an *ego network*, which is based on term similarities within the neural embedding space, as follows:

Definition 2. ***(Ego Network)*** *An ego network for term t_i, denoted as $\xi(t_i) = (\mathsf{V}, \mathsf{E}, g)$, is a weighted undirected graph where $\mathsf{V} = \{t_i\} \cup N_\varepsilon(t_i)$, and $\mathsf{E} = \{e_{t_i,t_j} : \forall t_i, t_j \in \mathsf{V}\}$. The function $g : \mathsf{E} \rightarrow [0, 1]$ is the cosine similarity between the embedding vectors of two incident terms of an*

edge e_{t_i,t_j}, i.e., v_{t_i} and v_{t_j}. We re ne $\xi(t_i)$ by pruning any edge with a weight below $\varepsilon \times \mu(t_i)$.

In the above definition, we propose to build an ego network for term t_i such that t_i is the ego node and is connected directly to other terms only if the degree of similarity between the ego and its neighbors is above a given threshold. For instance, assuming 'Pizza Hut' is the ego and $\varepsilon = 0.9$, given the fact that 'Subway (restaurant)' is the most similar term to the ego with a similarity of 0.82, the immediate neighbors of the ego node will consist of all the terms in the embedding space that have a similarity above 0.738 to 'Pizza Hut'. Figure 3.3 shows the ego networks for the specific term 'Pizza Hut' and the generic term 'Food and Drink'. As shown in the figure, for example, the immediate neighbors of Pizza Hut include terms such as 'Subway (restaurant)', 'KFC', '7-Eleven', 'Krispy Kreme', among others. In this ego network, the neighbors are also connected to each other if their similarity is more than 0.738, which is $\varepsilon \times \mu(t_i)$.

3.2 Specificity Metrics

Based on the developed ε-neighborhood and the ego network for a given term t_i, we propose to measure the specificity of the ego term. More concretely, based on our intuition that the characteristics of the local neighborhood of a term are potential indicators of its specificity, we measure term specificity as a function of the structure of the terms ego network. In the following, first we propose four categories of unsupervised specificity metrics, including:

1. Neighborhood based, which are based on the idea that a specific term is likely to be associated with a large number of specific terms in its neighborhood.

2. Graph-based metrics, which consider the structure of the ego network to estimate the specificity of the ego node.

3. Cluster-based metrics, which consider the term clusters around a term as potential indicators for the specificity of the term.

4. Vector-based metrics, which consider the term embedding vector's characteristics as of a specificity measure.

We propose a supervised method that incorporates all these metrics, as features, in a learning to rank model for estimating the specificity of a term based on neural embeddings.

3.2.1 Neighborhood-based

Neighborhood-based metrics only consider the connections between the ego node and its immediate neighbors. Our intuition is that as highly specific terms express precise semantics, they have a high likelihood of being surrounded, in the embedding space, by a higher number of specific terms compared to generic terms. For example, as in shown in Figure 3.1 the specific term 'Pizza Hut' which refers to a fast-food brand is highly similar to other terms referring to other fast-food chains such as 'KFC' and 'Burger King'. However, since a generic term, e.g., 'Food and Drink', is often related to many different terms with diverse senses, it would end up having weaker relationships with these diverse neighbors. In other words, generic terms are likely to be related to other generic terms that originate from various domains, and while the relation (i.e., semantic relatedness) between terms does exist, it is notably weaker than in the case of specific terms that are highly semantically related to one another. Therefore,

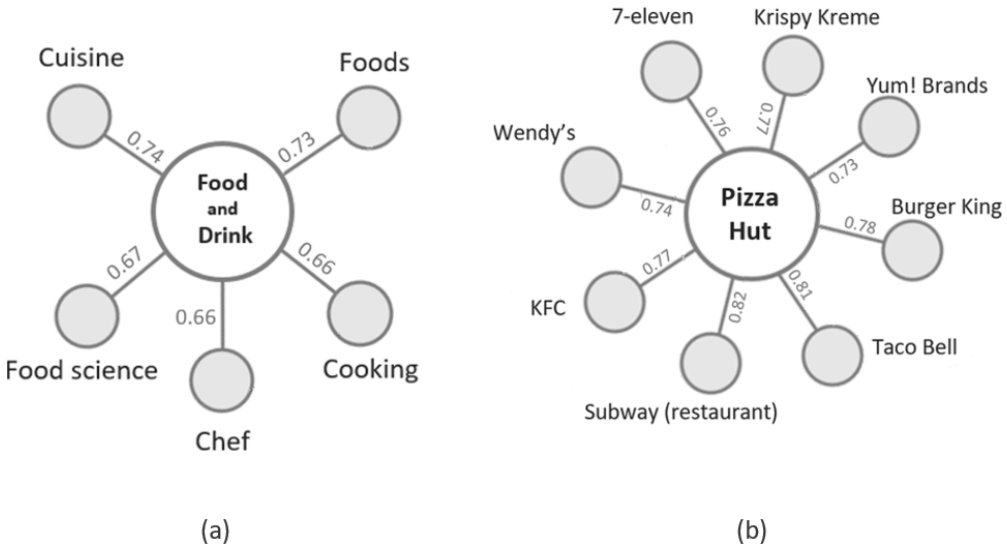

Figure 3.1: ε-neighborhood for: a) Food and Drink, b) Pizza Hut

by computing and considering the strength of the connection (edge weight) of a term with its immediate neighbors, it is possible to differentiate between specific and generic terms. Based on this idea, we define seven ego-node based metrics summarized in Table 3.1.

Neighborhood Size (NS). This metric measures specificity based on the idea that given a constant similarity threshold, specific terms are surrounded by more similar terms compare to generic terms. We support this notion by comparing the neighborhood of a generic concept 'Food and Drink', and a specific one 'Pizza Hut' in Figure 3.1.

Assuming ε is 0.9, Neighborhood Size of the generic concept "Food and Drink" is 5 while there are 8 neighbors in the 'Pizza Hut' neighborhood. In general, we can conclude that the size of the ε-neighborhood of term t_i relates directly to the specificity of t_i.

Weighted Degree Centrality (WDC). In network theory, the centrality of a node in a

network is usually a hint to the importance of the vertex [40]. Having said that, centrality can be mulled over as a degree of abstractness since more unique terms are more potent. Degree centrality is the simplest of the node centralities, which is measured only by using the local structure around nodes. In a weighted network, degree centrality has generally been extended to the sum of weights. If we look at the ε-neighborhood as a network, calculating Weighted Degree Centrality (WDC) of node t_i in this network can lead to proper measurement of t_i specificity. For instance in Figure 3.1, the WDC of the specific term "Pizza Hut" is 6.19 whereas the 'Food and Drink' weighted degree centrality in its ε-neighborhood is equal to 3.46. Hence, higher WDC can be interpreted as higher specificity.

Median Absolute Deviation (MAD). Median Absolute Deviation (MAD) represents how spread out a set of data is. MAD is more robust to outliers compared to other measures of spread such as variance. We prefer to utilize median absolute deviation instead of mean absolute deviation since median is more robust to outliers compared to mean. Eventually MAD it calculates how far each data point is from its median.

Since our work is based on the intuition that specific concepts have more similar neighbors compared to a generic one, we can present a hypothesis that the similarity in a neighborhood of a more specific concept should be less dispersed than a generic one. Therefore, it should have less MAD value. It might be a misunderstanding that MAD value in Figure 3.1 is higher for the more specific concept of Pizza Hut but it should be mentioned that in Figure 3.1, ε has been chosen at a very high-value rate; subsequently the number of neighbors are not too high. However, if we optimize the hyper-parameter i.e., ε, there would be an appropriate number of neighbors in the ε-neighborhood to make a proper judgment of the MAD metric for measuring

specificity.

Neighborhood Variance (NV). The variance of similarity of neighbors of term t, to term t could be an indicator of specificity. This is true if we assume that the dispersion of neighbors in a neighborhood is a gauge for specificity. Consider the two neighborhood in Figure 3.1, the variance in the neighborhood of the more generic term is 0.0012, while the variance for the other specific term is 0.0007. This can be justified by the intuition that although the number of neighbors in the more specific neighborhood is higher, the neighbors of the specific concept are more similar to each other. Consequently, the variance of their similarity should be less.

Most Similar Neighbor (MSN). As stated before, the similarity in a neighborhood of term t_i is correlated with its abstractness. This can also be inferred that if we consider the most similar neighbor M_{t_i} to term t_i, the similarity between M_{t_i} and t_i is highly related to how specific t_i is. This fact also is authenticated in Figure 3.1 as the most similar neighbor to 'Pizza Hut' is 'subway (restaurant)' with similarity of 0.82 but the most similar neighbor to 'Food and Drink' which is more general concept is "cuisine" with similarity of 0.74 . In general, we can state that general concepts have lower MSN compared to specific concepts.

Neighborhood Vector Similarity (NVS). A different approach to measuring the similarity of term t_i to its neighborhood is to treat the neighborhood as one unified concept and then calculate the cosine similarity of the neighborhood and the term t_i. To do so, first, we need to unify all the neighbors in the neighborhood. We propose to consider all the embedding vectors in the neighborhood and add all of the neighbors' vectors embeddings in the neighborhood to make a vector V_{N_ε} that is made of summing up all the neighbors in the neighborhood. Then, we can calculate the similarity between V_{N_ε} and t_i. The more these two vectors are close

together, the more specific the term t_i is. This metric is impressive since it considers specificity from the geometric aspects of embedding and their distribution of an embeddings' most similar vectors in the space.

Neighborhood Vector Magnitude (NVM).Similar to the background notion of NVS, in this proposed specificity metric, we unify the neighbors by adding all the neighbors' vectors to each other to get V_{N_ε}. However, to calculate specificity via the Neighborhood Vector Magnitude (NVM) metric, we should measure the size of V_{N_ε} vector. The idea behind this metric is presented in a 2-D space (we reduced dimension for drawing purposes) in Figure 3.2. If t_i is a general concept, then the neighbors are less similar to each other compared to neighbors of t_j which is a less general concept as shown in the boxes (a) and (b) in 3.2. Therefore, if we add all of the neighbors' vectors (as shown in the box (c) and (d) in Figure 3.2), we would end up having V_{N_ε} (the red dotted vectors) whose size could be an indicator of specificity. Because, if a concept is specific then the neighbors are more similar (They are in the same direction in Figure 3.2) so we will end up having a V_{N_ε} vector with a greater size. We should mention that it is assumed that all the vectors are normalized.

