Knowledge Based Text Representations for Information Retrieval

Chenyan Xiong
cx@cs.cmu.edu

May 2016

Language Technologies Institute
School of Computer Science
Carnegie Mellon University
5000 Forbes Ave., Pittsburgh, PA 15213
`www.lti.cs.cmu.edu`

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy
in Language and Information Technologies

Abstract

The successes of information retrieval (IR) in recent decades were built upon bag-of-words representations. Effective as it is, bag-of-words is only a shallow text understanding; there is a limited amount of information for document ranking in the word space. This dissertation goes beyond words and builds *knowledge based text representations*, which embed the external and carefully curated information from knowledge bases, and provide richer and structured evidence for more advanced information retrieval systems.

This thesis research first builds *query representations* with entities associated with the query. Entities' descriptions are used by query expansion techniques that enrich the query with explanation terms. Then we present a general framework that represents a query with entities that appear in the query, are retrieved by the query, or frequently show up in the top retrieved documents. A latent space model is developed to jointly learn the connections from query to entities and the ranking of documents, modeling the external evidence from knowledge bases and internal ranking features cooperatively. To further improve the quality of relevant entities, a defining factor of our query representations, we introduce learning to rank to entity search and retrieve better entities from knowledge bases. In the *document representation* part, this thesis research also moves one step forward with a bag-of-entities model, in which documents are represented by their automatic entity annotations, and the ranking is performed in the entity space.

This proposal includes plans to improve the quality of relevant entities with a co-learning framework that learns from both entity labels and document labels. We also plan to develop a hybrid ranking system that combines word based and entity based representations together with their uncertainties considered. At last, we plan to enrich the text representations with connections between entities. We propose several ways to infer entity graph representations for texts, and to rank documents using their structure representations.

This dissertation overcomes the limitation of word based representations with external and carefully curated information from knowledge bases. We believe this thesis research is a solid start towards the new generation of intelligent, semantic, and structured information retrieval.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The core task of information retrieval is document ranking. Consider the following query and document that a search engine may encounter:

> Query: "*Carnegie Mellon Location*"
>
> Document: "*Carnegie Mellon University is a private research university in Pittsburgh, Pennsylvania. Founded in 1900 by Andrew Carnegie as the Carnegie Technical Schools, the university became the Carnegie Institute of Technology in 1912 and began granting four-year degrees. In 1967, the Carnegie Institute of Technology merged with the Mellon Institute of Industrial Research to form Carnegie Mellon University."* [1]

The goal of search engine is to find the relevant documents, such as the example one, that satisfy the information needs behind the query, and rank them higher than the irrelevant ones. A typical ranking system can be divided into two components: *representation*, which transfers the natural language query and documents into computer understandable format, and *ranking model*, which models query-document pairs' relevancy upon their representations and produce document rankings.

In modern information retrieval, the representation is usually done by bag-of-words, in which a query or a document is converted to a vector of index terms derived from words. Each dimension of the vector corresponds to a term that appear in their text. Its weight records the importance of the term to the text. For example, the example query can be transferred into the vector "{Carnegie:1, Mellon:1, Location:1}". Bag-of-words breaks a text into terms, and assumes the terms are independent with each other. These simplifications make bag-of-words very cheap to construct and maintain, scaling up easily to the large query traffic and infinite web pages, and become the standard in most search engines.

Ranking models then rank documents based on their relevance to the query. With bag-of-words representations, several term-level statistics are widely used to model the query-document relevancy. For example, how many times the query terms appear in the document's vector (term frequency), the frequency of query terms in the entire corpus (document frequency), and the document length. Unsupervised ranking models (e.g. language models, vector space models, BM25,

[1] `https://en.wikipedia.org/wiki/Carnegie_Mellon_University`

and DPH [25]) developed various ways to combine these term-level statistics to produce document rankings. The recent development of learning to rank (LeToR) further improves ranking accuracy using machine learning techniques [50]. In learning to rank features from unsupervised ranking models and document qualities are combined with models trained on user feedback or expert labels.

Bag-of-words based search engines have been a huge success. They are serving billions of users every day. In recent decades, the improvement of ranking accuracy is more of a result from the development of ranking models, while there is a relatively smaller focus in the representation part. Various unsupervised retrieval models have been developed, but they all rely on the same set of term-level statistics: term frequency (TF), inverse document frequency (IDF) and document length. Many sophisticated learning to rank models brought in the cutting-edge machine learning technologies, but their features are almost identical, all using features from the same term-level statistics plus several document quality features. However, if limiting information retrieval with bag-of-words representation, there may be an upper bound on what fancy ranking models alone can achieve, as the available information from the representation part is limited, and intrinsic noises from bag-of-words' assumptions are hard to eliminate.

There are techniques developed to improve bag-of-words representation. For example, query expansion techniques expand the query with a larger set of related terms, with the hope that the expansion terms can better retrieve relevant documents. Sequential dependency model (SDM) takes phrases into consideration and matches query and documents not only by their words but also ngrams. However, within the bag-of-words scenario, the term vector based representation is only an approximation about how search engine users understand texts. Search engine users generate queries and read returned documents with their prior knowledge that is external to the query or corpus. Also, they understand the texts structurally instead of by flat term vectors. Individual term counts and statistic models provide an empirical approximation that works well, but the gap between user's and search engine's understanding of query and documents is inevitably there.

There has been a long history of incorporating human knowledge in information retrieval, bridging the gap between users and search systems. Since two thousand years ago, librarians have been using subject headings and thesauri to describe, index, and retrieval texts. Subject headings and thesauri, or more generally, *controlled vocabularies*, are a set of manually selected terms or phrases, referring to real world concepts like 'Education' and 'University'. The terms in a controlled vocabulary are grouped into carefully designed *ontology*. The ontology partitions the target domain into tree-structured categories. For example, a possible top-down path in the ontology is 'Education' $->$ 'University' $->$ 'Private university'. The earliest search engines adopted the librarian's approach and represented documents by controlled vocabularies [67]. In these retrieval systems, human editors manually annotate documents with terms from a controlled vocabulary, and the retrieval of documents are operated in the controlled vocabulary space.

Controlled vocabularies based representation has several advantages. The terms in controlled vocabularies are manually selected informative units, for example, concepts of a certain domain. The vocabularies are controlled to be clean and of real world meanings. The design of the ontology and the categorization of terms are done by experts using their domain knowledge. They provide search engines meaningful external information to refer to during retrieval. The annotation of controlled vocabulary to documents is done by editors according to their understandings about the documents. The vocabulary mismatch problem is also less severe as both query and documents are grounded manually to a small set of controlled vocabularies. The controlled vocabulary based search systems can retrieval document accurately with simple retrieval models, and is still being used by search engines in several domains [55].

The involvement of human experts makes controlled vocabulary representation more intelligent, but also restricts the scale of its usage. The manual construction of controlled vocabulary requires huge expert efforts. Thus, the coverage of controlled vocabularies over general domain texts is limited. It is impossible to manually assign controlled vocabulary terms to the infinite number of documents in modern search engines, and the automatic assignment was not effective enough for general domain search's requirements. The focus of ranking model research in recent decade is mainly on bag-of-words representations. It is unclear whether and how controlled vocabulary based search engines can benefit from them.

Recently, large scale *knowledge bases* or *knowledge graphs* have emerged. Knowledge bases store human knowledge into computer understandable format in graph-like structures. Each node in the knowledge graph records a basic information unit, called 'object' or 'entity'. An object or entity can be a named entity (e.g. Carnegie Mellon University), a concept (e.g. University), or anything that corresponds to a real-world stuff. The edges in the knowledge graph connect entities by their relationships or link entities to their attributes. The information represented by an edge is usually called 'fact'. A fact can be an attribute (e.g. the enrollment of Carnegie Mellon), a category (e.g. CMU is a University), or a relationship to another entity (e.g. CMU locates in Pittsburgh). Knowledge bases share the similar goal of storing human knowledge into structured formats with controlled vocabularies, but they are also different in many perspectives. Modern knowledge bases are curated by community efforts (e.g. Wikipedia), semi-automatically by both human and machines (e.g. Google's Knowledge Vault [30]), or automatically by information extraction systems (e.g. NELL [15]). They are carefully curated thus with a higher quality than document corpus, but are usually less precise than controlled vocabularies which are manually created by domain experts. The annotation of entities to texts can also be done automatically using entity linking systems [16, 37]. The automatic nature makes it possible to obtain knowledge bases at large scale with richer information (e.g. Freebase), and to annotate large web corpora (e.g. Google's FACC1 annotation [32]). This brings a new opportunity to explore knowledge bases' ability in information retrieval, but the noises and contradictions due to automation also raise new challenges to address.

This thesis research aims to improve the text representation of information retrieval with knowledge bases. The information stored in knowledge bases are semi-structured, carefully curated, and external to the query and corpus. We use them to build more structured and external knowledge embedded representations for both query and document. The new representations can directly improve the performance of existing ranking models, and can support more accurate ranking models. The knowledge based text representations and corresponding ranking models developed in this thesis research provide a better way to represent the query and documents, and promote the next generation of more structured and intelligent information retrieval systems.

Query representation is the first focus of this thesis research. Queries are usually short and carelessly written, unable to fully describe user's information needs behind them. We utilize the information about entities in knowledge bases to build better query representations. For example, entities such as Carnegie Mellon University, Andrew Carnegie, and Language Technologies

Institute are linked with their descriptions, attributes, types and related entities. This knowledge can be useful in explaining various queries about Carnegie Mellon. Many existing techniques can be used to find useful entities for a query. Query annotations provide entities that directly appear in the query string. Entity search retrieves entities that are related to the query. Document annotations provide entities that connect to the query through retrieved documents, which are also useful under the pseudo relevance feedback assumption.

We start by using query expansion, a widely used effective technique, to expand query with terms from relevant entity's textual descriptions. The carefully written textual descriptions of knowledge base entities can help explain corresponding queries. For example, the description of Carnegie Mellon University contains terms 'research', 'computer', 'Turing', and 'award', all useful for query 'CMU'. Our method first finds relate entities for a given query, for example, Carnegie Mellon University, Pittsburgh and Computer Science for query 'CMU', using entity search and document annotations. Terms that appear frequently in their descriptions, or fall into similar categories with the query are selected to expand the query. The expansion terms from knowledge base turned out to be very effective in both improving the accuracy of existing ranking models and reducing the risk of query expansion, This result motivates us to further explore the potential of knowledge bases in information retrieval.

Knowledge base entities are associated with much richer information than just textual descriptions. To better make use of them, we develop `EsdRank`, a framework that generally utilizes such external semi-structured data to improve ranking. In `EsdRank`, entities that appear in the query, are retrieved by the query, and appear frequently in query's top retrieved documents are selected to represent the query. EsdRank framework makes it possible to use all kinds of information associated with these entities to improve document ranking. Recall our example query 'Carnegie Mellon Location', relevant documents should have strong connections to the entity Carnegie Mellon University. The connection is not only about text similarity, but can also be determined by whether the document contains query entities and whether it falls into similar categories with the query entities. To make use of the new and heterogeneous evidence introduced by query entities, a novel learning to rank model is developed to learn the connection from query to entities and the rank of documents jointly. The new entity based query representation adds information retrieval with richer external evidence and a sophisticated but suitable learning to rank model, thus significantly improves the state-of-the-art of document ranking.

An important step in our knowledge based query representations is to find relevant entities for the query. In our previous systems, it was done by existing automatic approaches. One of them is entity search, which provides entities that are useful for our systems, but also found to be a bottleneck of the ranking performances. We address this problem by developing our own entity search system using learning to rank. The facts about an entity in the knowledge base are grouped into the fields of a virtual document. Text similarity features between the query and entity's virtual document are extracted and used in learning to rank models. We have successfully improved entity search on several test collections that are labeled according to *user's* information needs. This thesis research plans to continue this work by developing a relevant entity finding method that not only improves user's satisfaction, but also the performance of knowledge based information retrieval. We propose a co-ranking framework that uses supervisions from both entity part and document part, and learns how to find relevant entities and how to rank documents in a multitask learning setting.

Document representation is another focus of this thesis research. The texts in documents are usually long and complex. Breaking them to individual terms cannot fully represent their meanings. The thesis research aims to build better document representations with the information from the entities associated with documents. We first revisit controlled vocabulary based document representation, with larger knowledge bases and automatic entity linking tools. Given a document, we construct a *bag-of-entity* vector from its automatic entity annotations. Our preliminary experiments find that automatic entity linking systems can provide sufficient coverage on general domain texts. The accuracy of automatic annotation is not great, but simply using exact matches between query entities and document entities can already perform better than standard bag-of-words based retrieval.

The promising results motivate us to continue exploring the potential of bag-of-entities based ranking. To utilize more evidence from bag-of-entities, and to handle the uncertainties from entity linking errors, we plan to develop a hierarchical learning to rank framework that learns to rank with bag-of-entities and to combine with bag-of-words based ranking models. Its first layer models ranking in the two representations individually. More evidence from the bag-of-entities is incorporated, for example, connections, similarities, and relationships between entities. The state-of-the-art bag-of-words based ranking models are merged. The second layer models the uncertainties of the two representations and learns which one is more reliable when combining them for the final ranking. The bag-of-words representation and bag-of-entities representations have many differences. Bag-of-words has a higher recall but contains uninformative terms. Bag-of-entities is more focused but useful information may not be included. Since the errors made by bag-of-words and bag-of-entity are likely to be uncorrelated, our ranking framework has a higher chance succeed if combining them smartly.

The last goal of this thesis research is to build structured representations for texts using knowledge bases. Structure learning has been studied for a long time in nature language processing to help text understanding. It is successful in mapping short texts (sentences) to predefined structures, for example, syntactic parsing tree, dependency parsing tree, and semantic role labeling graphs. Documents are much longer and harder for this kind of structures. But knowledge bases provide a new opportunity. Their information is already organized as a graph: Ontology, relationships and attributes naturally from the knowledge graph. A large fraction of knowledge bases' information is about the interactions between entities but not each individual ones. Such information has already been successfully used in question answering. In state-of-the-art question answering systems, natural language questions are parsed into the knowledge base's subgraphs, and answers in the knowledge base are fetched by graph retrieval. Knowledge based question answering has already been integrated into commercial search engines and smartphone assistants like Siri, Google Now, and Cortana.

As appealing as they are, structured representations have not yet been effective for document ranking. This thesis research proposes to develop *entity graph* representations for query and documents. The nodes in the entity graph are the same with those in bag-of-entities. The edges between entities can be either manually defined close schema facts, or automatically extracted phrases by open information extraction. The graph based representation makes it possible to perform ranking in a more structured way. We propose to use random walk or graph kernel techniques to rank documents in the graph space. With the graph representation and ranking, more evidence from knowledge bases are incorporated and ranking accuracy can be further improved.

This thesis research aims to build better text representation for information retrieval using knowledge bases. We construct term based, entity based, and entity graph based text representations for query and documents, incorporating structured, carefully curated, and external information from knowledge bases into information retrieval. The new representations improve the performance of existing ranking models and make it possible to develop better ranking models. With the new text representations and better ranking models, this thesis research has already achieved state-of-the-arts in two search domains with two knowledge bases. We believe our research has opened a new space to explore for researchers in information retrieval, information extraction, and machine learning. For example, our work can motivate research works about embedding more external structured information into search engines, developing sophisticated ranking models, and constructing knowledge base specially towards real world applications' needs.

The rest of this thesis proposal organizes as follows. A brief overview of knowledge bases and related works are presented in Chapter 2. Our current research about knowledge based *query representation* is in Chapter 3, with query expansion with knowledge base in Session 3.1, EsdRank with query entities in Session 3.2, and relevant entity finding in Session 3.3. The preliminary research about bag-of-entities based *document representation* is discussed in Chapter 4. The proposed research topics are in Chapter 5. The last chapter concludes and discusses potential impacts of this thesis research.

Chapter 2

Background and Related Work

This chapter first provides some background about controlled vocabularies based information retrieval. Then it gives an overview of knowledge base and related works.

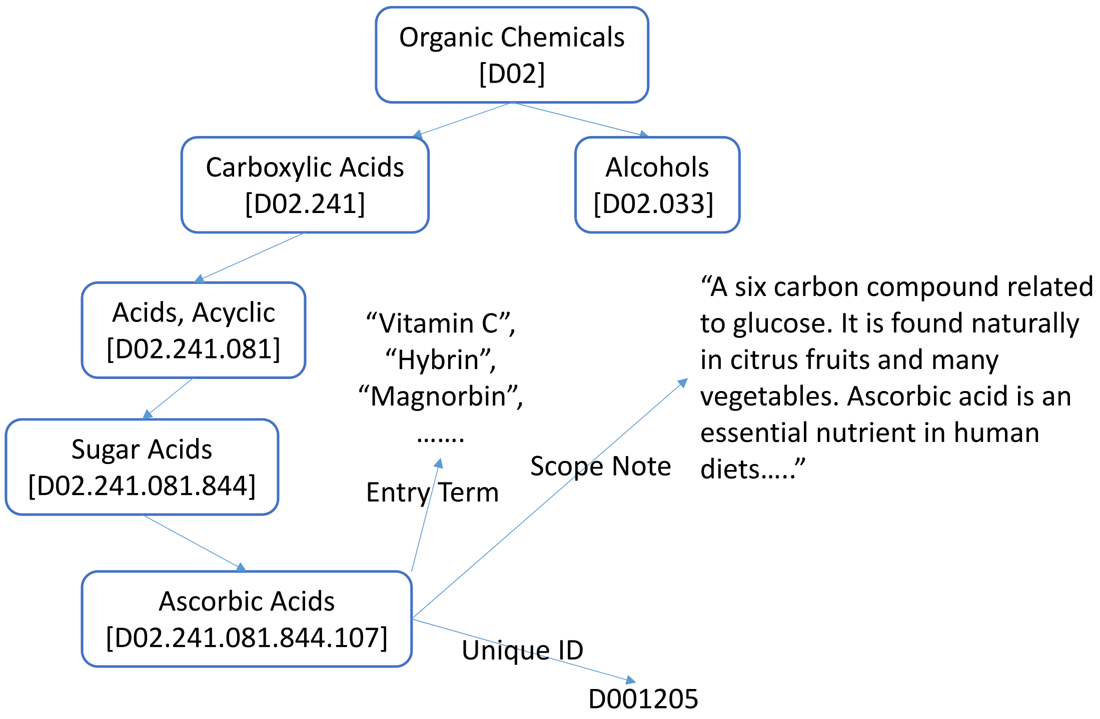
2.1 Controlled Vocabularies and Information Retrieval

The use of controlled vocabularies can date back to at least two thousand years ago, when the librarians in Egypt use them to organize books in the Library of Alexandria, 300 BCE. Instead of using all words, controlled vocabularies restrict themselves to a much smaller set of informative terms. These terms are carefully selected to represent real world objects, for example, concepts, common names, and domain terminologies. The controlled vocabularies are still being developed and used in the internet era. Famous representatives include World Bank Thesaurus, Medical Subject Headings (MeSH), and Library of Congress Subject Headings (LCSH).

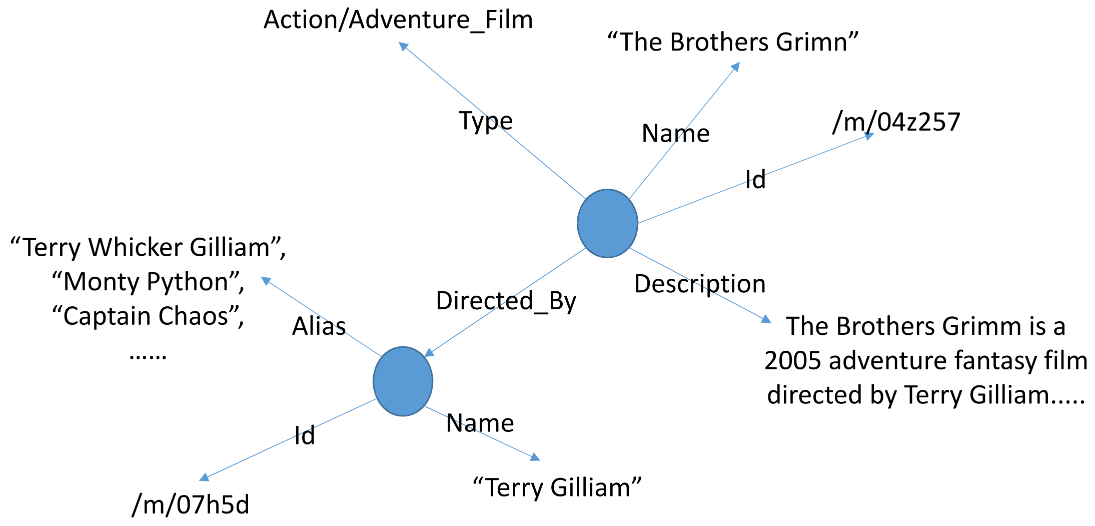
Although different controlled vocabulary datasets may make different choices on what to include based their application needs, there are some common ones among various datasets. Synonym, or alias, is one of the first things to include in a controlled vocabulary, since one major usage of controlled vocabulary is to resolve language variety. Ontology, or taxonomy, is another important component of many controlled vocabularies. Based on domain experts' understanding, ontology partitions the domain knowledge into a tree level structure, in order to organize, index, and retrieval target information. Textual descriptions are also often included. A text description defines or explains a controlled vocabulary term. The description helps users understand the term and facilitates the usage for domain experts. Figure 2.1a illustrates a part of MeSH's ontology and some facts associated with the term 'Ascorbic Acids', including its aliases ('Entry Term') and description ('Scope Note').

Information retrieval researchers inherited librarians' approach and built controlled vocabulary based search engines. In earliest search engines, documents were represented by terms from controlled vocabularies, e.g. Thesaurus or subject headings. The annotation of controlled vocabularies to documents was performed manually by domain experts, using their understandings of the documents and prior knowledge. The documents are then indexed based on their annotations, and the retrieval is done by matching query and documents' in the controlled vocabulary space.

As full-text search became popular, controlled vocabularies continued to be used in some


Organic Chemicals [D02]
Carboxylic Acids [D02.241]
Alcohols [D02.033]
Acids, Acyclic [D02.241.081]
Sugar Acids [D02.241.081.844]
Ascorbic Acids [D02.241.081.844.107]
"Vitamin C", "Hybrin", "Magnorbin", …….
Entry Term
Scope Note
"A six carbon compound related to glucose. It is found naturally in citrus fruits and many vegetables. Ascorbic acid is an essential nutrient in human diets….."
Unique ID
D001205

(a) Part of MeSH


Action/Adventure_Film
Type
"The Brothers Grimn"
Name
/m/04z257
Id
Description
The Brothers Grimm is a 2005 adventure fantasy film directed by Terry Gilliam…..
Directed_By
"Terry Whicker Gilliam", "Monty Python", "Captain Chaos", ……
Alias
Id
/m/07h5d
Name
"Terry Gilliam"

(b) Part of Freebase

Figure 2.1: Examples of a controlled vocabulary (MeSH) and a knowledge base (Freebase)

systems. The use of controlled vocabularies is almost a necessity in medical search engines. Medical queries are often about diseases, treatments or genes. Their special meanings may not be covered by their names. Typical procedures include using controlled vocabularies as alternative representations to match query and document [55, 64], and expanding queries using synonyms [54]. There is also a rich literature to overcome vocabulary mismatch by adding synonyms and related concepts to queries and documents [48]. These approaches can be effective in enterprise search and domain-specific search environments [35], or improving recall of relevant documents in general search environments [48]. On smaller collections where manually labeling is possible, one can also improve the ranking accuracy by combining controlled vocabulary and bag-of-words [64].

Nevertheless, bag-of-words plays a more important role in current search engines, because the use of controlled vocabularies faces many obstacles, especially in general domains and large scale search environments. It is very expensive for experts to construct a large enough controlled vocabulary to sufficiently cover general domains, if ever possible. The construction of controlled vocabulary representation requires annotations. In search environments where the sizes of corpora are very large, manually annotation is not feasible. The annotation can be done automatically using large scale multiclass classification techniques, which is a challenging task itself [34]. Another still on-going research topic is how to automatically leverage the information in controlled vocabulary in search engines. For example, synonyms of medical terms are very important for medical search, in which vocabulary mismatch problem is severe. However, the results of TREC Genomics Track illustrate the difficulty of using controlled vocabularies in medical search; most systems require human efforts to be successful [70]; one particular challenge is how to correctly pick and weight the synonyms [54]. The ontology and cross-reference of controlled vocabulary terms provide structural information that connects individual terms together. Using these connections intuitively should be able to improve the matching between query and documents, however, the experiments in recent works only show mixed results [41, 48].

2.2 Knowledge Base Overview

The term 'Knowledge Base', or 'Knowledge Graph' is a relatively new name that is mostly used for current large scale collections that store human knowledge. The collections can be manually or semi-automatically curated, for example, Freebase [10], YAGO [72], and DBPedia [47], or automatically constructed, for example, NELL [15] and OpenIE [3]. They are usually organized semi-structurally as a graph. For example, a sub-graph of Freebase is shown in Figure 2.1b. In the graph an entity (object) is expressed as a node with a unique Machine Id. An edge in the graph links an entity to its attribute or another entity. There are many kinds of edges in Freebase, representing different facts. For example, in Figure 2.1b, `The Brothers Grimm` is connected to `Terry Gilliam` by a `Directed_by` edge, showing that the movie `The Brothers Grimm` is directed by the director `Terry Gilliam`; the two nodes are also associated with its attributes such as aliases (synonyms), types (category) and textual descriptions.

