Inf Retrieval (2014) 17:265–294
DOI 10.1007/s10791-013-9237-0

Exploiting entity relationship for query expansion in enterprise search

Xitong Liu · Fei Chen · Hui Fang · Min Wang

Received: 13 June 2013 / Accepted: 30 December 2013 / Published online: 10 January 2014

Abstract Enterprise search is important, and the search quality has a direct impact on the productivity of an enterprise. Enterprise data contain both *structured* and *unstructured* information. Since these two types of information are complementary and the structured information such as relational databases is designed based on ER (entity-relationship) models, there is a rich body of information about entities in enterprise data. As a result, many information needs of enterprise search center around *entities*. For example, a user may formulate a query describing a problem that she encounters with an entity, e.g., the web browser, and want to retrieve relevant documents to solve the problem. Intuitively, information related to the entities mentioned in the query, such as related entities and their relations, would be useful to reformulate the query and improve the retrieval performance. However, most existing studies on query expansion are term-centric. In this paper, we propose a novel entity-centric query expansion framework for enterprise search. Specifically, given a query containing entities, we first utilize both unstructured and structured information to find entities that are related to the ones in the query. We then discuss how to adapt existing feedback methods to use the related entities and their relations to improve search quality. Experimental results over two real-world enterprise collections show that the proposed entity-centric query expansion strategies are more effective and robust to improve the search performance than the state-of-the-art pseudo feedback methods for long

X. Liu (✉) · H. Fang
Department of Electrical and Computer Engineering, University of Delaware, Newark,
DE 19716, USA
e-mail: xtliu@udel.edu

H. Fang
e-mail: hfang@udel.edu

F. Chen
HP Labs, Palo Alto, CA 94304, USA
e-mail: fei.chen4@hp.com

M. Wang
Google Research, Mountain View, CA 94043, USA
e-mail: minwang@google.com

natural language-like queries with entities. Moreover, results over a TREC ad hoc retrieval collections show that the proposed methods can also work well for short keyword queries in the general search domain.

Keywords Entity centric · Enterprise search · Retrieval · Query expansion · Combining structured and unstructured data

1 Introduction

Today any enterprise has to deal with a sheer amount of information such as emails, wikis, Web pages, relational databases, etc. The quality of enterprise search is critical to reduce business costs and produce positive business outcomes.

Despite the great progress on Web search, there are still many unsolved challenges in enterprise search (Hawking 2004). In particular, enterprise data contain not only unstructured information such as documents and web pages, but also a rich set of structured information such as relational databases. These structured data usually center around entities since relational databases are designed based on Entity-Relation models. Furthermore, the unstructured data, which capture information complementary to structured data, also contain rich information about entities and their relations, embedded in text. Clearly, a large portion of enterprise information centers around entities. Moreover, recent studies (Hearst 2011; Lin et al. 2012) show that there is a trend that users are more likely to issue long, natural language like entity-bearing queries. Therefore, it would be interesting to study how to fully utilize the unique characteristic of enterprise data, i.e., *entities and their relations*, as a bridge to seamlessly combine both *structured* and *unstructured* data to improve enterprise search quality.

One of the important search problems in every enterprise is to provide effective self-service IT support, where an enterprise user submits a query to describe a problem and expects to find relevant information for solving the problem from a collection of knowledge documents.

Example 1 (Problem): A user cannot access the enterprise intranet with her PC whose hostname is "XYZ.A.com" and needs to search for documents helping her solve this problem. The enterprise maintains a knowledge document collection, which is a set of how-to documents and frequent Q&A. As the user does not know what could potentially cause the problem, or which computer components are related to the problem, she submits a query "XYZ cannot access intranet" to search over the document collection, and expects to find solutions in the search results.

The example query centers around an entity, i.e., computer "XYZ", and the relevant documents are expected to contain information that is relevant to this specific entity. However, as the knowledge documents seldom cover information about specific IT assets such as "XYZ", there might be many documents relevant to "cannot access intranet" but not to "XYZ". With existing search engines, the user has to go through several returned documents, check her computer to verify each possible cause, and may even need to reformulate the query with additional knowledge to retrieve more relevant documents.

The challenge here is how to automatically reformulate the query so that documents related to the query entity (i.e., computer "XYZ") can be brought up in the ranking list.

Query expansion is a commonly used technique to reformulate a query. Unfortunately, most existing work on query expansion are based on terms (Fang and Zhai 2006; Lavrenko and Croft 2001; Rocchio 1971; Zhai and Lafferty 2001). In particular, expansion terms are selected from either feedback documents or external sources and added to the query based on the proposed weighting strategies. Although these term-centric expansion methods work well for short keyword queries, they may not be the optimal solution for longer queries that involve entities. In a way, this is caused by the limitation that the information about the entity in a query is ignored and terms that are used to describe an entity are treated in the same way as other query terms. As a result, the selected expansion terms may introduce noise and are less relevant to the query as a whole. Let us re-visit the previous example. Existing query expansion techniques may add terms such as "connect" to the original query. However, these terms are useful to retrieve documents relevant to "cannot access intranet" but not those related to the specific entity "XYZ".

It is clear that query entities, i.e., those mentioned in a query, should play an important role in the query expansion process, since they often represent one or multiple aspects of the information need. Intuitively, a document mentioning entities that are related to the query entities is more likely to be relevant to the query than those not mentioning any related entities. Thus, it would be interesting to study how to find related entities and exploit them to improve search accuracy.

In this paper, we study the problem of entity-centric query expansion for enterprise search. Given a query involving entities, the goal is to utilize the entity relations embedded in both *structured* and *unstructured* information to find entities and their relations that are related to the query and use them to improve the enterprise search performance.

We now explain the proposed method in the context of the previous example scenario.

Example 2 (Entity-centric expansion): Enterprise data contain both structured and unstructured information, and structured data contain rich information that should be exploited to improve search accuracy but is often being under-utilized by existing enterprise search engines. In additional to the document collection mentioned in Example 1, the enterprise data often contain structured information, such as table Asset containing information about all IT assets and Dependency containing the dependency relationships between the IT assets. As shown in step 1 of Fig. 1, "XYZ.A.com" is an asset with ID equal to "A103" and its category is a PC. We can then exploit both structured and unstructured data to find entities related to the query entity "XYZ". For example, one related entity is "proxy.A.com" because it is the web proxy server for all the PCs (including "XYZ") to access webpages according to the join relations between the two tables. "ActivKey" is another related entity because it is required for the authentication of employees so that PCs can access the intranet according to the information from both table "Asset" and the unstructured knowledge documents. Both the related entities and their relations are useful to retrieve relevant documents that the original query fails to retrieve, as illustrated in Step 2 of the Fig. 1.

The first challenge is how to identify related entities. The structured data contain explicit information about relations among entities such as key-foreign key relationships. However, the entity relation information can also be hidden in unstructured data. We apply Condition Random Fields (CRFs) model to learn a domain-specific entity recognizer, and apply the entity recognizer to documents and queries to identify entities from the unstructured information. If two entities co-occur in the same document, they are deemed to be related. The relations can be discovered by the context terms surrounding their occurrences.

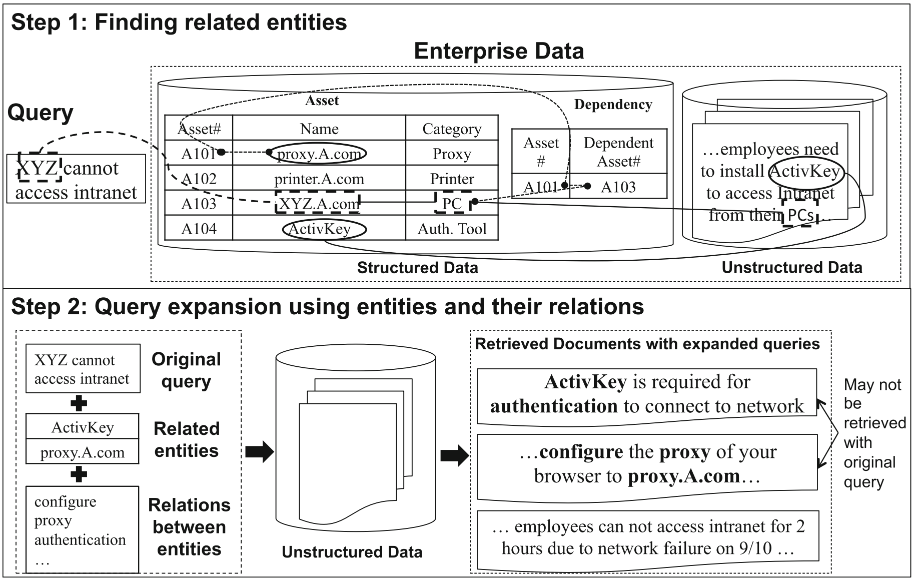

Step 1: Finding related entities
Enterprise Data
Query
XYZ cannot access intranet
Asset
Asset#
Name
Category
A101
proxy.A.com
Proxy
A102
printer.A.com
Printer
A103
XYZ.A.com
PC
A104
ActivKey
Auth. Tool
Dependency
Asset #
Dependent Asset#
A101
A103
…employees need to install ActivKey to access Intranet from their PCs…
Structured Data
Unstructured Data
Step 2: Query expansion using entities and their relations
XYZ cannot access intranet
Original query
ActivKey
proxy.A.com
Related entities
configure proxy authentication …
Relations between entities
Unstructured Data
Retrieved Documents with expanded queries
ActivKey is required for authentication to connect to network
…configure the proxy of your browser to proxy.A.com…
… employees can not access intranet for 2 hours due to network failure on 9/10 …
May not be retrieved with original query

Fig. 1 An example scenario: basic idea of the proposed entity-centric query expansion

With the entities and relations identified in both structured and unstructured data, we propose a general ranking strategy that systematically integrates the entity relations from both data types to rank the entities which have relations with the query entities. Intuitively, related entities should be relevant not only to the entities mentioned in the query but also the query as a whole. Thus, the ranking strategy is determined by not only the relations between entities, but also the relevance of the related entities for the given query and the confidence of entity identification results.

After that, we then study how to exploit related entities and their relations for query expansion. In particular, we explore three options: (1) using only related entities; (2) using related entities and their relations; and (3) using the relations between query entities.

We conduct extensive experiments over real-world enterprise data collections to evaluate the proposed methods. We find that the performance of entity identification is satisfying, and the proposed entity ranking methods are effective to find related entities for a given query. In particular, the relations hidden in the unstructured information are more useful than those in the structured information due to the sparseness of the relationship data in the structured information. Moreover, experimental results show that entity relation based query expansion methods are more effective than state-of-the-art pseudo feedback methods to improve the retrieval performance over longer, natural language-like queries with entities. Result analysis suggests that entity-centric methods make it possible to select more informative and less distracting expansion terms.

To examine whether the proposed methods can work well for the general ad hoc search problem with short keyword queries, we also conduct experiments over a standard TREC collection. Results reveal that our entity-centric query expansion methods can deliver better or comparable performance than the-state-of-the-art feedback methods in term of effectiveness, and our methods demonstrate stronger robustness for performance improvement.

The rest of the paper is organized as follows. We discuss the related work in Sect. 2. Section 3 describes the overview of our work. We then discuss how to find related entities in Sect. 4 and how to use the related entities and their relations for query expansion in Sect. 5. Experiment design and results are discussed in Sect. 6 for enterprise search and Sect. 7 for traditional ad hoc search. Finally, we conclude in Sect. 8.