Table 3.1: The set of Neighborhood-based specificity metrics to estimate the specificity of the term t_i

ABV	**Metric Name**	**Description**
NS	Neighborhood Size	The number of terms in -neighborhood of term t_i
WDC	Weighted Degree Centrality	The average weight of edges directly connected to t_i
MAD	Median Absolute Deviation	The Median Absolute Deviation (MAD) of weight of edges directly connected to t_i
NV	neighborhood Variance	The variance of weight of edges directly connected to t_i
MSN	Most Similar neighbor	The maximum weight of edges directly connected to t_i
NVS	Neighborhood Vector Similarity	How similar a term is to the vector that is made of all the terms neighbors.
NVM	Neighborhood Vector Magnitude	Size of neighbors vectors

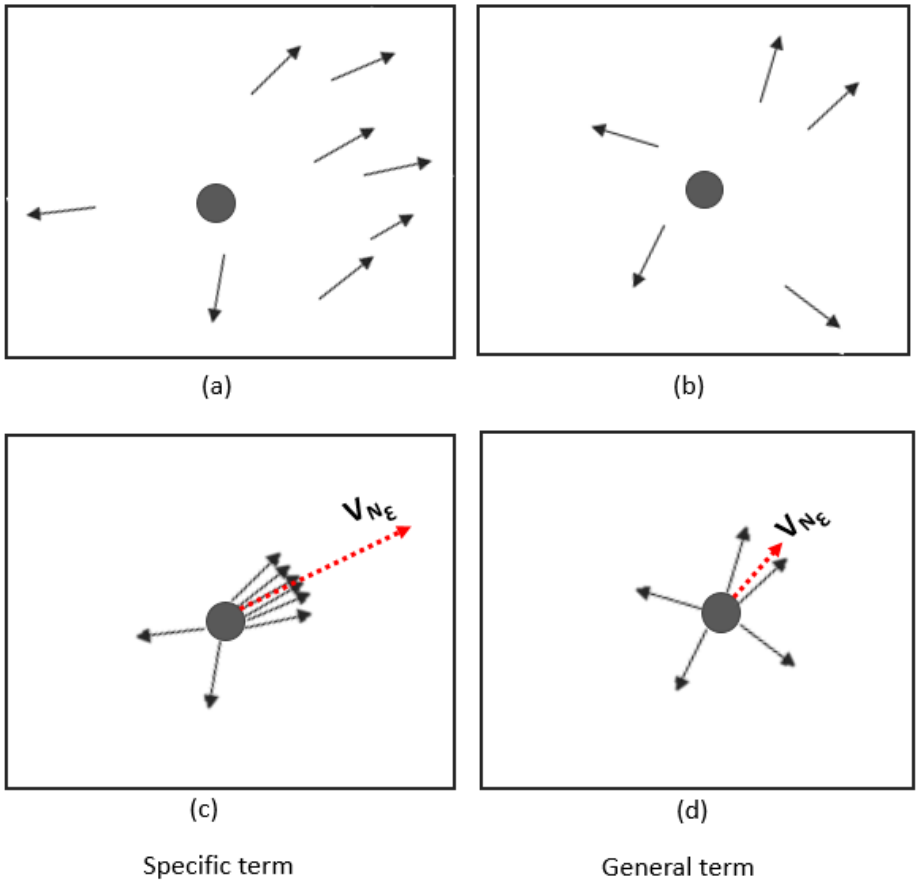

Figure 3.2: Neighborhood Vector Magnitude (NVM): Boxes (a) and (b) represent the neighbors of two different concept. The former is more specific than the latter. Consequently, their neighbors in the embedding space are also similar (here in the same direction). Adding all the neighbors vectors would result in the red dotted vectors. We hypothesize that the magnitude of the more specific concept neighborhood is greater than the generic one.

3.2.2 Graph-based metrics

While neighborhood-based metrics only focus on the connections of neighborhood terms with the ego node, in graph-based metrics, we take all the connections in the ego-network into account. Our intuition for defining these metrics is that the denser an ego-network is, the more specific the ego term would be. In other words, not only a specific term is surrounded by a higher number of neighbors in the embedding space, but also its neighbors are also highly similar to each other. Hence, we might want to consider the neighbors' relationship together as well as considering the neighbors' relationship to term t_i. We formalize these intuitions by defining ego-network based metrics summarized in Table 3.2. Also, we represent the ego network that the graph-based metrics are all working on in Figure 3.3.

We can categorize the graph-based metrics by their intuition into two categories:

1. Node importance metrics that measures the importance of a node in a graph and

2. Edge importance metrics that focus on the importance of edges and considers the relationships in the graph.

Node Importance Metrics

Based on [40], centrality measures the importance of each node in a graph. Therefore, we consider three most common centrality measures, including Betweenness centrality, Degree centrality, and Closeness centrality in our graph-based specificity metrics. Moreover, PageRank [41], is one other way of measuring the importance of nodes in a graph. We utilize as well in order to measure the importance of nodes in a graph.

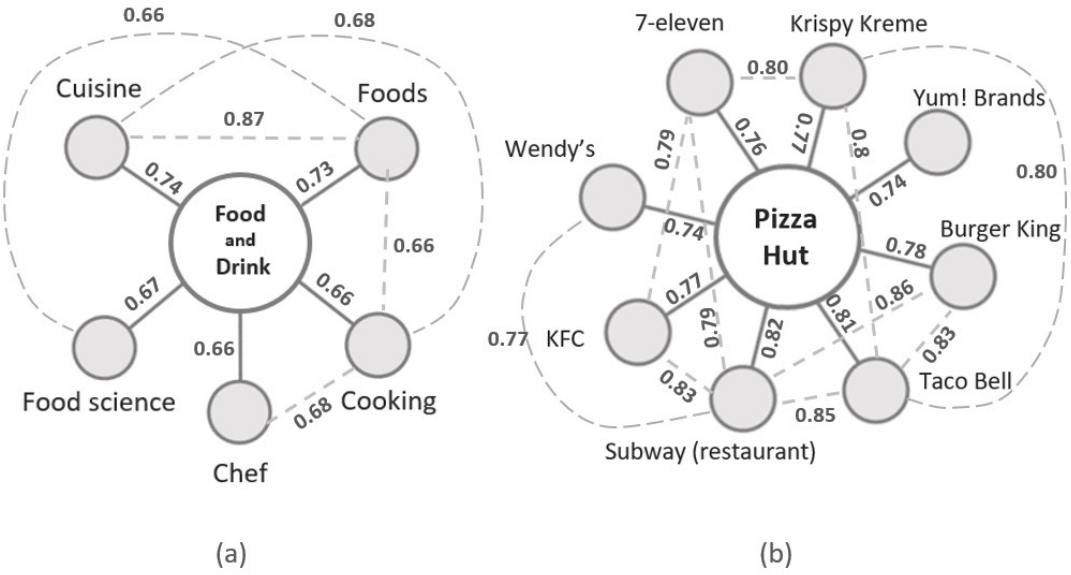

Figure 3.3: Ego-Network for: a) Food and Drink, b) Pizza Hut

Betweenness Centrality (BC). Given Π the set of all the shortest paths among every pair of nodes in V in graph $\zeta(t_i)$, for each vertex υ , the betweenness centrality of υ is defined as the number of paths in Π that include υ. In simple words, betweenness centrality of a node υ is the number of the shortest paths in the graph $\zeta(t_i)$ that the node υ has participated in them. If a node is engaged in more shortest paths, it means it is a more critical node or more central node. Since we interpreted the importance of a node in a graph to its specificity, therefore we can infer that the node is more specific if it has higher betweenness centrality.

Degree Centrality (DC).Previously we explained the notion of Degree centrality while explaining Weighted Degree Centrality (WDC). This metric distinguishes itself from WDC as we are measuring the node centrality in a more complex network, i.e, ego network, compared to the ε-neighborhood. Degree centrality, or the number of links incident upon the node υ, can be interpreted in terms of the importance of the node.

Closeness Centrality (CC). Closeness centrality of node v in a graph $\zeta(t_i)$ is the average length of all the shortest paths between v and all the other nodes in V. If an ego node ζ_i in graph $\xi(\zeta_i)$ has a lower closeness centrality that another ego node ζ_j in graph $\xi(\zeta_j)$, it is more central. It can be construed that the role of ζ_i in $\xi(\zeta_i)$ is more important that the role of ζ_j in $\xi(\zeta_j)$. Hence, ζ_i is a more specific concept.

Inverse Edge Frequency (IEF). In Section 2.1.6, we described how IDF has been considered to be an applicable indicator of specificity in the literature. Based on this perception, we generalize the idea of IDF into a graph. Instead of Inverse Document frequency in a corpus, we calculate Inverse Edge Frequency in a graph as follows :

$$IEF(t_i) = log(\frac{|\mathsf{E}|}{|\mathsf{E}_{t_i}|}) \tag{3.2}$$

As stated in Definition 2.2, E is the set of all edges in the graph $\xi(t_i)$ and E_{t_i} is set of all edges in the graph $\xi(t_i)$ that include the node t_i.

Page Rank (PR. Intuitively, PageRank is a link analysis method, and allocates a numerical weighting to every component of a hyperlinked set of documents, for example, the World Wide Web, to quantify its relative importance within the set. A PageRank results from the calculation dependent on the webgraph which are made by pages as hubs and hyperlinks as edges. The rank worth demonstrates a significance of a specific page. A hyperlink to a page is considered as a vote towards importance of the page. The PageRank of a page is characterized recursively and relies on the number and PageRank metric of all the incoming links. A page that is connected to numerous pages with high PageRank values receives a high position itself. The PageRank for

t_i in the defined ego network can imply as to the importance of the ego node or the specificity of ego node.

Edge Importance Metrics

This group of metrics is mainly focused on two important concerns: (1) the relationship among neighbors and (2) the relationship among neighbors and the ego node. The relationship among neighbors and the ego node is different from the neighborhood-based specificity metrics since the network that we are working on for this study is a different one which can be explained by comparing Figure 3.1 and Figure 3.3.

Edge Count (EC). The denser a graph is, the more number of edges it has. Accordingly, $|\mathsf{E}|$ in graph $\xi(t_i)$ can measure specificity. This metric intuition is analogous to NSmetric but instead of working on the neighborhood, we are calculating the same idea in a more complex graph, ego-network, by considering the bonds within neighbors.

Edge Weight Sum (EWS). Specific concepts have stronger links with their neighbors. As the neighbors are more similar to each other, we would achieve a higher total summation of weights in the graph of more specificity concept compared to a less specific one.

Edge Weight Average_ego (EWAe). The notion behind this metric is the same as EWS. However, this metric is normalized meaning that we are calculating the average number of weights in the ego network $\zeta(t_i)$. Lower average weight in $\zeta(t_i)$ implies more generality of the term t_i.

Edge Weight Average (EWA) . This variation of EWAe, which is EWA, neglect all the edges that are connected to the ego node t_i in order to focus on the neighbors' relationship with

each other. Because the more close to each other the neighbors are, the more specific the ego node t_i is.

Edge Weight Max_ego (EWXe). EWXe attempt to find the most similarity or the edge with the maximum weight in the ego network. This metric is comparable to MSN; however, the networks are different. The higher the maximum weight of the ego network, the more specific the ego term t_i.