The knowledge graph is usually stored as RDF triples. Each triple of subject-predicate-object corresponds to a head, edge, and tail in the knowledge graph. The head is an entity, or object, which can be a named entity, general domain entity, or just a noun phrase. The edge stores the type of the connection. Its type can be chosen from a vocabulary predefined manually by domain experts, called closed schema (e.g. in Freebase and DBPedia). It can also be verb phrases automatically extracted by information extraction techniques, called open schema (e.g. in NELL and OpenIE). The tail can be a related entity, or an attribute such as name, description or category of the subject entity.

The modern knowledge bases make different choices with classic controlled vocabularies in recording real world semantics. Controlled vocabularies are carefully edited by domain experts, more precise but mainly designed for specific domains at a smaller scale. Modern knowledge bases choose a looser schema to facilitate semi-automatic or automatic construction, which also introduces noises and contradictions. For example, MeSH, a widely used medical controlled vocabulary, contains about 27 thousand descriptors (terms), while Freebase contains more than 58 millions entities. They also favor different semantic information. Controlled vocabularies focus more on the ontologies. For example, MeSH has a carefully defined ontology that partitions medical knowledge into a thirteen-level hierarchical tree. In comparison, although storing general domain knowledge at much larger scale, Freebase only has a two level ontology. But modern knowledge bases include a wider range of attributes and relationships, for a more thorough coverage of an entity's semantic facts. For example, the entity Carnegie Mellon University is associated with 640 facts from 74 types in Freebase, while most MeSH terms only have less than ten attributes.

The automatic mapping from natural language texts to knowledge bases is also drawing more and more attentions from both academia and industry. This task is popularly addressed as Entity linking. Its goal is to recognize the appearance of entities in a text and link them to corresponding entries in a knowledge base. For example, given the first sentence of our example document in Chapter 1, an entity linking system may annotate it as "Carnegie Mellon University is a private research university in Pittsburgh, Pennsylvania.", with the underlined phrases linked to corresponding entities in a knowledge base. A standard way to perform entity linking is to first match ngrams in the text to names and aliases of entities in the knowledge base (spotting), and then pick the right one for each spot from all entities have the same name (disambiguation). In the linking process, various information from the knowledge base and the text are considered. The decision is made based on both local evidence about the entity and the ngram, and the global evidence across the entire text and all other possible entity links. Famous entity linking systems include (but not limited to as there are so many) TagMe [31], DBPedia spotlight [56], Wikification [59] and S-MART [83]. Linking long documents is the first focus of entity linking research [37], while recently research about linking on shorter text such as tweet and query has also emerged [16, 36]. After years of development, now entity linking systems can be reliable and fast enough to annotate knowledge bases as large as Freebase to large scale web corpora such as ClueWebs' (e.g. Google's FACC1 annotation [32]). A more detailed study of entity linking's coverage and accuracy will be provided in Chapter 4. It will also evaluate whether current entity linking systems are satisfying enough for general domain search engine's needs.

The large scale, richness, and flexibility of modern knowledge bases, together with the rapid developments of automatic grounding techniques, bring a new opportunity to reconsider their potential in information retrieval. However, through these divergences, modern knowledge bases, and classic controlled vocabularies share the same spirits: storing human knowledge into structured formats, carefully maintained quality, and centered around semantically informative objects (controlled vocabulary terms or entities). They are all ‘external’ to the user, query, and corpus, from search engine’s perspective. This thesis research uses a broader definition of knowledge bases that includes all these external and semi-structured collections, with the goal of developing general solutions to utilize these resources in modern information retrieval systems. We will also use the term ‘entity’ and ‘object’ interchangeably when referring to the ‘basic’ information unit of controlled vocabularies or modern knowledge bases, such as a concept, an noun phrase, an named entity, or an general entity.

2.3 Related Work

The exploration of modern knowledge bases’ ability in information retrieval has just started in recent years. Besides this thesis research, there are several parallel research that also focuses on improving information retrieval with modern knowledge bases.

One of the recent breakthroughs is the Entity Query Feature Expansion (EQFE) work by Dalton et al. [26, 28]. They study several ways to link Freebase entities to a query using query annotation, textual similarity, entity context model, and annotations from top retrieved documents. The name, alias and type fields of these entities are considered as possible expansion candidates. The combination of different linking methods, expansion fields, and hyperparameters in the expansion are enumerated to get various expanded queries. These enumerated expansion queries are then used to calculate ranking scores for each document using a query likelihood or sequential dependency model. A learning to rank model uses these ranking scores as features for each document and produces final document rankings. Their results on news retrieval (Robust 04) and web search (TREC Web Track) are among one of the first to demonstrate that knowledge bases can compete with state-of-the-art bag-of-words based ranking methods on general domain search tasks.

Another related work in knowledge based information retrieval is the Latent Entity Space (LES) model by Liu et al., which performed the best in TREC 2014 Web Track ad-hoc task [51, 52]. They use entities that are manually labeled to a query as a latent space between query and documents. The latent entities provide alternative connections between query and documents. With the latent entities, the ranking of documents is determined not only by query-document matching, but also by the textually similarities between those documents to latent entities, and between latent entities to the query. This evidence is incorporated in an unsupervised latent space language model to rank documents. Their extensive experiments on multiple test collections and in the TREC Web Track competition provide solid evidence of knowledge bases’ potential in modern information retrieval.

These related works share the same goal of utilizing modern knowledge bases in information retrieval systems. The development of knowledge base and automatic grounding techniques reveal new opportunities for information retrieval and inspired this thesis research. This vision is shared by many other information retrieval researchers, resulted in several parallel works that advance the state-of-the-arts from different places. Their research also reveals several important questions in the current stage of knowledge based information retrieval, for example, how to select proper entities for query and documents, how to incorporate the structured data in knowledge bases, and how to integrate them into current ranking systems. Their works and ours have demonstrated the effectiveness and potential of knowledge bases in information retrieval. We believe more research works have been and will be motivated, and the related work discussed in this section is just a beginning.

There are also other works related to specific sub-tasks of this thesis research. Later chapters of this proposal provide more in detail discussions about such related works. Section 3.1 discusses related works in query expansion. Section 3.2 revisits and compares our systems with learning to rank methods. Section 3.3 presents related work in entity search.

2.4 Summary

This chapter first reviews controlled vocabularies and their usage in information retrieval. Then it provides an overview of modern knowledge bases, and discusses their divergence and commonness with controlled vocabularies. The last section presents several related works about using modern knowledge bases to improve information retrieval. These related works serve as baselines of our systems, but more importantly, help strengthen and provide inspirations for the further pursuit of this thesis research.

Chapter 3

Knowledge Based Query Representation

Although knowledge bases contain information that can improve understanding of a topic, how to use it effectively for information retrieval tasks is still an open problem. In this chapter we study how to improve *query representation* with knowledge bases. Queries are short, ambiguous and usually carelessly written, thus often insufficient to fully describe the information needs behind the keywords. The rich semantic information stored in knowledge bases, for example, synonyms, ontologies, entities, and relationships, gives search engine a different perspective to understand the query, and provides a new opportunity for improving the retrieval accuracy.

In Section 3.1 we develop several query expansion methods that select terms from related entities in a knowledge base to expand the original query. Then Section 3.2 presents our `EsdRank` framework that represents queries by query entities from knowledge bases, and jointly learns how to connect query to query entities and how to rank documents in a novel learning to rank model. In both works, we used existing methods to find related entities for the query, and we found the quality of them plays an important role in determining the final ranking performances. In last part of this chapter, Section 3.3 presents our preliminary research about how to better find related entities for a query using entity search.

3.1 Query Expansion with Knowledge Base

Query expansion techniques, which generates expansion terms to enhance the original query, have been widely used to find better term based query representations. This section presents a simple and effective method of using one such knowledge base, *Freebase*, to improve query representation using query expansion techniques. We decompose the problem into two components. The first component identifies query-specific entities to be used for query expansion. We present implementations that retrieve entities directly, or select entities from retrieved documents. The second component uses information about these entities to select potential query expansion terms. We present implementations that select terms with a tf.idf method, or using category information. Finally, a supervised model is trained to combine information from multiple sources for better expansion.

Our experiments on the TREC Web Track ad-hoc task demonstrate that all our methods, when used individually, are about 20% more effective than previous state-of-the-art query expansion methods, including Pseudo Relevance Feedback (PRF) on Wikipedia [82] and supervised query expansion [14]. In addition to these improvements, experimental results show that our methods are more robust and have better *win/loss* ratios than state-of-the-art baseline methods, reducing the number of damaged queries by 50%. This makes query expansion using Freebase more appealing, because it is well-known that most query expansion techniques are 'high risk / high reward' insofar as they often damage as many queries as they improve, which is a huge disadvantage in commercial search systems. The supervised model also successfully combines evidence from multiple methods, leading to 30% gains over the previous state-of-the-art. Besides being the first to improve query expansion this much on the widely used ClueWeb09 web corpus, the methods presented here are also fully automatic. This work appears in the proceedings of ICTIR 2015 [79] and is the current state-of-the-art in query expansion.

Section 3.1.2 discusses our new methods of using Freebase for query expansion. Experimental methodology and evaluation results are described in Sections 3.1.3 and 3.1.4 respectively. The last part of this section summarizes contributions.

3.1.1 Related Work in Query Expansion

Queries are usually short and not written carefully, which makes it more difficult to understand the intent behind a query and retrieve relevant documents. A common solution is query expansion, which uses a larger set of related terms to represent the user's intent and improve the document ranking.

Among various query expansion techniques, Pseudo Relevance Feedback (PRF) algorithms are the most successful. PRF assumes that top ranked documents for the original query are relevant and contain good expansion terms. For example, Lavrenko et al.'s RM model selects expansion terms based on their term frequency in top retrieved documents, and weights them by documents' ranking scores:

$$s(t) = \sum_{d \in D} p(t|d) f(q,d)$$

where D is the set of top retrieved documents, $p(t|d)$ is the probability that term t is generated by document d's language model, and $f(q,d)$ is the ranking score of the document provided by the retrieval model [45]. Later, Metzler added inverse document frequency (IDF) to demote very frequent terms:

$$s(t) = \sum_{d \in D} p(t|d) f(q,d) \log \frac{1}{p(t|C)} \tag{3.1}$$

where $p(t|C)$ is the probability of term t in the corpus language model C [29].

Another famous PRF approach is the Mixture Model by Tao et al. [73]. They assume the terms in top retrieved documents are drawn from a mixture of two language models: query model θ_q and a background model θ_B. The likelihood of a top retrieved document d is defined as:

$$\log p(d|\theta_q, \alpha_d, \theta_B) = \sum_{t \in D} \log(\alpha_d \, p(t|\theta_q) + (1 - \alpha_d) \, p(t|\theta_B)).$$

α_d is a document-specific mixture parameter. Given this equation, the query model θ_q can be learned by maximizing the top retrieved documents' likelihood using EM. The terms that have non-zero probability in θ_q are used for query expansion.

Although these two algorithms have different formulations, they both focus on term frequency information in the top retrieved documents. So do many other query expansion algorithms [23, 46, 58, 86]. For example, Robertson et al.'s BM25 query expansion selects terms based on their appearances in relevant (or pseudo-relevant) documents versus in irrelevant documents [65]. Lee et al. cluster PRF documents and pick expansion terms from clusters [46]. Metzler and Croft include multi-term concepts in query expansion and select both single-term concepts and multi-term concepts by a Markov Random Field model [58].

The heavy use of top retrieved documents makes the effectiveness of most expansion methods highly reliant on the quality of the initial retrieval. However, web corpora like ClueWeb09 are often noisy and documents retrieved from them may not generate reasonable expansion terms [5, 61]. Cao et al.'s study shows that top retrieved documents contain as many as 65% harmful terms [14]. They then propose a supervised query expansion model to select good expansion terms. Another way to avoid noisy feedback documents is to use an external high quality dataset. Xu et al. proposed a PRF-like method on top retrieved documents from Wikipedia, whose effectiveness is verified in TREC competitions [61, 82]. Kotov and Zhai demonstrated the potential effectiveness of concepts related to query terms in ConceptNet for query expansion, and developed a supervised method that picks good expansion concepts for difficult queries [42].

Another challenge of query expansion is its 'high risk / high reward' property, that often as many queries are damaged as improved. This makes query expansion risky to use in real online search service because users are more sensitive to failures than successes [20]. Collins-Thompson et al. [23] address this problem by combining the evidence from sampled sub-queries and feedback documents. Collins-Thompson also proposes a convex optimization framework to find a robust solution based on previous better-on-average expansion terms [21, 22]. The risk is reduced by improving inner difference between expansion terms, and enforcing several carefully designed constraints to ensure that expansion terms provide good coverage of query concepts.

3.1.2 Expansion Using Freebase

In this section, we introduce our methods of using Freebase for query expansion. We first discuss our unsupervised expansion methods utilizing different information from Freebase. Then we propose a supervised query expansion method to combine evidence from our unsupervised methods.

3.1.2.1 Unsupervised Expansion Using Freebase

We perform unsupervised query expansion using Freebase in two steps: object linking and term selection. In object linking, we develop implementations that retrieve objects directly, or select them from annotations in top ranked documents. In term selection, we also present two implementations: one uses the tf.idf information from object descriptions; the other uses similarity of the query and the term's distributions in Freebase's categories.

Formally, given a query q, and a ranked list of documents from initial retrieval $D = \{d_1, ...d_j..., d_N\}$, the goal of the object linking step is to generate a ranked list of Freebase objects $O = \{o_1, ...o_k..., o_K\}$, with ranking scores $r(O) = \{r(o_1), ...r(o_k)..., r(o_K)\}$. The goal of term selection is to find a set of expansion terms $T = \{t_1, ...t_i..., t_M\}$ and their scores $s(T) = \{s(t_1), ...s(t_i)..., s(t_M)\}$ from linked objects using their descriptions $e(O) = \{e(o_1), ...e(o_k)..., e(o_K)\}$ and Freebase categories $C = \{c_1, ...c_u..., c_U\}$.

Linking Freebase Objects to the Query

Our first linking method retrieves objects directly. The query q is issued to the Google Freebase Search API[1] to get its ranking of objects O with ranking scores $r_s(O)$. The ranking score ranges from zero to several thousands, with a typical long tailed distribution. We normalize them so that the ranking scores of each query's retrieved objects sum to one.

Our second approach selects related objects from the FACC1 annotations in top retrieved documents. It is a common assumption that top retrieved documents are a good representation of the original query. Intuitively the objects that appear frequently in them shall convey meaningful information as well. We utilize such information by linking the query to objects that are frequently annotated to top retrieved documents.

Specifically, for a query q's top retrieved documents D, we fetch their FACC1 annotations, and calculate the ranking score for object o_k as:

$$r_f(o_k) = \sum_{d_j \in D} tf(d_j, o_k) \log \frac{|F|}{df(o_k)}. \tag{3.2}$$

In Equation 3.2, $tf(d_j, o_k)$ is the frequency of object o_k in document d_j's annotations, and $df(o_k)$ is the total number of documents o_k is annotated to in the whole corpus. $|F|$ is the total number of documents in the corpus that have been annotated in the FACC1 annotation. $\frac{|F|}{df(o_k)}$ in Equation 3.2 serves as inverse document frequency (IDF) to demote objects that are annotated to too many documents. $r_f(o_k)$ is normalized so that ranking scores of each query's objects sum to one.

Selecting Expansion Terms from Linked Objects

We develop two methods to select expansion terms from linked objects.

The first method does tf.idf based Pseudo Relevance Feedback (PRF) on linked objects' descriptions. PRF has been successfully used with Wikipedia articles [5, 61, 82]. It is interesting to see how it works with Freebase.

Given the ranked objects O and $r(O)$, a term's score is calculated by:

$$s_p(t_i) = \sum_{o_k \in O} \frac{tf(e(o_k), t_i)}{|e(o_k)|} \times r(o_k) \times \log \frac{|E|}{df(t_i)} \tag{3.3}$$

where $tf(e(o_k), t_i)$ is the term frequency of t_i in o's description, $|e(o_k)|$ is the length of the description, $df(t_i)$ is the document frequency of t_i in the entire Freebase's description corpus E. $|E|$ is the total number of entities in Freebase that have a description.

[1]`https://developers.google.com/freebase/v1/getting-started`

Our second term selection method uses Freebase's entity categories. Freebase provides an ontology tree that describes entities at several levels of abstraction. We use the highest level in the ontology tree, such as $/people$ and $/movie$, to make sure sufficient instances exist in each category. There are in total $U = 77$ first level categories in Freebase. The descriptions of entities in these categories are training data to learn the language models used to describe these categories.

Our second approach estimates query and terms distributions on categories, and selects terms that have similar category distributions with the query.

The distribution of a term in Freebase categories is estimated using a Naive Bayesian classifier. We first calculate the probability of a term t_i generated by a category c_u via:

$$p(t_i|c_u) = \frac{\sum_{o_k \in c_u} tf(e(o_k), t_i)}{\sum_{o_k \in c_u} |e(o_k)|}$$

where $o_k \in c_u$ refers to objects in category c_u.

Using Bayes' rule, the probability of term t_i belonging to category c_u under uniform priors is:

$$p(c_u|t_i) = \frac{p(t_i|c_u)}{\sum_{c_u \in C} p(t_i|c_u)}.$$

Similarly, the category distribution of a query q is:

$$p(q|c_u) = \prod_{t_i \in q} p(t_i|c_u),$$
$$p(c_u|q) = \frac{p(q|c_u)}{\sum_{c_u \in C} p(q|c_u)}.$$

The similarity between the two distributions $p(c_u|t_i)$ and $p(c_u|q)$ is evaluated by negative Jensen-Shannon divergence:

$$s_c(t_i) = -\frac{1}{2}\text{KL}(p(C|q)||p(C|q,t_i)) - \frac{1}{2}\text{KL}(p(C|t_i)||p(C|q,t_i))$$

where:

$$p(C|q,t_i) = \frac{1}{2}(p(C|q) + p(C|t_i))$$

and $\text{KL}(\cdot||\cdot)$ is the KL divergence between two distributions. $s_c(t_i)$ is the expansion score for a term t_i. We use a min-max normalization to re-range all $s_c(t_i)$ into $[0, 1]$.

As a result, we have two methods that link related Freebase objects to a query, and two methods to select expansion terms from linked objects. They together form four unsupervised expansion methods, as listed in Table 3.1.

Table 3.1: Unsupervised Query Expansion Methods Using Freebase.

	Link by Search	Link by FACC1
Select by PRF	`FbSearchPRF`	`FbFaccPRF`
Select by Category	`FbSearchCat`	`FbFaccCat`

3.1.2.2 Supervised Expansion Using Freebase

Different object linking and term selection algorithms have different strengths. Object search links objects that are directly related to the query by keyword matching. FACC1 annotation provides objects that are more related in meanings and does not require exact textual matches. In expansion term selection, PRF picks terms that frequently appear in objects' descriptions. The category similarity method selects terms that have similar distributions with the query in Freebase's categories. They together provide three scores describing the relationship between a query-term pair: tf.idf Pseudo Relevance Feedback score in retrieved objects, tf.idf Pseudo Relevance Feedback score in top retrieved documents' FACC1 annotations, and a negative Jensen-Shannon divergence score between category distributions.

The three scores are used as features for a supervised model that learns how to select better expansion terms. All terms in linked objects' descriptions are used as candidates for query expansion. The ground truth score for a candidate term is generated by its influence on retrieved documents, when used for expansion individually. If a term increases the ranking scores of relevant documents, or decreases the ranking scores of irrelevant documents, it is considered to be a good expansion term, and vice versa.

The influence of a term t_i over retrieved documents is calculated as:

$$\begin{aligned} y(t_i) = & \frac{1}{|R|}\sum_{d_j \in R}(f(q+t_i, d_j) - f(q, d_j)) \\ & - \frac{1}{|\bar{R}|}\sum_{d_j \in \bar{R}}(f(q+t_i, d_j) - f(q, d_j)) \end{aligned}$$

where R and $\bar{R}$ are the sets of relevant and irrelevant documents in relevance judgments. $f(q, d_j)$ is the ranking score for document d_j and query q in the base retrieval model. $f(q+t_i, d_j)$ is the ranking score for d_j when the query is expanded using expansion term t_i individually. Binary labels are constructed using $y(t)$. Terms with $y(t) > 0$ are treated as good expansion terms and the rest as bad expansion terms.

Our ground truth label generation is a little different than Cao et al.'s [14]. Their labels were generated by a term's influence on documents' ranking positions: if relevant documents are moved up or irrelevant document are moved down by a term, it is considered a good expansion term, otherwise a bad one. In comparison, we use influence on ranking scores which reflect an expansion term's effectiveness more directly. Our preliminary experiments also confirm that both their method and our method work better with our ground truth labels.

We used a linear SVM classifier to learn the mapping from the three features of a term t to its binary label. To get the expansion weights, we used the probabilistic version of SVM in the LibSVM [17] toolkit to predict the probability of a term being a good expansion term. The predicted probabilities are used as terms' expansion scores, and those terms with highest scores are selected for query expansion.

3.1.2.3 Ranking with Expansion Terms

We use the selected expansion terms and their scores to re-rank the retrieved documents with the RM model [45]:

$$f^*(d_j, q) = w_q f(q, d_j) + (1 - w_q)(\sum_{t_i \in T} s(t_i) f(t_i, d_j))). \tag{3.4}$$

In Equation 3.4, $f^*(q, d_j)$ is the final ranking score to re-rank documents. $f(q, d_j)$ and $f(t_i, d_j)$ are the ranking scores generated by the base retrieval model, e,g, BM25 or query likelihood, for query q and the expansion term t_i respectively. w_q is the weight on the original query. T is the set of selected expansion terms and $s(t_i)$ is the expansion score of the term t_i. Expansion scores are normalized so that the scores of a query's expansion terms sum to one.

3.1.3 Experimental Methodology

In this section, we introduce our experimental methodology, including dataset, retrieval model, baselines, hyper-parameters, and evaluation metrics.

Dataset: Our experiments use ClueWeb09, TREC Web Track 2009-2012 adhoc task queries and the relevance judgments provided by TREC annotators. This dataset models a real web search scenario: queries are selected from the search log from Bing, and ClueWeb09 is a widely used web corpus automatically crawled from the internet by Carnegie Mellon University. ClueWeb09 is known to be a hard dataset for query expansion [5, 26, 61], because it is much noisier than carefully edited corpora like the Wall-street Journal, news and government web sets.

We use Category B of ClueWeb09 and index it using the Indri search engine [71]. Typical INQUERY stopwords are removed before indexing. Documents and queries are stemmed using the Krovetz stemmer [43]. Spam filtering is very important for ClueWeb09 and we filter the 70% most spammy documents using the Waterloo spam score [24].

We retrieved Freebase objects and fetched their descriptions using the Google Freebase API on July 16th, 2014. Entity linking from documents to entities are found in FACC1 annotation [32], which was published in June 2013. Corpus statistics such as term IDF and categories' language models were calculated from the April 13th, 2014 Freebase RDF dump.

Retrieval Model: We use Indri's language model [29] as our base retrieval model. The ranking score of a document is the probability of its language model generating the query. Dirichlet smoothing is applied to avoid zero probability and incorporate corpus statistics:

$$p(q|d_j) = \frac{1}{|q|} \sum_{t_i \in q} \frac{tf(d_j, t_i) + \mu p(t_i|\mathcal{C})}{|d_j| + \mu}, \tag{3.5}$$

where $p(t_i|\mathcal{C})$ is the probability of seeing term t_i in the whole corpus, and μ is the parameter controlling the smoothing strength, set to the Indri default: 2500.

Baselines: We compare our four unsupervised expansion methods (as listed in Table 3.1) and the supervised method described in Section 3.1.2.2 (`FbSVM`) with several baselines. The first baseline is the Indri language model (`IndriLm`) as in Equation 3.5. All relative performances and Win/Loss evaluations of other methods are compared with `IndriLm` if without specific reference. Our second baseline is the Sequential Dependency Model (`SDM`) [57], a strong competitor in TREC Web Tracks.

We also include two well-known state-of-the-art query expansion methods as baselines. The first one is Pseudo Relevance Feedback on Wikipedia (`RmWiki`) [5, 61, 82]. We indexed the Oct 1st 2013 Wikipedia dump using same setting we used for ClueWeb09. Standard Indri PRF with the IDF component [61] was performed to select expansion terms.

The other query expansion baseline is the supervised query expansion (`SVMPRF`) [14]. We extracted the 10 features described in their paper and trained an SVM classifier to select good expansion terms. We used our term level ground truth labels as discussed in Section 3.1.2.2, because their model performs better with our labels. Following their paper, the RBF kernel was used, which we also found necessary for that method to be effective.

For clarity and brevity, we do not show comparison with other methods such as RM3 [45], Mixture Model [73], or EQFE [26] because they all perform worse on ClueWeb09 than `RmWiki` and `SDM` in our experiment, previous TREC competitions [61], or in their published papers.