2 Related work

Enterprise search is an important topic as the failure of finding relevant information will significantly cause the loss of productivities and therefore profit (Feldman and Sherman 2003). However, compared with Web search, enterprise search has received little attention in the research community. Hawking (2004) discussed several challenges in enterprise search, but it remains unclear what are the effective strategies to improve search accuracy. The enterprise track in TREC (Bailey et al. 2007; Balog et al. 2008; Craswell et al. 2005; Soboroff et al. 2006) has attracted more research efforts on improving enterprise search quality including expert finding (Balog 2007; Balog et al. 2006; Balog and de Rijke 2008; Brunnert et al. 2007; Serdyukov et al. 2008), user behavior study (Freund and Toms 2006), metadata extraction (Şah and Wade 2010) and document search (Bailey et al. 2006; Kolla and Vechtomova 2007; Macdonald and Ounis 2006). However, to our best knowledge, little work has been done on *explicitly* utilizing entities and their relationships to improve document search.

Our work is also related to entity retrieval. The entity track of the TREC conference focused on the problem of finding related entities (Balog et al. 2010, 2011). The goal is to retrieve entities that are related to a structured query from a document collection. The query specifies an input entity, the type of the related entities to be found, and the relation between the input and the related entities. The related entities are expected to have the specified type and the specified relation with the input entity. The Entity Ranking track of INEX (Demartini et al. 2009, 2010) also focused on retrieving related entities with the emphasis on the type of target entities (i.e., categories) rather than the relation between the target and input entities. It is interesting to note that Weerkamp et al. (2009) estimated the entity-entity probability based on standard set similarity measure to model the inter-entity relationship, however, the estimated probability is query independent and thus may not reflect the relation specified in query. Besides these, Liu et al. (2011) studied the problem of finding relevant information with specified types. Unlike these previous studies, we used unstructured queries to find the related entities, and no entity type is specified in the query and no explicit relation is specified either. Moreover, the related entities are not returned directly but utilized to improve document search accuracy. Entity retrieval is related to keyword search over relational databases (Coffman and Weaver 2013) in the sense that join operations are performed across different tuples to form an entity-like entries, which will be ranked and retrieved based on relevance with keyword query on syntactic level without leveraging query structure. Instead, we extract the entities in query and rank entity candidates based on their relevance with the query entities estimated on both unstructured data and structured database on semantic level.

Another body of related work is entity linking, particularly on linking entities from free text to structured data. There have been studies on on linking entities to open domain knowledge bases like DBpedia (Auer et al. 2007), YAGO (Suchanek et al. 2007). Mihalcea and Csomai (2007) first moved beyond entity disambiguation and solved the entity linking problem by identifying important concepts from text based on collection statistics and linking them to corresponding Wikipedia pages through existing word sense disambiguation methods. Shen et al. (2012) proposed a novel framework which leverages rich semantic knowledge in Wikipedia and YAGO to achieve superior performance. However, little work has been done on linking entities from free text to relational databases as what we focused on in this paper. Moreover, the primary focus of this paper is to study how to leverage entities and their relations to improve search performance, and we can easily leveraging existing work on entity linking to achieve better accuracy in the entity identification step.

Query expansion is a well known strategy to improve retrieval performance (Cao et al. 2008; Lv and Zhai 2009; Miller et al. 1999; Tao and Zhai 2006). A common strategy in most existing query expansion methods is term-based. Specifically, they use different strategies to select expansion *terms* from feedback documents, user feedback or external sources, and update the existing query through some re-weighting strategies. Zhai and Lafferty (2001) proposed two different approaches (i.e. generative model and risk minimization) to estimate the query language model which fits the relevant documents best with collection background model. Instead of modeling the (pseudo-)relevant documents explicitly, Lavrenko and Croft (2001) proposed to estimate relevance model from feedback documents in a more generalized way, and Lafferty and Zhai (2001) developed an approach which uses Markov chains to estimate query models. Tan et al. (2007) used a cluster-based method to select terms from documents, presented the terms to users for feedback, and used the selected terms to update query models. More recently, Weerkamp et al. (2012) proposed a query-dependent expansion method that enables each query to have its own mixture of external collections for expansion. Lv and Zhai (2010) presented a positional relevance model which leverages proximity and selects useful expansion terms based on their distance from query terms in feedback documents. Metzler and Croft (2007) proposed a Markov random fields based query expansion technique which is capable of incorporating arbitrary features to model the term dependency for query expansion. Instead of selecting expansion terms from feedback documents, we study the feasibility of exploiting related entities and the relations among them for query expansion.

Recently, some research efforts have been done on addressing the challenge of expansion on long verbose queries on which standard term-based query expansion techniques do not perform well. Bendersky et al. (2010, 2011) assumed each query is associated with a set of concepts related to the information need and estimated the weight of each concept in a parameterized fashion. Instead of modeling query term dependency as many recent effective models do, Bendersky and Croft (2012) proposed to model the high-order dependency between arbitrary query concepts through generalization of query hypergraphs. Despite their superior performance on verbose query expansion, they are supervised models and may not be easily extended to other collections when training data is not available. On the contrary, our model works in unsupervised manner and thus can be ported to other collections with moderate efforts.

We summarize our contributions as follows: (1) We focus on the problem of enterprise search which is important but receives little attention; (2) We use both structured and unstructured information to improve the retrieval performance over either short keyword or long narrative queries with entities; (3) To our best knowledge, we are the first one to use entities and their relations for query expansion.

3 Overview of our work

One unique characteristic of enterprise data is the rich information about entities and their relations. As a result, many information needs in enterprise search often center around entities. For example, in the self service IT support scenario, queries may describe the problem of different entities, i.e., IT assets.

In this paper, we focus on the problem of *entity-centric search*, where queries contain at least one entity. In particular, we propose to reformulate entity-centric queries by utilizing the entity relation information embedded in both structured and unstructured information in the enterprise data. This is based on the assumption that entities related to a query should be useful to reformulate the query and improve the retrieval performance.

As shown in Fig. 1, the query contains one entity "XYZ". We can find related entities "ActivKey" and "proxy.A.com" from the relationships specified in both structured and unstructured information in the enterprise data. These related entities together with their relations are able to provide useful information (i.e., terms such as "ActivKey", "proxy", "authentication", "configure", etc.) to retrieve more relevant documents.

Figure 2 illustrates the basic idea of the proposed *entity-centric query expansion* method. Let us first explain the notations.

- Q denotes an entity-centric query;
- E_Q denotes a set of entities in query Q;
- E_R denotes the related entities for query Q;
- Q_E denotes the expanded query of Q;
- $\mathcal{D}$ denotes an enterprise data collection;
- $\mathcal{D}_{TEXT}$ denotes the unstructured information in $\mathcal{D}$;
- $\mathcal{D}_{DB}$ denotes the structured information in $\mathcal{D}$;
- e_i denotes an entity in the structured information $\mathcal{D}_{DB}$;
- $\bigcup\{e_i\}$ denotes the list of all entities complied from $\mathcal{D}_{DB}$;
- m denotes an entity mention in a piece of text;
- $M(T)$ denotes the set of entity mentions in the text T;
- $E(m)$ denotes the set of top K similar candidate entities from $\mathcal{D}_{DB}$ for entity mention m.

The first challenge is how to retrieve a set of entities E_R that are relevant to query Q. Intuitively, the relevance score of an entity should be determined by the relations between the entity and the entities in the query. The entity relation information exists not only *explicitly* in the structured data such as databases in the format of entity relationship (ER) models (Garcia-Molina et al. 2008), but also *implicitly* in the unstructured data such as documents. To identify the entities in the unstructured documents, we go through every document and identify whether it contains any occurrences of entities in the structured databases. Note that this step is done offline. We use a similar strategy to identify E_Q in query Q, and then propose a ranking strategy that can retrieve E_R for the given query Q based on the relations between E_R and E_Q based on information from both $\mathcal{D}_{TEXT}$ and $\mathcal{D}_{DB}$. The details of the entity identification and ranking methods are described in Sect. 4.

Given the related entities E_R, we can use them to estimate the entity relations from both the structured and unstructured data, and use both the entities and relations between entities to expand the original query Q and retrieve documents with the expanded query Q_E. Since the expanded query contains related entities and their relations, the retrieval performance is expected to be improved. This method is described in Sect. 5.

4 Finding related entities

Since structured information is designed based on entity-relationship models, it is straightforward to identify entities and their relations there. However, the problem is more challenging for the unstructured information, where we do not have any information about

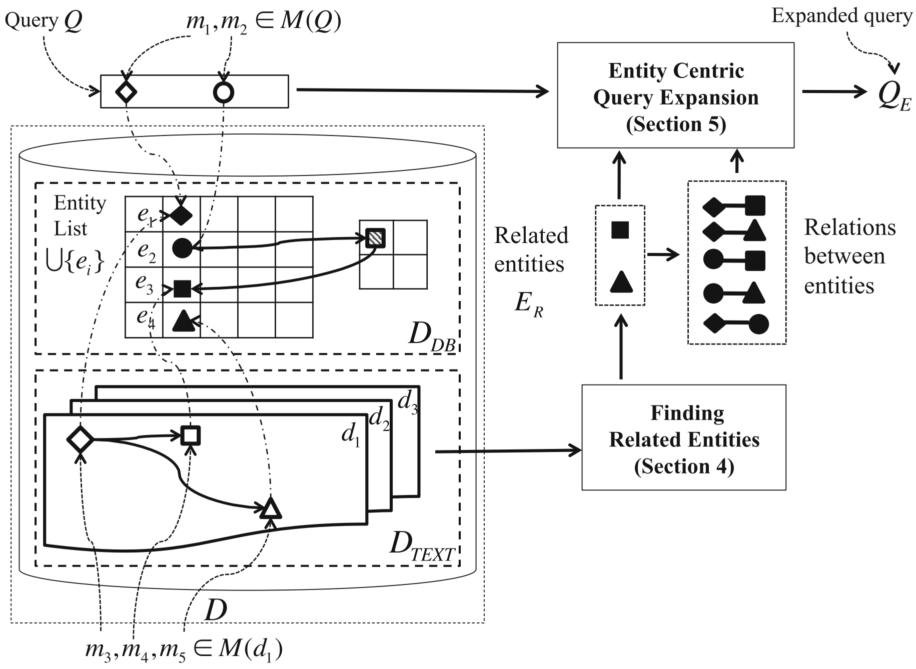


Fig. 2 Overview of the proposed approach

the semantic meaning of a piece of text. In this section, we will first discuss how to identify entities in the unstructured information and then propose a general ranking strategy to rank the entities based on the relations in both unstructured and structured information.

4.1 Entity identification in unstructured data

Unlike structured information, unstructured information does not have semantic meaning associated with each piece of text. As a result, entities are not explicitly identified in the documents, and are often represented as sequences of terms. Moreover, the mentions of an entity could have more variants in unstructured data. For example, the entity “Microsoft Outlook 2003” could be mentioned as “MS Outlook 2003” in one document but as “Outlook” in another.

The majority of entities in enterprise data are *domain specific* entities such as IT assets. These domain specific entities have more variations than the common types of entities. To identify entity mentions from the unstructured information, following existing studies on named entity identification (Carlson et al. 2010; Doan et al. 2009; Sarawagi 2008), we train a model based on conditional random fields (CRFs) (Lafferty et al. 2001) with various features. The model makes binary decision for each term in a document, and the term will be labeled as either an entity term or not.

In particular, we implemented the named entity recognizer based on the open source CRF package[1] with the following three domain specific features:

[1] http://crf.sf.net

- Dictionary feature: the statistics of dictionary terms in the training document;
- POS feature: the Part of Speech (POS) tag of a given term, generated by Illinois Part of Speech Tagger (Rizzolo and Roth 2010);
- Version feature: whether the digital numbers in a given string is the version number of software product.