Table 3.2: The set of Graph-based specificity metrics considered to estimate the specificity of the term t_i

ABV	**Metric Name**	**Description**
BC	Betweenness Centrality	Betweenness centrality of node t_i in the ego network
DC	Degree Centrality	Degree centrality of node t_i in the ego network
CC	Closeness centrality	Closeness centrality of node t_i in the ego network
IEF	Inverse Edge Frequency	The Number of edges in the ego network divided by the number of edges connected to node t˙i in the ego network
PR	PageRank	The value of PageRank of term ti in the ego network
EC	Edge Count	The number of edges in the ego network
EWS	Edge Weight Sum	The sum of edge weights in the ego network
EWAe	Edge Weight Avg˙ego	The average of all edge weights in the ego network
EWA	Edge Weight Avg	The average of all edge weights in the ego network except the ones connected to ego
EWXe	Edge Weight Max˙ego	The minimum edge weight in the ego network

3.2.3 Cluster-based Metrics

These metrics are based on the idea that the characteristics of term clusters within the neighborhood of a given term are potential indicators of its specificity. Therefore, to extract the term clusters around t_i, we apply a clustering algorithm, such as the K-means algorithm, to the embedding vectors of terms in its -neighborhood, i.e., $N_\varepsilon(t_i)$, which results in K clusters for t_i, i.e., $C_{t_i}^1$, ..., $C_{t_i}^K$. Then, we estimate the specificity of the term t_i by calculating the variance of the number of elements in the obtained clusters. As mentioned before, a generic term is more likely to be related to many terms from different domains. If we cluster the neighborhood terms of a given term, we may expect that each cluster will be associated with one domain. Low variance of the number of elements in the clusters shows that there is no dominant cluster in the neighborhood of the term. Consequently, it is probably a more generic term.

The association between the term clusters can also be considered to be an indicator of specificity. Each cluster is defined with its *centroid* c, which is a vector in the embedding space that indicates the center of the cluster. Therefore, for term t_i, given its term clusters, i.e.,$C_{t_i}^1$, ..., $C_{t_i}^K$, we define its centroid network, denoted $\xi(t_i)$, as follows:

Definition 3. *(Centroid Network) A centroid network for term t_i, denoted as $\zeta(t_i) = (\mathsf{V}, \mathsf{E}, g)$, is a weighted undirected graph in which V includes the centroid points of the term's clusters $C_{t_i}^1, ..., C_{t_i}^K$, and $\mathsf{E} = \{e_{c_i,c_j} : \forall c_i, c_j \in \mathsf{V}\}$. The weight function $g : \mathsf{E} \rightarrow [0, 1]$ is the cosine similarity between the vectors of two centroid points of an edge e_{c_i,c_j}, i.e., v_{c_i} and v_{c_j}.*

Edge Weight Centriod (Avg, Min, Max). Based on the centroid network definition, we can define specificity as follows: The more the term clusters of a given term are similar, the

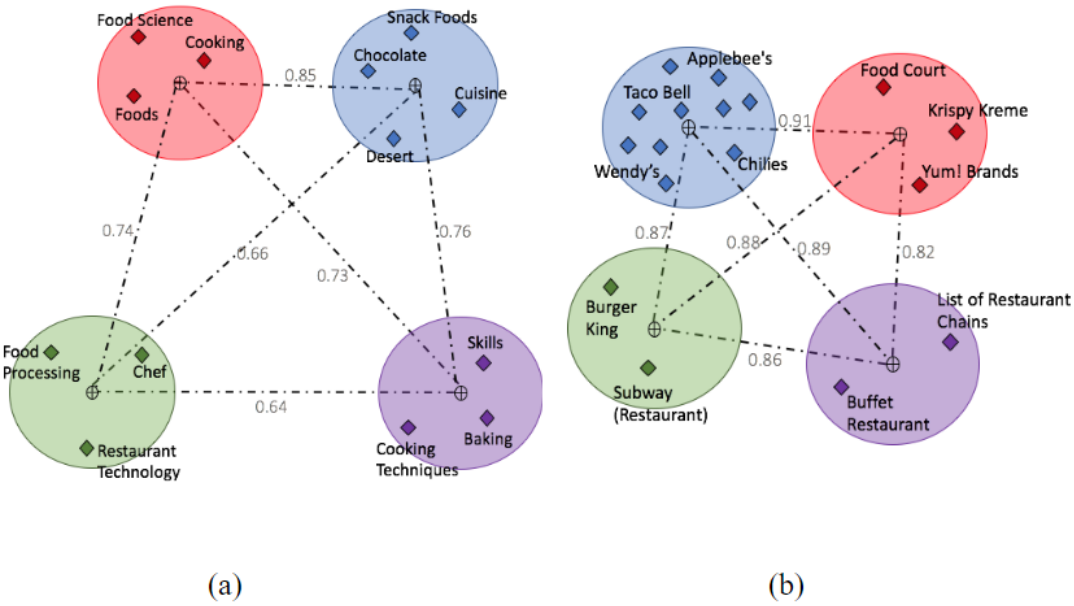

Figure 3.4: The term clusters and the centroid network for (a) the generic term 'Food and drink' and (b) the specific term 'Pizza Hut'. $\oplus$ represents centroid of each cluster. The edges in the centroid network is based on the distance between two centroids.

more specific the terms would be. Therefore, edge weights in the centroid network play a crucial role in estimating specificity because they show how clusters are distributed in the embedding space. Thus, we define three metrics by aggregating edge weights in the centroid network using min, max, and average functions. To compare the cluster-based metrics of a general term, e.g., 'Food and Drink' to a more specific one, such as 'Pizza Hut', we illustrate their clusters and centroid networks in Figure 3.4. Let K be 4, the variance of the number of elements of the clusters of the generic term 'Food and Drink' is 0.5, which is far less than that of 'Pizza Hut', which is 6. In addition, Figure 3.4 shows that clusters of a specific term are positioned close to one another in the embedding space, which implies higher semantic relatedness. Thus, the average edge weights of a centroid network of 'Food and Drink' is 0.73, which is less than that

of the average edge weights of Pizza Hut at 0.87. The details of the proposed cluster-based metrics are provided in Table 3.3.

Cluster Elements Variance (CEV). According to Figure 3.4 and the intuition behind it,the number of members in each cluster has, the potential to be an indicator of specificity. We elaborate on this idea by using following example: given that we have a constant number of clusters, if a term is specific, it should have one dominant cluster in its neighborhood, such as centroid network (b) in Figure 3.4. The blue cluster has more number of members compared to the other clusters in (b) network. Whereas, if a term is a vague one, such as sample (a) in Figure 3.4 can have multiple meanings; therefore, the number of elements in the clusters should be more balanced. We can measure the distribution of members in the clusters by the variance. Thus, low variance among several members in the clusters show that the word is an ambiguous one while high variance represents the unbalanced distribution of members in the clusters and therefore there exists a dominant cluster and a specific term.

In Figure 3.4, the CEV ('Food and Drink'), (a) is variance of [3,4,3,3] which is 0.1875 and the CEV('Pizza Hut'), (b), is variance of [10,3,2,2] which is 11.18. The more specific term, 'Pizza Hut', has the higher CEV. Thus, higher CEV is an indicator of higher specificity.

Non-Specific Clustering (NSC). So far, we have considered clustering in the neighborhood of a term in an embedding space with a defined number of clusters. In the NSC metric, unlike EWAc, EWNc, EWMc, and CEV, we apply clustering without pre-defining the number of clusters. We use hierarchical clustering to cluster the neighbors, and then the number of the generated clusters can be used as a specificity metric. Because a higher number of clusters shows different concepts around the term t, which means term t is a more general concept.

Table 3.3: The set of Cluster-based specificity metrics considered in our work. The metrics estimate the specificity of the term t_i

ABV	**Metric Name**	**Description**
EWAc	EdgeWeight Avg˙centroid	The average edge weight in the centroid network of the term ti
EWNc	Edge Weight Min˙centroid	The minimum edge weight in the centroid network of the term ti
EWXc	Edge Weight Max˙centroid	The maximum edge weight in the centroid network of the term ti
CEV	Clusters Elements Variance	The variance of the number of elements in the extracted clusters of the ego network
NSC	Non-Specific Clustering	Number of clusters can we find among the neighbors

3.2.4 Vector-based metrics

If we consider the featurized vector representation of words, we can assume that high values in each element of the vector means that this word is highly dependent on this feature. So, we might conclude that if a vector contains lots of elements with high values or very low values, this means that it is associated with those features (or not associated with those features at all). Both of this information could be helpful as they can add information toward the word by emphasizing or omitting a feature. There are existing work in the literature that support the idea that the norm of word vectors can correlate with some statistical features such as word frequency in the trained corpus [42]. We define Vector-based metrics to explore whether the properties of the embedding vector can be an indicator of any statistical feature of the words.

Based on the explained intuition, we define three metrics Sum Embedding (SE), Sum Em-

Table 3.4: The set of Vector-based specificity metrics considered in our work. The metrics estimate the specificity of the term t_i

ABV	**Metric Name**	**Description**
SE	Sum Embedding	Sum of an embedding vector elements
SEA	Sum Embedding Absolute	Absolute Sum of a vector elements
VE	Var Embedding	Variance among embedding vector

bedding Absolute (SEA), and Variance Embedding (VE in Table 3.4. The result of the proposed metric will be discussed in the next chapter, and we will experiments whether the embedding vector elements can mean anything independent or not.

3.3 Summary

In this chapter, we have proposed four different groups of specificity metrics for the neural embedding representation of words, namely (1) Neighborhood-based metrics (2) Graph-based metrics (3) Cluster-based metrics and (4) Vector-based metrics. We discussed the intuition behind each group, and we propose more detailed metrics in each group. In total, we proposed 25 different metrics. In the next chapter, we will report the results of the experiments that have been conducted on these metrics, and we will evaluate them in the application of query performance prediction to see which of the proposed metrics perform well for this task.

Chapter 4

Evaluation

Our work in this thesis is concerned with the design of efficient metrics for pre-retrieval QPP based on pre-trained neural embeddings. Existing work in the literature [43, 9] have already shown that measures of term specificity are suitable indicators of query difficulty, i.e., more specific terms are more discriminative and are hence easier to handle as queries while more generic terms are less distinct and as such form harder queries. For this reason, our objective is to define QPP metrics based on our intuition for term specificity, which we define by relying on geometric properties of neural embeddings.

Having said that, the first step in our work should be introducing specificity metrics (Chapter 3). Using embeddings, we proposed four different categories of specificity-based metrics, including neighborhood-based metrics, graph-based metrics, cluster-based metrics, and vector-based metrics. We aim to use these proposed metrics to enhance the performance of query difficulty estimation.

It should be noted that as mentioned in Section 2.2.4, specificity metrics can be used in

other applications as well as query performance prediction. Therefore, it is beneficial to assess specificity metrics directly before evaluating them in a special application such as QPP. We define the *speci city metrics evaluation* problem as ranking different concepts with a variety of specificity range. However, we need a gold standard with a proper specificity-based ranking of different concepts. Similar to [24, 21], for evaluating the specificity metrics directly, we employed hierarchies as the gold standard for specificity because the level of each concept in a hierarchy could be a potential indicator of specificity.

