Chapter 8
Leveraging Entities in Document Retrieval

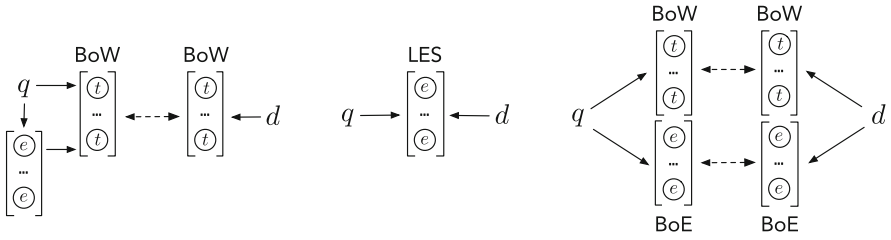
This chapter focuses on the classic IR task of *ad hoc document retrieval* and discusses how entities may be leveraged to improve retrieval performance. At their core, all document retrieval methods compare query and document representations. Traditionally, these representations are based on terms (words). Entities facilitate a semantic understanding of both the user's information need, as expressed by the keyword query, and of the document's content. Entities thus may be used to improve query and/or document representations. As a first step of that process, entities that are related to the query need to be identified, thereby establishing a mapping between queries and entities. As we shall explain in Sect. 8.1, one may go beyond considering only those entities that are explicitly mentioned in the query. We next present three different families of approaches, which are illustrated in Fig. 8.1.

- *Expansion-based* methods utilize entities as a source of expansion terms to enrich the representation of the query (Sect. 8.2).
- *Projection-based* methods treat entities as a latent layer, while leaving the original document/query representations intact (Sect. 8.3).
- *Entity-based* methods consider explicitly the entities that are recognized in documents, as first-class citizens, and embrace entity-based representations in "duet" with traditional term-based representations in the retrieval model (Sect. 8.4).

This particular order corresponds to the temporal evolution of research in this area, where the tendency toward more and more explicit entity semantics is clearly reflected. Throughout this chapter, we shall assume that both queries and documents have been annotated with entities, using entity linking techniques we have discussed before (see Chap. 5 for documents and Sect. 7.3 for queries). A particular challenge involved here is how to deal with the uncertainty of these automatic annotations.

In practice, necessitated by efficiency considerations, all methods described in this chapter are implemented as re-ranking mechanisms. The details are found in Sect. 8.5. Finally, we present standard datasets and useful resources in Sect. 8.6. We refer to Table 8.1 for the notation used throughout this chapter.

K. Balog, *Entity-Oriented Search*, The Information Retrieval Series 39,
https://doi.org/10.1007/978-3-319-93935-3_8



(a) Expansion-based (b) Projection-based (c) Entity-based

Fig. 8.1 Three main ways to leverage entities for improved document retrieval. The representations are bag-of-words (BoW), bag-of-entities (BoE), and latent entity space (LES). The main difference between projection-based (**b**) and entity-based (**c**) methods is that the former treats entities as a latent layer between queries and documents, while the latter explicitly models the entities mentioned in the document and complements the traditional bag-of-words (BoW) representations with bag-of-entities (BoE) representations

Table 8.1 Notation used in this chapter

Symbol	Description
$c(t;x)$	Count (raw frequency) of term t in the x
d	Document ($d \in \mathcal{D}$)
$\mathcal{D}$	Collection of documents
$\mathcal{D}_q(k)$	Top-k ranked documents in response to query q
e	Entity ($e \in \mathcal{E}$)
$\mathcal{E}$	Entity catalog (set of all entities)
$\mathcal{E}_q(k)$	Top-k ranked entities in response to query q
$\mathcal{E}_q$	Set of query entities
l_x	Length of the description of x ($l_x = \sum_{t \in x} c(t;x)$)
q	Query ($q = \langle q_1, \ldots, q_n \rangle$)
t	Term ($t \in \mathcal{V}$)
$\mathcal{T}$	Type taxonomy
$\mathcal{T}_e$	Set of types assigned to e
$\mathcal{V}$	Vocabulary of terms

8.1 Mapping Queries to Entities

A common component that is shared by all approaches that follow later in this chapter is the mapping of queries to entities. The goal is to identify a set of entities that may be semantically related to the query.[1] We shall refer to this set $\mathcal{E}_q$ of related entities as *query entities*. Naturally, not all query entities are equally strongly related to the query. Therefore, we use the probability $P(e|q)$ to express the likelihood of

[1]Notice that this is highly related to the task of *ad hoc entity retrieval* (cf. Chap. 3), as well as to the *candidate entity ranking* and *semantic linking* subtasks in query entity annotation (cf. Sect. 7.3).

entity e being related to query q. The estimation of this probability may be based on (1) entities mentioned in the query, (2) entities retrieved directly from a knowledge base, and/or (3) entities retrieved indirectly, through pseudo-relevant documents. Let us look at these in order.