Besides these, we also included one built-in feature provided in the package: ConcatRegexFeatures, which matches the term with character patterns based on regular expressions (e.g., capitalized word, a number, small case word, special character, ect.) We trained the model on a training document set with their entity mentions manually labeled. Note that the training set is different from the test collections we used in the experiments.

After identifying entity mentions in the unstructured data (denoted as m), we need to connect them with the entities in the structured data (denoted as e) to make both the unstructured and structured data integrated. Specifically, by joining all the tables in a structured database, we get a list of tuples, each of which represents the profile of one entity, and a list of candidate entities will be compiled from the tuple list. Given an entity mention in a document, we calculate its string similarity with every entity in the candidate list and select the most similar candidates. To minimize the impact of entity identification errors, we map one entity mention to multiple candidate entities, i.e., the top K ones with the highest similarities. Each mapping between entity mention m and a candidate entity e is assigned with a mapping confidence score, i.e., $c(m, e)$, which can be computed based string similarity metic. A simple strategy is to employ cosine similarity (or TFIDF) which is widely used in information retrieval community. However, since cosine similarity is based on overlapped terms between strings, it may not work in some cases of our problem setup: one mention for one entity may have the same similarity score with other entities. For example, “Outlook 03” has same cosine similarity with entity “Microsoft Outlook 2003” and “Microsoft Outlook 2007”. The SoftTFIDF string similarity function proposed by Cohen et al. (2003) extends the overlapped term set by incorporating other non-overlapped term for similarity estimation and thus is a good solution to mitigate the problem. We use the implementation provided in the SecondString package[2] in our experiment.

As shown in Fig. 3, $\bigcup\{e_i\}$ is the list of candidate entities compiled from $\mathcal{D}_{DB}$ and m_i is an mentions identified from $\mathcal{D}_{TEXT}$. “Outlook 2003” is an entity mention, and it can be mapped to two entities, i.e. “Microsoft Outlook 2003” and “Microsoft Outlook 2007”. The numbers near the arrows denote the confidence scores of the entity mapping.

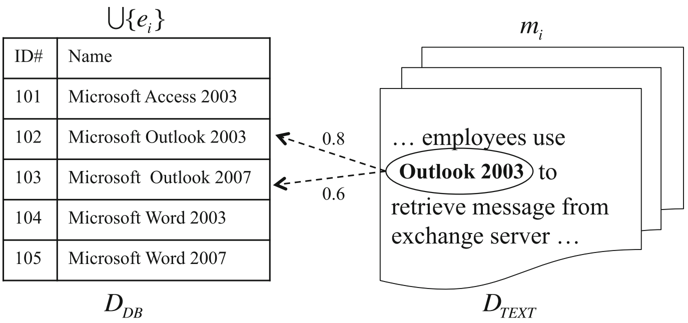
4.2 Entity ranking

The next challenge is how to rank candidate entities for a given query. The underlying assumption is that the relevance of the candidate entity for the query is determined by the relations between the candidate entity and the entities mentioned in the query. If a candidate entity is related to more entities in the query, the entity should have higher relevance score. Formally, the relevance score of a candidate entity e for a query Q can be computed as follows:

$$R_e(Q, e) = \sum_{e_Q \in E_Q} R(e_Q, e). \tag{1}$$

where E_Q denotes the set of query entities in Q, $R(e_Q, e)$ is the relevance score between query entity e_Q and a candidate entity e based on their relations in collection $\mathcal{D}$. As the

[2] SecondString Java package: http://secondstring.sourceforge.net/



ID#	Name
101	Microsoft Access 2003
102	Microsoft Outlook 2003
103	Microsoft Outlook 2007
104	Microsoft Word 2003
105	Microsoft Word 2007

$\bigcup\{e_i\}$
m_i
0.8
0.6
… employees use Outlook 2003 to retrieve message from exchange server …
D_{DB}
D_{TEXT}

Fig. 3 Entity identification and mapping

example shown in Fig. 2, there are two query entities e_1 and e_2 in $\mathcal{D}_{DB}$, and they are mapped from entity mentions m_1 and m_2 in query Q, respectively. There are two candidate entities e_3 and e_4, and e_3 is related to e_1 and e_2 through $\mathcal{D}_{TEXT}$ and $\mathcal{D}_{DB}$, and e_4 is related to e_1 through $\mathcal{D}_{TEXT}$ only.

Recall that, for every entity mention in the query, there could be multiple (i.e., K) possible matches from the entity candidate list and each of them is associated with a confidence score. Thus, the relevance score of candidate entity e for a query entity mention m can be computed using the weighted sum of the relevance scores between e and the top K matched candidate entity of the query entity mention. Thus, Eq. (1) can be rewritten as:

$$R_e(Q,e) = \sum_{m \in M(Q)} \sum_{e_Q \in E(m)} c(m, e_Q) \cdot R(e_Q, e), \tag{2}$$

where $M(Q)$ denotes the set of entity mentions in Q, $E(m)$ denotes the set of K candidate entities for entity mention m in the query, e_Q denotes one matched candidate entity of m, and $c(m, e_Q)$ is the SoftTFIDF string similarity (Cohen et al. 2003) between m and e_Q.

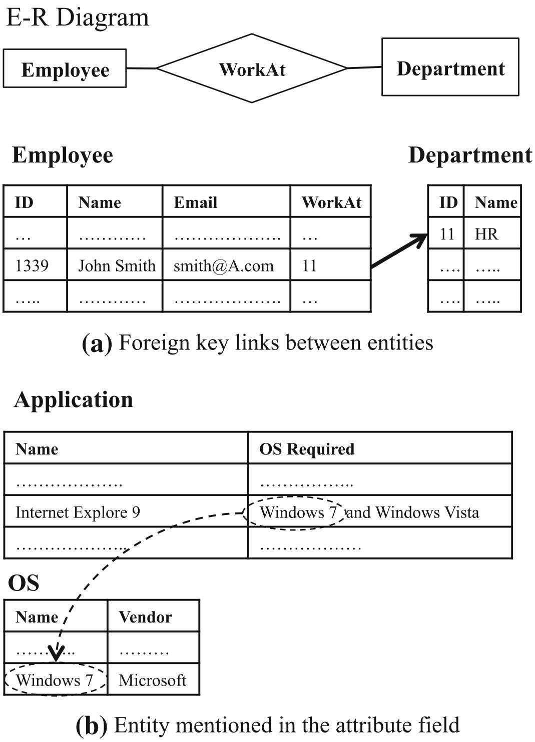
We now discuss how to exploit the characteristics of both unstructured and structured information to compute the relevance score between two entities, i.e., $R(e_Q, e)$ based on their relations.

4.2.1 Using relationships from the structured data

In relational databases, every table corresponds to one type of entities, and every tuple in a table corresponds to an entity. The database schema describes the relations between different tables as well as the meanings of their attributes.

We consider two types of entity relations. First, if two entities are connected through foreign key links between two tables, these entities will have the same relation as the one specified between the two tables. For example, as shown in Fig. 4a, entity “John Smith” is related to entity “HR”, and their relationship is “WorkAt”. Second, if one entity is mentioned in an attribute field of another entity, the two entities have the relation specified in the corresponding attribute name. As shown in Fig. 4b, entity “Windows 7” is related to entity “Internet Explorer 9” through relation “OS Required”. We now discuss how to compute the relevance scores between entities based on these two relation types.

Fig. 4 Entity relations in structured data



Employee

ID	Name	Email	WorkAt
…	………..	……………….	…
1339	John Smith	smith@A.com	11
…..	………..	……………….	…

Department

ID	Name
11	HR
….	…..
….	…..

(a) Foreign key links between entities

Application

Name	OS Required
……………….	……………..
Internet Explore 9	Windows 7 and Windows Vista
……………….	………………

OS

Name	Vendor
………..	………
Windows 7	Microsoft

(b) Entity mentioned in the attribute field

The relevance scores based on *foreign key relations* are computed as:

$$R^{LINK}(e_Q, e) = \begin{cases} 1 & \text{if there is a link between } e_Q \text{ and } e \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

The relevance scores based on *field mention relations* are computed as:

$$R^{FIELD}(e_Q, e) = \sum_{m \in M(e_Q.text)} c(m, e) + \sum_{m \in M(e.text)} c(m, e_Q), \tag{4}$$

where *e.text* denotes the union of text in the attribute fields of *e*.

We can get the final ranking score by combining the two types of relevance score through linear interpolation:

$$R^{DB}(e_Q, e) = \alpha R^{LINK}(e_Q, e) + (1 - \alpha) R^{FIELD}(e_Q, e) \tag{5}$$

where α serves as a coefficient to control the influence of two components. Note that both $R^{LINK}(e_Q, e)$ and $R^{FIELD}(e_Q, e)$ are normalized to the same range before linear interpolation.

4.2.2 Using relationships from unstructured data

Unlike in the structured data where entity relationships are specified in the database schema, there is no explicit entity relationship in the unstructured data. Since the co-occurrences of entities may indicate certain semantic relations between these entities, we use the co-occurrence relations in this paper. Our experimental results in Sect. 6 show that such co-occurrence relations can already deliver good performance in entity ranking and query expansion. We may also apply advanced NLP techniques to automatically extract relations (Zelenko et al. 2003; Zhu et al. 2009), and we leave it as our future work.

After identifying entities from unstructured data and connecting them with candidate entities as described in the previous subsections, we are able to get the information about co-occurrences of entities in the document sets. If an entity co-occurs with a query entity in more documents and the context of the co-occurrences is more relevant to the query, the entity should have higher relevance score.

Formally, the relevance score can be computed as follows:

$$\begin{aligned} R^{TEXT}(e_Q, e) = & \sum_{d\in\mathcal{D}_{TEXT}} \sum_{m_Q\in M(d) e_Q\in E(m_Q)} \sum_{m\in M(d) e\in E(m)} \\ & S\Big(Q, WINDOW(m_Q, m, d)\Big) \cdot c(m_Q, e_Q) \cdot c(m, e), \end{aligned} \tag{6}$$

where d denotes a document in the unstructured data of enterprise collection, and $WINDOW(m_Q, m, d)$ is the context window of the two entities mentions in d and it is centered at m_Q. The underlining basic assumption is that the relations between two entities can be captured through their context. Thus, the relevance between the query and context terms can be used to model the relevance of the relations between two entities for the given query. As the example shown in Fig. 2, the query entity e_1 is mentioned in d_1 as m_3, and candidate entity e_3 is mentioned in d_1 as m_4. Assuming d_1 is the only document in which e_1 and e_3 co-occur, the relevance between e_1 and e_3 can be estimated as:

$$R^{TEXT}(e_1, e_3) = S\Big(Q, WINDOW(m_3, m_4, d_1)\Big) \cdot c(m_3, e_1) \cdot c(m_4, e_3).$$

The context window size is set to 64 based on preliminary results. If the position of m is beyond the window, it will be considered as non-related. $S(Q, WINDOW(m_Q, m, d))$ measures the relevance score between query Q and content of context window $WINDOW(m_Q, m, d)$. Since both of them are essentially bag of words, the relevance score between them can be estimated with any existing document retrieval models.

5 Entity centric query expansion

We now discuss how to utilize the related entities and their relations to improve the performance of document retrieval. As shown in Fig. 1, we observe that the related entities, which are relevant to the query but are not directly mentioned in the query, as well as the relations between the entities, can serve as complementary information to the original query terms. Therefore, integrating the related entities and their relations into the query can help the query to cover more information aspects, and thus improve the performance of document retrieval.