Given a hierarchy of different concepts, the length of the shortest path from concept c to the root of hierarchy, can be considered as the specificity of c. Therefore, in Section 4.1.1, we introduce two different test collections from two comprehensive hierarchies, i.e, Wikipedia and DMOZ [44, 27] for the purpose of assessing the specificity metrics. The results of evaluation on all the 25 proposed metrics from Chapter 3 on the mentioned test collections, are presented in Section 4.1.3. In addition, in order to investigate the robustness of our proposed specificity metrics to different embeddings, we have conducted experiments on three different pre-trained embeddings including Hierarchy Category Embedding (HCE), FastText Pre-trained embedding and Word2Vec. Furthermore, we employ a learning to rank strategy, where we incorporate all the proposed specificity metrics and find the most pertinent combination of metrics in a supervised manner.

In the end, used in the literature; to use aggregation functions, such we evaluate all of the unsupervised and supervised metrics by predicting the performance of all the following corpora topics: Robust04, ClueWeb09, and GOV2. The proposed specificity metrics are defined for individual terms. Given queries can be composed of more than one term, we adopt the approach

that is widely as sum, maximum, minimum, and average [45], over the specificity of individual query terms. More specifically, aggregation functions, including sum, maximum, minimum, and average, are used to integrate individual term specificity values over the whole query. In Section 4.2, we compare our proposed metric with the state-of-the-art pre-retrieval QPP methods and show how and when our method outperform the baselines.

4.1 Evaluation on Specificity Metrics

The goal of this section is to evaluate the specificity metrics proposed in Chapter 3, regardless of their impact on QPP. To do so, first, we propose two test collections in Subsection 4.1.1. We explain the pre-trained embeddings that we use for each of the test collections, and describe how we can measure the performance of specificity metrics based on the suggested test collections. Eventually, we report the results in Subsection 4.1.3.

4.1.1 Test Collections

In this section, we introduce two test collections to evaluate the proposed measures of term specificity. In order to avoid the biases associated with manually curated gold standard data sets, our test collection is based on the Wikipedia category hierarchy and DMOZ taxonomy, which formally organize knowledge in degrees of specificity. The most generic category of the hierarchy sits at the topmost level, and the most specific categories are located at the leaves of the hierarchy. Since the level of each node in the hierarchy is an appropriate indicator of the nodes specificity with regards to its parent and child nodes, it can also be used as ground truth to evaluate specificity metrics.

We consider the level of each category, i.e., its shortest path to the root, as an indicator for the specificity of that category. We have randomly sampled unique paths, each with a length of 5, which form our test collection. We extracted 713 paths from Wikipedia and 171 paths from DMOZ taxonomy. The reasons we opted to extract and include paths in our test collection, as opposed to pairs of categories, were two fold:

- Paths generalize the concept of pairs as they are a collection of multiple pairs
- The consideration of paths ensures that the test collection only seeks to compare categories that are semantically related to each other and avoids the comparison of two or more categories that are not on at least one shared path from the root to a leaf

The objective of this experiment is to evaluate the ability of the proposed metrics to rank the categories in each path the same as the actual order in the hierarchy. We assess this ranking ability by Kendall Tau, which is a rank correlation metric. The more the ranking is similar to the actual one in the hierarchy, the higher the performance of the proposed metric is.The test collection and the results of our experiments are made available [1].

Wikipedia Category Hierarchy

Kapanipathi et al. [44] have empirically found that while hierarchical, the Wikipedia category can potentially include cyclic references between categories. Therefore, to transform the Wikipedia category structure into a strict hierarchy, we adopt the approach proposed by Kapanipathi et al. [44]. To do so, we first downloaded the freely available English version of

[1] https://github.com/WikipediaHierarchyPaths/WikiPedia_Hierarchy_Paths

DBpedia [2], which is extracted from Wikipedia dumps dated April 2016. This dataset consists of $1,411,022$ categories with $2,830,740$ subcategory relations between them. Then, we removed the Wikipedia admin categories that are used only to manage Wikipedia. Next, we selected Category: Main Topic Classifications, as the root node of the hierarchy and assigned the hierarchical level of each category based on its shortest path length to the root node. As the last step, we removed all directed edges from a category of lower hierarchical level (specific) to categories at higher hierarchical levels. The outcome of this process is a hierarchy with a height of 26 and $1,016,584$ categories with $1,486,019$ links among them. As such, and in order to evaluate specificity metrics, we have randomly sampled 713 unique paths, each with a length of 5, which form our test collection.

DMOZ Hierarchy

DMOZ which was also generally known as the Open Directory Project (ODP), was the largest, most comprehensive human-edited directory of the Web. It was built and kept up by an energetic, worldwide community of volunteer editors however it was owned by AOL. DMOZ closed in 2017 since AOL did not wish to back the venture. DMOZ is free of charge, and it is the foremost broadly dispersed information base of Web substance classified by humans. DMOZ is organized as a tree-structured taxonomy with 16 top categories and over $1,014,849$ categories in-depth and more than 5.1 million sites categorized. We utilized DMOZ taxonomy to assess the specificity of categories by looking at their position in the DMOZ hierarchical structure. We assumed that categories in a hierarchy in a position close to the root are broader than ones

[2] http://wiki.dbpedia.org/Downloads2015-10

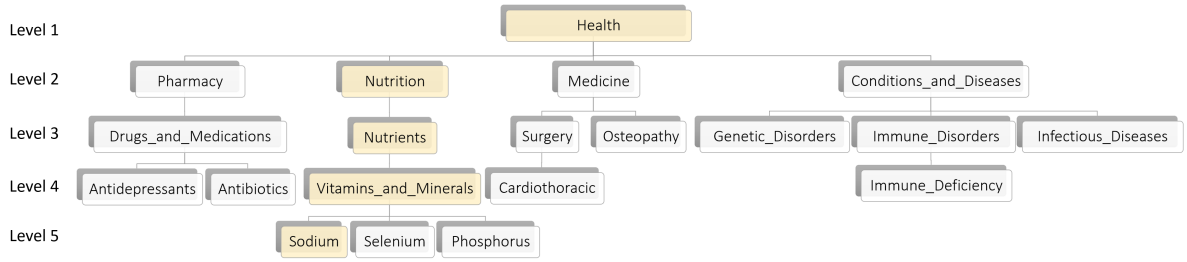

Figure 4.1: A part of DMOZ hierarchy and a randomly selected unique path. The selected path with the length of 5 is shown in yellow.

in positions close to the leaves.

In order to generate paths from DMOZ and assign embedding to them, first, We remove the less meaningful categories such as ones that are categorized by alphabet or name of cities, then we matched DMOZ categories to word2Vec embedding. We could only match almost 20% of the categories. Finally, from the remaining categories whose embedding exist, we were able to selected 171 paths with a length of 5. An example of a selected path from DMOZ taxonomy is represented in Figure 4.1 by yellow blocks.

4.1.2 Experimental setup

For estimating the specificity of a term, based on the proposed metrics, we utilize a pre-trained embedding model to identify the ε-neighborhood of that term. Given our test collections are based on DMOZ and Wikipedia, to calculate the specificity of each category in the DMOZ, we utilized Word2Vec embedding that had been trained on Google News since it is a general embedding and it is not biased to any specific subject. On the other hand, for Wikipedia category hierarchy, we needed an embedding model based on Wikipedia content and its categories. We chose two pre-trained embeddings that were trained on Wikipedia content:

1. HCE: Li et al. [29] have proposed a method that simultaneously learns entity and category embedding vectors from Wikipedia. Therefore, we adopt their trained Hierarchical Category Embedding (HCE) model to extract the required embedding vectors.

2. fastText: We used fastText pre-trained embedding [30] on Wikipedia content. Having said that, we run the experiments on the same hierarchy with two different embeddings in order to investigate the impact of embeddings on our metrics. This way we can find how robust our metrics are to embedding perturbations.

Now based on the paths included in the test collection and given a set of categories, the objective of an effective specificity metric would be to produce the correct specificity-based ordering of these categories, the correct ordering of which exists in the test collection. It is then possible to evaluate the performance of each specificity metric using rank correlation measures such as Kendall's Tau to determine the relationship between the actual order of categories observed in the test collection and the list of categories ranked based on the specificity metric.

4.1.3 Comparative Analysis

In this subsection, we present the results of the experiments that compare the proposed unsupervised specificity metrics based on the Kendall Tau rank correlation. We evaluate all the proposed neighborhood-based, graph-based, cluster-based and vector-based specificity metrics on two different hierarchies i.e, Wikipedia and DMOZ within three different embedding spaces i.e, HCE, fastText and Word2Vec and we report the results in Tables 4.1, 4.2, 4.3 and 4.4. It should be noted that to extract the ε-neighborhood of each term and build the ego network in the embedding space, the value of ε is set based on five-fold cross-validation optimized for

Kendall Tau. Almost all the results (except two) have P-value less than 0.05.

We investigate the consistency among the results in Table 4.1.3 by comparing the rank of each metric in the three evaluation, which is DMOZ hierarchy with Word2Vec embeddings and Wikipedia hierarchy with both fastText embedding and HCE embedding. We rank all the proposed metrics by their performance to see if they have consistent results on all the three experiments. Based on the results in Table 4.1.3, we can conclude that the metrics are performing coherent among the three experiments. There is some deviation among their performance that we elaborate on them in the following by analyzing each group of metrics.

1. **Vector-based metrics**: Vector-based metrics are the only metrics that are not performing consistently among different experiments. Previously, we discussed whether these metrics could be sufficiently beneficial for estimating the specificity or not. Since this group of specificity metrics is highly dependent on the embeddings, they are not reliable. Table 4.1.3 demonstrates a deviation in vector-based metrics performance. This group of metrics has the worst performance in all the three experiments except the HCE embeddings. This fact reveals that this group of metrics due to (1) their reliance on the embedding vectors and (2) lack of considering the relationship among embedding vectors in embedding space, performs poorly. On the other hand, the other three groups of metrics, i.e., neighborhood-based metrics, graph-based metrics, and cluster-based metrics perform consistently among the three experiments.

2. **Cluster-based metrics**: Among all the cluster-based metrics, EWAc, EWNc, and EWXc, which are a different variation of the same idea on the centroid network, are per-

forming well and the rest of them do not show acceptable performance. EWAc, EWNc, and EWXc have a decent performance among all the three experiments. We can reach the conclusion that these metrics are robust to both hierarchy and embeddings as they are among the top-10 of each of the experiments. More specifically, EWAc has outstanding performance and leads this group of metrics.

3. **Graph-based metrics**: There is a significant discrepancy among metrics in this groups. Some metrics have satisfactory performance in this category whereas others have lower performance. Ones that mostly are concerned with centrality measure, i.e., DC and CC perform well among all the three experiments. IEF, which was constructed based on the IDF idea, also performs among Top-3 in the Wikipedia hierarchy and has acceptable performance on the DMOZ taxonomy as well. Other metrics in this category, do not show adequate performance. Thus, among all the graph-based metrics, we can conclude that DC, CC, and IEF work the best.

4. **Neighborhood-based metrics**: In general neighborhood-based metrics perform modestly. Weighted Degree Centrality WDC works the best among this category.