- *Entities mentioned in the query.* The presence of an entity mention in the query provides a unique opportunity for improving the understanding of the user's information need [5]. Entities can be identified and disambiguated using entity linking techniques, cf. Sect. 7.3. Let $\mathcal{E}_q$ be the set of entities that have been identified in query q. For each of the query entities $e \in \mathcal{E}_q$, we let $score_{ELQ}(e;q)$ be the associated confidence score. We note that the annotated entity mentions may overlap (i.e., we are not concerned with forming interpretation sets). For queries that are associated with a specific entity (i.e., $|\mathcal{E}_q| = 1$), it makes sense to use that entity's description as pseudo-feedback information [56]. More generally, we consider a single entity that has the highest annotation score:

$$P(e|q) = \begin{cases} 1\,, & e = \arg\max_{e \in \mathcal{E}_q} score_{ELQ}(e;q) \\ 0\,, & \text{otherwise}\,. \end{cases} \tag{8.1}$$

  Further generalization to an arbitrary number of entities can easily be done, by introducing a minimum confidence threshold on the annotations:

$$P(e|q) = \begin{cases} \frac{1}{Z}\, score_{ELQ}(e;q)\,, & score_{ELQ}(e;q) > \gamma \\ 0\,, & \text{otherwise}\,, \end{cases}$$

  where γ is a score threshold parameter, and Z is a normalization factor, such that $0 \leq P(e|q) \leq 1$. Additionally, entities relevant to those mentioned in the query may also be considered [34].

- *Entities retrieved from a knowledge base.* An alternative route may be taken by querying a knowledge base directly for relevant entities [18]. We let $score_{ER}(e;q)$ be the relevance score of entity e given q. This score may be computed using any of the entity retrieval methods introduced in Chap. 3. For pragmatic considerations, only the top-k entities are considered, denoted as $\mathcal{E}_q(k)$. $P(e|q)$ then becomes:

$$P(e|q) = \begin{cases} \frac{1}{Z}\, score_{ER}(e;q)\,, & e \in \mathcal{E}_q(k) \\ 0\,, & \text{otherwise}\,, \end{cases}$$

  where Z is a normalization coefficient.

- *Entities from pseudo-relevant documents.* The third method uses the top-ranked documents retrieved in response to the query, in the spirit of pseudo relevance feedback [40]. This corresponds to the setting of ranking entities without direct representations (cf. Sect. 3.4). Formally:

$$P(e|q) \propto \sum_{d \in \mathcal{D}_q(k)} P(e|d) P(d|q)\,, \tag{8.2}$$

where $\mathcal{D}_q(k)$ denotes the set of top-k highest scoring documents retrieved in response to query q, $P(d|q)$ corresponds to document d's relevance to the query, and $P(e|d)$ is the probability of observing the entity in d. $P(e|d)$ may be taken as a maximum-likelihood estimate:

$$P(e|d) = \frac{c(e;d)}{\sum_{e' \in d} c(e';d)},$$

where $c(e;d)$ is the number of times entity e occurs in document d. Additionally, the frequency of e across the document collection may also be taken into account (to demote entities that occur in too many documents) by adding an IDF-like component, see Eq. (3.6). One alternative to relying on maximum-likelihood estimation is presented by Meij et al. [40], who re-estimate the probability mass of the entities using parsimonious language models.

We note that entity relevance may be estimated selectively or jointly using the above methods, depending on the type of the query. For example, Xu et al. [56] employ Eq. (8.1) for queries that can be associated with a single entity; for other queries, the top-k ranked documents are considered, i.e., Eq. (8.2) is used.

8.2 Leveraging Entities for Query Expansion

Keyword queries are typically too short to describe the underlying information need accurately. Query expansion is one of the classical techniques used in document retrieval, dating all the way back to the 1970s [43]. The idea is to supplement the keyword query with additional terms, thereby having a more elaborate expression of the underlying information need. These additional terms may be extracted from documents that are deemed relevant. In most cases, however, there is no explicit feedback from the user as to which documents are relevant and which are not. Instead, one may "blindly" assume that the top-ranked documents are relevant, and extract expansion terms from these. This technique is known as *pseudo* (or *blind*) *relevance feedback* and has been thoroughly investigated in the past. In general, pseudo relevance feedback helps more queries than it hurts [39]. Clearly, it can only be effective when the initial set of retrieved documents is good, otherwise it merely introduces noise. Prior work has demonstrated the benefits of exploiting external collections for query expansion [2, 20, 49]. In this section, we will leverage a knowledge base as an external resource, and utilize entities for query expansion. This can bring in external semantic signals that may not be available within feedback documents.