Language modeling (Ponte and Croft 1998) has been a popular framework for document retrieval in the recent decade. One of the popular retrieval models is KL-divergence (Zhai and Lafferty 2001), where the relevance score of document D for query Q can be estimated based on the distance between the document and query models, i.e.

$$S(Q, D) = \sum_{w} p(w|\theta_Q) \log p(w|\theta_D).$$

To further improve the performance, Zhai and Lafferty (2001) proposed to update the original query model using feedback documents as follows:

$$\theta_Q^{new} = (1 - \lambda)\theta_Q + \lambda\theta_{\mathcal{F}}, \tag{7}$$

where θ_Q is the original query model, $\theta_{\mathcal{F}}$ is the estimated feedback query model based on feedback documents, and λ controls the influence of the feedback model.

Unlike previous work where the query model is updated with terms selected from feedback documents, we propose to update it using the related entities and their relations. Following the sprit of model-based feedback methods (Zhai and Lafferty 2001), we propose to update the query model as follows:

$$\theta_Q^{new} = (1 - \lambda)\theta_Q + \lambda\theta_{ER}, \tag{8}$$

where θ_Q is the query model, θ_{ER} is the estimated expansion model based on related entities and their relations and λ controls the influence of θ_E. Given a query Q, the relevance score of a document D can be computed as:

$$S(Q, D) = \sum_w \Big((1 - \lambda)p(w|\theta_Q) + \lambda p(w|\theta_{ER})\Big) \log p(w|\theta_D) \tag{9}$$

The main challenge here is how to estimate $p(w|\theta_{ER})$ based on related entities and their relations.

5.1 Entity name based expansion

Given a query, we have discussed how to find related entities E_R in the previous section. We think the top ranked related entities can provide useful information to better reformulate the original query. Here we use "bags-of-terms" representation for entity names, and a name list of related entities can be regarded as a collection of short documents. Thus, we propose to estimate the expansion model based on the related entities as follows:

$$p\big(w|\theta_{ER}^{NAME}\big) = \frac{\sum_{e_i \in E_R^L} c(w, N(e_i))}{\sum_{w'} \sum_{e_i \in E_R^L} c(w', N(e_i))} \tag{10}$$

where E_R^L is the top L ranked entities from E_R, $N(e)$ is the name of entity e and $c(w, N(e))$ is the occurrence of w in $N(e)$.

5.2 Relation based expansion

Although the names of related entities provide useful information, they are often short and their effectiveness to improve retrieval performance could be limited. Fortunately, the relations between entities could provide additional information that can be useful for query reformulation. We focus on two relation types: (1) *external relations:* the relations between a query entity and its related entities; (2) *internal relations:* the relations between two query entities. For example, consider the query in Fig. 1 "XYZ cannot access intranet": it contains only one entity "XYZ", the *external relation* with the related entities, e.g. "ActivKey", would be: "ActivKey is required for authentication of XYZ to access the intranet". Consider another query "Outlook can not connect to Exchange Server", there are two entities "Outlook" and "Exchange Server", and they have an *internal relation*, which is "Outlook retrieve email messages from Exchange Server".

The key challenge here is how to estimate a language model based on the relations between two entities. As discussed earlier, the relation information exists as co-occurrence context about entities in documents of unstructured data. To estimate the model, we propose to pool all the relation information together, and use maximum likelihood estimation to estimate the model.

Specifically, given a pair of entities, we first find all the relation information from the enterprise collection $\mathcal{D}$, and then estimate the entity relation as follows:

$$p\big(w|\theta_{ER}^{R}, e_1, e_2)\big) = p_{ML}(w|CONTEXT(e_1, e_2)), \tag{11}$$

where *CONTEXT*(e_1,e_2) is the set of documents mentioning both entities, and p_{ML} is the maximum likelihood estimate of the document language model.

Thus, given a query Q with E_Q as a set of query entities and E_R^L as a set of top L related entities, the external entity relation $\theta_{ER}^{R_{ex}}$ can be estimated by taking the average over all the possible entity pairs, showed as below:

$$p\big(w|\theta_{ER}^{R_{ex}}\big) = \frac{\sum_{e_r \in E_R^L} \sum_{e_q \in E_Q} p\big(w|\theta_{ER}^{R}, e_r, e_q\big)}{|E_R^L| \cdot |E_Q|}, \tag{12}$$

where $|E_Q|$ denotes the number of entities in the set E_Q. Note that $|E_R^L| \leq L$ since some queries may have less than L related entities.

Similarly, the internal relation entity relation $\theta_{ER}^{R_{in}}$ is estimated as:

$$p\big(w|\theta_{ER}^{R_{in}}\big) = \frac{\sum_{e_1 \in E_Q} \sum_{e_2 \in E_Q, e_2 \neq e_1} p\big(w|\theta_{ER}^{R}, e_1, e_2\big)}{\frac{1}{2} \cdot |E_Q| \cdot (|E_Q| - 1)}. \tag{13}$$

Note that $\frac{1}{2} \cdot |E_Q| \cdot (|E_Q| - 1) = \binom{|E_Q|}{2}$ as we only count the co-occurrences of different entities.

Compared with the entity name based expansion, the relation based expansion method can be viewed as a generalization of entity named based expansion in the sense that *CONTEXT*(e_1, e_2) is the extension from entity name to the context of entities. In fact, the expansion terms generated from the relation-based expansion form a superset of those from entity name based method.

5.3 Discussions

Efficiency is a critical concern for information retrieval systems. The proposed entity-centric query expansion methods can be implemented as efficiently as traditional methods. First, entity identification for documents can be done offline, and we can build an entity-based inverted index which can make the data access more efficiently. The cost of entity identification on a query is negligible since the query is relatively short. Second, finding related entities from structured information could be rather fast given the efficiency support provided by existing relational databases. And finding those from unstructured information could be implemented efficiently through building the entity-based inverted index so that the cost of searching for documents covering both query entities and candidate entities could be minimized. Finally, traditional pseudo feedback methods require two rounds of retrieval, i.e., to get initial ranking for term selection and to generate the final ranking. However, our methods do not require the first round of initial ranking.

Although we focus on extending feedback methods in language models only in this paper, we expect that other retrieval models can be extended similarly and leave this as our future work.

6 Experiments in enterprise search domain

6.1 Experiment design

To evaluate the proposed methods, we have constructed two enterprise data sets using real-world data from HP. They are denoted as **ENT1** and **ENT2**. Each data set consists of two parts: unstructured documents and structured databases.

- The unstructured documents are knowledge base documents which are provided by IT support department of HP. Most of the documents are talking about how-to and troubleshooting for the software products used in HP. More specifically, **ENT1** contains the information about HP's products while **ENT2** contains the information about the Microsoft and IBM's products.
- The structured data include a relational database which contains information about 2,628 software products.

Queries are collected from HP's internal IT support forum as the query set. Almost all the queries are described in natural languages, and the average query length is 8 terms, which is much longer than keyword queries used in Web search. The queries are selected based on the criterion that each query contains at least one entity. Let us consider a query from the query set, i.e., "Office 2003 SP3 installation fails on Windows XP". It mentions two entities: "Office 2003" and "Windows XP". For each query, we employ assessors to manually label the relevance of each entity for the evaluation of finding related entities. We also follow the pooling strategy used in TREC to get the top 20 documents from each of the evaluated methods as candidates and ask human assessors to manually label their relevance. All results are reported in MAP (Mean Average Precision). The statistics of two data sets are summarized in Table 1.

Note that we did not use existing TREC enterprise data sets because both W3C and CSIRO collections (Bailey et al. 2007; Craswell et al. 2005) contain unstructured data (e.g,. documents) only and do not have the complementary structured data such as the ones we have in our collections.

6.2 Effectiveness of finding related entities

6.2.1 Entity identification

In order to test the accuracy of our entity identification approach, we manually labeled the occurrences of the 200 software products in a randomly selected set of documents in **ENT1**. This sample contained 2,500 documents, and we found 3,252 occurrences of software products. We then did a fivefold cross-validation on this sample. The results showed that the precision of our entity identification is 0.852, the recall is 0.908, and the $F1$ is 0.879. This indicates that we can effectively find entity mentions in the unstructured documents.

6.3 Entity ranking

We evaluate the effectiveness of our entity ranking methods. By plugging Eqs. (5) and (6) into Eq. (2), we can get different entity ranking models, which are denoted as R_e^{DB} and R_e^{TEXT}, respectively. Moreover, structured and unstructured data may contain different

Table 1 Statistics of two enterprise data sets

Data set	# Q	# Doc	Avg. Doc. length	Avg. Rel. entity	Avg. Rel. Doc
ENT1	60	59,706	117	6.1	3.2
ENT2	100	262,894	330	9.7	2.8

relations between entities. Thus, it would be interesting to study whether combining these relations could bring any benefits. We can combine them through a linear interpolation:

$$R_e^{BOTH}(Q,e) = \beta R_e^{DB}(Q,e) + (1-\beta)R_e^{TEXT}(Q,e) \tag{14}$$

where β balances the importance of the relations from two sources. Both $R_e^{DB}(Q, e)$ and $R_e^{TEXT}(Q, e)$ are normalized to the same range before linear interpolation.

Table 2 presents the results under optimized parameter settings (denoted as "Optimized") and fivefold cross-validation (denoted as "fivefold")[3]. We notice that the performance of R_e^{TEXT} is much better than R_e^{DB} on both data sets, implying the relations in the unstructured documents are more effective than those in the structured data. The R_e^{BOTH} model can reach the best performance on both data sets, and its improvement over R_e^{TEXT} is statistically significant on **ENT2**.

By analyzing the data, we find that the main reason for the worse performance of structured data based entity ranking (i.e. R_e^{DB}) is that the number of relations between entities (either foreign key links or entity mention in the attribute field) is much smaller than that in the unstructured data. Only 37.5% of entities have relationships in the structured data. We expect that the performance of R_e^{DB} could be improved if the structured data can provide more information about entity relations.

The parameter values used to achieve the optimized performance are similar on both data collections, which indicates that using the parameters trained on one collection would get near-optimal performance on the other data set. Specifically, K is set to 4, which means that we have 1–4 mapping between an entity mention from the documents and the candidate entities from the databases. α in Eq. (5) is set to 0.7 indicating that the foreign link relations is more important than entity mention relations. And β in Eq. (14) is set to 0.3, which suggests that the unstructured data contributes most to rank the related entities.

6.4 Effectiveness of query expansion in enterprise search

We design four sets of experiments to evaluate the effectiveness of the proposed entity-centric query expansion methods. First, we compare the proposed entity name based expansion methods. Second, we evaluate the effectiveness of the two relation-based expansion methods. Third, we compare the proposed expansion methods with the state-of-the-art feedback methods. Finally, we construct a small data set to understand the effectiveness of internal relation models.

The entity-centric expansion function is shown in Eq. (9). In the experiments, we estimate $p(w|\theta_Q)$ by maximum likelihood, i.e. $p(w|\theta_Q) = \frac{count(w,Q)}{|Q|}$, where $count(w, Q)$ is the number of occurrences of w in Q and $|Q|$ is the query length. And $p(w|\theta_D)$ can be estimated using smoothing methods such as Dirichlet Prior (Zhai and Lafferty 2001).

[3] The notations will be used throughout the remaining of the paper.