By analyzing the left two-column of Table 4.1.3, i.e., the results on Wikipedia hierarchy with two different embeddings, we can note that the TOP-5 metrics performance-wise are interestingly from three different groups. In general, in this experiments, WDC from neighborhood-based metrics, IEF, DC, and CC from Graph-based and EWAc from cluster-based metrics are chosen as the best-performing metrics. These metrics are also robust to the embedding as well. In simple words, by using these metrics, we can estimate the specificity of terms, no matter

which embedding is used.

Another interesting finding is that the specificity computed from an ego-network works better than when it is estimated by the corresponding neighborhood-based metric. A potential explanation may be that while the number of neighbors computed in the neighborhood-based metrics is intuitively an indicator for specificity, the connections between neighboring nodes captured in the graph-based metrics can reinforce and strengthen specificity estimation. This hypothesis is boosted when comparing EWAe and WDC. WDC that overlooks the connections between neighboring nodes performs weaker than EWAe that considers such connections.

Table 4.1: The performance of the neighborhood-based specificity metrics in terms of Kendall Tau rank correlation. All values are statistically significant at alpha=0.05 except the italic one.

		Neighborhood-based Metrics						
Hierarchy	**Embedding**	**NS**	**WDC**	**MAD**	**NV**	**MSN**	**NVS**	**NVM**
Wikipedia	HCE	0.146	0.322	0.179	0.145	0.142	0.195	0.177
Wikipedia	fastText	0.086	0.249	0.176	0.176	0.088	0.175	0.168
DMOZ	Word2Vec	0.06	0.309	0.189	0.193	*0.016*	0.216	0.291

Table 4.2: The performance of the graph-based specificity metrics in terms of Kendall Tau rank correlation. All values are statistically significant at alpha=0.05.

		Graph-based Metrics									
Hierarchy	**Embedding**	**BC**	**CC**	**DC**	**IEF**	**EC**	**EWS**	**EWA**	**EWAe**	**EWXe**	**PR**
Wikipedia	HCE	0.137	0.339	0.341	0.336	0.207	0.139	0.288	0.315	0.291	0.131
Wikipedia	fastText	0.093	0.247	0.217	0.257	0.234	0.182	0.202	0.246	0.174	0.164
DMOZ	Word2Vec	0.25	0.339	0.32	0.328	0.22	0.162	0.367	0.379	0.318	0.279

Table 4.3: The performance of the cluster-based specificity metrics in terms of Kendall Tau rank correlation. All values are statistically significant at alpha=0.05.

		Cluster-based Metrics					
Hierarchy	Embedding	**CEV**	**EWAc**	**EWNc**	**EWXc**	**PC**	**NSC**
Wikipedia	HCE	0.213	0.320	0.321	0.295	0.110	0.116
Wikipedia	fastText	0.087	0.186	0.162	0.170	0.085	0.129
DMOZ	Word2Vec	0.213	0.409	0.419	0.371	0.295	0.369

Table 4.4: The performance of the vector-based specificity metrics in terms of Kendall Tau rank correlation. All values are statistically significant at alpha=0.05 except the italic one.

		Vector-based		
Hierarchy	Embedding	**SE**	**SEA**	**VE**
Wikipedia	HCE	0.237	0.035	0.208
Wikipedia	fastText	*0.010*	0.035	0.037
DMOZ	Word2Vec	0.068	0.084	0.091

Table 4.5: Ranking the result of specificity evaluations for investigating consistency among the results. The rankings are colored by their performance from white (the lowest performance) to the darkest color (highest performance).From Column left to right: Results on Wikipedia hierarchy using fastText embedding, results on Wikipedia hierarchy using HCE embedding, and results on DMOZ using word2Vec embedding

	ABV	Fasttext	HCE	Word2Vec
Neighborhood-based Metrics	NS	22	18	24
	WDC	2	4	11
	MAD	12	15	19
	NV	16	19	18
	MSN	20	20	25
	NVS	13	14	16
	NVM	15	16	12
Graph-based Metrics	DC	8	1	9
	CC	3	2	7
	BC	19	22	14
	IEF	1	3	8
	EC	7	13	15
	EWS	11	21	20
	PR	17	24	13
	EWA	10	10	6
	EWAe	5	7	3
	EWXe	14	9	10
Cluster-based Metrics	CEV	21	25	17
	EWAc	4	6	2
	EWNc	6	5	1
	EWXc	9	8	4
	NSC	18	17	5
Vector-based Metrics	SE	25	11	23
	SEA	24	23	22
	VE	23	12	21

4.1.4 Learning Specificity

All the metrics introduced above estimate the specificity of a term in an unsupervised manner. We additionally apply a supervised strategy to collectively incorporate all the unsupervised specificity metrics to predict the specificity of a term and find the most pertinent combination of unsupervised specificity metrics in a supervised manner. Due to the advantage of learning-to-rank strategy to combine a large number of features, it has attracted attention for different applications [46, 47]. We also take advantage of the learning to rank strategy to predict the specificity of terms by representing each term under investigation by a vector of features. In our model, features are the same as the specificity metrics in Table 4.1.3.

To apply learning to rank algorithm, we used RankLib[3], that includes the implementation of different learning to rank methods. In this section, we first exploit two well-known and frequently used learning-to-rank methods, including RankBoost and RandomForest to find the best method for ranking the categories based on their specificity by optimizing ndcg@5. Table 4.6 reports the performance of the models obtained by measuring the Kendall Tau rank correlation between the actual order of categories observed in the test collection and the list of categories ranked based on the output of each learning to rank model. Note that the results reported in Table 4.6 are calculated by applying five-fold cross-validation for each learning-to-rank method. For easier comparison, the table also includes the results of the Top-five specificity metrics observed in Table 4.1.3. As the table shows, the application of learning-to-rank methods that incorporate all of the proposed specificity metrics in a single model outperforms the Top-5 specificity metrics when used in isolation in ranking category paths.

[3] https://sourceforge.net/p/lemur/wiki/RankLib/

To further analyze the relative effectiveness of each feature (i.e, specificity metric defined in Table 4.1.3) in our supervised model, we rank all features based on their feature frequency in the Random Forest model. We start with the basic model consisting of 4 most important (i.e., most frequent) features and proceed to extend this model by incrementally introducing additional features, based on their frequency. In Tables 4.7 and 4.8, the results of this ablation study on Wikipedia hierarchy for both HCE embeddings and fastText embeddings are reported respectively in terms of Kendall Tau rank correlation. In Table 4.7,when Edge Weight Max_ego (EWXe) is added as a feature to the basic model, we observe a significant improvement of about 7.5%. Other significant improvements are also observed by adding Weighted Degree Centrality (WDC) (about 8% performance enhancement). Adding the rest of the features does not lead to significant changes in performance. Therefore, we did not list all the features and we only report the first 16 features out of all 25 features. In other words, using the top-8 most frequent features in RandomForest model would be sufficient to have the same performance when using all the 25 features. In Table 4.8 , the significant improvements occur when adding EWXe,CC,DC and SEA. Results in Table 4.8, shows that using the top-12 most frequent features is sufficient to have the performance quality same as using all the features in RandomForest model. Because, no significant improvement is observed when using more than top-12 features.

Table 4.6: Comparison of the performance of learning-to-rank models in terms of Kendall Tau rank correlation to the top-5 best performing metrics.The best model is indicated in bold. All the results are statistically significant at alpha=0.5.

	Supervised Methods		**Unsupervised Methods**				
	RankBoost	Random Forest	WDC	DC	CC	IEF	EWAc
HCE	0.350	**0.409**	0.322	0.341	0.339	0.336	0.320
fastText	0.389	**0.426**	0.249	0.247	0.217	0.257	0.186

Table 4.7: The performance of Random Forest model after incrementally adding features on HCE embedding. Basic model includes 4 most important features: SEA, SE, NSC and MSN. The * symbol indicates statistical significance on a paired t-test p-value<0.05.

	Basic Model	**+EWXe**EWXe	**+NVS**	**+EWXc**	**+WDC**	+EWA	+EWAe	+MAD	+EWNc	+EWS	+EWAc	+NVM	+EC
Kendall Tau	0.358	0.385	0.393	0.397	0.426	0.416	0.409	0.407	0.399	0.404	0.410	0.405	0.408
△		+7.54%*	+2.07%	+1.01%	+8.31%*	-2.34%	-1.68%	-0.48%	-1.67%	+1.25%	+1.48%	-1.21%	+0.074%

Table 4.8: The performance of Random Forest model after incrementally adding features on fastText embedding. Basic model includes 4 most important features: BC, PR, NV and NVS. The * symbol indicates statistical significance on a paired t-test p-value<0.05.

	Basic Model	+CEV	+EWXe	**+CC**	**+VE**	+DC	+NS	+IEF	+SEA	+EC	+NVM	+MSN	+EWXc
Kendall Tau	0.326	0.334	0.349	0.367	0.370	0.381	0.383	0.382	0.403	0.403	0.414	0.424	0.422
△		+2.4%	+4.4%*	+5.2%*	+0.81%	+3.0%*	+0.52%	-0.26%	+5.2%*	0%	+2.7%	+1.9%	-0.47%

4.1.5 Comparison with Baselines

In this section, we compare the performance of our model with two well-known frequency-based specificity metrics, i.e., Average IDF and Simplified Clarity Score (SCS), as our baselines. As

another baseline metric, we adopt the recently proposed approach by Roy et al. [8] that utilizes embedded word vectors to predict query performance. Their specificity metric, known as $P_{clarity}$, is based on the idea that the number of clusters around the neighborhood of a term t_i is a potential indicator of its specificity. To apply their approach to our embedding vectors, we have used the implementation provided by the authors.

In order to apply the frequency-based metrics, i.e., SCS and IDF, to estimate the specificity of a Wikipedia category, we consider Wikipedia as a collection of documents (each Wikipedia article is a document) and compute Average IDF and Simplified Clarity Score (SCS). We only compare our method with the baselines on Wikipedia hierarchy and we left the experiments on DMOZ for future works.

The results of the baselines are compared to the best variations of our supervised and unsupervised models. As shown, both of our supervised and unsupervised methods outperform the baselines despite the fact that the baseline methods have access to corpus-specific frequency information whereas our methods do not and are solely based on pre-trained neural embeddings.

Table 4.9: Comparison with Baselines on Wikipedia Hierarchy using HCE embedding. (P-values are less than 0.05)

Baselines			Our method (best variation)	
IDF_{max}	SCS	$P_{Clarity}$	Supervised	Unsupervised
0.304	0.144	0.121	0.409	0.341

4.2 Evaluation on Pre-retrieval QPP

In this section, we evaluate all the proposed specificity estimation metrics to predict TREC topics performance. First, we describe how we evaluate specificity metric in the context of query performance prediction. Then, we explain the employed corpora and topics in Subsection 4.2.1. In Subsection 4.2.3, we compare the proposed metrics with the baselines on three different TREC collections, including Robust04, ClueWeb09, and GOV2.