Parameter Setting: Hyper parameters in our experiment, including the number of expansion terms (M), number of objects (K) in Freebase linked for expansion, and number of PRF documents for `RmWiki` and `SVMPRF`, are selected by maximizing the performance on training folds in a five-fold cross validation. The number of expansion terms is selected from $\{1, 3, 5, 10, 15, 20, 30, 40, 50, 60, 70, 80, 90, 100\}$, the number of entities is selected from $\{1, 3, 5, 10, 15, 20, 30, 40, 50\}$ and the number of PRF documents is selected from $\{5, 10, 15, 20, 25, 30\}$.

Parameters of SVM in supervised expansion (`FbSVM` and `SVMPRF`) are selected by another five-fold cross validation. In each of the five folds of the outside cross validation that were used to select expansion parameters, we performed a second level cross validation to select the parameters of SVM. The explored range of cost c of linear kernel and RBF kernel is $\{0.01, 0.1, 1, 10, 100\}$. The range of γ in RBF kernel is $\{0.1, 1, 10, 100, 1000, 10000\}$.

To keep the experiment tractable, other parameters were fixed following conventions in previous work [14, 61, 82]. The weight of the original query w_q is set to 0.5, and the re-rank depth is 1000. We chose re-ranking instead of retrieval again in the whole index because the latter is very expensive with the large set of expansion terms and did not show any significant difference in our experiments. When using FACC1 annotations to link objects, we used the FACC1 annotations in the top 20 retrieved documents provide by `IndriLm`. The candidate terms for `FbSVM` were generated from the top 20 retrieved objects and top 20 linked FACC1 annotations. To reduce noise in object description, we ignored terms that contained less than three characters.

Evaluation Metric. Our methods re-ranked the top retrieved documents, so we mainly focus evaluation on the top 20 documents in the re-ranked list. We chose ERR@20 as our main evaluation metric, which is the main metric of the TREC Web Track adhoc task. We also show the evaluation results for MAP@20 and NDCG@20.

Table 3.2: Performance of unsupervised expansion using Freebase. Relative gain is calculated using ERR over IndriLm. Win/Loss/Tie is the number of queries helped, hurt, and not changed comparing with IndriLm. †, ‡, § and ¶ mark the statistic significant improvements ($p < 0.05$) over IndriLm, SDM, RmWiki and SVMPRF respectively. The best results in each column are marked **bold**.

Method	MAP@20	NDCG@20	ERR@20	Relative Gain	Win/Loss/Tie
IndriLm	0.357	0.147	0.116	NA	NA
SDM	$0.387^{\dagger}$	$0.166^{\dagger}$	$0.122^{\dagger}$	5.52%	58/27/115
RmWiki	0.362	$0.161^{\dagger}$	0.114	-1.70%	67/72/61
SVMPRF	0.367	$0.158^{\dagger}$	0.125	8.00%	63/72/65
FbSearchPRF	$\mathbf{0.436}^{\dagger,\ddagger,\S,\P}$	$\mathbf{0.186}^{\dagger,\ddagger,\S,\P}$	$\mathbf{0.152}^{\dagger,\ddagger,\S,\P}$	30.80%	84/30/86
FbSearchCat	$0.421^{\dagger,\ddagger,\S,\P}$	$0.182^{\dagger,\ddagger,\S,\P}$	$0.144^{\dagger,\ddagger,\S,\P}$	23.99%	67/43/90
FbFaccPRF	$0.428^{\dagger,\ddagger,\S,\P}$	$0.184^{\dagger,\ddagger,\S,\P}$	$0.145^{\dagger,\ddagger,\S,\P}$	24.71%	97/55/48
FbFaccCat	$0.400^{\dagger,\S,\P}$	$0.173^{\dagger}$	$0.136^{\dagger,\ddagger,\S}$	17.25%	88/67/45

Table 3.3: Query level Win/Loss/Tie comparison between unsupervised query expansion methods. Each cell shows the number of queries helped (Win), damaged (Loss) and not changed (Tie) by row method over column method.

	FbSearchPRF	FbSearchCat	FbFaccPRF	FbFaccCat
FbSearchPRF	NA/NA/NA	73/47/80	82/74/44	95/65/40
FbSearchCat	47/73/80	NA/NA/NA	72/86/42	87/72/41
FbFaccPRF	74/82/44	86/72/42	NA/NA/NA	84/67/49
FbFaccCat	65/95/40	72/87/41	67/84/49	NA/NA/NA

3.1.4 Evaluation Results

In this section, we first discuss the average performance of our unsupervised expansion methods using Freebase, comparing with current state-of-the-art baselines. Then we evaluate our supervised expansion method. Besides average performances, we also analysis our methods' robustness at the individual query level. We conclude our analysis with case studies and discussions.

3.1.4.1 Performance of Unsupervised Expansion

The average performances on MAP, NDCG and ERR are shown in Table 3.2. The relative gain and Win/Loss ratio are compared with IndriLm on ERR. Statistical significance tests are performed using the Permutation test. Labels †, ‡, § and ¶ indicate statistical significance ($p < 0.05$) over IndriLm, SDM, RmWiki and SVMPRF respectively.

Our unsupervised expansion methods outperform all state-of-the-art baselines by large margins for all evaluation metrics. All the gains over IndriLm are statistically significant, while SVMPRF and RmWiki are only significantly better on NDCG. Three of the methods, FbSearchPRF, FbSearchCat and FbFaccPRF, are significantly better than all baselines.

Table 3.4: Performance of supervised expansion using Freebase. Relative gain and Win/Loss/Tie are calculated comparing with `IndriLm` on ERR. †, ‡, § and ¶ mark the statistically significant improvements over `FbSearchPRF`, `FbSearchCat`,`FbFaccPRF` and `FbFaccCat` respectively. Best results in each column are marked **bold**.

Method	MAP@20	NDCG@20	ERR@20	Relative Gain	Win/Loss/Tie
`FbSearchPRF`	0.436	0.186	0.152	30.80%	84/30/86
`FbSearchCat`	0.421	0.182	0.144	23.99%	67/43/90
`FbFaccPRF`	0.428	0.184	0.145	24.71%	97/55/48
`FbFaccCat`	0.400	0.173	0.136	17.25%	88/67/45
`FbSVM`	**0.444**	**0.199**†,‡,§,¶	**0.165**‡,§,¶	42.42%	96/63/41

`FbFaccCat`'s improvements do not always pass the statistical significance test, even when the relative gains are almost 10%. This reflects the high variance of query expansion methods, which is addressed in Section 3.1.4.3.

Comparing the performances of our methods, linking objects by search works better than by FACC1 annotations, and selecting expansion terms by PRF works better than using category similarity. One possible reason is that objects from FACC1 annotation are noisier because they rely on the quality of top retrieved documents. Also, the category similarity suffers because suitable categories for query or terms may not exist.

We further compare our unsupervised expansion methods at the query level. The results are shown in Table 3.3. Each cell shows the comparison between the method in the row and the method in the column. The three numbers are the number of queries in which the row method performs better (win), worse (loss), and equally (tie) with the column method respectively. The results demonstrate that our methods do perform differently. The two most similar methods are `FbSearchPrf` and `FbSearchCat`, doing the same on 80 queries out of 200. But 36 queries have no returned objects from the Google Search API, on which two methods retreat to `IndriLm`. Otherwise, our four unsupervised methods perform the same for at most 49 queries.

These results showed the different strengths of our unsupervised methods. The next experiment investigates whether they can be combined for further improvements by a supervised method.

3.1.4.2 Performance of Supervised Expansion

The performance of our supervised method `FbSVM`, which utilized the evidence from our unsupervised methods, is shown in Table 3.4. To investigate whether the combination of multiple sources of evidence is useful, we conduct statistical significance tests between `FbSVM` with our unsupervised methods. †, ‡, § and ¶ indicates statistical significance in the permutation test over `FbSearchPRF`, `FbSearchCat`,`FbFaccPRF` and `FbFaccCat` correspondingly.

The results demonstrate that evidence from different aspects of Freebase can be combined for further improvements: `FbSVM` outperforms `IndriLm` by as much as 42%. Statistical significance is observed over our unsupervised methods on $NDCG$, but not always on MAP and ERR. We have also run statistical significance tests between `FbSVM` and all other baselines, which are all statistically significant as expected.

FbSVM and SVMPRF differ in their candidate terms and features. FbSVM selects terms from Freebase, while SVMPRF selects from web corpus. FbSVM uses features from Freebase's linked objects' descriptions and categories, while FbSVM uses term distribution and proximity in top retrieved documents from web corpus. Table 3.5 shows the quality of candidate terms from two sources. Surprisingly, Freebase' candidate terms are slightly weaker in quality (39.4% vs. 41.4%) and there are more of them. However, FbSVM's classification Precision is about 10% relatively better than SVMPRF, as shown in Table 3.6. The Recall of FbSVM is lower, but FbSVM still picks more good expansion terms given the larger number of good candidate terms in Freebase.

Nevertheless, the marginal gains of FbSVM over our best performing unsupervised method FbSearchPRF are not as high as expected. Our preliminary analysis shows that one possible reason is the features between query and terms are limited, i.e. only three dimensions. Another possible reason is the way of using the supervised information (document relevance judgments). Document relevance judgments are used to generate labels at the term level using heuristics, while the final document ranking is still computed using unsupervised retrieval models. A more powerful machine learning framework seems necessary to better utilize Freebase information. This would be a good topic for further research.

3.1.4.3 Query Level Analysis

A common disadvantage of query expansion methods is their high variances: they often hurt as many queries as helped. To evaluate the robustness of our methods, we compare the query level performance of each method versus IndriLm and record the Win/Loss/Tie numbers. The results are listed in the last columns of Tables 3.2 and 3.4. Table 3.2 shows that SDM, which is widely recognized as effective across many datasets, is reasonably robust and hurts only half as many queries as it helps. It also does not change the performance of 115 queries partly because 53 of them only contain one term on which nothing can be done by SDM. In comparison, RmWiki and SVMPRF hurt more queries than they help, which is consistent with observations in prior work [20].

Our methods have much better Win/Loss ratios than baseline query expansion methods. When selecting terms using PRF from linked objects' descriptions, FbSearchPRF and FbFaccPRF improve almost twice as many queries as they hurt. The variance of term selection by category is higher, but FbSearchCat and FbFaccCat still improve at least 30% more queries than they hurt. Linking by object retrieval has slightly better Win/Loss ratios than by FACC1 annotation, but it also helps a smaller number of queries. One reason is that for some long queries, there is no object retrieved by Google API.

More details of query level performance can be found in Figure 3.1. The x-axis is the bins of relative performances on ERR compared with IndriLm. The y-axis is the number of queries that fall into corresponding bins. If the performance is the same for a query, we put it into 0 bin. If a query is helped by 0 to 20%, we put it into bin 20%, etc. Figure 3.1 confirms the robustness of our methods. Especially for FbSearchPRF and FbFaccPRF, more queries are helped, fewer queries are hurt, and much fewer queries are extremely damaged.

FbSVM's robustness is average among our expansion methods, and is better than RmWiki and SDM. Fewer queries fall into bin 0, as it is rare that none of our evidence affects a query. However, the number of damaged queries is not reduced. One possible reason is that the ground

Table 3.5: Candidate term quality from top retrieved documents (Web Corpus) and linked objects' descriptions (Freebase). Good and bad refer to the number of terms that have positive and negative influences on ranking accuracy respectively.

Source	**Good**	**Bad**	**Good Fraction**
`Web Corpus`	$9,263$	$13,087$	41.4%
`Freebase`	$19,247$	$29,396$	39.6%

Table 3.6: Classification performance of supervised query expansion.

Method	**Precision**	**Recall**
`SVMPRF`	0.5154	0.0606
`FbSVM`	0.5609	0.0400

truth we used to train the SVM classifier is the individual performance of each candidate term, and only the average performance is considered in model training/testing. As a result, our model might focus more on improving average performance but not on reducing risk.

3.1.4.4 Case Study and Discussion

To further understand the properties of our object linking and term selection methods, Table 3.7 lists the queries that are most helped or hurt by different combinations of methods. The $\uparrow$ row shows the most helped queries and $\downarrow$ row shows those most hurt[2]. The comparison is done on `ERR` compared to `IndriLm` too.

Table 3.7 shows the different advantages of linking by object search and FACC1 annotation. For example, the query 'fybromyalgia' is damaged by `FbFaccPRF`, while improved by `FbSearchPRF` and `FbSearchCat`. The FACC1 annotation leads to a set of weakly related objects, like doctors and organizations focused on diseases, which generate overly-general expansion terms. Instead, object search is precise and returns the object about 'fybromyalgia'. Sometimes the generality of FACC1 annotations can help instead. For example, for query 'rock art' whose main topic is about rock painting, object search links to objects about rock music, while FACC1 annotation is more general and links to related objects for both rock painting and rock music.

Our two term selection methods also have different behaviors. An exemplary case is the query 'computer programming', on which `FbSearchPRF` and `FbFaccPRF` perform very well, while `FbSearchCat` and `FbFaccCat` do not. The linked objects of two methods are both reasonable: object search mostly links to programming languages, and FACC1 annotation brings in programming languages, textbooks, and professors. With the good quality of linked objects, PRF selects good expansion terms from their descriptions. However, category similarity picks terms like: 'analysis', 'science', 'application' and 'artificial', which are too general for this query. The granularity of the Freebase ontology's first level is too coarse for some queries, and lower levels are hard to use due to insufficient instances. Nevertheless, when the linked objects are

[2]More details including linked objects and expansion terms are available at `http://boston.lti.cs.cmu.edu/appendices/ICTIR2015/`.

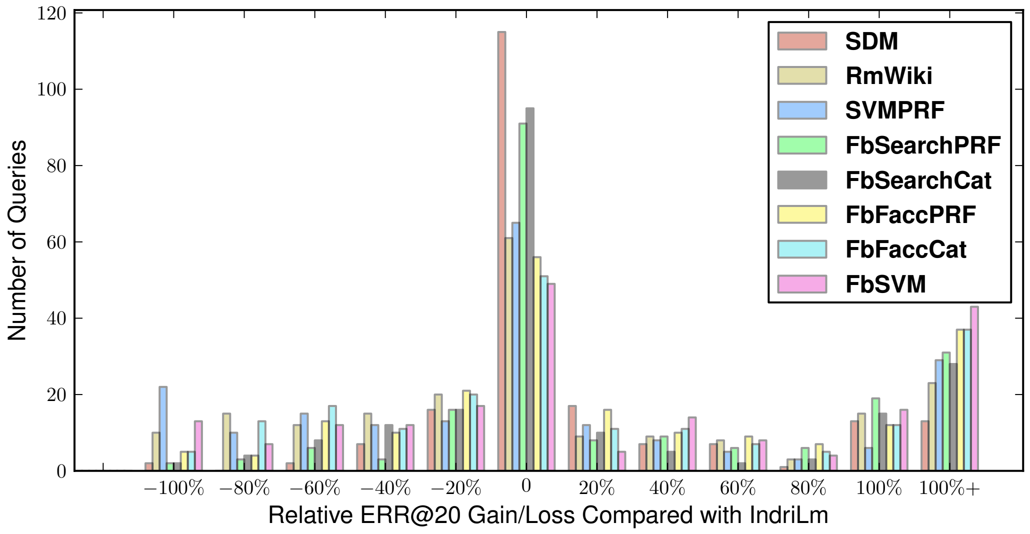


Figure 3.1: Query level relative performance. X-axis is the bins of relative performance on ERR compared with `IndriLm`. Y-axis is the number of queries that fall into each bin. Bin 0 refers to queries that were not changed, 20% refers to queries that improved between $(0\%, 20\%]$, etc. The left-to-right ordering of histograms in each cell corresponds to the top-to-bottom ordering of methods shown in the key.

noisy, like for the query ‘sore throat’, the category information helps pick more disease-related terms using the ‘/medicine’ category and provides better performance.

Some queries difficult for all methods. For example, ‘wedding budget calculator’ contains the entities ‘wedding’, ‘budget’ and ‘calculator’, but actually refers to the concept ‘wedding budget’ and how to calculate it. Similar cases are ‘tangible personal property tax’ and ‘income tax return online’, whose meanings cannot be represented by a single Freebase object.

There are also queries on which Freebase is very powerful. For example, the query ‘UNC’ asks for the campuses of the University of North Carolina. Freebase contains multiple objects about UNC campuses, and campuses of other related universities, which generate good expansion terms. Freebase is also very effective for ‘Figs’, ‘Atari’, ‘Hoboken’ and ‘Korean Language’, whose meanings are described thoroughly by linked Freebase objects.

To sum up, our object linking and term selection methods utilize different parts of Freebase, and thus have different specialties. In object linking, object search is aggressive and can return the exact object for a query, when there are no ambiguities. FACC1 annotation relies on top retrieved documents and usually links to a set of related objects. Thus, it is a better choice for queries with ambiguous meanings. In term selection, Pseudo Relevance Feedback via tf.idf directly reflects the quality of linked objects, and is better when the linked objects are reasonable. In contrast, category similarity offers a second chance to pick good expansion terms from noisy linked objects, when proper category definition exists for the query. `FbSVM` offers a preliminary way to combine the strength from different evidence and does provide additional improvements. Next in Chapter 3.2 we develop a more sophisticated method that better use supervision and richer evidence from Freebase, and further improves the ranking accuracy.

Table 3.7: The queries most helped and hurt by our methods. ↑ row shows the five most-helped queries for each method, and ↓ shows the most-hurt queries.

	`FbSearchPRF`	`FbSearchCat`
↑	porterville	unc
	hobby stores	porterville
	fybromyalgia	fybromyalgia
	computer programming	bellevue
	figs	figs
↓	von willebrand disease	rock art
	website design hosting	espn sports
	403b	ontario california airport
	ontario california airport	computer programming
	rock art	bobcat
	`FbFaccPRF`	`FbFaccCat`
↑	signs of a heartattack	porterville
	computer programming	idaho state flower
	figs	bellevue
	idaho state flower	flushing
	hip fractures	atari
↓	wedding budget calculator	poem in your pocket day
	poem in your pocket day	ontario california airport
	fybromyalgia	computer programming
	ontario california airport	bobcat
	becoming a paralegal	blue throated hummingbird

3.1.5 Summary of Query Expansion with Knowledge Base

In this work, we use Freebase, a large public knowledge base, to improve query representation using query expansion techniques. We investigate two methods of identifying the entities associated with a query, and two methods of using those entities to perform query expansion. A supervised model combines information derived from Freebase descriptions and categories to select terms that are effective for query expansion. Experiments on the ClueWeb09 dataset with TREC Web Track queries demonstrate that these methods are almost 30% more effective than strong, state-of-the-art query expansion algorithms. In addition to improving average performance, some of these methods have better win/loss ratios than baseline algorithms, with 50% fewer queries damaged. To the best of our knowledge, this work is the first to show the effectiveness of Freebase for query expansion on the widely used ClueWeb09 web corpus.

3.2 EsdRank: Connecting and Ranking Query-Documents with Query Entities

In last section we discussed our methods about finding better *term* based query representations with query expansion techniques. Effectively as they are, only the terms in entity descriptions are used to represent the query, and still, only term frequency and ontology information from Freebase are utilized. On the other hand, the information in knowledge bases is centered around *entities*. If we can directly represent queries by entities, it may be easier to fully utilize all information in knowledge bases to improve search engine's ranking accuracy.

This section presents `EsdRank`, a new technique that learns to enrich the query representation with entities and to rank documents in one unified framework. `EsdRank` treats entities from external knowledge base as latent objects connecting query and documents. The information from knowledge base used for better query representation is used as features between the query and entities. The document ranking evidence used in LeToR research is used as features between entities and documents. Instead of treating the features about query-entity and features about entity-document individually, `EsdRank` uses a *latent-listwise* LeToR model, Latent-ListMLE. The model treats the entities as a latent layer between query and documents, and learns how to handle the features between query, entities, and documents in one unified procedure directly from document relevance judgments.

One major challenge in using external data is to find related entities for a query and documents. Several methods have been used by prior research [26, 82] and by our prior work [79], but it is not clear how each of them contributes to final performance. This section explores three popular methods to select related objects from external data, including query annotation, entity search, and document annotation. To investigate their effectiveness, we apply EsdRank with related entities generated by these methods on one newer knowledge base, Freebase [10], and one classic controlled vocabulary, MeSH[3], which can also be considered as a knowledge base but of specific domain and smaller scale, in web search and medical search respectively. Experiments on TREC Web Track and OHSUMED datasets show EsdRank's significant effectiveness over state-of-the-art ranking baselines, especially when using entities from query annotations. Experiments also show that the effectiveness not only comes from the additional information from external data, but also our Latent-ListMLE model that uses it properly. This work is published in the proceedings of CIKM 2015 [78] and holds the current state-of-the-art in ranking accuracy on ClueWeb09-B and ClueWeb12-B13.

Section 3.2.1 discusses EsdRank, its Latent-ListMLE ranking model, related objects selection, and features. The experimental methodology and evaluation results are described in Sections 3.2.2 and 3.2.3 respectively. The last part of this section summarizes EsdRank's contributions.

3.2.1 EsdRank

EsdRank is intended to be a general technique for using external semi-structured data to improve ranking. External data elements are modeled as *objects*. An object could be a term from another

[3]http://www.nlm.nih.gov/mesh/meshhome.html

corpus or a knowledge base entity. The evidence from the query, external data, and the corpus is incorporated as features to express the relationship between query, object and document. A ranking model is then learned to rank documents with these objects and evidence.

In the first part of this section, we propose a novel latent listwise learning to rank model, Latent-ListMLE, as our ranking model. Latent-ListMLE handles the objects as a latent space between query and documents. The evidence between query-object and object-document is naturally expressed as features connecting the query to the latent space, and then connecting latent space to documents.

Prior research found that a major challenge in using external data is to find related objects [16, 26, 82]. In the second part of this section, we explore three popular related object selection methods used in prior research. In the final part of this section, we describe the features used to connect query, objects and documents.

3.2.1.1 Latent-ListMLE

Given a query q and an initial set of retrieved documents D, a set of objects $O = \{o_1, ..., o_j, ..., o_m\}$ related to q and D is produced by one of the related object selection methods as described in Section 3.2.1.2. Features that describe relationships between query q and object o_j are denoted as vector v_j. Features that describe relationships between object o_j and document d_i are denoted as u_{ij}. The goal of Latent-ListMLE, like other learning to rank methods, is to re-rank D, but with the help of the related objects O and feature vectors $U = \{u_{11}, ..., u_{ij}, .., u_{nm}\}$ and $V = \{v_1, ..., v_j, ..., v_m\}$.

Latent-ListMLE treats O as the latent space between q and D and uses V, U as features to describe the relationships between query-object, and object-document. We will first revisit ListMLE [77], the listwise learning to rank model which Latent-ListMLE is built upon. Then we discuss the construction, learning, and ranking of Latent-ListMLE.

ListMLE Revisited

ListMLE defines the probability of a ranking (a list) being generated by a query in a parametric model. Maximum likelihood estimation (MLE) is used to find parameters that maximize the likelihood of the best ranking(s). However, the sample space of all possible rankings is the permutation of all candidate documents D, which is too large. One contribution of ListMLE is that it reduces the sample space by assuming the probability of a document being ranked at position i is independent of those ranked at previous positions.

Specifically, with a likelihood loss and a linear ranking model, ListMLE defines the probability of a document d_i being ranked at position i as:

$$p(d_i|q, S_i) = \frac{\exp(w^T x_i)}{\sum_{k=i}^{n} \exp(w^T x_k)}, \tag{3.6}$$

where $S_i = \{d_i \dots d_n\}$ are the documents that were not ranked in positions $1 \dots i-1$, x_i is the query-document feature vector for d_i, and w is the parameter vector to learn.

Equation 3.7 defines the likelihood of a given ranking $\vec{D}$ of candidate documents.

$$p(\vec{D}|q;w) = \prod_{i=1}^{n} \frac{\exp(w^T x_i)}{\sum_{k=i}^{n} \exp(w^T x_k)}. \tag{3.7}$$

The parameter vector w is learned by maximizing the likelihood for all given queries q_k and their best rankings $\vec{D}_k^*$ given document relevance judgments:

$$\hat{w} = \arg\max_{w} \prod_{k} p(\vec{D}_k^*|q_k;w). \tag{3.8}$$

This is an unconstrained convex optimization problem that can be solved efficiently by gradient methods.

Latent-ListMLE Construction

Latent-ListMLE extends ListMLE by adding a latent layer in the ranking generation process. The latent layer contains related objects O as possible representations of the original query q.

With the latent layer O, the ideal generation process of a ranking $\vec{D}$ is to first sample a ranking of objects $\vec{O}$, and then sample document ranking $\vec{D}$ based on $\vec{O}$. However, this process is also impractical due to the huge sampling space. Similarly to ListMLE, we assume that the probabilities of picking objects and documents at each position are independent with those at previous positions. Thus, the generative process is redefined to be:

For each position from 1 to N:

1. Sample o_j from multinomial distribution $Multi(O|q)$, with probability $p(o_j|q)$; and
2. Sample d_i from multinomial distribution $Multi(S_i|o_j)$, with probability $p(d_i|o_j, S_i)$.

We further define:

$$p(o_j|q) = \frac{\exp(\theta^T v_j)}{\sum_{k=1}^{m} \exp(\theta^T v_k)} \tag{3.9}$$

$$p(d_i|o_j, S_i) = \frac{\exp(w^T u_{ij})}{\sum_{k=i}^{n} \exp(w^T u_{kj})}, \tag{3.10}$$

where v_j is the query-object feature vector for o_j; u_{ij} is the object-document feature vector between d_i and o_j; m is the number of objects; and θ and w are the model parameters.