Table 2 Results of finding related entities

Models	Equations	ENT1		ENT2	
		Optimized	Fivefold	Optimized	Fivefold
R_e^{DB}	Plugging (5) in (2)	0.1246	0.1198	0.1695	0.1695
R_e^{TEXT}	Plugging (6) in (2)	$0.5740^{\triangle}$	$0.5740^{\triangle}$	$0.6448^{\triangle}$	$0.6448^{\triangle}$
R_e^{BOTH}	(14)	$\mathbf{0.5907}^{\triangle}$	$\mathbf{0.5804}^{\triangle}$	$\mathbf{0.6614}^{\triangle\blacktriangle}$	$\mathbf{0.6614}^{\triangle\blacktriangle}$

$^{\triangle}$ and $^{\blacktriangle}$denote the improvements over R_e^{DB} and R_e^{TEXT} are statistically significant at 0.05 level using Wilcoxon signed-rank test, respectively

Table 3 Results of *entity name* based query expansion

Models	ENT1		ENT2	
	Optimized	Fivefold	Optimized	Fivefold
NoFB	0.2165	0.2165	0.4272	0.4272
QE_{DB}^{NAME}	0.2347	0.2274	0.4272	0.4138
QE_{TEXT}^{NAME}	$0.2557^{\triangle}$	$\mathbf{0.2557}^{\triangle}$	$\mathbf{0.4335}^{\triangle}$	0.4219
QE_{BOTH}^{NAME}	$\mathbf{0.2561}^{\triangle}$	$0.2528^{\triangle}$	$0.4328^{\triangle}$	**0.4311**

$^{\triangle}$ denotes improvement over *NoFB* is statistically significant at 0.05 level based on Wilcoxon signed-rank test

Thus, the basic retrieval model (i.e., when $\lambda = 0$) is the KL-divergence function with Dirichlet prior smoothing (Zhai and Lafferty 2001), which is one of the state-of-the-art retrieval functions. We denote it as *NoFB*. The smoothing parameter μ is set to 250 in all experiments based on the optimized setting for *NoFB* (tuned from 100 to 5,000).

6.4.1 *Entity name based expansion*

As described in Sect. 5.1, we can expand queries with the names of entities that are related to the query. Specifically, the entity name based expansion model (i.e., Eq. 10) using entity lists from R_e^{DB}, R_e^{TEXT} and R_e^{BOTH} are denoted as QE_{DB}^{NAME}, QE_{TEXT}^{NAME} and QE_{BOTH}^{NAME} respectively. The results are reported in Table 3. It is clear that QE_{TEXT}^{NAME} and QE_{BOTH}^{NAME} can improve the retrieval performance over *NoFB* significantly, and they are more effective than QE_{DB}^{NAME}.

6.4.2 *Relation based expansion*

For the relation based expansion method, we use the related entity list of R_e^{BOTH} as E_R. The expanded query models using $\theta_{ER}^{R_{ex}}$ and $\theta_{ER}^{R_{in}}$ are denoted as $QE^{R_{ex}}$ and $QE^{R_{in}}$, respectively. Besides these, since the information from these two models is complementary to each other, we could combine them through linear interpolation as follows:

$$p(w|\theta_{ER}) = \gamma p\left(w|\theta_{ER}^{R_{ex}}\right) + (1-\gamma)p\left(w|\theta_{ER}^{R_{in}}\right), \tag{15}$$

and use the combined θ_{ER} to do query expansion, which is denoted it as $QE^{R_{ex}+R_{in}}$.

The optimized results are reported in Table 4. We can find that all of the relation based query expansion models can outperform *NoFB*, and the improvements of all models are

Table 4 Results of *relation* based query expansion

Models	ENT1		ENT2	
	Optimized	Fivefold	Optimized	Fivefold
NoFB	0.2165	0.2165	0.4272	0.4272
$QE^{R_{ex}}$	$0.2792^{\triangle}$	$0.2629^{\triangle}$	$0.4628^{\triangle}$	$0.4560^{\triangle}$
$QE^{R_{in}}$	$0.2442^{\triangle}$	$0.2442^{\triangle}$	$0.4450^{\triangle}$	$0.4425^{\triangle}$
$QE^{R_{ex}+R_{in}}$	$\mathbf{0.2920}^{\triangle}$	$\mathbf{0.2780}^{\triangle}$	$\mathbf{0.4634}^{\triangle}$	$\mathbf{0.4574}^{\triangle}$

$^{\triangle}$ denotes improvement over *NoFB* is statistically significant at 0.05 level based on Wilcoxon signed-rank test

statistically significant. It shows the effectiveness of relation-based expansion methods. Moreover, $QE^{R_{ex}}$ outperforms $QE^{R_{in}}$, and combining both relations yields the best performance.

6.4.3 Performance comparison with existing feedback methods

We compare the best method of the proposed entity name based expansion (i.e., QE^{NAME}_{BOTH}) and that of the proposed relation based expansion (i.e., $QE^{R_{ex}} + QE^{R_{in}}$) models with a set of state-of-the-art feedback methods. The first one is model-based feedback method (Zhai and Lafferty 2001), denoted as *ModFB*. The second one is the relevance model (Lavrenko and Croft 2001), denoted as *RelFB*. The third one is the latent concept expansion (Metzler and Croft 2007)[4], denoted as *LCE*, which incorporates term dependence and has been shown to perform well on long queries (Bendersky et al. 2011).

The optimized performance is shown in Table 5. The corresponding parameter settings for *ModFB* are to select top 10 feedback documents and top 20 terms for expansion, set the weight for feedback model $\alpha = 0.1$ and the weight for collection language model $\lambda = 0.3$. Those for *RelFB* are to select top 10 feedback documents and top 25 terms for expansion, set the smoothing parameter $\lambda = 0.6$. Those for *LCE* are to select top 25 feedback documents and top 50 terms for expansion, set the weight for unigram potential function $\lambda_{T_D} = 0.32$, the weight for bigram potential functions $\lambda_{O_D} = \lambda_{U_D} = 0.04$ and the weight for feedback model $\lambda'_{T_D} = 0.60$. Those for QE^{NAME}_{BOTH} are to select top 4 entities for expansion and set the weight for feedback model $\lambda = 0.4$. Those for $QE^{R_{ex}+R_{in}}$ are to select top 5 entities for expansion and set the weight for feedback model $\lambda = 0.6$.

We observe that our best query expansion method $QE^{R_{ex}+R_{in}}$ significantly outperforms three baselines methods, proving the effectiveness of our entity-centric query expansion approach. Furthermore, $QE^{R_{ex}+R_{in}}$ outperforms QE^{NAME}_{BOTH}, implying the entity relations contain more useful information than entity names. Finally, we notice that the improvements of *ModFB* and *RelFB* over *NoFB* are marginal, implying that they are not effective for expanding long queries, while *LCE* demonstrates much better performance (although still not statistically significant over *NoFB*). The superior performance of *LCE* over *RelFB* is consistent with the observations in the previous study (Bendersky et al. 2011), and its main advantage is contributed by the incorporation of term dependence as *LCE* is a generalization of *RelFB* from unigram language model to Markov Random Field (Metzler and Croft 2005, 2007).

[4] Implementation provided by Ivory: http://lintool.github.io/Ivory/

Table 5 also shows the results of fivefold cross-validation. And the results reveal that $QE^{R_{ex}+R_{in}}$ is more robust to parameter settings and performs better than all four baselines as well.

We also use one data set for training to get the optimized parameter settings for each of our query expansion models, and apply it to the other data set accordingly. The results are summarized in Table 6. We can find that $QE^{R_{ex}+R_{in}}$ is robust and can still outperform most baselines, which is consistent with our observation in Table 5, and QE^{NAME}_{BOTH} is sensitive to the parameter settings. Furthermore, among the three baseline feedback methods, *RelFB* and *ModFB* do not perform well under testing parameter setting and cross-validation, implying they are more sensitive to the parameter setting, while *LCE* exhibits much stronger robustness. Finally, the performance differences between $QE^{R_{ex}+R_{in}}$ and *LCE* are not statistically significant. One advantage of our method is the lower computational cost since *LCE* takes all the bigrams from query for relevance estimation while ours focuses only on important concepts (i.e., entities) in the query. Also, our models involves fewer parameters than *LCE*, which means less tuning effort.

6.4.4 *Robustness of query expansion methods*

Robustness of a query expansion method is also important since a robust expansion method is expected to improve the performance for more queries and hurt the performance for

Table 5 Performance comparison with existing feedback methods

Models	**ENT1**		**ENT2**	
	Optimized	Fivefold	Optimized	Fivefold
NoFB	0.2165	0.2165	0.4272	0.4272
ModFB	0.2210	0.1988	0.4279	0.4265
RelFB	0.2443	0.2277	$0.4385^{\triangle\blacktriangle}$	0.4147
LCE	0.2727	$0.2559^{\star}$	0.4559	0.4354
QE^{NAME}_{BOTH}	$0.2561^{\triangle\blacktriangle}$	$0.2528^{\triangle\blacktriangle\star}$	$0.4328^{\triangle\blacktriangle}$	$0.4311^{\blacktriangle}$
$QE^{R_{ex}+R_{in}}$	$\mathbf{0.2920}^{\triangle\blacktriangle\star}$	$\mathbf{0.2780}^{\triangle\blacktriangle\star}$	$\mathbf{0.4634}^{\triangle\blacktriangle\star}$	$\mathbf{0.4574}^{\triangle\blacktriangle\star}$

$^{\triangle}, ^{\blacktriangle}, ^{\star}$ and † denote improvements over *NoFB*, *ModFB*, *RelFB* and *LCE* are statistically significant at 0.05 level based on Wilcoxon signed-rank test, respectively

Table 6 Testing performance comparison with existing methods

Test collection Parameters trained on	**ENT1** **ENT2**	**ENT2** **ENT1**
NoFB	0.2165	0.4272
ModFB	0.2184	0.4227
RelFB	0.2266	0.4001
LCE	0.2517	0.4473
QE^{NAME}_{BOTH}	0.2446	0.4241
$QE^{R_{ex}+R_{in}}$	**0.2485**	$\mathbf{0.4487}^{\triangle\blacktriangle\star}$

$^{\triangle}, ^{\blacktriangle}, ^{\star}$ and † denote improvements over *NoFB*, *ModFB*, *RelFB* and *LCE* are statistically significant at 0.05 level based on Wilcoxon signed-rank test, respectively

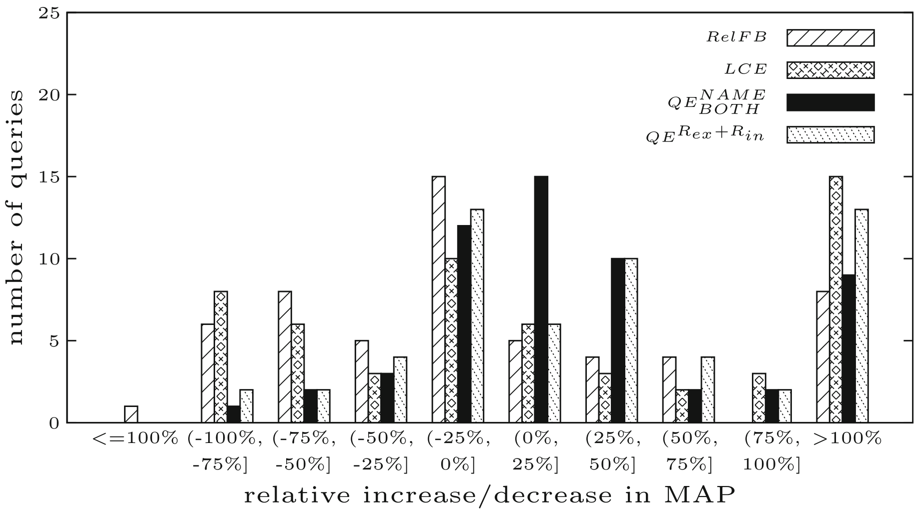


Fig. 5 Histogram of queries when applied with *RelFB*, *LCE*, QE^{NAME}_{BOTH} and $QE^{R_{ex}+R_{in}}$ compared with *NoFB* on **ENT1**