A common approach for measuring the performance of a QPP metric is to use rank correlation metrics to measure the correlation between the list of queries (1) ordered by their difficulty for the retrieval method (ascending order of AP, and (2) ordered by the QPP metric. Kendall's tau, which is a non-parametric measure of the relationship between two ranked lists and Pearson's rho coefficient, which is a measure of the linear correlation between two variables, are common correlation metrics in this space. To be able to rank the different metrics over a range of topics, and given the fact that no individual or a subset of metrics outperform the others overall topic sets and document collections, we compute the rank of each metric in each topic set and report the rank of the median of each metric overall topics of each document collection. This is specified as rank and is reported separately for Kendall's tau and Pearson's rho in the tables. It should be noted that our metrics are dependent on some hyper-parameters, which were set using five-fold cross-validation optimized for Pearson correlation.

4.2.1 Dataset

We employed three widely used corpora, namely Robust04, ClueWeb09, and GOV2. For Robust04, TREC topics $301-450$ and $601-650$, for GOV2, topics $701-850$ and for ClueWeb09,

topics $1-200$ are used. The topic difficulty was based on Average Precision (AP of each topic computed using Query Likelihood (QL) implemented in Anserini for the three collections. The three widely used corpora, Robust04, which consists of $528,155$ documents, ClueWeb09, which consists of 50 million English web pages, and Gov2, which consists of over 25 million documents have been extensively used in the literature for performing pre-retrieval query performance prediction experiments.

4.2.2 Baselines

For comparative analytics, we adopt the most widely used specificity metrics reported in the literature [2]. Previously we explained all of them in section 2.2.2. The formulation of these metrics is provided in chapter 2. We use aggregation functions (e.g., sum, avg, min, and max) to apply the term-based metrics to the whole query and we report the best result among all the different varieties of aggregation functions.

As another baseline, we used the recent work of Roy et al. [8] who proposed a novel query performance prediction predictor which similar to our work, utilize neural embedding representation of terms. While their work outperforms all the state-of-the-art post-retrieval query performance prediction metrics, it does not show acceptable performance in pre-retrieval QPP metrics. They also compared aggregation functions versus query term composition for applying term-based metrics to the whole query. Their experiments showed their approach worked better with the latter, which represents the query by summing up the query terms embedding vectors. Therefore, all the reported result for $P_{clarity}$ are based on query term composition.

4.2.3 Results and Discussion

Empirical studies on pre-retrieval QPP metrics have shown that while some metrics show better performance on some corpora and topic sets, there is no single metric or a set of metrics that outperforms the others on all topics and corpora [2]. Our empirical experiments confirm this. That is why we investigate the QPP results on different topics and corpora.

Table 4.10 , 4.11 and 4.12 illustrate the results on Robust04, ClueWeb09 and GOV2 respectively. In Tables 4.10,4.11, and 4.12, the top-3 best-performing metrics on Kendall tau and Pearson rho have been indicated using *. In the following, we will discuss the results on each of the collections. Later, we reach to a conclusion for the top performed metrics.

- *Robust04*: As shown in Table 4.10, our proposed QPP metrics are among the top-3 metrics (performance-wise) on both Person correlation and Kendal Tau correlation on. If we consider a balance between Kendall tau and Pearson rho and look for those metrics that perform relatively well on both, ,CC, IEF and DC seem to have a moderate consistent high performance. While VAR is among top-3 in 5 out of 8 experiments, it has poor performance on the other three, i.e, Pearson and Kendal Tau correlation on topic 300-350 and Pearson correlation on topic 401-450. On the other hand, our best performed proposed metrics i.e., IEF, CC, and DC have more consistent performance among all the topics. In general, CC, DC and IEF show satisfactory performance among all the topics on Robust 04.

- *ClueWeb09*:On ClueWeb09 neighborhood based metrics are performing poorly. On the other hand, graph-based metrics are showing noticeable performance. From baselines,

VAR, and from our proposed metrics, BC, CC, and DC which measure centralities in a graph, are among the best-performed metrics on this collection.

- *GOV2*: Unlike ClueWeb09 and Robust04, our proposed metrics perform a little inconsistent on GOV2. For instance, most of the graph-based metrics including BC, DC, CC, EC and IEF do not have excellent performance on topic 701-750. Whereas they are among top-3 metrics on the rest of the topics, i.e., 751-850. From the baselines, SCQ has outstanding performance on GOV2 collection. Since there are no metrics that have an acceptable performance on all of the topics on this collection, by considering a balance among all the topics and evaluation metrics, we conclude PR, CC and DC to be among the top performing metrics.

Overall, when considering a balance between the two evaluation measures and the performance of the metrics on all topics and corpora, we find our CC and DC metric to be a well-performing metric across the board. It is among the best metrics on Gov2 and ClueWeb09 and has a reasonably balanced performance on Robust04. However, CC tends to have a high time complexity of $O(V^3)$. On the other hand, the DC metric performs quite well on both ClueWeb09 and Gov2 but less effectively on Robust04. The added benefit of the DC metric is that it is inexpensive to compute with O(1). As such, while CC is the preferred metric given most metric computations are performed offline for QPP, DC can serve as an alternative if computation limitations exist.

Table 4.10: Results of QPP on Robust04 . Gray rows are baselines. Bold numbers indicate statistical significance at alpha=0.05. Top-3 metrics in each column are selected by *.

agg	Method name	Pearson	Kendal Tau	Pearson	Kendal Tau	Pearson	Kendal Tau	Pearson	Kendal Tau
		300-350		351-400		401-450		600-650	
avg	**VAR**	0.08	0.17	**0.46***	**0.36***	0.23	**0.35***	**0.55***	**0.34***
max	**SCQ**	0.01	0.13	**0.5**	**0.42***	**0.66***	**0.5**	**0.35**	**0.26**
avg	**IDF**	**0.52***	**0.22**	**0.42**	**0.29**	**0.43***	**0.28**	**0.36**	**0.32**
-	**SCS**	**0.43**	**0.21**	**0.35**	**0.24**	**0.34**	**0.23**	**0.42**	**0.3**
max	**PMI**	0.06	0.04	0.12	0.18	**0.28**	**0.18**	0.08	0.2
comp	**PC**	**0.34**	**0.28**	**0.24**	0.20	**0.36**	0.25	**0.38**	**0.26**
avg	**NS**	0.02	0.07	**0.4**	**0.25**	0.05	0.02	**0.39**	**0.22**
sum	**WDC**	**0.47**	**0.22**	0.00	0.02	**0.40**	0.12	0.13	0.02
max	**MAD**	0.09	0.10	0.19	0.10	**0.35**	**0.21**	**0.28**	0.16
sum	**NV**	**0.40**	**0.25**	0.24	0.13	0.18	0.18	**0.31**	**0.26**
sum	**MSN**	**0.48**	**0.24**	0.04	0.01	**0.42**	**0.19**	0.18	0.07
sum	**NVS**	**0.42**	**0.23**	0.06	0.03	**0.36**	0.18	0.14	0.04
max	**NVM**	**0.10**	**0.02**	**0.31**	0.20	0.39	0.08	0.06	**0.20**
max	**BC**	**0.35**	**0.31**	0.17	**0.20**	**0.39**	**0.30***	**0.40**	**0.42***
avg	**CC**	**0.26**	**0.24**	**0.41**	**0.29**	**0.42**	**0.27**	**0.45***	**0.37***
max	**DC**	**0.28**	**0.20**	**0.42**	**0.29**	**0.44***	**0.30***	**0.44**	**0.33**
min	**EC**	**0.37**	0.17	**0.41**	0.15	**0.26**	**0.27**	**0.40**	0.19
min	**EWS**	**0.34**	0.17	**0.49***	**0.24**	**0.26**	**0.29**	**0.46***	**0.22**
avg	**IEF**	**0.41**	**0.47***	**0.44**	**0.36***	**0.40**	**0.30**	**0.43**	**0.33**
max	**PR**	**0.33**	**0.25**	**0.47***	**0.40***	**0.31**	**0.24**	**0.31**	0.16
avg	**EWA**	0.30	0.15	**0.42**	**0.23**	0.18	0.09	0.27	0.18
avg	**EWAe**	**0.33**	0.19	**0.45**	**0.18**	0.18	**0.29**	**0.33**	0.17
avg	**EWXe**	**0.29**	0.12	**0.30**	**0.14**	0.21	**0.22**	**0.34**	0.15
avg	**CEV**	0.15	0.01	**0.38**	**0.21**	0.09	0.01	**0.36**	0.18
sum	**EWAc**	**0.54***	**0.35***	0.19	0.12	**0.32**	**0.17**	**0.29**	**0.17**
sum	**EWNc**	**0.54***	**0.37***	**0.28**	**0.17**	**0.34**	**0.22**	**0.38**	**0.23**
sum	**EWXc**	**0.52***	**0.32**	0.15	0.09	**0.29**	**0.16**	0.26	0.15
sum	**NSC**	**0.50**	**0.29**	0.07	0.04	**0.43***	**0.25**	0.18	0.11
max	**SE**	0.07	0.06	0.10	0.04	0.19	0.14	**0.09**	**0.03**
sum	**SEA**	**0.50**	**0.28**	0.07	0.02	**0.35**	**0.24**	0.18	0.10
sum	**VE**	**0.50**	**0.27**	0.07	0.04	**0.34**	**0.18**	0.18	0.06

Table 4.11: Results of QPP on ClueWeb09 . Gray rows are baselines. Bold numbers indicate statistical significance at alpha=0.05. Top-3 metrics in each column are selected by *.