In this generative process, the latent layer is the sampled objects produced by query q, and the document ranking probability is conditioned on the sampled objects instead of the query. With this extension, the probability of picking d_i at position i is:

$$p(d_i|q, S_i) = \sum_{j=1}^{m} p(d_i|o_j, S_i)p(o_j|q) \tag{3.11}$$

$$= \sum_{j=1}^{m} \frac{\exp(w^T u_{ij})}{\sum_{k=i}^{n} \exp(w^T u_{kj})} \frac{\exp(\theta^T v_j)}{\sum_{k=1}^{m} \exp(\theta^T v_k)} \tag{3.12}$$

and the probability of ranking $\vec{D}$ given q is:

$$p(\vec{D}|q; w, \theta) = \prod_{i=1}^{n} \sum_{j=1}^{m} p(d_i|o_j, S_i)p(o_j|q). \tag{3.13}$$

Latent-ListMLE also uses query-document features by adding a 'query node' o_0 to O that represents query q. The features relating query q to o_0 are set to 0, thus, o_0 is the 'origin point' for related objects. The features relating o_0 to documents are typical LeToR query-document features. The combination of query-document features and object-document features is done by treating them as individual dimensions in U, and setting missing feature dimensions' values to zero. So the dimensions referring to object-document similarities for the query node are set to zero, and vice versa.

Our idea of representing the query via related objects ($p(o|q)$) is similar to query expansion. In fact, if we use expansion terms as our related objects, and discard the document ranking part, Latent-ListMLE becomes a supervised query expansion method. On the other hand, if we only use the query node as the related object, Latent-ListMLE is exactly the same as ListMLE. The difference is that, in Latent-ListMLE, the connections from query to latent layer, and from latent layer to documents are learned together in one unified procedure, which finds the best combination of query representation ($p(o|q)$) and document ranking ($p(d|o)$) together, instead of only focusing on one of them.

Learning

The parameters w and θ are learned using MLE with given queries and their best rankings. To keep the notation clear, we present the derivation with one training query q, without loss of generality.

For a training query q and its best ranking $\vec{D}^*$ derived from relevance labels, their log likelihood is:

$$l(\vec{D}^*|q; w, \theta) = \log p(\vec{D}^*|q; w, \theta) \tag{3.14}$$

$$= \sum_{i=1}^{n} \log \sum_{j=1}^{m} \left(\frac{\exp(w^T u_{ij})}{\sum_{k=i}^{n} \exp(w^T u_{kj})} \times \frac{\exp(\theta^T v_j)}{\sum_{k=1}^{m} \exp(\theta^T v_k)} \right). \tag{3.15}$$

The goal of MLE is to find parameters w^*, θ^* such that:

$$w^*, \theta^* = \arg\max_{w,\theta} l(\vec{D}^*|q; w, \theta). \tag{3.16}$$

Directly maximizing Equation 3.15 is difficult due to the summation of the latent variables inside the log, thus we use the EM algorithm to solve this optimization problem.

The **E step** finds the posterior distribution of hidden variable o_j for each ranking position given the current parameters θ_{old} and w_{old}.

$$\pi(o_j|d_i, q) = p(o_j|d_i, q; \theta_{old}, w_{old})$$
$$= \frac{p(d_i|o_j, S_i; w_{old})p(o_j|q; \theta_{old})}{\sum_{k=1}^{m} p(d_i|o_k, S_i; w_{old})p(o_k|q; \theta_{old})}.$$

The **M step** maximizes the expected log complete likelihood.

$$
\begin{aligned}
E(\tilde{l}) =& E_{\pi(\vec{O}|\vec{D^*},q)} \log p(\vec{D^*}, \vec{O}|q; w, \theta) \\
=& \sum_{i=1}^{n} \sum_{j=1}^{m} \pi(o_j|d_i, q) \log p(d_i|o_j, S_i) p(o_j|q) \\
=& \sum_{i,j} \pi(o_j|d_i, q) \log \frac{\exp(w^T u_{ij})}{\sum_{k=i}^{n} \exp(w^T u_{kj})} \frac{\exp(\theta^T v_j)}{\sum_{k=1}^{m} \exp(\theta^T v_k)}.
\end{aligned}
$$

We use gradient ascent to maximize the expectation. The gradients are:

$$
\frac{\partial E(\tilde{l})}{\partial w} = \sum_{i,j} \pi(o_j|d_i, q) \{ u_{ij} - \frac{\sum_{k=i}^{n} u_{kj} \exp w^T u_{kj}}{\sum_{k=i}^{n} \exp w^T u_{kj}} \} \tag{3.17}
$$

$$
\frac{\partial E(\tilde{l})}{\partial \theta} = \sum_{i,j} \pi(o_j|d_i, q) \{ v_j - \frac{\sum_{k=1}^{m} v_k \exp(\theta^T v_k)}{\sum_{k=1}^{m} \exp(\theta^T v_k)} \}. \tag{3.18}
$$

The E step is very efficient with the closed form solution. The M step is an easy convex optimization problem. Intuitively, the E step finds the best assignment of object probabilities under current parameters and best document rankings, thus transferring document relevance information to latent objects. The M step learns the best parameters that fit the object probabilities provided by the E step. EM iterations are guaranteed to improve the likelihood until convergence. However, the overall optimization is not convex and has local optima. In practice, the local optima problem can be suppressed by repeating the training several times with random initial w and θ, and using the result that converges to the largest likelihood.

Ranking

Given a learned model, a query, and an initial ranking, a new ranking is constructed by picking the document that has the highest probability $p(d_i|q, S_i)$ at each position from 1 to n. The complexity of picking one document is $O(nm)$, thus the total cost of ranking n documents with m related objects is $O(n^2m)$, which is slower than ListMLE's $O(n \log n)$ ranking complexity. We can restrict the number of documents n (e.g. 100) and related objects m (e.g. < 5) to maintain reasonable efficiency.

3.2.1.2 Related Objects

How to find related objects O given query q and documents D is important for using external data. Many options have been proposed by prior research [26, 52, 62, 82], but it is not clear which is the most reliable. This research studies the following three popular automatic methods to find related objects.

Query Annotation selects the objects that directly appear in the query. Based on specific object types in the external data, one can choose corresponding techniques to 'annotate' them to the query, for example, entity linking techniques [16] to find entities that appear in the query, or all query terms when the objects are terms from an external corpus.

Object Search selects the objects that are textually similar to the query. Search engines can be used to find such objects. We can build an index for all the objects in external data, in which each document is the textual data about the object, for example, name, alias, description and context words of an entity. Then textually similar objects can be retrieved by running the query to the index.

Document Annotation selects the objects that appear in retrieved documents D. The terms in retrieved documents have been widely used in query expansion. The entities that are annotated to D were also useful in prior work [26]. This method introduces objects that are more indirectly related to the query, such as 'President of United States' for the query 'Obama family tree'. A typical method to score and select objects from retrieved documents is the RM3 pseudo relevance feedback model [45].

3.2.1.3 Features

Representing the relationship between query and related objects is the major focus of prior research in using external data. We explore the following query-object features in EsdRank.

Features between Query and Objects

Object Selection Score features are the scores produced by the object selection step: Annotator confidence, object ranking score, and RM3 model score $s(q, o)$.

Textual Similarity features cover the similarities between the query and the object's textual fields. For example, one can use coordinate matches, BM25 scores, language model scores, and sequential dependency model (SDM) scores between the query and the object's textual fields, such as name, alias (if any) and descriptions if using entities.

Ontology Overlap features use the ontology, a common type of information for some external semi-structured datasets. When an ontology is available, one can build a multiclass classifier that classifies the query into the ontology, and use the overlaps with the object's ontology as features.

Object Frequency is the number of documents in the corpus the object appears in (e.g. term) or is annotated to (e.g. entity). This feature is query independent and distinguishes frequent objects from infrequent ones.

Similarity with Other Objects are the max and mean similarity of this object's name with the other related objects'. These features distinguish objects that have similar names with other related objects from those that do not.

Features between Objects and Documents

In learning to rank, rich query-document ranking features, such as multiple retrieval algorithms on different document fields, have been shown very important for ranking performance [49]. Our object-document features are similar to query-document features widely used in LeToR research. But objects may have richer contents than queries, enabling a larger set of features.

Textual Similarity features measure the similarity of a document and an object's textual fields. BM25, language model, SDM, coordinate match, and cosine similarity in the vector space model are used to calculate similarities between all combinations of an object's textual fields and a document's fields. These retrieval models are applied by using the object to 'retrieve' the document. The fusion of these similarity features provides multiple ways to express the object-document similarities for EsdRank.

Ontology Overlap features are the same as the ontology overlap features between query and object. The same multiclass classifier can be used to classify documents, and overlaps in the object's and document's top categories can be used as features.

Graph Connection features introduce information about the relationships in the external data and document annotations. If such graph-like information is available, one can start from the object, traverse its relations, and record the number of annotated objects reached at each step (usually within 2 steps) as features. These features model the 'closeness' of the object and the document's annotations in the external data's relationship graph.

Document Quality features are commonly used in learning to rank. We use classic document quality features such as document length, URL length, spam score, the number of inlinks, stop word fraction, and whether the document is from Wikipedia (web corpus only).

3.2.1.4 Discussion

EsdRank provides general guidance about how to use external semi-structured data in ranking. When an external dataset is available, to use it in ranking, one can first use object selection methods to find related objects for each query, and then extract query-object and object-document features. Although the detailed object selection methods and features may vary for different datasets, we list some common related object selection methods and features that have been used widely in prior research to start with. One can also derive new related object selection methods and features based on other available information.

Related objects and features are handled by our latent listwise LeToR model, Latent-ListMLE. It models the objects as the latent space. As a result, the evidence is decoupled into a query-object part and an object-document part, which is easier to learn than mixed together (as shown in Section 3.2.3.2). Instead of solely focusing on the query-object part, or the document ranking part, Latent-ListMLE learns them together using EM. Our experiments show that the additional evidence from external data and Latent-ListMLE are both necessary for EsdRank's effectiveness.

3.2.2 Experimental Methodology

EsdRank is intended to be a general method of using external semi-structured data, and experiments test it with two types of semi-structured data and search tasks: Web search using the Freebase knowledge base, and medical search using the MeSH controlled vocabulary. The former uses a newer type of semi-structured data, a noisy corpus, and queries from web users; the latter uses a classic form of semi-structured data, a clean corpus, and queries from domain experts. Datasets and ranking features are determined by these two choices, as described below.

Data: Experiments on web search using Freebase were conducted with ClueWeb09-B and ClueWeb12-B13, two web corpora used often in IR research. Each corpus contains about 50 million English web documents. The 2009-2013 TREC Web Tracks provided 200 queries with relevance assessments for ClueWeb09 and 50 queries with relevance assessments for ClueWeb12. We used a snapshot of Freebase provided by Google on Oct 26th, 2014.[4]

Experiments on medical search using the MeSH controlled vocabulary were conducted with the OHSUMED corpus, a medical corpus used often in IR research. It contains abstracts of medical papers that are manually annotated with MeSH terms. The OHSUMED dataset includes 106 queries with relevance assessments. We used the 2014 MeSH dump provided by the U.S. National Library of Medicine (NIH).[5]

Indexing and Base Retrieval Model: The ClueWeb and OHSUMED corpora were indexed by the Indri search engine, using default stemming and stopwords. ClueWeb documents contain title, body and inlink fields. OHSUMED documents contain title and body fields.

Freebase and MeSH also were indexed by Indri, with each object (entity, controlled vocabulary term) treated as a document containing its associated text fields (name, alias, description). Objects without descriptions were discarded. Retrieval was done by Indri using its default settings.

Spam filtering is important for ClueWeb09. We removed the 70% spammiest documents using Waterloo spam scores [24]. Prior research questions the value of spam filtering for ClueWeb12 [26], thus we did not filter that dataset.

Related Objects: We implement three related generation methods as described in Section 3.2.1.2.

Query annotation (`AnnQ`) was done by TagMe [31], one of the best systems in the Short (Query) Track at the ERD14 workshop competition [16]. TagMe annotates the query with Wikipedia page ids. Wikipedia page ids were mapped to Freebase objects using their Wikipedia links and to MeSH controlled vocabulary terms using exact match.

Object search (`Osrch`) was done with the Indri search engine for both external semi-structured datasets. We investigated using Google's Freebase Search API to search for Freebase objects and PubMed's MeSH query annotations to annotate MeSH objects. Neither was significantly better than Indri search or TagMe annotation. Thus, we only report results with public solutions.

Google's FACC1 dataset provided annotations of Freebase entities in ClueWeb documents[6] [32]. The OHSUMED dataset contains MeSH annotations for each document. Document annotation (`AnnD`) objects are generated by first retrieving documents from the ClueWeb or OHSUMED corpus, and then using the RM3 relevance model [45] to select annotated objects from these documents.

Thus, the experiments consider query annotation (`EsdRank-AnnQ`), object search (`EsdRank-Osrch`) and document annotation (`EsdRank-AnnD`) methods to select related objects.

Feature Extraction: We extract features described in Section 3.2.1.3. Our feature lists are shown in Tables 3.8–3.10.

For object-document text similarity features, each object field is dynamically expanded with a virtual field that contains any query terms that the field does not contain. For example, given an

[4]https://developers.google.com/freebase/data
[5]http://www.nlm.nih.gov/mesh/filelist.html
[6]http://lemurproject.org/clueweb09/

Table 3.8: Query-object features. 'Web' and 'Medical' indicate the number of features in each corpus.

Feature	Web	Medical
Score from Query Annotation	1	1
Score from Object Search	1	1
Score from Document Annotation	1	1
Language model of object's description	1	1
SDM of object's description	1	1
BM25 of object's description	1	1
Coordinate match of object's text fields	3	3
Top 3 categories overlap	1	1
Object's idf in corpus's annotation	1	1
Similarity with other objects	2	2
Total Dimensions	13	13

object with name 'Barack Obama' and query 'Obama family tree', the text 'family tree' is added to the object as a virtual field called 'query minus name'. Similar expansion is performed for 'query minus alias' and 'query minus description' fields. As a result, Latent-ListMLE also knows which part of the query is not covered by an object. This group of features is very important for long queries, in order to make sure documents only related to a small fraction of the query are not promoted too much.

The ontology features use the linear multiclass SVM implementation of SVMLight [38] as the classifier. The Freebase classes are the 100 most frequent (in the FACC1 annotation) bottom categories in Freebase's two-level ontology tree. The classes for MeSH are all of the top level categories in MeSH's ontology. The training data is the descriptions of objects in each class. Objects are classified by their descriptions. Documents are classified by their texts. Queries are classified using voting by the top 100 documents that Indri retrieves for them [13]. The accuracy of Multiclass SVM in cross-validating on training data is above 70%.

The graph connection features between Freebase entities and ClueWeb documents are calculated using Freebase's knowledge graph. We treat all edges the same, leaving further exploration of edge type to future work. We only use the reachable at zero hop for OHSUMED, that is whether the MeSH term is annotated to the document.

Evaluation Metrics: We use ERR@20 and NDCG@20, which are the main evaluation metrics in the TREC Web Track ad hoc task. Besides these two metrics that focus on the top part of the ranking, we also use MAP@100, which considers the quality over the entire reranked section.

Statistical Significance was tested by the permutation test (Fisher's randomization test) with p-value < 0.05, which is a common choice in IR tasks.

Hyper-Parameters and Training: We follow standards in prior work to set hyper-parameters. The number of documents retrieved and re-ranked is set to 100. The same set of documents is used to generate document annotation `AnnD`. All supervised ranking models are evaluated on the testing folds in 10-fold cross validation. Cross-validation partitioning is done randomly and kept the same for all experiments. To suppress the local optima problem, in each

Table 3.9: Object-document features. ‘Web’ and ‘Medical’ indicate the number of features in each corpus. ‘Field combinations’ refers to combinations of an object’s and a document’s fields. An object’s fields include name, alias, description, query minus name, query minus alias, and query minus description. ClueWeb documents (for Fb) contain title, body and inlink fields. OSHUMED documents (for MeSH) contain title and body fields.

Feature	Web	Medical
Language model of field combinations	18	12
SDM of field combinations	18	12
Cosine correlation of field combinations	18	12
Coordinate match of field combinations	18	12
BM25 of field combinations	18	12
Top 3 cateogories overlap	1	1
Is annotated to document	1	1
Connected in graph at 1,2 hop	2	0
Total Dimensions	94	62

Table 3.10: Query-document and document quality features used by EsdRank and learning to rank baselines. ‘Web’ and ‘Medical’ indicate the number of features in each corpus.

Feature	Web	Medical
Language model of doc’s fields	3	2
SDM of doc’s fields	3	2
Cosine correlation of doc’s fields	3	2
Coordinate match of doc’s fields	3	2
BM25 of doc’s fields	3	2
Spam Score	1	0
Number of inlinks	1	0
Stop word fraction	1	1
URL length	1	0
Document length	1	1
Is Wikipedia	1	0
Total Dimensions	21	12

cross-validation fold Latent-ListMLE is trained ten times with random initialization. The training result with the largest likelihood on training data is used for testing. In order to maintain reasonable ranking efficiency and avoid too many noisy objects, the number of related objects from `AnnD` and `Osrch` are restricted to at most 3. This restriction does not change `AnnQ` as none of our queries has more than 3 annotations.

Baselines: The first baseline is the Sequential Dependency Model (`SDM`) [58] approach which is widely recognized as a strong baseline. For ClueWeb datasets, we find that Dalton et al’s SDM results obtained with Galago[7] are stronger than our SDM results obtained with In-

[7]http://ciir.cs.umass.edu/downloads/eqfe/runs/

dri, thus we use their SDM results as a baseline. We also use EQFE rankings provided by them as the second baseline on ClueWeb data sets. On OHSUMED, we use the Indri query language to obtain SDM results. EQFE is not included for OHSUMED as it is specially designed for knowledge bases. The third and fourth baselines are two state-of-the-art learning to rank baselines: RankSVM (pairwise) [39] and ListMLE (listwise) [77]. Their features are shown in Table 3.10 and are trained and tested exactly the same as EsdRank.

Three of our baseline algorithms were used widely in prior research; the fourth (EQFE) is included because it has important similarities to EsdRank. We could have included other methods that use external data, for example, Wikipedia for query expansion [12, 82], Wikipedia as one resource for query formulations [6], and MeSH as an alternate document representation [55]. These methods mainly focus on using external data to better represent the query; they rank documents with unsupervised retrieval models. In recent TREC evaluations, supervised LeToR systems that use rich ranking features have shown much stronger performance than unsupervised retrieval systems. Thus, we mainly compare with LeToR models with document ranking features, e.g., those in Table 3.10, which we believe are the strongest baselines available now.

3.2.3 Evaluation Results

This section first presents experiment results about EsdRank's overall accuracy. Then it investigates the effectiveness of our Latent-ListMLE model, together with the influence of related entity quality on EsdRank's performance.

3.2.3.1 Overall Performance

Tables 3.11a, 3.11b, and 3.11c show the retrieval accuracy of several baseline methods and three versions of EsdRank on ClueWeb09, ClueWeb12, and OHSUMED datasets. The three variants of EsdRank differ only in how related entities are selected. $\dagger, \ddagger$, § and ¶ indicate statistical significance over SDM, RankSVM, ListMLE and EQFE. The change relative to ListMLE is shown in parentheses. Win/Tie/Loss is the number of queries improved, unchanged or damaged as compared to ListMLE by ERR@20. The best performing method according to each evaluation metric is marked **bold**.

On all three data sets, the best EsdRank methods outperform all baselines on all metrics. The gain over ListMLE varies from 3.33% (OHSUMED, MAP@100) to 21.85% (ClueWeb09, ERR@20), with 5–7% being typical. We note that ListMLE is a very strong baseline; there is little prior work that provides statistically significant gains of this magnitude over ListMLE on the ClueWeb datasets. These improvements show the effectiveness of external data in ranking, which is EsdRank's only difference with ListMLE.

On ClueWeb data sets, all three versions of EsdRank outperform EQFE, and the best performing version improves over EQFE by 10%-30%. Both EQFE and EsdRank use Freebase as external data and learning to rank models to rank documents. The features between query and object in EsdRank are very similar with those used in EQFE. Why does EsdRank perform better than EQFE?

One difference is in the object-document features. EQFE uses the language model or SDM score between an entity's textual fields and the whole document to connect the entity to the

Table 3.11: Performance of EsdRank. AnnQ, Osrch and AnnD refer to object selection methods: Query annotation, object search and document annotation. Relative changes compared to ListMLE are shown in parentheses. Win/Tie/Loss is the number of queries improved, unchanged or hurt compared to ListMLE. †, ‡, § and ¶ show statistic significance ($p < 0.05$ in permutation test) over SDM, RankSVM, ListMLE and EQFE. The best method for each evaluation metric is marked **bold**.

(a) ClueWeb09

Method	**MAP@100**	**NDCG@20**	**ERR@20**		**Win/Tie/Loss**
SDM	0.375	0.214	0.135	(−13.52%)	75/30/95
EQFE	0.364	0.213	0.139	(−11.21%)	77/27/96
RankSVM	$0.352^{\dagger}$	$0.193^{\dagger}$	0.147	(−6.10%)	59/37/104
ListMLE	$0.410^{\dagger,\ddagger,\P}$	$0.221^{\dagger,\ddagger}$	$0.156^{\dagger}$	–	–
EsdRank-AnnQ	$\mathbf{0.437}^{\dagger,\ddagger,\S,\P}$	$\mathbf{0.237}^{\dagger,\ddagger,\S,\P}$	$\mathbf{0.190}^{\dagger,\ddagger,\S,\P}$	(**21.85%**)	88/42/70
EsdRank-Osrch	$0.397^{\dagger,\ddagger,\P}$	$0.219^{\dagger,\ddagger}$	$0.155^{\dagger}$	(−0.75%)	71/47/82
EsdRank-AnnD	$0.403^{\dagger,\ddagger,\P}$	$0.221^{\dagger,\ddagger}$	$0.167^{\dagger,\P}$	(6.70%)	75/36/89

(b) ClueWeb12

Method	**MAP@100**	**NDCG@20**	**ERR@20**		**Win/Tie/Loss**
SDM	0.293	0.126	0.091	(−25.66%)	18/7/25
EQFE	0.319	$0.146^{\dagger}$	0.106	(−13.61%)	17/7/26
RankSVM	$0.383^{\dagger}$	$0.161^{\dagger}$	0.114	(−7.48%)	16/13/21
ListMLE	$0.397^{\dagger,\P}$	$0.174^{\dagger,\P}$	$0.123^{\dagger}$	–	–
EsdRank-AnnQ	$0.410^{\dagger,\P}$	$0.181^{\dagger,\P}$	$\mathbf{0.133}^{\dagger,\ddagger,\S,\P}$	(**8.39%**)	20/12/18
EsdRank-Osrch	$0.391^{\dagger,\P}$	$0.167^{\dagger}$	$0.121^{\dagger}$	(−1.26%)	22/10/18
EsdRank-AnnD	$\mathbf{0.440}^{\dagger,\ddagger,\S,\P}$	$\mathbf{0.186}^{\dagger,\ddagger,\S,\P}$	$0.132^{\dagger,\P}$	(7.67%)	24/14/12

(c) OHSUMED

Method	**MAP@100**	**NDCG@20**	**ERR@20**		**Win/Tie/Loss**
SDM	0.413	0.332	0.453	(−6.14%)	38/11/57
RankSVM	0.396	0.336	0.453	(−6.12%)	43/14/49
ListMLE	0.414	0.343	0.483	–	–
EsdRank-AnnQ	$\mathbf{0.428}^{\ddagger,\S}$	$\mathbf{0.355}^{\dagger,\ddagger,\S}$	$\mathbf{0.511}^{\dagger,\ddagger,\S}$	(**5.77%**)	56/16/34
EsdRank-Osrch	$0.427^{\ddagger,\S}$	0.347	$0.489^{\ddagger}$	(1.29%)	44/17/45
EsdRank-AnnD	0.416	0.342	$0.500^{\dagger,\ddagger}$	(3.59%)	44/20/42

document, while EsdRank uses a much larger feature set (Table 3.9). The object-document features in Table 3.9 provide multiple different views of object-document relevancy, which has been shown very effective in LeToR research [49]. More object-document features also better propagate document relevance judgments to latent objects in Latent-ListMLE's EM training procedure, and help it learn better weights for query-object features. We have tried EsdRank with only language model scores between an object's textual fields and the document as object-document features, however, results with this smaller feature set are no better than EQFE. In

Section 3.2.3.2, our second experiment also shows that our Latent-ListMLE model is another essential factor for EsdRank's performance with its ability to handle features in the two parts properly.

On ClueWeb09 and OHSUMED, query annotation (EsdRank-AnnQ) performs the best with statistically significant improvements on almost all evaluations over all baselines, indicating query annotation is still the most effective in finding reliable objects. On ClueWeb12, EsdRank-AnnQ also outperforms all baselines, although less statistical significance is observed due to fewer training queries (only 50). It is interesting to see that EsdRank-AnnD performs better than EsdRank-AnnQ on ClueWeb12. This is consistent with results of EQFE [26], showing that FACC1 annotation might be of better quality for ClueWeb12 queries than ClueWeb09 queries. EsdRank-Osrch performs the worst on all three data sets. The reason is that object search provides related entities that might be textually similar to the query, but are actually noise. The ranking models designed to rank document use assumptions that may not be suitable for objects, leaving room for improvement in future research.