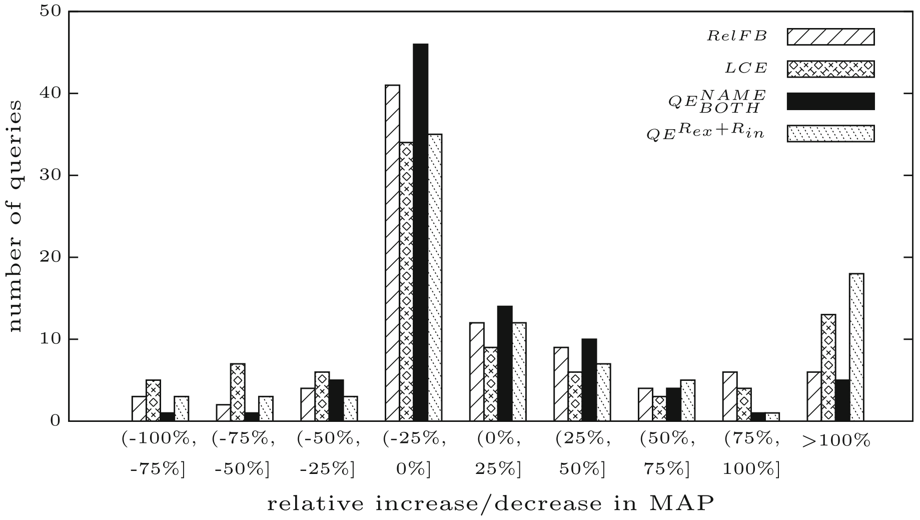


Fig. 6 Histogram of queries when applied with *RelFB*, *LCE*, QE^{NAME}_{BOTH} and $QE^{R_{ex}+R_{in}}$ compared with *NoFB* on **ENT2**

fewer queries (Wang et al. 2012). To investigate the robustness of our models, we report the number of queries which are improved/hurt (and by how much) after applying different query expansion methods.

The results over the two collections are shown in Figs. 5 and 6. The x-axis represents the relative increases/decreases in MAP clustered in several groups, and y-axis represents the number of queries in each group. The bars to the left of (0, 25 %] represent queries whose performance are hurt by using the query expansion methods, and the other bars represent queries whose performance are improved using expansion methods. We choose both *RelFB* and *LCE* as the feedback baselines to be compared with our methods, as *ModFB* could only improve over *NoFB* marginally.

Clearly, both of our methods are more robust than both *RelFB* and *LCE*. For **ENT1**, QE^{NAME}_{BOTH} improves 38 queries and hurts 12, $QE^{R_{ex}+R_{in}}$ improves 35 queries and hurts 17, whereas *RelFB* improves 23 queries and hurts 28, *LCE* improves 30 queries and hurts 22. For **ENT2**, QE^{NAME}_{BOTH} improves 35 queries and hurts 24, $QE^{R_{ex}+R_{in}}$ improves 46 queries and hurts 18, whereas *RelFB* improves 39 queries and hurts 21, *LCE* improves 36 and hurts 32.

6.4.5 Result analysis on expansion terms

We analyze the expansion terms generated by different methods, and find that the relation based expansion can provide higher quality terms than *ModFB* and *RelFB*. Table 7 shows the top 5 weighted expansion term by different methods for query "Internet Explorer can not list directory of FTP". It is clear that *ModFB* cannot find a useful term, *RelFB* and QE^{NAME}_{BOTH} can find a useful term "server", while $QE^{R_{ex}+R_{in}}$ can find more useful terms as the problem may be caused by 'file" permission "property" or "connection" settings to the "server". The main difference between our methods and *ModFB* is the estimation of expansion models, i.e., θ_{ER} estimated based on entity relations vs. $\theta_{\mathcal{F}}$ estimated from feedback documents in *ModFB*. Thus, it is clear the our proposed entity-centric models are effective in extracting high quality terms.

6.4.6 Further analysis on internal relation expansion

We notice that in Table 4, the performance improvement of applying internal relation for query expansion (i.e., $QE^{R_{in}}$) is much smaller than that of applying external relation

Table 7 Top 5 weighted expansion terms for query "Internet Explorer can not list directory of FTP"

Models	Expansion terms
ModFB	client, open, site, data, process
RelFB	file, **server**, site, click, name
QE^{NAME}_{BOTH}	**server**, windows, xp, vista, 2003
$QE^{R_{ex}+R_{in}}$	**file, connect, property**, xp, **server**

Terms denoted in **bold** font are potentially helpful to improve performance

Table 8 Performance comparison over a set of 29 queries, each of which contains multiple query entities

Models	Optimized parameter	Fivefold cross-validation
NoFB	0.3855	0.3855
ModFB	0.3899	0.3497
RelFB	0.4184	0.4095
LCE	0.4168	0.4059
$QE^{R_{ex}}$	0.4693$^{\triangle\blacktriangle}$	0.4286$^{\triangle\blacktriangle}$
$QE^{R_{in}}$	0.4858$^{\triangle\blacktriangle\bigstar}$	0.4762$^{\triangle\blacktriangle}$
$QE^{R_{ex}+R_{in}}$	**0.4939**$^{\triangle\blacktriangle\bigstar}$	**0.4920**$^{\triangle\blacktriangle\bigstar}$

$^{\triangle}$, $^{\blacktriangle}$, $^{\bigstar}$ and † denote improvements over *NoFB*, *ModFB*, *RelFB* and *LCE* are statistically significant at 0.05 level based on Wilcoxon signed-rank test, respectively

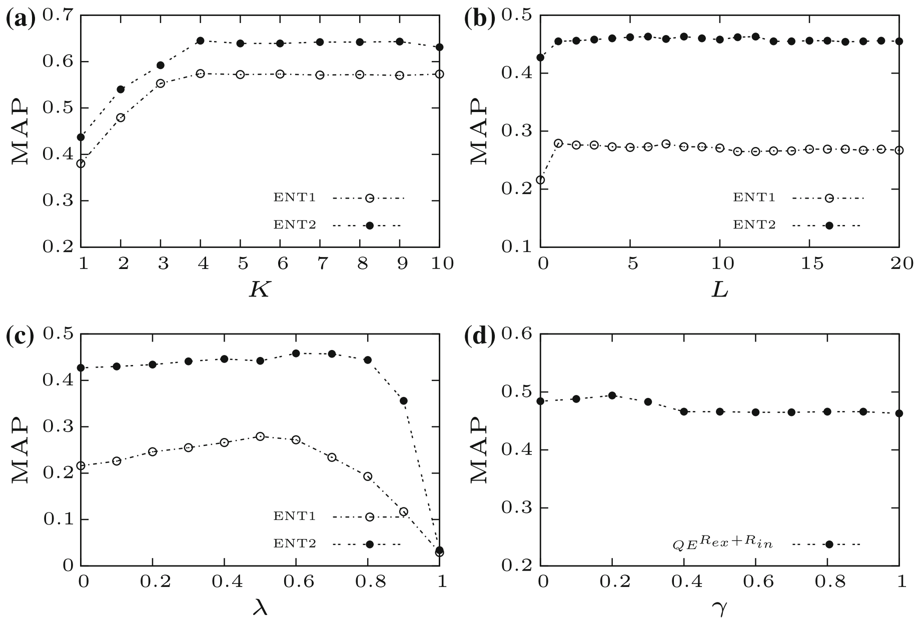


Fig. 7 Parameter sensitivity on enterprise collection

(i.e., $QE^{R_{ex}}$). This may be caused by the fact that not all the queries have more than one entity, and only those queries with more than one entity would benefit from the expansion using internal relation. Among all the 100 queries in **ENT2**, there are 29 queries qualified for internal relation expansion[5].

To validate our hypothesis, we evaluate the performance of two baselines as well as $QE^{R_{ex}}$, $QE^{R_{in}}$ and $QE^{R_{ex}+R_{in}}$ on these 29 queries and summarize the results in Table 8. Clearly, when queries have multiple entities, using internal relations can significantly improve the performance.

6.5 Parameter sensitivity

We now report the performance sensitivity of parameters used in our methods.

The first parameter is K for finding related entity models. K is the number of candidate entities from the structured data that an entity mention can be mapped to. As shown in Fig. 7a, when K is larger than 4, the performance of R_e^{TEXT} remains stable. This suggests that the confidence scores associated with the mapping are reasonable. Even if we include more candidates, the confidence scores are able to reduce the impact of noisy entities. Moreover, we observe that when K is smaller than 4, the performance decreases, which implies that one to multiple mapping enables us to find more relevant entities. As the computational cost increases with K, and when K is greater than 4 it would not yield any further improvement, 4 would be the optimal suggested value.

[5] Actually there are 9 queries qualified for internal relation expansion in **ENT1**. However, since the query set is too small to construct working set, we do not report the results.

The second parameter is L for query expansion models. L is the number of related entities we will use for query expansion. Figure 7b presents the performance of $QE^{R_{ex}}$. We observer that when L is larger than 2, the performance is insensitive to it. Using only two related entities yields the optimized performance. The observations on other models are similar.

Another parameter is λ in Eq. (8), where it controls the influence of the entity-centric expansion model (i.e., θ_{ER}). Figure 7c illustrates the performance of $QE^{R_{ex}}$. When λ is set to 0, we use original queries. And when λ is set to 1, we use only the terms from expansion models. It is not surprising that the performance decreases as the value of λ is close to 1, since the expanded queries are "drifted" away from the original query intent. Setting λ to 0.5 often leads to reasonable performance, which means that both original query model and the expansion model are equally important. The observations on other models are similar.

The last parameter is γ in Eq. (15), where it balances the weight of expanded query models $\theta_{ER}^{R_{ex}}$ and $\theta_{ER}^{R_{in}}$. We report the performance of $QE^{R_{ex}+R_{in}}$ on the 29 queries which qualify for internal relation expansion in Fig. 7d. We observe optimal performance can be reached when γ is less than 0.4 and $\theta_{ER}^{R_{in}}$ is favored over $\theta_{ER}^{R_{ex}}$, implying that internal relation contributes more than external relation. It suggests that if a query qualifies both external and internal relation expansion, the internal relation expansion should be favored more.

7 Experiments in general search domain

To examine how our methods would perform beyond enterprise search and longer queries, we also evaluate the proposed methods in the general search domain using a data set constructed based on a standard TREC collection.

7.1 Data collection

- The unstructured data consist of 528,155 documents (1,904MB text) from TREC disks 4&5 without the Congressional Record. This data collections is used in TREC 2004 Robust Track (Voorhees and Harman 2005).
- The structured data comes from the English version of DBpedia. It has a wide coverage of entities on the Web (i.e., 3.77 million "things" with 400 million "facts"), which is the best resource that we can find to cover entities from the general domain.

We use the official query set which consists of all the 250 topics (i.e., 301–450 & 601–700) used in TREC 2004 Robust Track. For each topic, we use only title field to construct a query because we want to evaluate the effectiveness of our methods on short keyword queries, which are commonly used in Web search. The data set is essentially the data set used in TREC 2004 Robust Track extended with DBpedia (Auer et al. 2007), and we denote it as **robust04**.

7.2 Experiments setup

Since this data set is not an enterprise collection, we use slightly different strategies in the entity identification step. The number of entities in DBpedia is huge (nearly 3.77 million), so the computational cost of estimating the relevance scores between the query entity and each of the entities from DBpedia could be very high. Thus, our candidate entity set only includes neighboring entities which have either incoming or outgoing links to the query entities on the RDF graph. To further reduce the computational cost, we only consider 1 to 1 mapping between an entity mention in the document and the candidate entity in the DBpedia graph. Because of the lack of training data, we did not use CRFs to do the mapping. Instead, we use exact matching.