agg	Method name	Pearson	Kendal Tau	Pearson	Kendal Tau	Pearson	Kendal Tau	Pearson	Kendal Tau
		1-50		51-100		101-150		151-200	
max	**VAR**	0.14	**0.28***	**0.04**	**0.23**	**0.42***	**0.27***	0.08	0.01
max	**SCQ**	0.22	**0.19**	**0.24**	**0.25**	**0.33**	**0.30***	0.09	0.03
avg	**IDF**	0.18	**0.24***	**0.21**	**0.25**	0.27	0.16	0.01	0.05
-	**SCS**	0.18	**0.23***	**0.19**	**0.24**	0.16	0.10	0.05	0.07
comp	**PC**	**0.29***	**0.3**	**0.27**	0.15	0.18	0.1	0.24	**0.24**
max	**PMI**	0.2	0.15	0.12	0.16	0.04	0.07	0.04	0.03
avg	**NS**	0.08	0.08	0.13	0.09	0.11	0.14	0.08	0.03
avg	**WDC**	0.188	0.175	0.131	0.002	0.154	0.037	0.0302	0.055
sum	**MAD**	0.082	0.025	0.087	0.073	0.138	0.07	0.135	0.12
sum	**NV**	0.044	0.054	0.136	0.167	0.27	**0.209**	0.65	0.013
avg	**MSN**	0.177	0.144	0.115	0.022	0.135	**0.214**	0.01	0.006
max	**NVS**	0.124	0.05	0.171	0.145	0.109	0.045	0.037	0.027
mean	**NVM**	0.06	0.07	0.00	0.03	0.22	0.12	0.13	0.04
avg	**BC**	**0.28**	**0.22**	**0.29***	**0.28***	**0.37**	**0.26**	**0.28**	**0.22**
min	**CC**	**0.31***	**0.22**	0.17	**0.2**	**0.47***	**0.28***	**0.33***	**0.36***
avg	**DC**	**0.25**	**0.22**	**0.31***	**0.27***	**0.39***	0.15	**0.34***	**0.38***
max	**EC**	0.06	0.14	0.27	0.16	0.29	**0.27***	0.14	0.05
max	**EWS**	0.07	0.13	0.27	0.18	0.27	**0.26**	0.14	0.06
avg	**IEF**	**0.29***	0.17	0.26	**0.2**	0.17	0.17	**0.36***	**0.39***
max	**PR**	**0.36***	0.13	**0.31***	**0.29***	**0.37**	0.18	**0.31**	**0.25**
avg	**EWA**	0.22	**0.21**	0.13	0.01	0.13	0.02	0.00	0.02
avg	**EWAe**	**0.19**	**0.22**	0.08	0.03	**0.17**	0.03	0.08	0.07
mean	**EWXe**	**0.25**	**0.15**	0.01	0.07	**0.21**	0.03	0.03	0.04
avg	**CEV**	0.02	0.7	0.02	0.01	**0.30**	0.16	0.03	0.01
avg	**EWAc**	0.03	0.03	0.18	0.20	0.22	0.02	0.13	0.03
sum	**EWNc**	0.18	0.09	0.12	0.08	0.24	0.14	0.06	0.08
sum	**EWXc**	0.08	0.04	0.16	0.13	0.19	0.06	0.00	0.16
avg	**NSC**	0.14	0.13	0.15	0.04	**0.31**	0.07	0.06	0.2
sum	**SE**	0.13	0.05	0.04	0.00	0.24	0.13	0.13	0.07
sum	**SEA**	0.13	0.06	0.09	0.08	0.19	0.11	0.11	0.17
sum	**VE**	0.13	0.03	0.08	0.10	0.19	0.13	0.10	0.10

Table 4.12: Results of QPP on GOV2 . Gray rows are baselines. Bold numbers indicate statistical significance at alpha=0.05. Top-3 metrics in each column are selected by *.

agg	Method name	Pearson	Kendal Tau	Pearson	Kendal Tau	Pearson	Kendal Tau
		701-750		751-800		801-850	
max	**VAR**	0.27	**0.2**	**0.06**	0.02	0.1	**0.05**
max	**SCQ**	**0.53***	**0.36***	**0.32**	**0.29***	**0.35**	**0.23**
avg	**IDF**	**0.4**	**0.27**	**0.25**	**0.22**	0.17	0.14
-	**SCS**	**0.34**	**0.23**	**0.19**	**0.19**	0.12	0.11
max	**PMI**	**0.44***	**0.28***	0.26	**0.22**	0.22	**0.16**
-	**PC**	**0.33**	**0.25**	**0.33***	**0.24**	**0.38**	**0.25**
	NS	0.11	0.11	0.08	0.11	0.09	0.01
max	**WDC**	0.11	0.13	0.01	0.01	0.24	**0.20**
mean	**MAD**	0.21	0.11	0.01	0.03	0.15	0.05
avg	**NV**	0.01	0.02	0.12	0.07	**0.28**	0.11
max	**MSN**	0.16	0.12	0.04	0.03	**0.26**	**0.22**
sum	**NVS**	0.08	0.06	0.04	0.08	**0.24**	0.06
avg	**NVM**	0.17	0.12	0.13	0.09	0.02	0.06
min	**BC**	0.18	0.18	**0.28**	0.11	**0.5***	**0.3**
max	**CC**	0.14	0.11	**0.42***	**0.36***	**0.51***	**0.41***
max	**DC**	0.15	0.12	**0.41***	**0.36***	**0.52***	**0.41***
min	**EC**	0.11	0.12	**0.24**	**0.26**	**0.38**	**0.3**
min	**EWS**	0.12	**0.2**	**0.33***	**0.2**	**0.44**	**0.31**
max	**IEF**	0.18	0.09	**0.33***	**0.25**	**0.47**	**0.34***
avg	**PR**	**0.38***	**0.28***	**0.32**	**0.19**	**0.32**	0.15
avg	**EWA**	0.15	0.11	0.16	0.15	0.11	0.10
avg	**EWAe**	**0.09**	**0.10**	**0.17**	0.14	0.16	**0.17**
avg	**EWXe**	**0.06**	0.09	**0.15**	0.15	0.17	**0.13**
max	**CEV**	0.06	**0.21**	0.10	0.01	0.09	0.07
sum	**EWAc**	0.19	0.13	0.18	0.12	0.21	0.10
sum	**EWNc**	0.16	0.10	0.23	0.16	0.18	0.12
sum	**EWXc**	0.13	0.10	0.15	0.11	0.19	0.09
sum	**NSC**	0.10	0.10	0.14	0.08	0.17	0.07
max	**SE**	0.08	0.03	0.10	0.10	**0.37**	**0.22**
max	**SEA**	0.24	0.15	0.07	0.03	0.05	0.02
sum	**VE**	0.10	0.09	0.14	0.11	0.17	0.13

4.3 Summary

To sum up this chapter, we introduced two test collections on Wikipedia and DMOZ hierarchy in order to evaluate specificity metrics directly. We generated 713 paths from Wikipedia and 171 Paths from DMOZ with the length of 5. The paths consists of concepts from distinct level of the hierarchy, which indicate different range of specificity. We utilized the extracted paths as a gold standard to evaluate specificity metrics. Specificity metrics evaluation task is defined as ranking the concepts in a path based on the specificity metrics. The more the ranking of concepts is similar to the actual ranking in the extracted path, the higher the performance of specificity metric is. We report the result of this experiment among all the proposed metrics in Chapter 3. We discussed the performance of different groups of metrics and compared their performance individually. In the direct specificity metrics evaluation, the set of top-5 performing metrics is consisted of the following metrics: Weighted Degree Centrality (WDC), Inverse Edge Frequency (IEF), Closeness Centrality (CC), Degree Centrality (DC) and Edge Weight Average_centroid (EWAc).

We also proposed a supervised algorithm to estimate specificity. We used all the proposed metrics as features in order to apply the learning to the rank algorithm to measure the specificity of terms. Our results show that using all the proposed metrics can lead to a more accurate estimation of specificity.

Finally, we used specificity metrics in order to predict query performance before retrieval happens. We run the experiments on three different TREC corpus, including robust04, ClueWeb09, and GOV2. We investigate the consistency among the performance of specificity metrics on

different corpora and topics. We conclude that DC, CC and IEF from graph-based specificity metrics are having the best performance. They even outperform the state-of-the-art baselines on the pre-retrieval QPP. Considering the two evaluation tasks, i.e, direct specificity evaluation and QPP task, we can reach the conclusion that DC and CC, which are centrality measures have most satisfactory performance in both tasks.

Chapter 5

Conclusions and Future Work

5.1 Concluding Remarks

In this thesis, we explored how the geometric properties of neural embeddings can be used to estimate the term specificity metrics in order to predict query performance. The motivation is that while corpus-level frequency information is not accessible from neural embeddings, other forms of inter-term associations could be identified and measured based on neural embeddings for determining characteristics such as term specificity, which have shown to be correlated with query difficulty [9]. While it is not possible to measure frequency information from neural embeddings, they are convenient for identifying the set of highly similar terms to a term based on vector similarity. We formalize our recursive definition of specificity, i.e., the extent to which a term is specific can be determined from the context created by the surrounding highly similar terms within the neural embedding space. In order to formalize specificity, we first define the notion of an ego network built based on term similarities within the embedding space. Such a

network provides a context for defining specificity-based QPP metrics.

Based on the ego network and similar term in the neighborhood of the ego concept, we proposed four different categories of specificity metrics including neighborhood-based, graph-based, cluster-based and vector-based. We evaluated the proposed metrics directly and indirectly.

In order to evaluate the proposed metrics directly, we provided two test collections based on two legit hierarchies i.e, Wikipedia and DMOZ. The two test collections are consisted of paths with the length of 5 that are ordered from the most generic to the most specific concept. By ranking the concepts on each path base on the proposed specificity metric and compare it to the actual ranking of the concepts, we can assess the performance of the specificity metrics.

We conclude that among all the proposed specificity metrics, Weighted degree centrality (WDC) from neighborhood-based , Inverse Edge Frequency (IEF) metric, Degree Centrality (DC) metric , Closeness Centrality (CC) metric from graph-based and Edge Weight Average_centroid (EWAc) metric from cluster-based category performed the best.

We have shown that it is possible to devise metrics based on the neural embedding-based representation of terms to perform pre-retrieval QPP. More specifically, we have shown that specificity of a query term, estimated based on its ego network, can lead to stronger performance on QPP compared to the state of the art that considers term clusters based on neural embeddings [8]. Closeness Centrality showed a balanced performance over different topics sets and corpora, although with high time complexity. The alternative, computationally inexpensive, Degree Centrality had a competitive performance on ClueWeb09 and Gov2 but weaker performance on Robust04.

In this thesis, we have introduced four types of unsupervised metrics, as well as a supervised,

learn to rank strategy for measuring term specificity based on a collection of neural embeddings. We have also created and publicly shared a test collection derived from the Wikipedia category hierarchy to serve as the gold standard for this task.

In summary, our key findings include:

1. Pre-trained, corpus-independent neural embedding representations of terms enable accurate estimates of term specificity

2. Graph-based specificity metrics that consider term neighborhood, as well as term associations, have the best performance

3. The four proposed types of metrics have synergistic impact on specificity estimation, as demonstrated through the ablation study

4. The corpus-independent metrics perform better than traditional frequency-based corpus-dependent specificity metrics to predict query performance.

5.2 Future Work

To further this research, we intend to pursue the following tasks:

1. Post-retrieval QPP: We aim to expand our work by proposing post-retrieval QPP methods based on neural embeddings.

2. Collection difficulty: Instead of estimating query difficulty, we can incorporate neural embeddings in order to measure the collection difficulty. Same as query difficulty estimation, this can be beneficial to predict a retrieval performance.

3. Accounting for the information need : Recently, some works [48] have defined a retrieval system quality as the ability to satisfy the information need behind the query. Therefore, instead of focusing on query difficulties, they propose to concentrate on the information need behind the query. In the future, we can explore estimating information need difficulty using neural embeddings.

4. Parameter tuning: As a next step to our work, we can look into the impact of parameter tuning on the proposed specificity metrics. We can investigate which of the metrics are robust to the parameter tuning and which ones can drift by changing the hyper-parameters such as ε in ε-neighborhood.

5. Complexity analysis: A complexity analysis can be conducted on the proposed specificity metrics to compare them by their computation expenses. This could be beneficial to chose the optimal specificity metric in different applications.

6. Specificity metrics applications: As the next step of our work, we can leverage the proposed specificity metrics in different applications. For instance, to expand the query by more specific terms in order to enhance the retrieval performance or to recommend more specific entities to users on social media in order to improve the personalized recommendation.

References

[1] Christopher Manning, Prabhakar Raghavan, and Hinrich Schütze. Introduction to information retrieval. *Natural Language Engineering*, 16(1):100–103, 2010.