The different effectiveness of EsdRank with different related object selection methods shows the importance of related object quality. It is also well recognized as one of the most essential factors in determining the usefulness of external data in prior research [16, 28, 52, 79]. In Section 3.2.3.3, our third experiment studies the correlation between related entities' quality and EsdRank's performance.

3.2.3.2 Effectiveness of Latent-ListMLE

To investigate the effectiveness of Latent-ListMLE, in this experiment, we compare it with ListMLE, using the same external evidence, but with features generated by feature engineering techniques similar to those in Dalton et al. [26].

Formally, for each query q and document d_i pair, we have objects $\{o_1, ..., o_j, ..., o_m\}$, query-object features $V = \{v_1, ..., v_j, ..., v_m\}$ and object-document features $U_i = \{u_{i1}, ..., u_{ij}, ..., u_{im}\}$. Let $|u|$ and $|v|$ be the feature dimensions for query-object features and object-document features. We generate $|u| \times |v|$ features $x_i = \{x_i(1,1), ..., x_i(k_1,k_2), ..., x_i(|u|,|v|)\}$ by enumerating all combination of features in U_i and V. The combination of k_1-th feature in V and k_2-th feature U_i is defined as: $x_i(k_1,k_2) = \sum_j v_j(k_1) \times u_{ij}(k_2)$, the summation of corresponding features over all possible related entities. One may notice that if we only use language model scores as object-document features, this set up is exactly the same as EQFE. We name these 'flat' features in comparison with Latent-ListMLE's latent space model.

We put these 'flat' features into ListMLE and use the same training-testing procedure to evaluate them on our datasets. Evaluation results are shown in Table 3.12. ListMLE-AnnQ, ListMLE-Osrch and ListMLE-AnnD refer to the flat model with features from related entities selected by query annotation, object search and document annotation respectively. Relative performance (in brackets) and Win/Tie/loss values are compared with ListMLE, which uses the same model but query-document features. $\triangledown$ indicates significantly worse performance than ListMLE in the permutation test ($p < 0.05$). ▼ indicates significantly worse performance compared to the corresponding version of EsdRank. For example, ListMLE-AnnQ is significantly worse than EsdRank-AnnQ if marked by ▼.

Most times these flat features hurt performance, with significant drops compared to ListMLE and corresponding EsdRank versions. Even on ClueWeb09, where the most training data is available, it is typical for flat features to perform more than 10% worse than EsdRank. These results demonstrate the advantage of using a latent space learning to rank model instead of a flat model to handle external evidence: By modeling related entities as hidden variables in latent space, Latent-ListMLE naturally separates query-object and object-document evidence, and uses the EM algorithm to propagate document level training data to the object level. As a result, Latent-ListMLE avoids feature explosion or confusing machine learning algorithms with highly correlated features, while still only requiring document relevance judgments.

Table 3.12: Performance of ListMLE with flat features derived from external data. AnnQ, Osrch and AnnD refer to object selection methods: Query annotation, object search and document annotation. Relative changes and Win/Tie/Loss are compared to ListMLE. $\triangledown$ and $\blacktriangledown$ indicate statistic significantly weaker performance compared to ListMLE and EsdRank (which uses the same information but Latent-ListMLE).

(a) ClueWeb09

Method	MAP@100	NDCG@20	ERR@20
ListMLE-AnnQ	$0.389^{\blacktriangledown}$ (-4.96%)	$0.208^{\blacktriangledown}$ (-5.88%)	$0.170^{\blacktriangledown}$ (8.99%)
ListMLE-Osrch	$0.331^{\triangledown,\blacktriangledown}$ (-19.33%)	$0.168^{\triangledown,\blacktriangledown}$ (-23.79%)	$0.127^{\triangledown,\blacktriangledown}$ (-18.34%)
ListMLE-AnnD	$0.357^{\triangledown,\blacktriangledown}$ (-12.79%)	$0.197^{\triangledown,\blacktriangledown}$ (-10.63%)	0.155 (-0.61%)

(b) ClueWeb12

Method	MAP@100	NDCG@20	ERR@20
ListMLE-AnnQ	$0.355^{\blacktriangledown}$ (-10.71%)	$0.137^{\triangledown,\blacktriangledown}$ (-21.04%)	$0.098^{\triangledown,\blacktriangledown}$ (-20.37%)
ListMLE-Osrch	0.368 (-7.43%)	0.154 (-11.32%)	0.105 (-14.39%)
ListMLE-AnnD	$0.337^{\triangledown,\blacktriangledown}$ (-15.06%)	$0.149^{\triangledown,\blacktriangledown}$ (-14.38%)	$0.098^{\triangledown,\blacktriangledown}$ (-20.65%)

(c) OSHUMED

Method	MAP@100	NDCG@20	ERR@20
ListMLE-AnnQ	$0.388^{\triangledown,\blacktriangledown}$ (-6.13%)	$0.323^{\triangledown,\blacktriangledown}$ (-5.64%)	$0.444^{\triangledown,\blacktriangledown}$ (-7.94%)
ListMLE-Osrch	$0.400^{\blacktriangledown}$ (-3.37%)	$0.323^{\triangledown,\blacktriangledown}$ (-5.58%)	$0.445^{\triangledown,\blacktriangledown}$ (-7.84%)
ListMLE-AnnD	$0.397^{\triangledown,\blacktriangledown}$ (-4.12%)	0.339 (-1.14%)	0.494 (2.43%)

3.2.3.3 Influence of Related Entity Quality

Prior research [16, 28, 52, 79] and our evaluation results on EsdRank with different related entities all indicate the great influence of selected objects' quality on ranking accuracy. The third experiment further studies its influence on EsdRank's performance. Although directly measuring the quality is hard due to the lack of object level labels, we can indirectly characterize the quality by comparing the related entities selected by different methods, with the hypothesis that objects selected by multiple object selectors are more likely to have higher quality.

For each variant of EsdRank, queries were divided into two groups (Agree, Differ) based on whether the query had at least one related object in common with another variant.

The two groups were compared using the same methodology reported earlier. The results are summarized in Table 3.13. Each row is the performance of one variant (first column), divided by overlaps with another variant (second column). Relative gains, Win/Tie/Loss, and statistical significance (§) are calculated based on comparison with `ListMLE`. Results for ClueWeb09, ClueWeb12, and OSHUMED queries are combined due to space limitations and because their trends are similar.

The two groups clearly perform differently. On queries with common objects (`Agree`), `EsdRank` is much more effective. For example, when the less effective `AnnD` and `Osrch` object selectors agree with the more effective `AnnQ` selector, they both outperform `ListMLE` with statistical significance. When they differ, they may be worse than `ListMLE`. Although `AnnQ` is the most effective individual method, its performance is further improved when it agrees with `AnnD`, outperforming `ListMLE` by more than 10% on all metrics. This level of improvement is close to the gains reported by Liu et al. [52] with *manual* query annotations.

The results also show that `EsdRank` behaves predictably. When simple object selectors agree, `EsdRank` is more likely to improve retrieval quality; when they disagree, the search engine can simply retreat to `ListMLE`.

Finding related entities for query and document is still an open problem in the literature. In the SIGIR ERD'14 workshop [16], many teams built their query annotators upon TagMe and obtained about 60% accuracy. It is still not clear how to adapt full-text retrieval models to perform better object search. Document annotation can be done with very high precision but there is still room to improve recall, even on the widely used Google FACC1 annotations. Some prior research questions whether current Freebase query annotation is too noisy to be useful [28, 52]. With the help of features between query, object and document, `Latent-ListMLE` is able to improve retrieval accuracy even when object selection is noisy. As researchers improve the quality of object search and annotation, producing less noisy objects, we would expect `Latent-ListMLE`'s ranking accuracy to improve further.

3.2.4 EsdRank Summary

EsdRank is a general method of using external semi-structured data in modern LeToR systems. It uses objects from external data, such as vocabularies and entities, as an interlingua between query and documents. Query-object and object-document relationships are expressed as features. Latent-ListMLE treats objects as latent layers between query and document, and learns how to use evidence between query, objects and documents in one unified procedure. This general method can be applied to a variety of semi-structured resources, and is easily trained using ordinary query-document relevance judgments.

Experiments with two rather different types of external semi-structured data – a classic controlled vocabulary and a newer knowledge base – and three well-known datasets show that EsdRank provides statistically significant improvements over several state-of-the-art ranking baselines.

Experiments with the single-layer LeToR model, which uses exactly the same information but expressed by 'flat' features, show much weaker performance than Latent-ListMLE. This result confirms the advantage of separating evidence about query-object and object-document relationships with the latent space, instead of mixing them together. The single-layer approach

Table 3.13: The performances of `EsdRank`'s variants when they agree or differ with other variants, combined from three data sets. 'Agree' is defined by at least one overlap in related entities. Relative gain, Win/Tie/Loss and statistical significance § are compared with `ListMLE` on corresponding queries. The best relative gains for each variant are marked **bold**.

(a) EsdRank-AnnQ

Compare With	Group	MAP@100	NDCG@20	ERR@20	Win/Tie/Loss
`AnnD`	`Agree`	$0.541^{\S}$ (**14.80%**)	$0.299^{\S}$ (**12.89%**)	$0.254^{\S}$ (**24.18%**)	54/14/26
	`Differ`	0.390 (0.90%)	0.252 (2.45%)	$0.286^{\S}$ (8.14%)	110/56/96
`Osrch`	`Agree`	$0.495^{\S}$ (7.10%)	$0.328^{\S}$ (6.69%)	$0.347^{\S}$ (7.53%)	74/17/39
	`Differ`	0.393 (3.75%)	$0.227^{\S}$ (4.28%)	$0.238^{\S}$ (15.31%)	90/53/83

(b) EsdRank-Osrch

Compare With	Group	MAP@100	NDCG@20	ERR@20	Win/Tie/Loss
`AnnD`	`Agree`	0.490 (1.81%)	0.287 (1.66%)	0.210 (0.08%)	29/9/21
	`Differ`	0.388 (−1.70%)	0.243 (−0.56%)	0.258 (0.45%)	108/65/124
`AnnQ`	`Agree`	$0.482^{\S}$ (**4.41%**)	$0.320^{\S}$ (**4.02%**)	$0.340^{\S}$ (**5.37%**)	72/16/42
	`Differ`	0.361 (−4.81%)	0.210 (−3.53%)	0.198 (−4.08%)	65/58/103

(c) EsdRank-AnnD

Compare With	Group	MAP@100	NDCG@20	ERR@20	Win/Tie/Loss
`Osrch`	`Agree`	0.504 (4.71%)	0.292 (3.49%)	$0.233^{\S}$ (11.41%)	30/7/22
	`Differ`	0.394 (−0.28%)	0.244 (0.09%)	0.267 (3.92%)	113/63/121
`AnnQ`	`Agree`	$0.527^{\S}$ (**11.69%**)	$0.285^{\S}$ (**7.74%**)	$0.248^{\S}$ (**21.19%**)	50/12/32
	`Differ`	0.371 (−4.12%)	0.241 (−1.99%)	0.266 (0.47%)	93/58/111

may result in a large and highly correlated feature space, which is hard for machine learning models to deal with.

Finding related entities for query and documents is a very important step for using external data in ranking. EsdRank is tested with three popular related entity selection methods: query annotation, entity search, and document annotation. The evaluation results demonstrate the essential effect of related entity quality on final ranking accuracy, and suggest that query annotation is the most reliable source for related entities under current techniques. Section 3.3 further discusses our work in finding better related information from knowledge bases for queries with entity search.

3.3 Learning to Rank Related Entities

Previous studies in this chapter demonstrate the influence of related entities' quality to knowledge based information retrieval systems' performance. However, entity linking and entity search are both on-going research topics themselves. There is a huge room to improve in both techniques in order to acquire better entities to represent query.

This section presents our research about using entity search to find better related entities. Prior state-of-the-arts represent each entity as a structured document by grouping RDF triples of an entity in knowledge base into fields [2, 88] or a tree [53]. Then entities can be retrieved by conventional document retrieval algorithms such as BM25, query likelihood, or sequential dependency models. However, the problem is far from solved. The current P@10 of prior state-of-the-art [88] is at most 25%, introducing many noises in our knowledge based query representations. In this preliminary work, we study whether it is possible to utilize supervisions to improve entity search accuracy. We represent entities following the previous state-of-the-art's multi-field representations [88], extracts text similarity features for query-entity pairs, and use learning to rank models trained on entity's relevance judgments to rank entities.

Experimental results on an entity search test collection based on DBpedia [2] confirm that learning to rank is as powerful for entity ranking as for document ranking, and significantly improves the previous state-of-the-art. The results also indicate that learning to rank models with text similarity features are especially effective on keyword queries. The work presented in this section is done by collaborating with Jing Chen, a master student in MIIS program, and will appear in SIGIR 2016 as a short paper [18].

3.3.1 Related Work in Entity Search

Ad-hoc entity search was originally a task widely studied in the semantic web community, and recently draws attentions from information retrieval researchers as knowledge bases became popular. A major challenge in entity search discovered prior research is how to represent the entities. In knowledge bases, there are many kinds of information available for each entity, stored by RDF triples with different types of predicates. Recently Neumayer et al. found it more effective to just group the facts into two fields: title (name) and content (all others), and use standard field based document retrieval models such as BM25F and fielded language model [60]. Balog and Neumayer later produced a test collection that combines various previous entity search benchmarks, with different search tasks including single entity retrieval, entity list retrieval, and natural language question answering [2]. They also provide thorough experimental studies of baselines. Their experiment results demonstrate that in ad-hoc entity search, it is effective to combine entities' facts into fields of virtual documents, treat entity search as a document retrieval problem, and apply field based document retrieval models.

More recently, Zhiltsov et al. introduce the widely used fielded sequential dependence model (FSDM) to entity search [88]. They combine entities' facts into five manually defined fields. The five-fielded representation groups entities' RDF triples more structurally than only using a name field and a body field, and is also not too complex for learning to rank models to learn. FSDM achieves the state-of-the-arts with FSDM on multiple entity search benchmarks, especially on keyword queries [88]. On natural language question queries, Lu et al. shows that it is more effective to keep the original graph structure of knowledge bases as question queries are more complex than short keyword queries [53]. They parse the question into a tree structure, and finds the answer by mapping the parsed question tree to entity's facts, very similar to knowledge based question-answer technique (OpenQA) [4, 84]. As our focus is more on keyword queries, this work follows ad-hoc entity search methods [2, 88] and introduces the widely used and effective learning to rank techniques from document ranking to entity search.

Another related work is Schuhmacher et al. [69], in which the authors propose a new task of ranking entities specially for web queries, from web search's point of view. They use learning to rank model to rank the entities that are annotated to top retrieved documents for a web query, instead of retrieving entities directly from the knowledge base. In this way, it is not directly comparable with prior work about entity search but more related to our EsdRank work. Their findings and studied evidence are also valuable for our future research about finding better related entities to improve document ranking.

3.3.2 Learning to rank entities

The first question in entity search is how to represent entities. We follow Zhiltsov et al. [88] and group RDF triples into five fields: `Name`, which contains the entity's names; `Cat`, which contains its categories; `Attr`, which contains all attributes except name; `RelEn`, which includes the names of its neighbor entities; and `SimEn`, which contains its aliases. We only include RDF triples whose predicates are among the top 1,000 most frequent in DBpedia in the fields [2].

In state-of-the-art learning to rank systems for document ranking, most features are the scores of common unsupervised ranking algorithms applied to different document representations (fields). The different ranking algorithms and representations provide different views of the relevance of the document to the query. The multiple perspectives represented by these features are the backbone of any learning to rank system.

This approach can be applied to entity search by extracting features for query-entity pairs. We use the following ranking algorithms on each of an entity's five fields: `Language model` with Dirichlet smoothing ($\mu = 2500$), `BM25` (default parameters), `coordinate match`, `cosine correlation`, and `sequential dependency model (SDM)`. We also include Zhiltsov et al's. [88] `fielded sequential dependency model (FSDM)` score for the full document as a feature. As a result, there are in total 26 features as listed in Table 3.15.

With features extracted for all query-entity pairs, all learning to rank models developed for ranking documents can be used to rank entities. We use two widely-used LeToR models: `RankSVM`, which is an SVM-based pairwise method, and `Coordinate Accent`, which is a gradient-based listwise method that directly optimizes mean average precision (MAP). Both of these LeToR algorithms are robust and effective on a variety of datasets.

3.3.3 Experimental Methodology

This section describes our experiment methodology in studying the effectiveness of learning to rank in entity search.

Dataset: Our experiments are conducted on the entity search test collection provided by Balog and Neumayer [2], which others also have used for research on entity retrieval [53, 88].

The dataset has 485 queries with relevance judgments on entities from DBpedia version 3.7. These queries come from seven previous competitions and are merged into four groups based on their search tasks [88]. Table 3.14 lists the four query groups used in our experiments.

Base Retrieval Model: We use the `fielded sequential dependency model` (`FSDM`) as the base retrieval model to enable direct comparison to prior work [88]. All learning to rank methods are used to rerank the top 100 entities per query retrieved by `FSDM`, as provided by Zhiltov et al. [88].

Ranking Models: `RankSVM` implementation is provided by SVMLight toolkit[8]. `Coordinate Ascent` implementation is provided by RankLib[9]. Both methods are trained and tested using five fold cross validation. We use linear kernel in `RankSVM`. For each fold, hyper-parameters are selected by another five fold cross validation on the *training* partitions only. The 'c' of `RankSVM` is selected from range $1 - 100$ using step size of 1. The number of random restarts and iterations of `Coordinate Ascent` are selected from range $1 - 10$ and $10 - 50$ respectively using step size of 1.

Baselines: The main baseline is `FSDM`, the state-of-the-art for the benchmark dataset [88]. We also include `SDM-CA` and `MLM-CA` results as they perform well in this test coolection [88].

Evaluation Metrics: All methods are evaluated by MAP@100, P@10, and P@20 following previous work [88]. We also report NDCG@20 because it provides more details. Statistical significance tests are performed by Fisher Randomization (permutation) tests with $p < 0.05$.

Table 3.14: Entity Search Query Sets.

Query Set	Queries	Search Task
`SemSearch ES`	130	Retrieve one entity
`ListSearch`	115	Retrieve a list of entities
`INEX-LD`	100	Mixed keyword queries
`QALD-2`	140	Natural language questions

Table 3.15: Query-Entity features used in Learning to Rank Entity.

Features	Dimension
FSDM	1
SDM on all fields	5
BM25 on all fields	5
Language model on all fields	5
Coordinate match on all fields	5
Cosine on all fields	5

Table 3.16: Performance of learning to rank methods on entity search. Relative improvements over FSDM are shown in parentheses. Win/Tie/Loss show the number of queries improved, unchanged and hurt, comparing with FSDM. All section combines the evaluation results of the other four sections. $\dagger, \ddagger, \S$ indicate statistical significance over SDM-CA, MLM-CA, and FSDM respectively. The best method for each metric is marked **bold**.

	SemSearch ES					
	MAP@100	**P@10**	**P@20**	**NDCG@20**		**Win/Tie/Loss**
SDM-CA	0.254	0.202	0.148	0.355	(-29.5%)	26/15/89
MLM-CA	$0.320^{\dagger}$	$0.250^{\dagger}$	$0.178^{\dagger}$	$0.443^{\dagger}$	(-12.0%)	30/32/68
FSDM	$0.386^{\dagger\ddagger}$	$0.286^{\dagger\ddagger}$	$0.203^{\dagger\ddagger}$	$0.503^{\dagger\ddagger}$	-	-
RankSVM	$\mathbf{0.410}^{\dagger\ddagger\S}$	$\mathbf{0.304}^{\dagger\ddagger\S}$	$\mathbf{0.213}^{\dagger\ddagger\S}$	$\mathbf{0.527}^{\dagger\ddagger\S}$	(+4.7%)	65/27/38
Coor-Ascent	$0.396^{\dagger\ddagger}$	$0.295^{\dagger\ddagger}$	$0.206^{\dagger\ddagger}$	$0.511^{\dagger\ddagger}$	(+1.5%)	48/32/50
	ListSearch					
	MAP@100	**P@10**	**P@20**	**NDCG@20**		**Win/Tie/Loss**
SDM-CA	0.197	0.252	0.202	0.296	(+1.7%)	55/23/37
MLM-CA	0.190	0.252	0.192	0.275	(-5.3%)	39/28/48
FSDM	0.203	0.256	0.203	0.291	-	-
RankSVM	$0.224^{\dagger\ddagger\S}$	$\mathbf{0.303}^{\dagger\ddagger\S}$	$\mathbf{0.235}^{\dagger\ddagger\S}$	$\mathbf{0.332}^{\dagger\ddagger\S}$	(+14.3%)	61/23/31
Coor-Ascent	$\mathbf{0.225}^{\dagger\ddagger\S}$	$0.300^{\dagger\ddagger\S}$	$0.229^{\dagger\ddagger\S}$	$0.328^{\dagger\ddagger\S}$	(+12.9%)	62/21/32
	INEX-LD					
	MAP@100	**P@10**	**P@20**	**NDCG@20**		**Win/Tie/Loss**
SDM-CA	$0.117^{\ddagger}$	0.258	0.199	0.284	(-0.9%)	43/7/50
MLM-CA	0.102	0.238	0.190	0.261	(-8.8%)	34/13/53
FSDM	$0.111^{\ddagger}$	$0.263^{\ddagger}$	$0.214^{\dagger\ddagger}$	$0.287^{\ddagger}$	-	-
RankSVM	$\mathbf{0.126}^{\ddagger\S}$	$\mathbf{0.282}^{\ddagger}$	$\mathbf{0.231}^{\dagger\ddagger\S}$	$\mathbf{0.317}^{\dagger\ddagger\S}$	(+10.6%)	55/9/36
Coor-Ascent	$0.121^{\ddagger\S}$	$0.275^{\ddagger}$	$0.224^{\dagger\ddagger}$	$0.306^{\dagger\ddagger\S}$	(+6.7%)	53/7/40
	QALD-2					
	MAP@100	**P@10**	**P@20**	**NDCG@20**		**Win/Tie/Loss**
SDM-CA	0.184	0.106	0.090	$0.244^{\ddagger}$	(-6.8%)	36/66/38
MLM-CA	0.152	0.103	0.084	0.206	(-21.3%)	17/78/45
FSDM	$0.195^{\ddagger}$	$0.136^{\dagger\ddagger}$	$0.111^{\ddagger}$	$0.262^{\ddagger}$	-	-
RankSVM	$0.197^{\ddagger}$	$0.136^{\dagger\ddagger}$	$0.113^{\dagger\ddagger}$	$0.266^{\ddagger}$	(+1.6%)	31/74/35
Coor-Ascent	$\mathbf{0.208}^{\ddagger}$	$\mathbf{0.141}^{\dagger\ddagger}$	$\mathbf{0.115}^{\dagger\ddagger}$	$\mathbf{0.278}^{\ddagger}$	(+5.9%)	40/71/29
	All					
	MAP@100	**P@10**	**P@20**	**NDCG@20**		**Win/Tie/Loss**
SDM-CA	0.192	0.198	0.155	0.294	(-13.1%)	160/111/214
MLM-CA	0.196	0.206	0.157	0.297	(-12.1%)	120/151/214
FSDM	$0.231^{\dagger\ddagger}$	$0.231^{\dagger\ddagger}$	$0.179^{\dagger\ddagger}$	$0.339^{\dagger\ddagger}$	-	-
RankSVM	$\mathbf{0.246}^{\dagger\ddagger\S}$	$\mathbf{0.251}^{\dagger\ddagger\S}$	$\mathbf{0.193}^{\dagger\ddagger\S}$	$\mathbf{0.362}^{\dagger\ddagger\S}$	(+7.0%)	212/133/140
Coor-Ascent	$0.245^{\dagger\ddagger\S}$	$0.248^{\dagger\ddagger\S}$	$0.189^{\dagger\ddagger\S}$	$0.358^{\dagger\ddagger\S}$	(+5.7%)	203/131/151

3.3.4 Evaluation Results

We first present experimental results for learning to rank on entity search. Then we provide analysis of the importance of features and fields, and the influence of different query tasks on LeToR models.

3.3.4.1 Overall Performance

The ranking performances of learning to rank models are listed in Table 3.16. We present results separately for each query group and also combine the query groups together, shown in the `All` section of Table 3.16. Relative performances over `FSDM` are shown in parenthesis. †, ‡, § indicate statistical significance over `SDM-CA`, `MLM-CA`, and `FSDM` respectively. The best performing method for each metric is marked **bold**. Win/Tie/Loss are the number of queries improved, unchanged and hurt, also compared with `FSDM`.

The results demonstrate the power of learning to rank for entity search. On all query sets and all evaluation metrics, both learning methods outperform `FSDM`, defining a new state-of-the-art in entity search. The overall improvements on all queries can be as large as 8%. On `SemSearch ES`, `ListSearch` and `INEX-LD`, where the queries are keyword queries like 'Charles Darwin', LeToR methods show significant improvements over `FSDM`. However, on `QALD-2`, whose queries are questions such as 'Who created Wikipedia', simple text similarity features are not as strong.