After identifying entities, we then follow the same strategy to rank entities and do query expansion. Specifically, we evaluate the effectiveness of the following two methods, i.e., QE^{NAME}_{TEXT} and $QE^{R_{ex}}$. QE^{NAME}_{TEXT} is chosen over the other two entity name based expansion methods because it consistently performs better on the enterprise search collections. And $QE^{R_{ex}}$ is selected because the queries are keyword queries and most of them contain only one query entity. We also report the performance for the three baseline methods: *NoFB* (i.e., KL-divergence function with Dirichlet smoothing Zhai and Lafferty 2001) and *ModFB* (i.e., model-based feedback Zhai and Lafferty 2001), *RelFB* (i.e., relevance model Lavrenko and Croft 2001). The smoothing parameter μ is set to 1,000 in all experiments based on the optimized setting for *NoFB* (tuned from 500 to 5,000). All results are reported in MAP (Mean Average Precision).

7.3 Performance comparison over all the queries

Table 9 summarizes the performance of different models under optimized parameter settings and fivefold cross-validation. It is clear that QE^{NAME}_{TEXT} is more effective and robust than two state-of-the-art feedback methods including *ModFB* and *RelFB*. The optimized parameter settings for *ModFB* are to select top 20 feedback documents and top 100 terms for expansion, set the weight for feedback model $\alpha = 0.75$ and weight for collection language model $\lambda = 0.7$. Those for *RelFB* are to select top 30 feedback documents and top 25 terms for expansion, set the smoothing parameter $\lambda = 0.1$. Those for QE^{NAME}_{TEXT} are to select top 13 entities for expansion and set the weight for feedback model $\lambda = 0.5$. Those for $QE^{R_{ex}}$ are to select top 9 entities for expansion and set the weight for feedback model $\lambda = 0.9$.

We notice that $QE^{R_{ex}}$ is not as effective as QE^{NAME}_{TEXT}, which is inconsistent with our observation in the enterprise collection. This is because documents in **robust04** are much longer than those in the enterprise collections and may introduce more noise, making the quality of estimated entity relation lower. Therefore, entity name based expansion seems to be a better choice on ad hoc retrieval collections because of its lower computational cost and comparable effectiveness.

Our proposed models can be considered as a global expansion method (Xu and Croft 1996), which extracts expansion terms from documents across the whole collection. It

Table 9 Performance comparison on **robust04**

Models	Optimized parameter	Fivefold cross-validation
NoFB	0.2516	0.2516
ModFB	$0.2747^{\triangle}$	$0.2789^{\triangle}$
RelFB	$0.2823^{\triangle}$	$0.2823^{\triangle}$
QE^{NAME}_{TEXT}	$\mathbf{0.2878}^{\triangle\blacktriangle}$	$\mathbf{0.2878}^{\triangle\blacktriangle\star}$
$QE^{R_{ex}}$	$\mathbf{0.2871}^{\triangle}$	$\mathbf{0.2860}^{\triangle}$

$^{\triangle}$, $^{\blacktriangle}$ and * denote improvements over *NoFB*, *ModFB* and *RelFB* are statistically significant at 0.05 level based on Wilcoxon signed-rank test, respectively would be interesting to see how it would perform when used as a local expansion method, i.e., selecting expansion terms from top *K* documents of the initial retrieval. By limiting to the top 1,000 documents of *NoFB* for expansion term extraction, QE_{TEXT}^{NAME} and $QE^{R_{ex}}$ yield to 0.2865 and 0.2868 under optimized parameter settings, respectively. They are pretty close to the performance of corresponding global approaches reported in Table 9 and same significant improvements can be observed, implying our proposed models are robust with regard to expansion term extraction both locally and globally.

7.4 Performance comparison over only queries with related entities

Our proposed methods could only change the retrieval performance when a query has related entities. Among all the 250 queries in the **robust04** collection, 20 of them do not have valid related entities (i.e., entities with non-zero relevance score), which means that they will certainly not be able to benefit from our approaches and the performance of these queries would be the same as using *NoFB*.

To more accurately evaluate the effectiveness of our proposed methods, we conduct experiments over the 230 queries in which our methods can change the performance (either positively or negatively). The performance comparison are shown in Table 10. It is interesting to see that $QE^{R_{ex}}$ now outperforms three baseline methods significantly (i.e., *NoFB*, *ModFB* and *RelFB*), demonstrating the effectiveness of our approach.

7.5 Robustness

We conduct the similar analysis as in Sect. 6.3.4 to examine the robustness of our models. The histogram of queries in Fig. 8 demonstrates that $QE^{R_{ex}}$ and QE_{TEXT}^{NAME} show superior robustness over *RelFB*. More specifically, QE_{TEXT}^{NAME} improves 160 queries and hurts 68, $QE^{R_{ex}}$ improves 158 queries and hurts 71, whereas *RelFB* improves 135 queries and hurts 114. In addition, when $QE^{R_{ex}}$ and QE_{TEXT}^{NAME} hurt the performance, the decreases are much less than that of *RelFB*, confirming that our entity centric models are better choices for difficult queries.

7.5.1 Result analysis on expansion terms

We further conduct analyses on the expansion terms from these models, and observe that our models can extract more high quality terms which would potentially improve the performance. We list the top 5 weighted terms from different models for topic #362 "human smuggling" in Table 11. Terms in **bold** font are potentially helpful to improve the

Table 10 Performance comparison on **robust04** (230 queries with valid related entities)

Models	Optimized parameter	Fivefold cross-validation
NoFB	0.2515	0.2515
ModFB	$0.2793^{\triangle}$	$0.2732^{\triangle}$
RelFB	$0.2819^{\triangle}$	$0.2819^{\triangle}$
QE_{TEXT}^{NAME}	$\mathbf{0.2909}^{\triangle\blacktriangle}$	$\mathbf{0.2909}^{\triangle\blacktriangle}$
$QE^{R_{ex}}$	$\mathbf{0.2902}^{\triangle\blacktriangle\bigstar}$	$\mathbf{0.2893}^{\triangle\blacktriangle\bigstar}$

$^{\triangle}$, $^{\blacktriangle}$ and $^{\bigstar}$ denote improvements over *NoFB*, *ModFB* and *RelFB* are statistically significant at 0.05 level based on Wilcoxon signed-rank test, respectively

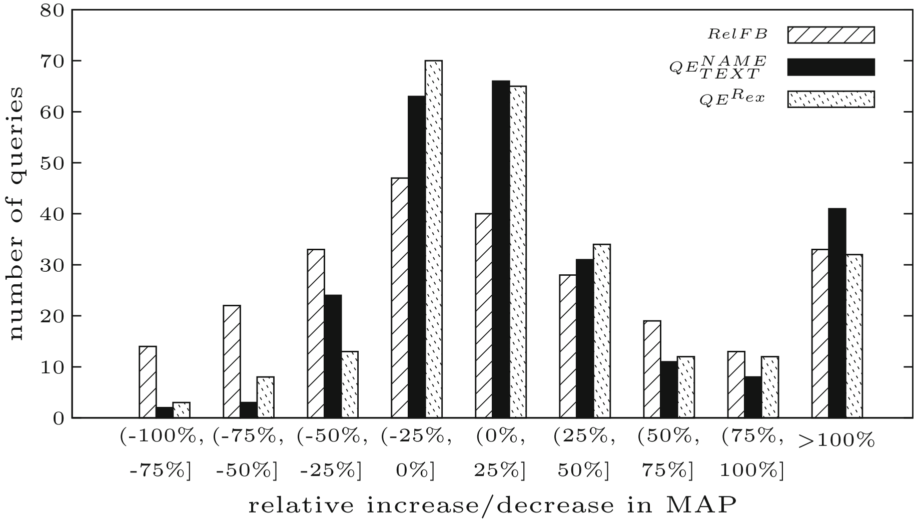


Fig. 8 Histogram of 250 queries when applied with *RelFB*, QE^{NAME}_{TEXT} and $QE^{R_{ex}}$ compared with *NoFB* on **robust04**

Table 11 Top 5 weighted expansion terms for topic #362 "human smuggling"

Models	Expansion terms
ModFB	case, **illegal, border**, chinese, criminal
RelFB	**illegal**, chinese, office, state, **country**
QE^{NAME}_{TEXT}	**illegal**, immigrate, **entry**, trafficking, **organize**
$QE^{R_{ex}}$	**illegal, police, people**, chinese, emigrate

Terms denoted in **bold** font are potentially helpful to improve performance

performance in the sense that "human smuggling", also called "**people** smuggling", is defined as the **organized** crime of **illegal entry** of **people** across international **border**.

Since these methods select different useful expansion terms, it would be interesting to see whether combining them could further improve the performance. In particular, we try to combine *ModFB* with our best entity centric approaches (i.e., QE^{NAME}_{TEXT} and $QE^{R_{ex}}$) through linear interpolation and denote them as *CombEnt*:

$$p(w|\theta_{ER}) = \gamma p(w|\theta_{\mathcal{F}}) + (1-\gamma)p(w|\theta^{NAME}_{ER})$$

and *CombRel*:

$$p(w|\theta_{ER}) = \gamma p(w|\theta_{\mathcal{F}}) + (1-\gamma)p(w|\theta^{R_{ex}}_{ER})$$

respectively. Results are presented in Table 12. Clearly combining *ModFB* with our entity centric approaches yields even better results, as the improvement of *CombEnt* over QE^{NAME}_{TEXT} and the improvement of *CombRel* over $QE^{R_{ex}}$ is statistically significant, respectively. The results indicate that our entity centric approaches are complementary to model-based feedback since they can extract different sets of useful expansion terms. Moreover,

Table 12 Performance comparison on **robust04** (250 queries)

Models	Optimized parameter	Fivefold cross-validation
NoFB	0.2516	0.2516
ModFB	$0.2747^{\triangle}$	$0.2789^{\triangle}$
RelFB	$0.2823^{\triangle}$	$0.2823^{\triangle}$
QE_{TEXT}^{NAME}	$0.2878^{\triangle\blacktriangle}$	$0.2878^{\triangle\blacktriangle\bigstar}$
$QE^{R_{ex}}$	$0.2871^{\triangle}$	$0.2860^{\triangle}$
CombEnt	**$0.2907^{\triangle\blacktriangle\alpha}$**	**$0.2898^{\triangle\blacktriangle\alpha}$**
CombRel	**$0.2917^{\triangle\blacktriangle\bigstar\beta}$**	**$0.2908^{\triangle\blacktriangle\bigstar\beta}$**

$^{\triangle}, ^{\blacktriangle}, ^{\bigstar}, ^{\alpha}$ and $^{\beta}$ denote improvements over *NoFB*, *ModFB*, *RelFB*, QE_{TEXT}^{NAME} and $QE^{R_{ex}}$ are statistically significant at 0.05 level based on Wilcoxon signed-rank test, respectively

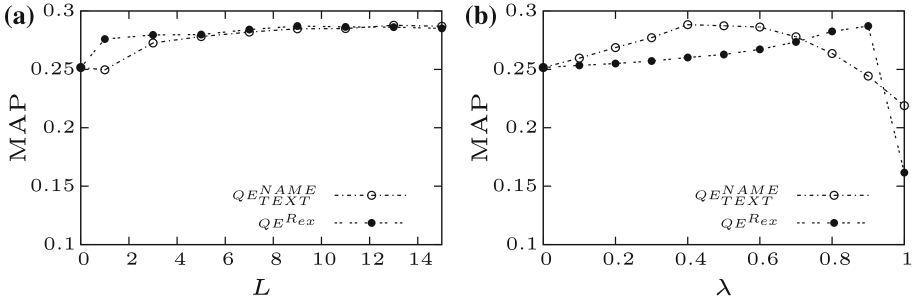


Fig. 9 Parameter sensitivity on **robust04**

the results of *CombEnt* and *CombRel* under fivefold cross-validation still outperform QE_{TEXT}^{NAME} and $QE^{R_{ex}}$, demonstrating the robustness of the combination based approaches.