[2] David Carmel and Elad Yom-Tov. Estimating the query difficulty for information retrieval. *Synthesis Lectures on Information Concepts, Retrieval, and Services*, 2(1):1–89, 2010.

[3] Claudia Hauff, Djoerd Hiemstra, and Franciska de Jong. A survey of pre-retrieval query performance predictors. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management, CIKM 2008, Napa Valley, California, USA, October 26-30, 2008*, pages 1419–1420, 2008.

[4] Karen Spärck Jones. A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, 60(5):493–502, 2004.

[5] Jiyin He, Martha Larson, and Maarten De Rijke. Using coherence-based measures to predict query difficulty. In *European conference on information retrieval*, pages 689–694. Springer, 2008.

[6] Bhaskar Mitra, Nick Craswell, et al. An introduction to neural information retrieval. *Foundations and Trends® in Information Retrieval*, 13(1):1–126, 2018.

[7] Hamed Zamani, W Bruce Croft, and J Shane Culpepper. Neural query performance prediction using weak supervision from multiple signals. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 105–114. ACM, 2018.

[8] Dwaipayan Roy, Debasis Ganguly, Mandar Mitra, and Gareth JF Jones. Estimating gaussian mixture models in the local neighbourhood of embedded word vectors for query performance prediction. *Information Processing & Management*, 56(3):1026–1045, 2019.

[9] Ben He and Iadh Ounis. Inferring query performance using pre-retrieval predictors. In *International symposium on string processing and information retrieval*, pages 43–54. Springer, 2004.

[10] Ellen M Voorhees. The trec robust retrieval track. In *ACM SIGIR Forum*, volume 39, pages 11–20. ACM, 2005.

[11] Ellen M Voorhees et al. Overview of the trec 2005 robust retrieval track. In *Trec*, 2005.

[12] Ellen M Voorhees. Overview of trec 2001 ellen m. voorhees, donna harman national institute of standards and technology gaithersburg, md 20899. 2003.

[13] Maurice G Kendall. A new measure of rank correlation. *Biometrika*, 30(1/2):81–93, 1938.

[14] Yun Zhou and W. Bruce Croft. Query performance prediction in web search environments. In *SIGIR 2007: Proceedings of the 30th Annual International ACM SIGIR Conference on*

Research and Development in Information Retrieval, Amsterdam, The Netherlands, July 23-27, 2007, pages 543–550, 2007.

[15] Anna Shtok, Oren Kurland, David Carmel, Fiana Raiber, and Gad Markovits. Predicting query performance by query-drift estimation. *ACM Trans. Inf. Syst.*, 30(2):11:1–11:35, 2012.

[16] Ying Zhao, Falk Scholer, and Yohannes Tsegay. Effective pre-retrieval query performance prediction using similarity and variability evidence. In *European conference on information retrieval*, pages 52–64. Springer, 2008.

[17] Lauri Kangassalo, Michiel M. A. Spapé, Giulio Jacucci, and Tuukka Ruotsalo. Why do users issue good queries?: Neural correlates of term specificity. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21-25, 2019.*, pages 375–384, 2019.

[18] R. Brown. *Words and things*. Macmillan, 1978.

[19] Qiao Zhang. Fuzziness - vagueness - generality - ambiguity. *Journal of Pragmatics*, 29(1):13 – 31, 1998.

[20] Robert B. Allen and Yejun Wu. Generality of texts. In *Digital Libraries: People, Knowledge, and Technology, 5th International Conference on Asian Digital Libraries, ICADL 2002 Singapore, December 11-14, 2002, Proceedings*, pages 111–116, 2002.

[21] Fabrizio Orlandi, Pavan Kapanipathi, Amit P. Sheth, and Alexandre Passant. Characterising concepts of interest leveraging linked data and the social web. In *2013*

IEEE/WIC/ACM International Conferences on Web Intelligence, WI 2013, Atlanta, GA, USA, November 17-20, 2013, pages 519–526, 2013.

[22] Vassilis Plachouras, Ben He, and Iadh Ounis. University of glasgow at trec 2004: Experiments in web, robust, and terabyte tracks with terrier. In *TREC*, 2004.

[23] Richard Kammann and Lynn Streeter. Two meanings of word abstractness. *Journal of Verbal Learning and Verbal Behavior*, 10(3):303 – 306, 1971.

[24] Dominik Benz, Christian Körner, Andreas Hotho, Gerd Stumme, and Markus Strohmaier. One tag to bind them all: Measuring term abstractness in social metadata. In *The Semanic Web: Research and Applications - 8th Extended Semantic Web Conference, ESWC 2011, Heraklion, Crete, Greece, May 29 - June 2, 2011, Proceedings, Part II*, pages 360–374, 2011.

[25] Christiane Fellbaum. Wordnet. In *Theory and applications of ontology: computer applications*, pages 231–243. Springer, 2010.

[26] Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*, pages 697–706. ACM, 2007.

[27] https://dmoz odp.org/.

[28] Simone Paolo Ponzetto and Michael Strube. Deriving a large scale taxonomy from wikipedia. In *AAAI*, volume 7, pages 1440–1445, 2007.

[29] Yuezhang Li, Ronghuo Zheng, Tian Tian, Zhiting Hu, Rahul Iyer, and Katia Sycara. Joint embedding of hierarchical categories and entities for concept categorization and dataless classification. *arXiv preprint arXiv:1607.07956*, 2016.

[30] Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. Advances in pre-training distributed word representations. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.

[31] Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751, 2013.

[32] David Mimno and Laure Thompson. The strange geometry of skip-gram with negative sampling. In *Empirical Methods in Natural Language Processing*, 2017.

[33] Matt Kusner, Yu Sun, Nicholas Kolkin, and Kilian Weinberger. From word embeddings to document distances. In *International conference on machine learning*, pages 957–966, 2015.

[34] Ebrahim Bagheri, Faezeh Ensan, and Feras Al-Obeidat. Neural word and entity embeddings for ad hoc retrieval. *Information Processing & Management*, 54(4):657–673, 2018.

[35] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*, 2016.

[36] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[37] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[38] Stefano Mizzaro and Josiane Mothe. Why do you think this query is difficult?: A user study on human query prediction. In *Proceedings of the 39th international ACM SIGIR conference on Research and development in information retrieval*, pages 1073–1076. ACM, 2016.

[39] Jamie Callan, Mark Hoy, Changkuk Yoo, and Le Zhao. Clueweb09 data set, 2009.

[40] Stanley Wasserman, Katherine Faust, et al. *Social network analysis: Methods and applications*, volume 8. Cambridge university press, 1994.

[41] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, 1999.

[42] Yoav Goldberg. Neural network methods for natural language processing. *Synthesis Lectures on Human Language Technologies*, 10(1):1–309, 2017.

[43] Paul Thomas, Falk Scholer, Peter Bailey, and Alistair Moffat. Tasks, queries, and rankers in pre-retrieval performance prediction. In *Proceedings of the 22nd Australasian Document Computing Symposium, ADCS 2017, Brisbane, QLD, Australia, December 7-8, 2017*, pages 11:1–11:4, 2017.

[44] Pavan Kapanipathi, Prateek Jain, Chitra Venkataramani, and Amit Sheth. User interests identification on twitter using a hierarchical knowledge base. In *European Semantic Web Conference*, pages 99–113. Springer, 2014.

[45] Claudia Hauff, Diane Kelly, and Leif Azzopardi. A comparison of user and system query performance predictions. In *Proceedings of the 19th ACM Conference on Information and Knowledge Management, CIKM 2010, Toronto, Ontario, Canada, October 26-30, 2010*, pages 979–988, 2010.

[46] Shuo Zhang and Krisztian Balog. Ad hoc table retrieval using semantic similarity. In *Proceedings of the 2018 World Wide Web Conference*, pages 1553–1562. International World Wide Web Conferences Steering Committee, 2018.

[47] Harrie Oosterhuis and Maarten de Rijke. Differentiable unbiased online learning to rank. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pages 1293–1302. ACM, 2018.

[48] Oleg Zendel, Anna Shtok, Fiana Raiber, Oren Kurland, and J Shane Culpepper. Information needs, queries, and query performance prediction. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 395–404. ACM, 2019.

Acronyms

AP Average Precision. 65, 66

BC Betweenness Centrality. x, 40, 43, 58, 63, 68–71

CC Closeness Centrality. 41, 43, 57, 58, 62, 63, 67–73

CEV Cluster Elements Variance. 46, 47, 59, 69–71

DC Degree Centrality. 40, 43, 57, 58, 62, 63, 67–73

EC Edge Count. 42, 43, 58, 63, 68–71

EEG Electro Encephalo Graphy. 19

EWA Edge Weight Average. 42, 43, 58, 63, 69–71

EWAc Edge Weight Average Centroid. 46, 47, 56, 57, 59, 63, 69–72

EWAe Edge Weight Average-ego. 42, 43, 58, 63, 69–71

EWNc Edge Weight Min Centroid. 46, 47, 56, 57, 59, 63, 69–71

EWS Edge Weight Sum. 42, 43, 58, 63, 69–71

EWXc Edge Weight Max Centroid. 47, 56, 57, 59, 63, 69–71

EWXe Edge Weight Max-ego. 43, 58, 62, 63, 69–71

HCE Hierarchy Category Embedding. x, 23, 25, 50, 55, 56, 58–60, 64

ICTF Inverse Collection Term Frequency. 20, 21

IDF Inverse Document Frequency. 3, 15, 20, 41, 57, 63, 64, 69–71

IEF Inverse Edge Frequency. 41, 43, 57, 58, 63, 67–73

IR Information Retrieval. 1, 10

MAD Median Absolute Deviation. 34, 37, 58, 63, 69–71

MSN Most Similar Neighbor. x, 35, 37, 43, 58, 63, 69–71

NS Neighborhood Size. 33, 37, 42, 58, 63

NSC Non-Specific Clustering. x, 46, 59, 63, 69–71

NV Neighborhood Variance. x, 35, 37, 58, 63, 69–71

NVM Neighborhood Vector Magnitude. xii, 36–38, 58, 63, 69–71

NVS Neighborhood Vector Similarity. x, 35–37, 58, 63, 69–71

ODP Open Directory Project. 53

PC P-Clarity. 59, 69–71

PMI Point-wise Mutual Information. 17, 18, 69–71

PR Page Rank. x, 41, 43, 58, 63, 68–71

QL Query Likelihood. 66

QPP Query Performance Prediction. vii, viii, xi, 2–4, 6–8, 11, 13–22, 25, 49–51, 65–71, 73, 75, 76

QS Query Scope. 20

SCQ Collection Query Similarity. 15, 16, 68–71

SCS Simplified Clarity Score. 21, 63, 64, 69–71

SE Sum Embedding. x, 47, 48, 59, 63, 69–71

SEA Sum Embedding Absolute. x, 48, 59, 62, 63, 69–71

VE Var Embedding. 48, 59, 63, 69–71

WDC Weighted Degree Centrality. 33, 34, 37, 57, 58, 62, 63, 69–72