Similar trends are also found in individual query performances. Figure 3.2 compares the best learning method, `RankSVM`, with `FSDM` at each query. The x-axis lists all queries, ordered by relative performance. The y-axis is the relative performance of `RankSVM` over `FSDM` on NDCG@20. On keyword queries more than half queries are improved while only about a quarter of queries are hurt. On questions (`QALD-2`), about the same number of queries are improved and hurt. A more effective method of handling natural question queries is developed recently by Lu et al. in which queries are parsed using question-answering techniques [53]. That method achieves 0.25 in P@10, but performs worse than `FSDM` on keyword queries. Section 3.3.4.3 further studies the influence of query types on entity-ranking accuracy.

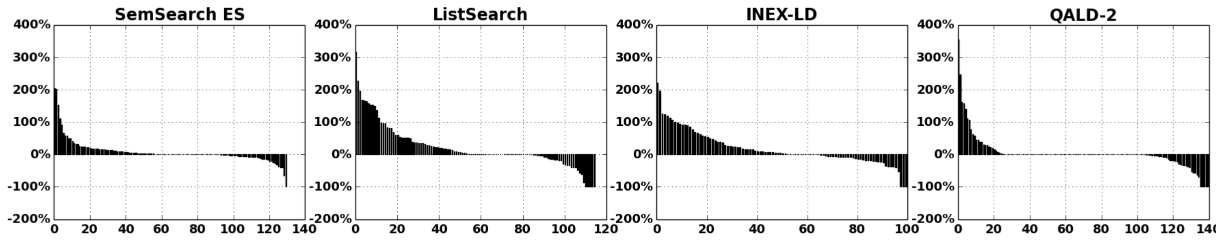
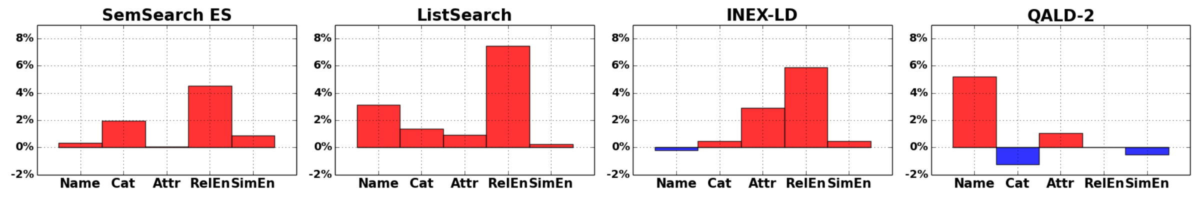
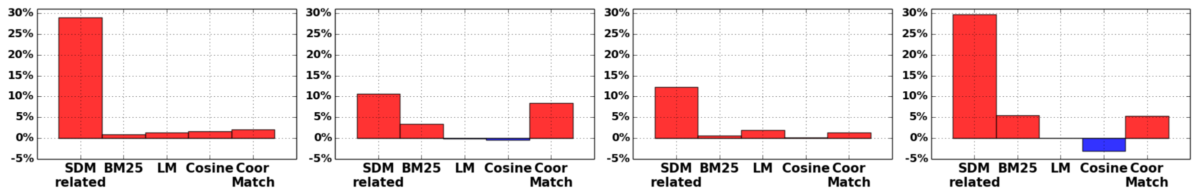


Figure 3.2: Query level relative performance in four query groups. X-axis lists all queries, ordered by relative performance. Y-axis is the relative performance of `RankSVM` comparing with `FSDM` in NDCG@20. Positive value indicates improvement and negative value indicates loss.

[8]`https://www.cs.cornell.edu/people/tj/svm_light/svm_rank.html`
[9]`http://sourceforge.net/p/lemur/wiki/RankLib/`



(a) Influence of fields



(b) Influence of feature groups

Figure 3.3: The contribution of fields and feature groups to `RankSVM`'s performance. X-axis lists the fields or feature groups. Y-axis is the relative NDCG@20 difference between `RankSVM` used with all fields or feature groups and without corresponding field or feature group. Larger values indicate more contribution.

3.3.4.2 Field and Feature Study

The second experiment studies the contribution of fields and feature groups to learning to rank models. For each field or feature group, we compare the accuracy of models when used without field or features from that group to those with all features. The change in accuracy indicates the contribution of the corresponding field or feature group. The field and feature studies for `RankSVM` are shown in Figures 3.3a and 3.3b respectively. The x-axis is the field or feature group studied. The y-axis is the performance difference between the two conditions (All versus held out). Larger values indicate greater contributions. Figure 3.3a organizes features by the fields they are extracted from, including `Name`, `Cat`, `Attr`, `RelEn`, and `SimEn`. Figure 3.3b organizes features into five groups, with one retrieval model per group. `SDM related` contains `FSDM` and `SDM` scores as they are very correlated.

Figure 3.3a shows that `RankSVM` favors different fields for different query sets. The `Name` field is useful for `ListSearch` and `QALD-2`, but does not contribute much to the other two query sets. `RelEn` provides the most gain to keyword queries, but is not useful at all for the natural language question queries in `QALD-2`. For feature groups we find that `SDM related` features are extremely important and provide the most gains across all query sets. This result is expected because all of the queries are relatively long queries and often contain phrases, which is where SDM is the most useful.

3.3.4.3 Query Type Differences

Previous experiments found that when different models are trained for different types of queries, each model favors different types of evidence. However, in live query traffic different types of queries are mixed together. The third experiment investigates the accuracy of a learning to rank entities system in a more realistic setting. The four query sets are combined into one query set, and new models are trained and tested using five fold cross validation as before.

Table 3.16 `All` shows the average accuracy when different models are trained for each of the four types of query. Table 3.17 shows the accuracy when a single model is trained for all types of queries. Despite being trained with more data, both learning to rank algorithms produce less effective models for the diverse query set than for the four smaller, focused query sets. Nonetheless, a single learned model is as accurate as the average accuracy of four carefully-tuned, query-set-specific `FSDM` models.

This results suggests that diverse query streams may benefit from query classification and type-specific entity ranking models. They may also benefit from new types of features or more sophisticated ranking models.

Table 3.17: Performance of learning to rank models when queries from different groups are all trained and tested together. Relative performances and Win/Tie/Loss are compared with the same model but trained and tested separately on each query group.

	`All Queries Mixed Together`				
	MAP@100	**P@10**	**P@20**	**NDCG@20**	**Win/Tie/Loss**
`RankSVM`	0.234	0.238	0.185	0.347 (-4.3%)	153/136/196
`Coor-Ascent`	0.233	0.232	0.179	0.344 (-3.8%)	148/129/208

3.3.5 Summary of Related Entities Finding

We use learning to rank models, the state-of-the-art in document ranking, for more accurate entity search. Entities are represented by multi-field documents constructed from RDF triples. How well a query matches an entity document is estimated by text similarity features and a learned model. Experimental results on a standard entity-oriented test collection demonstrate the power of learning to rank in entity retrieval. Statistically significant improvements over the previous state-of-the-art are observed on all evaluation metrics. Further analysis reveals that query types play an important role in the effectiveness of learned models. Text similarity features are very helpful for keyword queries, but less effective with longer natural language question queries. Learned models for different query types favor different entity fields because each query type targets different RDF predicates. The work in this section is our first step to finding better-related entities for information retrieval systems. Chapter 5 discusses our plans to continue this line of work.

3.4 Summary

The focus of this chapter is to enrich query representation with information from knowledge bases and to better retrieval related entities for a given query. We presented several query expansion methods that select expansion terms from the descriptions of retrieved entities and document annotations, using pseudo relevance feedback and ontology similarities with the query. Then we developed the EsdRank framework to represent query directly with entities, and a latent learning to rank model that jointly reasons the connection from query to query entities and document ranking. The experiments in both works demonstrate the advantage of using knowledge bases in query representation. But we also found the quality of related entities is a bottleneck of our systems' performances. In the last section, we show that related entity finding can be improved using learning to rank methods, which also motivated some of our proposed work in Chapter 5. Before proposing our research plans, Chapter 4 shifts the focus to document representations and study how to utilize the semantics in knowledge bases to help understand and represent texts in longer documents.

Chapter 4

Knowledge Based Document Representation

In last chapter, we demonstrate the effectiveness of representing query with terms and entities from knowledge bases. In both systems, the documents are represented by bag-of-words, and their relevance is modeled by their textual similarities with the query's enriched term based or entity based representations. In this chapter, we continue our exploration of knowledge based representation for documents, which contain longer text, and richer but more complex information than queries.

Recall that in the earliest information retrieval systems, query and documents were both represented by terms manually picked from predefined controlled vocabularies [67]. The controlled vocabulary representation conveys clean and distilled information and can be ranked accurately by simple methods. However, its usage is mainly limited to specific domains because manual annotations are required and the scale of classic controlled vocabulary dataset is usually small. Now that knowledge bases have grown to rather a large scale, and automatic entity annotation is made possible by entity linking research [16], it is time to reconsider whether a pure entity based representation can provide decent ranking performance.

This section presents a new bag-of-entities based representation for document ranking, as a heritage of the classic controlled vocabulary based representation, but with the aid of modern large scale knowledge bases and automatic entity linking systems. We represent query and documents by their bag-of-entities constructed from the annotations provided by three entity linking systems: Google's `FACC1` [32] with high precision, `CMNS` [36] with high recall, and `TagMe` [31] with balanced precision and recall. With the deeper text understanding provided by entity linking, documents can be ranked by their overlap with the query in the entity's explicit semantic space.

To investigate the effectiveness of bag-of-entities representations, we conducted experiments with a state-of-the-art knowledge base, Freebase, and two large scale web corpora, ClueWeb09-B and ClueWeb12-B13, together with their queries from the TREC Web Track. Our evaluation results first confirm that current entity linking systems can provide sufficient coverage over general domain queries and documents. Then we compare bag-of-entities with bag-of-words in standard document ranking tasks and demonstrate that although the accuracy of entity linking is not perfect (about $50\% - 60\%$ on TREC Web Track queries), the ranking performance can be improved by as much as 18% with bag-of-entities representations. This work is a preliminary work in knowledge based document representation and is under review as a short paper in ICTIR 2016 [81].

4.1 Bag-of-Entities Representation

We construct bag-of-entities representations for query and documents using several entity linking systems. When annotating texts, an entity linking system does not just match the n-grams with entity names, but makes the decision jointly by also considering external evidence such as the entity's descriptions and relationships in the knowledge base, and corpus statistics like commonness, linked probability and contexts of entities. As a result, representing texts by entities naturally incorporates deeper text understanding from the linking process: Entity synonyms are aligned, polysemy in entity mentions is disambiguated, and global coherence between entities is incorporated.

The choice of the knowledge base in this paper is Freebase, one of the largest public knowledge bases frequently used in recent IR research [26, 51, 78, 79]. Several entity linking systems have been developed for it. This work explores the following three popular ones to annotate query and documents with Freebase entities.

FACC1 is the Freebase annotation of TREC queries and ClueWeb corpora provided by Google [32]. It aims to achieve high precision, which is believed to be around 80-85%, based on a small-scale human evaluation[1].

TagMe is a system [31] widely used in entity linking competitions [16] and applications [78]. It balances precision and recall, both at about 60% in various evaluations.

CMNS is an entity linking system that spots texts using surface forms from the FACC1 annotation, and links all of them to their most frequently linked entities [36]. It can achieve almost 100% recall on some query entity linking datasets, but the precision may be lower [36].

Given the annotations of a query or document, we construct its bag-of-entities vector $\vec{E}_q$ or $\vec{E}_d$, in which each dimension ($\vec{E}_q(e)$ or $\vec{E}_d(e)$) refers to an entity e in Freebase, and its weight is the frequency of that entity being annotated to the query or document.

The bag-of-entities representation uses entities as its basic information unit. As with controlled vocabulary terms, entities are more informative than words. However, whereas controlled vocabulary terms are often assigned manually, entity linking is done automatically, which is more efficient but also more uncertain. With controlled vocabularies, simple ranking methods such as Boolean retrieval work well, because the representation is clean and distilled [67], while with bag-of-words more sophisticated ranking models are more effective. Given the heritage to the classic controlled vocabulary based search systems, we start simple and use the following two basic ranking models to study the power of bag-of-entities representations.

Coordinate Match (`COOR`) ranks a document by the number of query entities it contains:

$$f_{\text{COOR}}(q,d) = \sum_{e:\vec{E}_q(e)>0} \mathbb{1}(\vec{E}_d(e) > 0) \tag{4.1}$$

[1]`http://lemurproject.org/clueweb09/FACC1/`

Entity Frequency (EF) ranks document by the frequency of query entities in it:

$$f_{\text{EF}}(q, d) = \sum_{e:\vec{E}_q(e)>0} \vec{E}_q(e) \log(\vec{E}_d(e)) \tag{4.2}$$

$f_{\text{COOR}}(q, d)$ and $f_{\text{EF}}(q, d)$ are the ranking scores of document d for query q using coordinate match (COOR) and entity frequency (EF) respectively. $\mathbb{1}(\cdot)$ is the indicator function.

COOR performs Boolean retrieval, which is the most basic ranking method and often works well with controlled vocabularies. EF studies the value of term frequency information, another basis for document ranking. These simple models investigate basic properties of ranking with bag-of-entities, and provide understanding and intuition for the future development of more advanced ranking models.

4.2 Experiment Methodology

This section provides details about our experiments to study the quality of annotations and the effectiveness of bag-of-entities in ranking.

Dataset: Our experiments are conducted on TREC Web Track datasets. TREC Web Tracks use two large web corpora: ClueWeb09 and ClueWeb12. We use the ClueWeb09-B and ClueWeb12-B13 subsets. There are 200 queries with relevance judgments from TREC 2009-2012 for ClueWeb09, and 100 queries in 2013-2014 for ClueWeb12. Manual annotations provided by Dalton et al. [26] and Liu et al. [51] are used as query annotation labels.

Indexing and Base Retrieval Model: We indexed both corpora with Indri. Default stemming and stopword removal were used. Spam in ClueWeb09 was filtered using the default threshold (70%) for Waterloo spam scores. Spam filtering was not used for ClueWeb12 because its effectiveness is unclear. Indri's language model with default Dirichlet smoothing ($\mu = 2500$) was used to retrieve 100 documents per query for the evaluation of entity linking and ranking.

Entity Linking Systems: We used TagMe software provided by Ferragina et al. [31] to annotate queries and documents. It annotates text with Wikipedia entities. Following Xiong et al. [78], we aligned Wikipedia entities to Freebase entities using the Wikipedia ID field in Freebase.

CMNS is implemented by ourselves, following Hasibi et al. [36]. The boundary overlaps of surface forms are resolved by only linking the earliest and then the longest one [16, 36].

FACC1 entity annotations for ClueWeb documents are provided by Google [32]. They also annotated ClueWeb09 queries' intent descriptions, but not the queries. We used the descriptions' annotations as approximations of queries' annotations, and manually filtered out entities that did not appear in the original queries to reduce disturbance. ClueWeb12's queries are not annotated by Google so we are only able to study FACC1 annotations on ClueWeb09.

Baselines: We used two standard unsupervised bag-of-words ranking models as baselines: Indri's unstructured language model (Lm) and sequential dependency model (SDM), both with default parameters: $\mu = 2500$ for Lm, and query weights $(0.8, 0.1, 0.1)$ for SDM. Typically these baselines do well in competitive evaluations such as TREC. There are better rankers, for example learning to rank methods. However, methods that combine many sources of evidence usually

Table 4.1: Coverage of annotations. Freq and Dens are the average number of entities linked per query/document and per word respectively. Missed is the percentage of queries or documents that have no annotation at all. ClueWeb12 queries do not have FACC1 annotations as they are published later than FACC1.

(a) ClueWeb09

	Query			**Document**		
	Freq	Dens	Missed	Freq	Dens	Missed
FACC1	0.42	0.20	62%	15.95	0.13	30%
TagMe	1.54	0.70	1%	92.31	0.20	2%
CMNS	1.50	0.69	1%	252.41	0.55	0%

(b) ClueWeb12

	Query			**Document**		
	Freq	Dens	Missed	Freq	Dens	Missed
FACC1	NA	NA	NA	24.52	0.06	26%
TagMe	1.77	0.57	0%	246.76	0.37	0%
CMNS	1.75	0.55	0%	324.37	0.48	0%

Table 4.2: Entity linking performance on ClueWeb queries. All methods are evaluated by **Prec**ison, **Rec**all and **F1**.

	ClueWeb09 Query			**ClueWeb12 Query**		
	Prec	**Rec**	**F1**	**Prec**	**Rec**	**F1**
FACC1	0.274	0.236	0.254	NA	NA	NA
TagMe	0.581	0.597	0.589	0.460	0.555	0.503
CMNS	0.577	0.596	0.587	0.485	0.575	0.526

outperform methods that use a single source of evidence; such comparison would not reveal much about the value of the bag-of-entities representation.

There were three representations for ClueWeb09 (FACC1, TagMe and CMNS), and two for ClueWeb12 (TagMe and CMNS). COOR and EF were used to *re-rank* the top 100 documents per query retrieved by Lm. Ties were broken by Lm's score.

Evaluation Metrics: The entity annotations were evaluated by the lean evaluation metric from Hasibi et al. [36]. The ranking performance was evaluated by the TREC Web Track Adhoc Task's official evaluation metrics: ERR@20 and NDCG@20. Statistical significance was tested by the Fisher randomization test (permutation test) with $p < 0.05$.

4.3 Evaluation Results

This section evaluates the accuracy and coverage of entity annotations and bag-of-entities' performance in ranking.

4.3.1 Annotation Accuracy and Coverage

Table 4.2 shows the precision, recall, and F1 of `FACC1`, `TagMe` and `CMNS` on ClueWeb queries. `TagMe` performs the best on ClueWeb09 queries with higher precision, while `CMNS` performs better on ClueWeb12 queries. The ClueWeb09 queries are more ambiguous because they needed to support the TREC Web Track's Diversity task; `TagMe`'s disambiguation was more useful on this set. ClueWeb12 queries needed to support risk minimization research, and have been shown to be harder; both systems perform worse on them. `FACC1` query annotation does not perform well as its goal was to annotate the query's description, not the query itself.

There is no gold standard entity annotation for ClueWeb documents, preventing a quantitative evaluation. Nevertheless, our manual examination confirms that `FACC1` has high precision; `TagMe` performs a little better on documents with more contexts; and `CMNS` performs worse than `TagMe` on documents as it only uses the surface forms.

One concern of controlled vocabulary based search systems is the low coverage on general domain queries, restricting their usage mainly to specific domains. With the much larger scale of current knowledge bases, it is interesting to study whether their coverage is sufficient to influence the majority of general domain queries. Table 4.1 shows the coverage results of our entity annotations on ClueWeb queries and their top 100 retrieved documents. `Freq` and `Dens` are the average number of entities linked per query/document, and per word respectively. `Missed` is the percentage of queries/documents that have no linked entity. The results show that `TagMe` and `CMNS` have good coverage on ClueWeb queries and documents. Almost all queries and documents have at least one linked entity. The annotations are no longer sparse. There can be up to 324 entities linked per documents on average. However, precision and coverage have not been achieved together yet. `FACC1` has the highest precision but provides very few entities per document and misses many documents.

These results show that the entity linking itself is still an open research problem. Precision and coverage can not yet be achieved at the same time. Thus, the ranking method must be robust and able to accommodate a noisy representation.

4.3.2 Ranking Performance

The ranking performances of bag-of-entities based ranking models are shown in Table 4.3. `EF` and `COOR` are used to rerank top retrieved documents using the bag-of-entities representations built from `FACC1`[2], `TagMe` and `CMNS`. The percentages are relative performances over `SDM`. W/T/L refer to the number of queries improved (Win), unchanged (Tie) and hurt (Loss) comparing with `SDM`.

On ClueWeb09, both `TagMe` and `CMNS` work well with `EF` and `COOR`, and outperform all baselines on all evaluation metrics. The best method, `TagMe-EF`, outperforms `SDM` as much as 18% on ERR@20. On ClueWeb12, `COOR` outperforms all baselines on all evaluation metrics by about 12%. These results demonstrate that even with current imperfect entity linking systems, bag-of-entities is a valuable representation on which very basic ranking models can significantly outperform standard bag-of-words based ranking.

[2]`FACC1` annotations not available for ClueWeb12 queries.

Table 4.3: Ranking performance of bag-of-entity based ranking models. `FACC1`, `TagMe` and `CMNS` refer to the bag-of-entity representation constructed from corresponding annotations. `COOR` and `EF` refer to the coordinate match and entity frequency ranking models. Percentages show the relative changes compared to `SDM`. **Win/Tie/Loss** are the number of queries improved (Win), unchanged (Tie) and hurt (Loss) comparing with `SDM`. † and ‡ indicate statistic significance ($p < 0.05$ in permutation test) over `Lm` and `SDM`. The best method for each metric is marked **bold**.

(a) ClueWeb09

	NDCG@20		**ERR@20**		**Win/Tie/Loss**
`Lm`	0.176	-12.92%	0.119	-5.23%	39/88/71
`SDM`	0.202†	–	0.126†	–	–
`FACC1-COOR`	0.173	-14.16%	0.126	-0.21%	64/56/78
`FACC1-EF`	0.167	-17.32%	0.116	-8.14%	63/51/84
`TagMe-COOR`	0.211†	4.55%	0.133†	5.55%	108/35/55
`TagMe-EF`	**0.229**†,‡	**13.71%**	**0.149**†	**18.04%**	96/24/78
`CMNS-COOR`	0.210†	4.08%	0.131†	4.21%	105/37/56
`CMNS-EF`	0.216†	6.97%	0.136	7.52%	97/22/79

(b) ClueWeb12

	NDCG@20		**ERR@20**		**Win/Tie/Loss**
`Lm`	0.106	-2.10%	0.086	-4.69%	28/29/43
`SDM`	0.108	–	0.090	–	–
`FACC1-COOR`	NA	NA	NA	NA	NA
`FACC1-EF`	NA	NA	NA	NA	NA
`TagMe-COOR`	0.117†,‡	8.35%	0.095†	5.02%	42/20/38
`TagMe-EF`	0.107	-0.90%	0.091	1.08%	42/18/40
`CMNS-COOR`	**0.120**†,‡	**11.03%**	0.101†	11.20%	43/22/35
`CMNS-EF`	0.110	2.03%	**0.102**	**12.71%**	36/20/44

Table 4.4: Performances of bag-of-entity based ranking models on queries that are correctly annotated and not correctly annotated, compared with manual annotations. The relative performance and **Win/Tie/Loss**are calculated by comparing with `SDM` on the same query group. † and ‡ indicate statistic significance ($p < 0.05$ in permutation test) over `Lm` and `SDM`. The best method for each metric is marked **bold**.

(a) Correctly Annotated Query

	NDCG@20		**ERR@20**		**Win/Tie/Loss**
`TagMe-COOR`	$0.214^{\dagger,\ddagger}$	14.56%	$0.153^{\dagger,\ddagger}$	12.71%	59/16/17
`TagMe-EF`	$\mathbf{0.243}^{\dagger,\ddagger}$	**30.43%**	$\mathbf{0.178}^{\dagger,\ddagger}$	**31.19%**	53/10/29
`CMNS-COOR`	$0.211^{\dagger}$	4.28%	$0.146^{\dagger}$	4.89%	53/22/20
`CMNS-EF`	$0.240^{\dagger,\ddagger}$	18.44%	0.168	20.74%	52/11/32

(b) Mistakenly Annotated Query

	NDCG@20		**ERR@20**		**Win/Tie/Loss**
`TagMe-COOR`	$0.200^{\dagger}$	−3.28%	0.111	−1.93%	49/21/38
`TagMe-EF`	0.209	0.65%	0.118	4.30%	43/16/49
`CMNS-COOR`	$\mathbf{0.201}^{\dagger}$	**3.90%**	**0.112**	**3.40%**	52/17/36
`CMNS-EF`	0.185	−4.09%	0.100	−8.12%	45/13/47

Bag-of-entities' representation power correlates with the entity linking system's accuracy. ClueWeb09 queries are more ambiguous, favoring `TagMe` in annotation accuracy, and `TagMe` provides the most improvements when ranking for ClueWeb09 queries. ClueWeb12 queries have lower annotation quality, and bag-of-entities based ranking is not as powerful as on ClueWeb09 queries. `FACC1`'s coverage is too low and can not well represent the documents. This also explains why prior research mainly uses it as a pool to select query entities [26, 78, 79].

Entity linking is a rapidly developing area; improvement in the future is likely. To study how the bag-of-entities can benefit from improvements in annotation accuracy, we used the manual query annotations [26, 51] to divide queries into two groups: *Correctly Annotated*, and *Mistakenly Annotated*. Better ranking performance is expected for Correctly Annotated queries. Table 4.4 shows the ranking performances of `TagMe` and `CMNS` on the two groups in ClueWeb09. We omit `FACC1` as it always hurts, and ClueWeb12 queries as there are not enough queries in either group to provide reliable observations. The relative performance, W/T/L and statistical significance over `SDM` are calculated on the same queries. The results are as expected: On Correctly Annotated queries, bag-of-entities provides more accurate ranking; On Mistakenly Annotated queries, the improvements are smaller, and sometimes bag-of-entities reduces accuracy.