7.6 Parameter sensitivity

We also report the parameter sensitivity of L (the number of related entities used for query expansion) and λ (the weight of query expansion model θ_{ER}) in Eq. 8) in Fig. 9. The observations are similar to those discussed in Sect. 6.4 More specifically, performance increases slightly with L when L is less than 10, and optimized performance is reached when L is larger than 10. Optimized performance is reached when λ is set between 0.5 and 0.9.

8 Conclusion and future work

In this paper we study the problem of improving enterprise search quality using related entities to do query expansion. In particular, we propose a domain specific entity identification method based on CRF, a general ranking strategy that can find related entities based on different entity relations from both unstructured and structured data, and an entity-centric query expansion method that can utilize related entities and their relations to estimate a new query model. We then conduct experiments over two enterprise data sets to exam the effectiveness of both finding related entity and entity based query expansion methods. Experimental results over both enterprise collections and a standard TREC collection demonstrate that our proposed are more effective than state-of-the-art feedback models for both long natural language like queries and short keyword queries. Moreover, our methods are more robust than existing methods in terms of the risk minimization.

There are many interesting future research directions. First, it would be interesting to leverage relation extraction methods and utilize other types of relations extracted from unstructured information to further improve the performance. Second, we plan to study alternative ways of combining different types of relations. Third, we plan to study how to utilize the related entities to aggregate search results. Finally, it would be interesting to evaluate the effectiveness of our methods in other search domains.

Acknowledgments This material is based upon work supported by the HP Labs Innovation Research Program. We thank reviewers for their useful comments.

References

Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., & Ives, Z. (2007). DBpedia: A nucleus for a web of open data. In Aberer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., Cudré-Mauroux, P. (Eds.), *The semantic web, volume 4825 of lecture notes in computer science* (pp. 722–735). Berlin: Springer.

Bailey, P., Craswell, N., de Vries, A. P., & Soboroff, I. (2007). Overview of the TREC 2007 enterprise track. In: *Proceedings of TREC'07.*

Bailey, P., Hawking, D., & Matson, B. (2006). Secure search in enterprise webs: Tradeoffs in efficient implementation for document level security. In *CIKM* (pp. 493–502).

Balog, K. (2007). People search in the enterprise. In *SIGIR* (pp. 916–916).

Balog, K., Azzopardi, L., & de Rijke, M. (2006). Formal models for expert finding in enterprise corpora. In *SIGIR* (pp. 43–50).

Balog, K., & de Rijke, M. (2008). Non-local evidence for expert finding. In *CIKM* (pp. 489–498).

Balog, K., de Vries, A. P., Serdyukov, P., Thomas, P., & Westerveld, T. (2010). Overview of the TREC 2009 entity track. In *Proceedings of TREC.*

Balog, K., Serdyukov, P., & de Vries, A. P. (2011). Overview of the TREC 2010 entity track. In *Proceedings of TREC.*

Balog, K., Soboroff, I., Thomas, P., Bailey, P., Craswell, N., & de Vries, A. P. (2008) Overview of the TREC 2008 enterprise track. In *Proceedings of TREC'08.*

Bendersky, M., & Croft, W. B. (2012). Modeling higher-order term dependencies in information retrieval using query hypergraphs. In *SIGIR* (pp. 941–950).

Bendersky, M., Metzler, D., & Croft, W. B. (2010). Learning concept importance using a weighted dependence model. In *Proceedings of the third ACM international conference on web search and data mining, WSDM '10* (pp. 31–40).

Bendersky, M., Metzler, D., & Croft, W. B. (2011). Parameterized concept weighting in verbose queries. In *SIGIR* (pp. 605–614).

Brunnert, J., Alonso, O., & Riehle, D. (2007). Enterprise people and skill discovery using tolerant retrieval and visualization. In *ECIR* (pp. 674–677).

Cao, G., Nie, J.-Y., Gao, J., & Robertson, S. (2008). Selecting good expansion terms for pseudo-relevance feedback. In *SIGIR* (pp. 243–250).

Carlson, A., Betteridge, J., Kisiel, B., Settles, B., H. E. R. Jr., & Mitchell T. M. (2010). Toward an architecture for never-ending language learning. In *AAAI.*

Coffman, J., & Weaver, A. (2013). An empirical performance evaluation of relational keyword search techniques. *Knowledge and Data Engineering, IEEE Transactions on PP*(99), pp. 1–1.

Cohen, W. W., Ravikumar, P., & Fienberg, S. E. (2003). A comparison of string distance metrics for name-matching tasks. In *IJCAI* (pp. 73–78).

Craswell, N., de Vries, A. P., & Soboroff, I. (2005). Overview of the TREC 2005 enterprise track. In *Proceedings of TREC'05.*

Şah, M., & Wade, V. (2010). Automatic metadata extraction from multilingual enterprise content. In *CIKM*, (pp. 1665–1668).

Demartini, G., de Vries, A., Iofciu, T., & Zhu, J. (2009). Overview of the INEX 2008 entity ranking track. In *Focused Retrieval and Evaluation* (pp. 243–252).

Demartini, G., Iofciu, T., & de Vries, A. (2010). Overview of the INEX 2009 entity ranking track. In *Focused Retrieval and Evaluation* (pp. 254–264).

Doan, A., Ramakrishnan, L. G. R., & Vaithyanathan, S. (2009). Introduction to the special issue on managing information extraction. *SIGMOD Record, 37*(4).

Fang, H., & Zhai, C. (2006). Semantic term matching in axiomatic approaches to information retrieval. In *SIGIR* (pp. 115–122).

Feldman, S., & Sherman, C. (2003). The high cost of not finding information. In *Technical Report No. 29127*. IDC.

Freund, L., & Toms, E. G. (2006). Enterprise search behaviour of software engineers. In *SIGIR* (pp. 645–646).

Garcia-Molina, H., Ullman, J., & Widom, J. (2008). *Database systems: the complete book*. Upper Saddle River, NJ: Prentice-Hall.

Hawking, D. (2004). Challenges in enterprise search. In *Proceedings of ADC'04* (pp. 15–24).

Hearst, M. A. (2011). 'Natural' search user interfaces. *Communications of the ACM 54*(11), 60–67.

Kolla, M., & Vechtomova, O. (2007). Retrieval of discussions from enterprise mailing lists. In *SIGIR* (pp. 881–882).

Lafferty, J., & Zhai, C. (2001). Document language models, query models, and risk minimization for information retrieval. In *SIGIR* (pp. 111–119).

Lafferty, J. D., McCallum, A., & Pereira, F. C. N. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML* (pp. 282–289).

Lavrenko, V., & Croft, W. B. (2001). Relevance-based language models. In *SIGIR* (pp. 120–127).

Lin, T., Pantel, P., Gamon, M., Kannan, A., & Fuxman, A. (2012). Active objects: Actions for entity-centric search. In *WWW* (pp. 589–598).

Liu, X., Fang, H., Yao, C.-L., & Wang, M. (2011). Finding relevant information of certain types from enterprise data. In *CIKM* (pp. 47–56).

Lv, Y., & Zhai, C. (2009). A comparative study of methods for estimating query language models with pseudo feedback. In *SIGIR* (pp. 1895–1898).

Lv, Y., & Zhai, C. (2010). Positional relevance model for pseudo-relevance feedback. In *SIGIR* (pp. 579–586).

Macdonald, C., & Ounis, I. (2006). Combining fields in known-item email search. In *SIGIR* (pp. 675–676).

Metzler, D., & Croft, W. B. (2005). A Markov random field model for term dependencies. In *SIGIR* (pp. 472–479).

Metzler, D., Croft, & W. B. (2007). Latent concept expansion using Markov random fields. In *SIGIR* (pp. 311–318).

Mihalcea, R., & Csomai, A. (2007). Wikify! Linking documents to encyclopedic knowledge. In *Proceedings of CIKM* (pp. 233–242).

Miller, D. R. H., Leek, T., & Schwartz, R. M. (1999). A hidden Markov model information retrieval system. In *SIGIR* (pp. 214–221).

Ponte, J. M., & Croft, W. B. (1998.) A language modeling approach to information retrieval. In *SIGIR* (pp. 275–281).

Rizzolo, N., & Roth, D. (2010). Learning based Java for rapid development of NLP systems. In *LREC, 5*.

Rocchio, J. (1971). Relevance feedback in information retrieval. In: Salton G. (Eds.) *The SMART retrieval system: Experiments in automatic document processing, Prentice-Hall Series in Automatic Computation, chapter 14* (pp. 313–323). Englewood Cliffs, NJ: Prentice-Hall.

Sarawagi, S. (2008). Information extraction. *Foundations and Trends in Databases 1*(3), 261–377.

Serdyukov, P., Rode, H., & Hiemstra, D. (2008). Modeling multi-step relevance propagation for expert finding. In *CIKM* (pp. 1133–1142).

Shen, W., Wang, J., Luo, P., Wang, M. (2012). LINDEN: Linking named entities with knowledge base via semantic knowledge. In *Proceedings of the 21st international conference on world wide web, WWW '12* (pp. 449–458).

Soboroff, I., de Vries, A. P., & Craswell, N. (2006). Overview of the TREC 2006 enterprise track. In *Proceedings of TREC'06*.

Suchanek, F. M., Kasneci, G., & Weikum, G. (2007). YAGO: A core of semantic knowledge unifying WordNet and wikipedia. In *WWW* (pp. 697–706).

Tan, B., Velivelli, A., Fang, H., & Zhai, C. (2007). Term feedback for information retrieval with language models. In *SIGIR* (pp. 263–270).

Tao, T., & Zhai, C. (2006). Regularized estimation of mixture models for robust pseudo-relevance feedback. In *SIGIR* (pp. 162–169).

Voorhees, E. M., & Harman, D. K. (2005). *TREC: Experiment and Evaluation in Information Retrieval*. Cambridge: The MIT Press.

Wang, L., Bennett, P. N., & Collins-Thompson, K. (2012). Robust ranking models via risk-sensitive optimization. In *SIGIR* (pp. 761–770).

Weerkamp, W., Balog, K., & de Rijke, M. (2012). Exploiting external collections for query expansion. *ACM Transactions on the Web, 6*(4).

Weerkamp, W., Balog, K., & Meij, E. (2009). A generative language modeling approach for ranking entities. In *Focused Retrieval and Evaluation* (pp. 292–299).

Xu, J., & Croft, W. B. (1996). Query expansion using local and global document analysis. In *SIGIR* (pp. 4–11).

Zelenko, D., Aone, C., & Richardella, A. (2003). Kernel methods for relation extraction. *The Journal of Machine Learning Research 3*, 1083–1106.

Zhai, C., & Lafferty, J. (2001). A study of smoothing methods for language models applied to Ad Hoc information retrieval. In *SIGIR* (pp. 334–342).

Zhai, C., & Lafferty, J. (2001). Model-based feedback in the language modeling approach to information retrieval. In *CIKM*.

Zhu, J., Nie, Z., Liu, X., Zhang, B., & Wen, J.-R. (2009). StatSnowball: A statistical approach to extracting entity relationships. In *WWW* (pp. 101–110).