Our experiments show that intuitions developed for bag-of-words representations do not necessarily apply directly to bag-of-entities representations. A long line of research shows that frequency based (e.g., tf.idf) ranking models are superior to Boolean ranking models. Thus, one might expect `EF` to provide consistently more accurate ranking than `COOR`, however, that is not the case in our experiments. We found that the majority of annotation errors are missed annotations, which makes entity frequency counts less reliable. However, it is rare for the entity linker to miss every mention of an important entity in a document, thus, the Boolean representation is robust to this majority type of errors.

We also examined the effectiveness of other ranking intuitions, such as inverse document frequency (idf) and document length normalization. In our bag-of-entities representations, they did not provide improvements when used individually or in language modeling and BM25 rankers. We speculate that idf had less impact because most queries contained just one or two entities, thus most of the queries were 'short' in the bag-of-entities representation; idf is known to be less important for short queries. We also speculate that the lack of improvement from document length normalization is related to the lack of improvement from frequency-based weighting (`EF`), as discussed above. Our work suggests that better ranking will require thinking carefully about models designed for the unique characteristics of entities, rather than simply assuming that entities behave like words.

4.4 Summary

This chapter presents a new bag-of-entities representation for ranking documents. Query and documents are represented by bag-of-entities representations developed from entity annotations, and ranking is performed by matching them in the entity space. Experiments on TREC Web Track datasets demonstrate that the coverage of bag-of-entities representations is sufficient and bag-of-entities representations can outperform bag-of-words representations by as much as 18% in standard document ranking tasks.

The work done in this chapter is our preliminary progress towards more semantic and structured document representation for ranking. Chapter 5 proposes several future research topics along this line of work.

Chapter 5

Proposed Research Topics

Our preliminary research progresses successfully demonstrate the effectiveness and potential of knowledge bases in information retrieval. This chapter presents our proposed topics to finish this thesis research. Section 5.1 provides plans to continue our preliminary research about ranking with bag-of-entities representations. Then we will address the relevant entity finding task in Section 5.2. Last but not least, Section 5.3 discusses our plans to enrich the bag-of-entities representations to entity graphs.

5.1 Hierarchical Learning to Rank with Multiple Representations

The bag-of-entities representation is different with bag-of-words in many perspectives. Manually or automatically, the grounding from text to entities resolves language ambiguity, aligns synonyms, enforces consistency between entities, and embeds in external knowledge. The entities are more informative than raw terms, and the bag-of-entities is a more distilled representation than using all terms in the document. Each entity in the entity space is also associated with more semantic facts from knowledge bases: description, aliases, attributes, categories and related entities, to name a few. Although already outperforming standard bag-of-words based retrieval models, the simple exact match methods used in Chapter 4 have not fully utilized the bag-of-entities representations.

On the other hand, bag-of-entities also introduces new uncertainties in information retrieval. Its construction is done by entity linking systems. Noises are introduced by the entity linking errors. The meaning of the query or document might only be partially represented by their entities, especially for less entity oriented ones. How to handle this new uncertainty is another challenge to address to fully utilize bag-of-entities' advantages space while avoiding potential risks.

This section proposes a novel hierarchical learning to rank framework to utilize more evidence and to handle the new uncertainties of bag-of-entities representations. There are two layers in the model's hierarchy. The first layer learns to rank with bag-of-entities and bag-of-words individually. In the bag-of-entities part, more evidence is included as new features. In the bag-of-words part, the state-of-the-art learning to rank with bag-of-words is reused. The second layer learns how to combine the ranking models from the two representations. This layer models the uncertainties from two sides and favors the more reliable one in the combination.

5.1.1 Richer Ranking Features with Bag-of-Entities

The first layer of the hierarchical learning to rank includes individual ranking models of bag-of-entities and bag-of-words. Learning to rank with bag-of-words is a widely studied topic. We will reuse their features in the bag-of-words side of our model. In the bag-of-entities side, we will explore features including exact match and soft match to utilize the rich evidence associated with entities.

Exact Match Features between query and document's bag-of-entities can be extracted following previous learning to rank research. Besides coordinate match and entity frequency studied in Chapter 4, many other standard retrieval models can be used, for example, language models, BM25, vector space models and DPH. Features are extracted using the scores of these ranking models between query entities and entities from document's fields, such as title, body, and inlinks. Document quality features can also be extracted from the coherence of its entities.

Soft Match Features make use of the connections between query entities and document entities. The soft match in bag-of-words has been studied as translation models [7, 40]. However, its effectiveness is limited by two main difficulties. The first one is that there are not much evidence between words except context correlations. In comparison, there are many possible ways to model the connection between entities, for example, textual similarities, ontology similarities, relationships, random walk probabilities in the knowledge graph, etc. Though not yet used in ranking, this evidence has been successfully used on document similarity task [68]. The other difficulty is the unavailability of the ground truth about word-word relevancy. Instead of trying to develop an unsupervised entity based translation model, we will use the evidence between entities as features in a learning to rank model supervised by document relevance judgments.

5.1.2 Addressing the Uncertainties from Words and from Entities

The second layer of our model is about the combination of the two representations. Bag-of-words and bag-of-entities have different strength and weakness on different queries. If we can combine them smartly and favor the proper ones, the final ranking performance can benefit from the advantage of both representations and avoid their risks. We will explore various evidence about the uncertainties of bag-of-words and bag-of-entities, which will serve as features in the combination layer of the hierarchical model.

Uncertainties in Bag-of-Words has been studied in the query-performance prediction task. Prior research is accurate in predicting the ranking performance of each query, but have difficulties in improving the ranking accuracy, since there was no other choice when bag-of-words is not working well [63, 85]. With multiple representations available, these works can be used to guide which one to favor: when a query is difficult for bag-of-words based ranking, it is better to favor bag-of-entities, and vise versa. We will start with the query-performance prediction features in modeling the bag-of-words' uncertainties.

Besides query difficulty prediction, there are also many other possibilities. The features used to model how tightly the query terms are clustered can be used to model how valid bag-of-words' term independent assumption is [87]. The features used in query intent prediction can also be used as different query types (e.g. navigational or informational) may favor different representations.

Uncertainties in Bag-of-Entities can be reflected in many ways. The overlap between different entity sources is a good indicator of the entity representation's reliability, as shown in EsdRank's experiments (Section 3.2). The coverage of bag-of-entities on query and document's texts shows how entity-oriented the query and document are, and thus how useful bag-of-entities is for them. The confidence of entity linking system is another source of features for the uncertainties. It is also possible to train our own entity linking performance predictor. These indicators will be used as features in the combination layer of our hierarchical learning to rank model.

5.1.3 Attention Based Learning to Rank

Our experiments in Section 3.2 show that putting features from different parts in a 'flat' model may not be a good idea. Following the same intuition, we plan to develop a hierarchical learning to rank to model the evidence structurally. Its first layer shares the same setup with traditional learning to rank methods, and learns how to rank with the features from bag-of-entities or bag-of-words individually. Its second layer learns which representation to 'attend' by modeling their uncertainties, and combines multiple representations into the final ranking. This two-layer model can be considered as an attention-based learning to rank model, if using the terminologies from the deep learning community.

A basic set up of the attention based ranking model can be:

$$f(q, d) = \sum_k a(w_k, U(q_k, d_k)) \times s(\theta_k, X(q_k, d_k)). \tag{5.1}$$

k refers to the k-th representation, for example, bag-of-words or bag-of-entities. $a(\cdot)$ is the attention model that learns the weight to combine multiple representations. $s(\cdot)$ is the learning to rank model for each individual representation. w_k and θ_k are parameters to learn for the attention layer and the ranking layer. $U(q_k, d_k)$ and $X(q_k, d_k)$ are the uncertainty features and the ranking features respectively.

The choice of each sub-model is flexible. We can start with simple linear models. When necessary, we can extend them to more sophisticated models with kernels or Neural Networks. The model training can be done independently or jointly. We can first train the bag-of-words learning to rank and bag-of-entity learning to rank, and then train the combination of them. We can also train the two layers together with an EM method that treats the uncertainty models as latent mixtures. The loss function in training can be pairwise loss or listwise loss.

The proposed hierarchical learning to rank framework aims to advance the state-of-the-art in document ranking by including more evidence from bag-of-entities, carrying on the strength from current bag-of-words based ranking, and combining the two different representations with their uncertainties considered.

5.2 Joint Relevant Entity Finding and Document Ranking

To incorporate knowledge bases in information retrieval, a major component is to find relevant entities for query and documents. In all our current systems, we find that the quality of relevant entities determines whether introducing knowledge base can improve the document ranking or not. We have experimented with query annotations, retrieved entities, and document annotations. They do provide entities that are useful enough to improve the state-of-the-arts. However, their quality also appears to be the major bottleneck in our systems. Our experiments also found that when the quality of relevant entities is improved, the performances of all our systems are greatly boosted. Similar observations have also been made by other researchers [26, 28, 51]. Some of their systems use manual labels to ensure the quality of relevant entities [51].

As studied in Section 3.3 and Chapter 4, entity search, and entity linking themselves are still on-going research topics. Treating them as a black-box and fully trusting their results introduce their noises into the following search processes. Also, neither entity search nor entity linking is designed to satisfy search engine's needs. Instead, they are developed and optimized towards their own goals, e.g. to satisfy the user's needs for a specific type of queries [2], or to optimize the linking F-score on several specific domains [37]. It is not certain that their goals exactly align with the needs of knowledge based information retrieval systems.

This section discusses our research plans to find better relevant entities towards the end-to-end document ranking performance. We first propose a co-learning model that jointly learns how to find relevant entities and how to rank documents. Then we discuss how it improves current approaches with larger candidate entity set and richer features.

5.2.1 Co-Learning Model for Relevant Entity Finding

Our EsdRank work in Section 3.2 shows that it is better to jointly learn the connection from query to query entities and the ranking of documents. Our preliminary study in Section 3.3 shows the advantage of using supervisions in finding relevant entities directly from a knowledge base index.

We plan to integrate the advantages from both sides together in a co-learning model. Given a query, its top retrieved documents and a set of candidate entities, the model learns how to find relevant entities and how to rank documents jointly. The evidence about entity's relevancy is used as features in the entity side, the evidence about document ranking is used as features on the document side, and the connections between entities and documents are used to propagate the information between the two sides during training and testing.

Co-learning relevant entity finding and document ranking is an interesting multitask learning problem. There are many models developed to propagate the information around different tasks, effectively improving the performances of all tasks. The theoretical guarantees and empirical intuitions from prior multitask learning work provide us a good start towards a system that can find better relevant entities and rank documents more accurately at the same time.

The training of the co-learning model can be done with only document labels, treating entities as the latent layer using Latent-ListMLE (Section 3.2). Another perhaps more reliable way is to use labels from both the entity part and the document part. One possible training data is to use the TREC Web Track queries' labeled entities provided by Schuhmacher et al. [69], Dalton et al. [26], and Liu et al. [51]. Our preliminary experiments have found that using these manually labeled entities easily improve query expansion with knowledge base (Section 3.1), EsdRank (Section 3.2) and bag-of-entities based ranking (Chapter 4). It is also possible for ourselves to label entities for queries in document ranking test collections.

5.2.2 Larger Candidate Entity Set

EsdRank is the first step to learn the connections between query to entities towards better document ranking. It is conservative as it uses the output of external entity linking or entity search systems, only trying to down weight their noises. As a result, only a few entities, usually no more than five, are included in EsdRank. If a relevant entity does not appear in the top results of entity linking or entity search, it is not considered by EsdRank.

Our proposed co-learning model uses more supervisions and more evidence than EsdRank. Thus, a larger candidate entity set is possible. We plan to explore the following two resources for more candidate entities.

Directly Associated Entities: Instead of using the output of entity linking, we will include all spotted entities of query and documents, a.k.a all entities whose aliases appear in the query string or documents' texts. Also, a larger number of retrieved entities will be included, instead of only the top ranked ones from entity search. Thus, the errors of entity linking and entity search are avoided. There will be more noises, but they will be dealt with by the co-learning model that optimizes towards final document ranking while not 'intermediate' goals of entity linking or entity search.

Expanded Entities: Entities that are related with those directly associated ones can also be relevant. For example, besides the entity Carnegie Mellon University, the entity Pittsburgh is also relevant to the query 'CMU location'. We plan to expand the directly associated entities to include more entities as candidates. Possible ways to expand include the existing relationships in the knowledge graph, ontologies, context correlations and embedding similarities. Such second-order entities have been useful in other tasks like document similarity and entity linking [37, 68]. There are also many possible ways to avoid introducing too many noises: we can only expand from the highly confident entities, e.g. those associated with the query and multiple documents; we can only include entities that can be expanded from multiple initial entities. The expansion process is also be guided by supervisions from entity labels and document ranking labels.

5.2.3 Richer Features for Entities and Entity-Document Pairs

Including the 'input' of entity search and entity linking makes it possible to use a broader range of evidence, for example, linked probability, commonness, and entity search features. The weights of these features are now learned towards the final ranking performances. The global view over query, documents, possible annotations and retrieved entities also make it possible to consider global features. One possible feature group can be extracted from the consistency between candidate entities from different sources. Our experiments in Section 3.2 find that the agreement between different sources is a good indicator of relevant entity quality, so will also be good features.

We will start with the entity-document features used in EsdRank to model the connection between entities and documents. Schuhmacher et al. have also developed a wider range of features between entities and documents [69]. Their experiments demonstrated these features' effectiveness in ranking entities from document annotations. We will include these features in our model as well.

The proposed co-learning methods aim to find better relevant entities towards our end-to-end goal of improving document ranking. Possible baselines are state-of-the-art bag-of-words based learning to rank, EsdRank, and our preliminary work in bag-of-entities ranking. The advantages of this proposed work include more features, additional supervisions from entity labels, flexible candidate entity set, and the co-learning model that learns how to combine this information jointly.

5.3 Entity Graph Representation with Relationships

Representing texts by individual entities ignores the interaction between entities/terms. For example, in bag-of-entity representation, some parts of the text can be aligned to each entity, while some are hard to as they are discussing the interaction between entities. Modeling the interaction between entities is an important step towards deeper text understanding and more structured text representation. It is also a major focus of knowledge base to store human knowledge about entity's interactions. For example, a large fraction of facts stored in MeSH is about the belonging relationship in the ontology, and Freebase contains 3 billion facts about 58 million entities, most of which are about relationships.

Such information has already been successfully used in many tasks. In open question answering, the state-of-the-art is to parse the natural language question according to the subgraph in the knowledge bases around question entities, and to provide answers along the best path [84]. The relationships and subgraphs around entities in the knowledge base are also very useful in knowledge base inference [33]. It is also a standard to perform joint disambiguation among multiple surface forms by random walks in the knowledge graph in entity linking [37].

However, there is not much work that successfully models the interaction between entities in text representation nor ranking. This section proposes our plans to study how to model the interaction between entities, how to infer the entity graph representation, and how to rank with entity graphs.

5.3.1 Open Entity Interaction Modeling

One challenge is that the recall of existing relationships in knowledge bases is rather low, leaving most of the entities in a text unconnected. For example, in FACC1 annotation, only about 5% of entity pairs in a document's annotations is connected in Freebase. The construction of knowledge bases emphasizes high precision, trying to make all the defined and added relationships correct. Unlike open question answering in which search engine can choose not to answer and withdraw to keyword search, the low recall makes relationship information hard to be useful for general web search.

We plan to relax the requirement on entity relationships and incorporate more evidence between entities. Besides using the existing manually defined close schema typed relationships, and the verb-phrases extracted by open information extraction, we propose to explore an even wider set of evidence between any two entities. One type of evidence are all the texts that appear between entities. Instead of trying to extract a precise relationship using these texts for corresponding entity pair, we use them build a richer language model for their interactions or virtual documents [76]. The language model can also be jointly trained with the knowledge graph structure into relationship embedding [11, 74], facilitating soft matches and further reducing sparsity. This evidence provides higher recall in modeling the interaction between two entities but perhaps has lower precision. They will be a good complement for the high precision low recall relationships in current knowledge bases. Another type of evidence are the correlation and similarities between two entities: textual similarity, embedding similarity, random walk probability in the knowledge graph, category similarity, context correlations, PMI in annotations etc. This evidence provides information about the strength of the interaction between the two entities, while the previous evidence type is more about what the interaction is.

This new evidence ensures that there is enough information about entities' interactions for general search traffic. The rest of this section presents our plans about how to utilize them in representing the text and improve search.

5.3.2 Entity Graph Inference

In open question answering task, there is a ground truth subgraph to parse the question into, the one that leads to the correct answer. In structure learning tasks in natural language processing, the targeted structure is also known: syntactic parsing tree, dependency parsing tree, and semantic labeling graphs. The ground truth is also available. However, for information retrieval, it is unclear what a good document's structure is and there is no ground truth data to learn from.

We plan to begin our exploration following the word graph approach [8, 66]. The first goal is to infer an entity graph for a given text, whose nodes are entities, and edges store the strength of the connection between corresponding nodes. We plan to infer the entity graph representation using a latent CRF model that handles the node features between entity and documents, a.k.a those in bag-of-entity, and the edge features between entities (relationships). Based on the availability of supervisions, the inference of entity graph can be unsupervised, semi-supervised, or supervised.

Unsupervised: Unsupervised latent graph models have achieve good performance in correlated topic modeling [9] and unsupervised pos-tagging [1]. Similarly, we can treat the terms in the text as observations, and infer the most proper latent entity graph structure in a generative model. The features between an entity with the entire text are modeled as node features. Those between entities are edge features. And the observation (terms) are generated by the entities' language models (built from their descriptions or contexts), either using discrete bag-of-words, or continuous embedding. The parameters to combine features can be learned using maximum likelihood estimation (MLE).

Semi-Supervised: Another possible way is to use the widely available controlled vocabulary labels for scientific publications as partial supervisions. The publishers and focused domain search companies have enforced and manually labeled a large scale of documents with controlled vocabulary labels. These labels reflect human's understanding of the central idea of the document, and can be used as partial supervisions for our model. Similar generative modeling and MLE can be used. The partial supervisions can be added as pairwise preferences for the training process. It is also possible to transfer the controlled vocabulary labels to modern knowledge bases (e.g. Freebase), by directly align controlled vocabulary terms to knowledge base's entities, or using transfer learning techniques.

Supervised: As our research goes on, if enough evidence shows that graph based representation is effective, we may also manually label the entity graph for some documents. The approaches in developing semantic parsing labels can be borrowed, as we also want a graph-like representation, only at document level with more general semantics. With certain restrictions, for example, the edges can only be added if two entities appear closely together, or only relationships in the predefined closed schema can be used, the agreements between human labels could be reduced to a possible level. And then the inference of the entity graph becomes a fully supervised problem which is much easier to solve, and to evaluate.

5.3.3 Ranking with Entity Graph Representation

One straightforward way to use the entity graph is to use the node weights as weights in bag-of-entity representation. The weights inferred from entity graph incorporate additional evidence between entities and could be a better estimation of the nodes to the text [8].

The entity graph representation also enables query-document matching in the knowledge graph space. Random walk models can be used to model the connection strength from the query through the knowledge graph to the documents, which has been shown effective in modeling document semantic similarity [68]. Another possible choice is to use the graph similarity using graph kernels, or approximately with tree kernels which are more efficient [75]. It is also possible to embed the structured graph into a continuous space, using deep learning models like tree based LSTM. Then the ranking can be performed on their embedding.

5.3.4 Road Map for Entity Graph Representation Research

The proposed entity graph representation is the last and biggest research topic for this thesis research. We decompose the problem into several stages, each with its own evaluations and checkpoints. All of them have the potential to form an individual interesting work if can outperform their corresponding baselines.

The first stage is a latent space model that better infers bag-of-entities representations with unsupervised methods. In this stage the evidence from documents to entities and from entities to words are studied, so are the unsupervised inference models. The inferred bag-of-entities will be evaluated in their ability to represent the text using perplexities measures, and more importantly in their effectiveness to improve existing unsupervised bag-of-entities based ranking.

The second stage is to add in supervisions in the inference process. Semi-supervised and supervised representation learning are studied in this step. The goal is to figure out how to leverage supervisions to improve the representation ability. Evaluations are also carried out by their effectiveness in existing unsupervised bag-of-entities based ranking.

The third stage is to study the connections between entities, generating from bag-of-entities to entity graphs. With a clear understanding of what 'individual' features are useful and how to utilize supervisions, we can focus on the exploration of different evidence about the connections between entities. The evaluation is about whether adding connections can improve the representation ability and the performances of existing ranking models.

The last stage is to develop better ranking models upon the entity graph representations. With the entity graphs inferred and their ability to improve existing ranking models verified, the last goal is to build more sophisticated graph based ranking models to further improve document ranking.

5.4 Summary

This chapter proposes our future research topics. We plan to first work on hierarchical learning to rank with multiple representations. It will cost approximately four months from August 2016 to November 2016. From December 2016 to March 2017 we will work on relevant entity finding. After that, we will focus on entity graph representation. We plan to spend the rest of 2017 on this topic, following the proposed roadmap. The planned date to start dissertation writing and job hunting is January 2018, with the hope to finish in the summer of 2018.

Chapter 6

Conclusion

This final chapter first summarizes our current progress and proposed research plans. Then we discuss the contributions this thesis research going to achieve. The last part of this chapter wraps up this proposal with the potential impact of this thesis research.

6.1 Thesis Summary

This thesis research builds upon the successes of bag-of-words based information retrieval and marches towards more semantic and intelligent information retrieval with knowledge bases. We enrich the bag-of-words representations of query and documents with the knowledge bases, and formulate richer bag-of-words, bag-of-entities and entity graph representations for texts. The new knowledge based text representations incorporate a wide range of new evidence from the external knowledge bases, organize query and document texts more structurally, and can support more sophisticated ranking models. The state-of-the-arts, in both unsupervised and supervised ranking, in multiple mainstream ranking benchmarks across multiple search domains, are achieved or currently hold by this thesis research.

The thesis research begins with improving *query representation* with a widely used technique, query expansion, by finding better expansion terms from knowledge bases (Section 3.1). The query expansion is performed in two steps. The first step finds relevant entities by ad-hoc entity search, or from top retrieved documents' annotations. The second step selects expansion terms from related entities' textual descriptions, using pseudo relevance feedback, or by their ontology similarities with the query. A supervised model then combines these methods' scores and selects terms that are effective for query expansion. The experiments on TREC Web Track queries, ClueWeb09 corpus, and Freebase demonstrate that these methods improve the previous state-of-the-art query expansion methods effectively by almost 30%, and robustly with 50% fewer queries damaged. This performance shows the great potential of knowledge based in information retrieval, and motivated our pursuit of this thesis research.

The next step of this dissertation is to use the entities to represent the query, and moves from the word space to the entity space. Section 3.2 provides a general framework to improve document ranking with various kinds of knowledge bases. Query annotation, entity search and document annotation in the top retrieved documents are used to find relevant entities for the query. Features are extracted to describe the similarity between the query, query entities, and documents. The connection from query to query entities and the ranking of documents are jointly learned by a novel latent space listwise learning to rank model. Our method is evaluated on two different scenarios: a general domain search (TREC Web Track) with a large knowledge base (Freebase), and a medical search (OSHUMED) with a medical controlled vocabulary (MeSH). On both scenarios, EsdRank outperforms previous learning to rank methods on all evaluation metrics, and achieves the state-of-the-art in supervised ranking in these benchmark datasets.

In both term based and entity based query representation with knowledge base, a crucial component is relevant entity finding . We have explored query entity linking, document entity linking and entity search results as the input of our systems. It remains unclear which are the most suitable entities for knowledge based IR systems. Section 3.3 develops our own entity search system. It uses learning to rank to combine the text similarities features between query and entity's predefined fields. Simple and intuitive as it is, it is holding the current state-of-the-art in a popular entity search test collections.

Nevertheless, even the perfect entity linking or entity search results may not be the most *suitable* entities to be used for information retrieval systems. Their goals are different: to satisfy user's annotation and entity search preference, or to satisfy user's information needs for ad-hoc search. The dissertation proposes to direct address this key challenge by finding related entities that best optimizes *ranking* performances. Section 5.2 provides our research plans to learn which entities are relevant for the query towards better end-to-end document ranking performance.

Another equally important focus of this dissertation is *document representation* with knowledge bases. Together with knowledge based query representation, it moves information retrieval fully into the knowledge space. Our initial step is to represent documents by their annotations using the bag-of-entities model (Chapter 4). We first study the accuracy and coverage of three popular entity linking systems that annotate texts with Freebase entities. The results show that though the annotation accuracy is far from perfect (about 60% in the general domain), the coverage and density of annotations to web queries and documents are ready to generate reasonable bag-of-entity representations. Then we use simple unsupervised ranking models to rank documents based on their matches with the query in the bag-of-entity space. Our experiments on TREC Web Track queries and ClueWeb corpora demonstrate the effectiveness of entity based retrieval. Even though the accuracy of entity linking is imperfect, very simple unsupervised ranking methods can outperform standard bag-of-words based retrieval significantly. Section 5.1 proposes a hierarchical learning to rank framework that utilizes more evidence from bag-of-entities, and combines bag-of-entities with bag-of-words with their uncertainties considered.

The last part of this thesis research is about how to form better text representation with the connections between entities. Chapter 5.3 propose our plans to generate entity graph representations. The nodes in the graph are entities as in the bag-of-entity, and the edges model the connections between entities. Several different resources to collect the evidence between entities are proposed: the existing relationships in the knowledge graph, the context information in the corpus with open information extraction techniques, and the correlation and similarity of entities. This evidence serves as edge features in proposed machine learning models that infer the entity graph representations. With the entity graph representation, there are various possible ways to improve ranking. We plan to rank documents by their matches with the query in the knowledge graph space, using random walk or graph kernel methods.

6.2 Thesis Contributions

Bag-of-words based information retrieval systems have been successful for the past several decades. However, though various new ranking models have been proposed, they, unsupervised or supervised, are all built upon the same bag-of-words representation, or even more narrowly, the same term level statistics. With the same representation, the choices of information retrieval systems are restricted, and there are several intrinsic challenges hard to address, such as vocabulary mismatch. This thesis research goes beyond the bag-of-words representation and moves to the entities and knowledge graph based representation. With the new representation proposed in this thesis, the new, richer, and semi-structured information in knowledge bases are incorporated, new learning to rank models are developed, and better document ranking performance are achieved. More importantly, information retrieval receives a new opportunity to overcome the bottleneck of pure term based representation. This dissertation shares the same motivation with classic controlled vocabulary representations: incorporating human knowledge that is external to query and documents in information retrieval. It can be considered as a heritage of the classic controlled vocabulary based information retrieval, but equipped with modern large scale knowledge bases, automatic entity linking tools and more advanced machine learning techniques.

This dissertation demonstrates the potential of knowledge bases in modern information retrieval systems. We provide several simple, intuitive and powerful ways that utilize knowledge bases to better represent query and documents with terms, entities, and entity graphs. Effective and suitable ranking models are developed to handle the novel representations. Our systems achieve state-of-the-art performances in many standard ranking benchmark datasets in multiple search domains. The novelty, simplicity, intuitiveness, and demonstrated rich potentials have already and will continue to inspire new research in this direction from information retrieval and machine learning researchers.

We also provide valuable feedback to information extraction researchers about what information and techniques are the most desired to make use of knowledge bases in real world applications. For example, our query expansion work in Chapter 3.1 has inspired NELL researchers to study how to automatically generate textual descriptions (glosses) for NELL entities [27]. Our experiments about bag-of-entities show that the entity linking systems' coverage are very important, while solely focusing on extremely high precision may not be the best choice for document ranking. Also, rather than several specific domains of named entities, general domain entities with sufficient information are useful for all knowledge based information retrieval systems through this thesis research. This is a good signal for entity linking research community to embrace open domain entity linking. In our entity search work, we find that different predicates are of different importance in satisfying different information needs. This finding can also provide feedback about which predicates are more important to be extracted for knowledge base construction.

We believe that by developing knowledge based text representations, this thesis research provides a solid start towards real text understanding in which computers no longer just 'count' individual words, but treat text in a semantic, semi-structured, entity oriented, and prior knowledge incorporated way, just like we human beings.

6.3 Thesis Impact

This section wraps up with several possible impact of this thesis research. The first impact is to encourage more exploration of structured knowledge's usage in information retrieval, studying more evidence and developing better ranking models. Then we discuss the possible applications of our knowledge based text representations in a broader range of text related tasks.

Embedding More Knowledge and Developing Better Ranking Models

This thesis research opens a new gate for information retrieval towards more knowledge embedded and structured text understanding and document ranking. We have developed various simple, intuitive and effective ways to incorporate knowledge base's evidence in text representation and ranking models. However, the richness, heterogeneous, and semi-structured evidence in knowledge bases provide a large room to explore, well beyond one dissertation. For example, it is possible to build better language models for entities using their contexts in the corpus, their neighbor entities' description, or the ontology to guide the estimation of their language models. The better language models can lead to better expansion terms, better query-entity and entity-document connection features, and better inference of bag-of-entity and entity graph representations.

This thesis research has developed several new ranking models, and demonstrated the advantage of using *sophisticate and suitable* machine learning methods. It leads to another research possibility for machine learning community. For example, prior learning to rank models mostly face 'flag' features that work well in linear combinations. In the entity space, evidence distributes in multiple places, enabling more structured learning to rank model. Another new challenge is that the entity space is larger and sparser than the term space. For example, when using Freebase, there are more than five million possible entities (dimensions). In the meantime, the number of entities in a document is usually smaller than the number of terms. Also, the mapping from text to entities introduces a new uncertainty. The high dimensionality , the additional sparsity, and the new uncertainty can support fancier machine learning research, such as lasso, dimension reduction or embedding models, etc.

Broader Application in Text Related Tasks

The new knowledge based text representation is not restricted to ranking tasks. Instead, a much broader range of text-related applications can benefit from our new text representations.

Text classification and clustering tasks can directly use our new document representations. For example, it is intuitive that our bag-of-entity representation should improve *ontology* based classifications, as the entities are more semantic and directly associated with ontology types. Text clustering can also benefit from our work. One of the most suitable clustering tasks is shard partitioning in selective search, whose fundamental assumption is that documents of the same *topic* should be put into the same shard [44]. It is intuitive that the entity based or knowledge graph based representations provide new evidence that can lead to better shard partitioning.

User understanding is another area knowledge based text representation can help. Entities and knowledge graph provide a more distilled view of the texts in queries and browsed web pages. For example, in behavior targeting, an important task is to classify users into an ontology based on their query log and browse history [19], which is knowledge bases' specialty. Another example is sponsored search, which presents sponsored advertisements in search result pages [80]. Most commercial advertisements are about real world entities: restaurants, products, and brands. With the help of our knowledge based representation, search engines can better detect and understand user's commercial needs for better advertisements.

Bibliography

[1] Waleed Ammar, Chris Dyer, and Noah A Smith. Conditional random field Autoencoders for unsupervised structured prediction. In *Advances in 26th Neural Information Processing Systems (NIPS 2014)*, pages 3311–3319, 2014. 5.3.2

[2] Krisztian Balog and Robert Neumayer. A test collection for entity search in dbpedia. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval (SIGIR 2013)*, pages 737–740. ACM, 2013. 3.3, 3.3.1, 3.3.2, 3.3.3, 5.2

[3] Michele Banko, Michael J Cafarella, Stephen Soderland, Matthew Broadhead, and Oren Etzioni. Open information extraction for the web. In *Proceedings of the the 14th International Joint Conference on Artificial Intelligence (IJCAI 2007)*, pages 2670–2676. IJCAI, 2007. 2.2

[4] Hannah Bast and Elmar Haussmann. More accurate question answering on Freebase. In *Proceedings of the 24th ACM International Conference on Information and Knowledge Management (CIKM 2015)*, pages 1431–1440. ACM, 2015. 3.3.1

[5] Michael Bendersky, David Fisher, and W Bruce Croft. Umass at TREC 2010 Web Track: Term dependence, spam filtering and quality bias. In *Proceedings of The 19th Text Retrieval Conference, (TREC 2010)*. NIST, 2010. 3.1.1, 3.1.2.1, 3.1.3, 3.1.3

[6] Michael Bendersky, Donald Metzler, and W Bruce Croft. Effective query formulation with multiple information sources. In *Proceedings of the Fifth ACM International Conference on Web Search and Data Mining (WSDM 2012)*, pages 443–452. ACM, 2012. 3.2.2

[7] Adam Berger and John Lafferty. Information retrieval as statistical translation. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 1999)*, pages 222–229. ACM, 1999. 5.1.1

[8] Roi Blanco and Christina Lioma. Graph-based term weighting for information retrieval. *Information Retrieval*, 15(1):54–92, 2012. 5.3.2, 5.3.3

[9] David M Blei and John D Lafferty. A correlated topic model of science. *The Annals of Applied Statistics*, pages 17–35, 2007. 5.3.2

[10] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data (SIGMOD 2008)*, pages 1247–1250. ACM, 2008. 2.2, 3.2

[11] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *Proceedings of the 25th Advances in Neural Information Processing Systems 2013 (NIPS 2013)*, pages 2787–2795, 2013. 5.3.1

[12] Wladmir C Brandão, Rodrygo LT Santos, Nivio Ziviani, Edleno S Moura, and Altigran S Silva. Learning to expand queries using entities. *Journal of the Association for Information Science and Technology (JASIST)*, 65(9):1870–1883, 2014. 3.2.2

[13] Andrei Z Broder, Marcus Fontoura, Evgeniy Gabrilovich, Amruta Joshi, Vanja Josifovski, and Tong Zhang. Robust classification of rare queries using web knowledge. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2007)*, pages 231–238. ACM, 2007. 3.2.2

[14] Guihong Cao, Jian-Yun Nie, Jianfeng Gao, and Stephen Robertson. Selecting good expansion terms for pseudo-relevance feedback. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2008)*, pages 243–250. ACM, 2008. 3.1, 3.1.1, 3.1.2.2, 3.1.3

[15] Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R Hruschka Jr, and Tom M Mitchell. Toward an architecture for never-ending language learning. In *Proceedings of the 24th AAAI Conference on Artificial Intelligence*, volume 5, page 3, 2010. 1, 2.2

[16] David Carmel, Ming-Wei Chang, Evgeniy Gabrilovich, Bo-June (Paul) Hsu, and Kuansan Wang. ERD'14: Entity recognition and disambiguation challenge. In *Proceedings of the 37th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2014)*. ACM, 2014. 1, 2.2, 3.2.1, 3.2.1.2, 3.2.2, 3.2.3.1, 3.2.3.3, 4, 4.1, 4.2

[17] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`. 3.1.2.2

[18] Jing Chen, Chenyan Xiong, and Jamie Callan. An empirical study of learning to rank for entity search. In *Proceedings of the 39th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval,(SIGIR 2016)*. ACM, 2016. To Appear. 3.3

[19] Ye Chen, Dmitry Pavlov, and John F Canny. Large-scale behavioral targeting. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2009)*, pages 209–218. ACM, 2009. 6.3

[20] Kevyn Collins-Thompson. *Robust Model Estimation Methods for Information Retrieval*. PhD thesis, Carnegie Mellon University, December 2008. 3.1.1, 3.1.4.3

[21] Kevyn Collins-Thompson. Estimating robust query models with convex optimization. In *Proceedings of the 21st Advances in Neural Information Processing Systems (NIPS 2009)*, pages 329–336. NIPS, 2009. 3.1.1

[22] Kevyn Collins-Thompson. Reducing the risk of query expansion via robust constrained

optimization. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management (CIKM 2009)*, pages 837–846. ACM, 2009. 3.1.1

[23] Kevyn Collins-Thompson and Jamie Callan. Estimation and use of uncertainty in pseudo-relevance feedback. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2007)*, pages 303–310. ACM, 2007. 3.1.1

[24] Gordon V Cormack, Mark D Smucker, and Charles LA Clarke. Efficient and effective spam filtering and re-ranking for large web datasets. *Information Retrieval*, 14(5):441–465, 2011. 3.1.3, 3.2.2

[25] W Bruce Croft, Donald Metzler, and Trevor Strohman. *Search Engines: Information Retrieval in Practice*. Addison-Wesley Reading, 2010. 1

[26] Jeffrey Dalton, Laura Dietz, and James Allan. Entity query feature expansion using knowledge base links. In *Proceedings of the 37th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2014)*, pages 365–374. ACM, 2014. 2.3, 3.1.3, 3.1.3, 3.2, 3.2.1, 3.2.1.2, 3.2.2, 3.2.3.1, 3.2.3.2, 4.1, 4.2, 4.3.2, 5.2, 5.2.1

[27] Bhavana Dalvi, Einat Minkov, Partha P Talukdar, and William W Cohen. Automatic gloss finding for a knowledge base using ontological constraints. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining (WSDM 2015)*, pages 369–378. ACM, 2015. 6.2

[28] Laura Dietz and Patrick Verga. Umass at TREC 2014: Entity query feature expansion using knowledge base links. In *Proceedings of The 23st Text Retrieval Conference (TREC 2014)*. NIST, 2014. 2.3, 3.2.3.1, 3.2.3.3, 5.2

[29] Metzler Jr Donald A. *Beyond Bags of Words: Effectively Modeling Dependence and Features in Information Retrieval*. PhD thesis, University of Massachusetts Amherst, September 2007. 3.1.1, 3.1.3

[30] Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD 2014)*, pages 601–610. ACM, 2014. 1

[31] Paolo Ferragina and Ugo Scaiella. Fast and accurate annotation of short texts with Wikipedia pages. *arXiv preprint arXiv:1006.3498*, 2010. 2.2, 3.2.2, 4, 4.1, 4.2

[32] Evgeniy Gabrilovich, Michael Ringgaard, and Amarnag Subramanya. FACC1: Freebase annotation of ClueWeb corpora, Version 1 (Release date 2013-06-26, Format version 1, Correction level 0), June 2013. 1, 2.2, 3.1.3, 3.2.2, 4, 4.1, 4.2

[33] Matthew Gardner. *Reading and Reasoning with Knowledge Graphs*. PhD thesis, Carnegie Mellon University Pittsburgh, PA, 2015. 5.3

[34] Siddharth Gopal and Yiming Yang. Recursive regularization for large-scale classification with hierarchical and graphical dependencies. In *Proceedings of the 19th ACM SIGKDD*

International Conference on Knowledge Discovery and Data Mining (KDD 2009), pages 257–265. ACM, 2013. 2.1

[35] Gregory Grefenstette and Laura Wilber. *Search-Based Applications: At the Confluence of Search and Database Technologies*. Morgan & Claypool Publishers, 2010. 2.1

[36] Faegheh Hasibi, Krisztian Balog, and Svein Erik Bratsberg. Entity linking in queries: Tasks and evaluation. In *Proceedings of the Fifth ACM International Conference on The Theory of Information Retrieval (ICTIR 2015)*, pages 171–180. ACM, 2015. 2.2, 4, 4.1, 4.2

[37] Heng Ji, Joel Nothman, Ben Hachey, and Radu Florian. Overview of TAC-KBP 2015 tri-lingual entity discovery and linking. In *Proceedings of the 2015 Text Analysis Conference (TAC2015)*. NIST, 2015. 1, 2.2, 5.2, 5.2.2, 5.3

[38] T. Joachims. Making large-scale SVM learning practical. LS8-Report 24, Universität Dortmund, LS VIII-Report, 1998. 3.2.2

[39] Thorsten Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2002)*, pages 133–142. ACM, 2002. 3.2.2

[40] Maryam Karimzadehgan and ChengXiang Zhai. Estimation of statistical translation models based on mutual information for ad hoc information retrieval. In *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2010)*, pages 323–330. ACM, 2010. 5.1.1

[41] Bevan Koopman, Guido Zuccon, Peter Bruza, Laurianne Sitbon, and Michael Lawley. Information retrieval as semantic inference: A graph inference model applied to medical search. *Information Retrieval Journal*, 19:6–37, 2016. 2.1

[42] Alexander Kotov and ChengXiang Zhai. Tapping into knowledge base for concept feedback: Leveraging ConceptNet to improve search results for difficult queries. In *Proceedings of the Fifth ACM International Conference on Web Search and Data Mining (WSDM 2012)*, pages 403–412. ACM, 2012. 3.1.1

[43] Robert Krovetz. Viewing morphology as an inference process. In *Proceedings of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, (SIGIR 1993)*, pages 191–202. ACM, 1993. 3.1.3

[44] Anagha Kulkarni. *Efficient and Effective Large-Scale Search*. PhD thesis, Carnegie Mellon University, 2013. 6.3

[45] Victor Lavrenko and W Bruce Croft. Relevance based language models. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2001)*, pages 120–127. ACM, 2001. 3.1.1, 3.1.2.3, 3.1.3, 3.2.1.2, 3.2.2

[46] Kyung Soon Lee, W Bruce Croft, and James Allan. A cluster-based resampling method for pseudo-relevance feedback. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2008)*, pages 235–242. ACM, 2008. 3.1.1

[47] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N.

Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören Auer, and Christian Bizer. DBpedia - A large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web Journal*, 2014. 2.2

[48] Hang Li and Jun Xu. Semantic matching in search. *Foundations and Trends in Information Retrieval*, 8:89, 2014. 2.1

[49] Tie-Yan Liu. Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval*, 3(3):225–331, 2009. 3.2.1.3, 3.2.3.1

[50] Tie-Yan Liu, Jun Xu, Tao Qin, Wenying Xiong, and Hang Li. Letor: Benchmark dataset for research on learning to rank for information retrieval. In *Proceedings of SIGIR 2007 Workshop on Learning to Rank for Information Retrieval*, pages 3–10, 2007. 1

[51] Xitong Liu and Hui Fang. Latent entity space: a novel retrieval approach for entity-bearing queries. *Information Retrieval Journal*, 18(6):473–503, 2015. 2.3, 4.1, 4.2, 4.3.2, 5.2, 5.2.1

[52] Xitong Liu, Peilin Yang, and Hui Fang. Entity came to rescue - Leveraging entities to minimize risks in web search. In *Proceedings of The 23st Text Retrieval Conference, (TREC 2014)*. NIST, 2014. 2.3, 3.2.1.2, 3.2.3.1, 3.2.3.3

[53] Chunliang Lu, Wai Lam, and Yi Liao. Entity retrieval via entity factoid hierarchy. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics (ACL2015)*, pages 514–523. ACL, 2015. 3.3, 3.3.1, 3.3.3, 3.3.4.1

[54] Yue Lu, Hui Fang, and Chengxiang Zhai. An empirical study of gene synonym query expansion in biomedical information retrieval. *Information Retrieval Journal*, 12(1):51–68, 2009. 2.1

[55] Zhiyong Lu, Won Kim, and W John Wilbur. Evaluation of query expansion using MeSH in PubMed. *Information Retrieval Journal*, 12(1):69–80, 2009. 1, 2.1, 3.2.2

[56] Pablo N Mendes, Max Jakob, Andrés García-Silva, and Christian Bizer. DBpedia spotlight: shedding light on the web of documents. In *Proceedings of the 7th International Conference on Semantic Systems*, pages 1–8. ACM, 2011. 2.2

[57] Donald Metzler and W Bruce Croft. A Markov random field model for term dependencies. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2005)*, pages 472–479. ACM, 2005. 3.1.3

[58] Donald Metzler and W Bruce Croft. Latent concept expansion using Markov random fields. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2007)*, pages 311–318. ACM, 2007. 3.1.1, 3.2.2

[59] Rada Mihalcea and Andras Csomai. Wikify!: Linking documents to encyclopedic knowledge. In *Proceedings of the 16th ACM Conference on Information and Knowledge Management (CIKM 2007)*, pages 233–242. ACM, 2007. 2.2

[60] Robert Neumayer, Krisztian Balog, and Kjetil Nørvåg. When simple is (more than) good enough: Effective semantic search with (almost) no semantics. In *Proceedings of the 34th annual European Conference on Information Retrieval (ECIR 2012)*, pages 540–543. Springer, 2012. 3.3.1

[61] Dong Nguyen and Jamie Calan. Combination of evidence for effective web search. In *Proceedings of The 19th Text Retrieval Conference (TREC 2010)*. NIST, 2010. 3.1.1, 3.1.2.1, 3.1.3, 3.1.3

[62] Dazhao Pan, Peng Zhang, Jingfei Li, Dawei Song, Ji-Rong Wen, Yuexian Hou, Bin Hu, Yuan Jia, and Anne De Roeck. Using Dempster-Shafer's evidence theory for query expansion based on freebase knowledge. In *Information Retrieval Technology*, pages 121–132. Springer, 2013. 3.2.1.2

[63] Fiana Raiber and Oren Kurland. Query-performance prediction: Setting the expectations straight. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval (SIGIR 2014)*, pages 13–22. ACM, 2014. 5.1.2

[64] TB Rajashekar and Bruce W Croft. Combining automatic and manual index representations in probabilistic retrieval. *Journal of the American society for Information science*, 46(4): 272–283, 1995. 2.1

[65] Stephen E Robertson and Steve Walker. Okapi/Keenbow at TREC-8. In *Proceedings of The 8th Text Retrieval Conference (TREC 1999)*, pages 151–162. NIST, 1999. 3.1.1

[66] François Rousseau and Michalis Vazirgiannis. Graph-of-word and TW-IDF: New approach to ad hoc IR. In *Proceedings of the 22nd ACM International Conference on Conference on Information & Knowledge Management (CIKM 2013)*, pages 59–68. ACM, 2013. 5.3.2

[67] Gerard Salton and Michael J McGill. Introduction to modern information retrieval. 1986. 1, 4, 4.1

[68] Michael Schuhmacher and Simone Paolo Ponzetto. Knowledge-based graph document modeling. In *Proceedings of the 7th ACM International Conference on Web search and Data Mining (WSDM 2014)*, pages 543–552. ACM, 2014. 5.1.1, 5.2.2, 5.3.3

[69] Michael Schuhmacher, Laura Dietz, and Simone Paolo Ponzetto. Ranking entities for web queries through text and knowledge. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management (CIKM 2015)*, pages 1461–1470. ACM, 2015. 3.3.1, 5.2.1, 5.2.3

[70] Nicola Stokes, Yi Li, Lawrence Cavedon, and Justin Zobel. Exploring criteria for successful query expansion in the genomic domain. *Information Retrieval*, 12(1):17–50, 2009. 2.1

[71] Trevor Strohman, Donald Metzler, Howard Turtle, and W Bruce Croft. Indri: A language model-based search engine for complex queries. In *Proceedings of the International Conference on Intelligent Analysis*, volume 2, pages 2–6. Central Intelligence for Analysis and Production, 2005. 3.1.3

[72] 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 (WWW 2007)*, pages 697–706. ACM, 2007. 2.2

[73] Tao Tao and ChengXiang Zhai. Regularized estimation of mixture models for robust pseudo-relevance feedback. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2006)*, pages 162–169. ACM, 2006. 3.1.1, 3.1.3

[74] Kristina Toutanova, Danqi Chen, Patrick Pantel, Pallavi Choudhury, and Michael Gamon. Representing text for joint embedding of text and knowledge bases. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP 2015)*, pages 1499–1509. ACL, 2015. 5.3.1

[75] Kateryna Tymoshenko and Alessandro Moschitti. Assessing the impact of syntactic and semantic structures for answer passages reranking. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management (CIKM 2015)*, pages 1451–1460. ACM, 2015. 5.3.3

[76] Nikos Voskarides, Edgar Meij, Manos Tsagkias, Maarten de Rijke, and Wouter Weerkamp. Learning to explain entity relationships in knowledge graphs. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics (ACL 2015)*, pages 564–574. ACL, 2015. 5.3.1

[77] Fen Xia, Tie-Yan Liu, Jue Wang, Wensheng Zhang, and Hang Li. Listwise approach to learning to rank: Theory and algorithm. In *Proceedings of the 25th International Conference on Machine learning ïijĹICML 2008)*, pages 1192–1199. ACM, 2008. 3.2.1.1, 3.2.2

[78] Chenyan Xiong and Jamie Callan. EsdRank: Connecting query and documents through external semi-structured data. In *Proceedings of the 24th ACM International Conference on Information and Knowledge Management (CIKM 2015)*, pages 951–960. ACM, 2015. 3.2, 4.1, 4.2, 4.3.2

[79] Chenyan Xiong and Jamie Callan. Query expansion with Freebase. In *Proceedings of the fifth ACM International Conference on the Theory of Information Retrieval (ICTIR 2015)*, pages 111–120. ACM, 2015. 3.1, 3.2, 3.2.3.1, 3.2.3.3, 4.1, 4.3.2

[80] Chenyan Xiong, Taifeng Wang, Wenkui Ding, Yidong Shen, and Tie-Yan Liu. Relational click prediction for sponsored search. In *Proceedings of the Fifth ACM International Conference on Web Search and Data Mining (WSDM 2012)*, pages 493–502. ACM, 2012. 6.3

[81] Chenyan Xiong, Jamie Callan, and Tie-Yan Liu. Bag-of-entity representation for ranking. *Under Review*, 2016. 4

[82] Yang Xu, Gareth JF Jones, and Bin Wang. Query dependent pseudo-relevance feedback based on Wikipedia. In *Proceedings of the 32nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2009)*, pages 59–66. ACM, 2009. 3.1, 3.1.1, 3.1.2.1, 3.1.3, 3.2, 3.2.1, 3.2.1.2, 3.2.2

[83] Yi Yang and Ming-Wei Chang. S-MART: Novel tree-based structured learning algorithms applied to tweet entity linking. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics (ACL2015)*, pages 504–513. ACL, 2015. 2.2

[84] Wen-tau Yih, Ming-Wei Chang, Xiaodong He, and Jianfeng Gao. Semantic parsing via staged query graph generation: Question answering with knowledge base. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (ACL 2015)*, pages 1321–1331. ACL, 2015. 3.3.1, 5.3

[85] Elad Yom-Tov, Shai Fine, David Carmel, and Adam Darlow. Learning to estimate query

difficulty: Including applications to missing content detection and distributed information retrieval. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR 2005)*, pages 512–519. ACM, 2005. 5.1.2

[86] Chengxiang Zhai and John Lafferty. Model-based feedback in the language modeling approach to information retrieval. In *Proceedings of the 10th ACM Conference on Information and Knowledge Management (CIKM 2001)*, pages 403–410. ACM, 2001. 3.1.1

[87] Guoqing Zheng and Jamie Callan. Learning to reweight terms with distributed representations. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2015)*, pages 575–584. ACM, 2015. 5.1.2

[88] Nikita Zhiltsov, Alexander Kotov, and Fedor Nikolaev. Fielded sequential dependence model for ad-hoc entity retrieval in the web of data. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2015)*, pages 253–262. ACM, 2015. 3.3, 3.3.1, 3.3.2, 3.3.3