First, in Sect. 8.2.1, we describe how traditional document-based feedback works. The aim of that section is to show how an expanded query model $\hat{\theta}_q$ can be constructed and subsequently used for retrieval. Next, in Sect. 8.2.2, we present a general framework for performing entity-centric query expansion, i.e., estimating $\hat{\theta}_q$ with the help of entities. A core component of this framework is *term selection*, which may be approached using either unsupervised or supervised methods. These are discussed in Sects. 8.2.3 and 8.2.4, respectively.

8.2.1 Document-Based Query Expansion

To give an idea of how traditional (term-based) pseudo relevance feedback works, we present one of the most popular approaches, the *relevance model* by Lavrenko and Croft [33]. This method assumes that there exists some underlying relevance model R, which generates both the query and the relevant documents. Then, based on the observed query q (which is a sample from R), we attempt to learn the parameters of R. The probability of drawing a term t from R is approximated with the probability of observing that term, given the query:

$$P(t|R) \approx P(t|q_1, \ldots, q_n) \; .$$

Lavrenko and Croft [33] present two methods for estimating this conditional probability. The better performing of the two, referred to as *RM1*, assumes that the terms in the query and in relevant documents are sampled identically and independently from the relevance model (i.i.d. sampling):

$$P(t|R) \propto \sum_{d \in \mathcal{D}_q(m)} P(d) P(t|\theta_d) \prod_{i=1}^{n} P(q_i|\theta_d) \; , \tag{8.3}$$

where $\mathcal{D}_q(m)$ is the set of top-m highest ranked documents for the original query, according to some retrieval model (commonly query likelihood, i.e., $P(q|\theta_d)$).[2] These are used as evidence for estimating the relevance model, with m typically set between 10 and 50 [1, 17, 37, 56]. The prior document probability, $P(d)$, is usually assumed to be uniform. The term probabilities $P(t|\theta_d)$ and $P(q_i|\theta_d)$ are smoothed term probabilities from the document's language model (cf. Sect. 3.3.1.1).

The number of expansion terms has a direct impact on retrieval efficiency. Therefore, in practice, only the top-k expansion terms with the highest probability are used for expansion, with k typically ranging between 10 and 50 (see, e.g., [1, 5, 17]). Thus, the top-k highest scoring terms according to Eq. (8.3) are taken to form the expanded query model $\hat{\theta}_q$ (with the probabilities renormalized such that $\sum_t P(t|\hat{\theta}_q) = 1$).

To avoid the query shifting too far away from the user's original intent (an issue known as *topic drift*), it is common to define the final query model θ_q as a linear combination of the maximum likelihood and expanded query models [59]:

$$P(t|\theta_q) = (1 - \lambda) \frac{c(t;q)}{l_q} + \lambda P(t|\hat{\theta}_q) \; , \tag{8.4}$$

[2] We use m to denote the number of feedback documents, as the variable k will be used for the number of feedback terms.

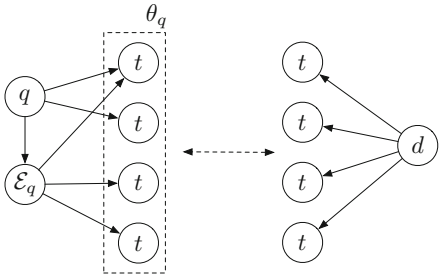


Fig. 8.2 Entity-based query expansion. Query entities ($\mathcal{E}_q$) are utilized to add and re-weigh terms in the original query q, resulting in an expanded query model θ_q

where $c(t;q)$ is the number of times t occurs in q, l_q is the total number of terms in the query, and λ is a mixture parameter. This parameter controls the influence of the expanded query model and is typically in the range $[0.4, 0.6]$ (see, e.g., [1, 56]). The combination of RM1 with the original query in Eq. (8.4) is commonly referred to in the literature as RM3 [37]. RM3 is widely regarded as a state-of-the-art model for pseudo relevance feedback.

Finally, the combined query model θ_q is used in a second retrieval round to obtain the final document ranking. For example, using the query likelihood retrieval model (a.k.a. the standard language modeling approach) the scoring of documents is done according to:[3]

$$\log P(q|\theta_d) = \sum_{t \in q} P(t|\theta_q) \log P(t|\theta_d) \,,$$

where $P(t|\theta_d)$ is the probability of term t in the (smoothed) language model of document d. Note that any other retrieval model can be used for obtaining the final document ranking by using θ_q as a (weighted) query.

8.2.2 *Entity-Centric Query Expansion*

Given that our focus is on entities, the question we ask is this: Can we use entities, instead of documents, for estimating an expanded query model? The idea of entity-centric query expansion is illustrated in Fig. 8.2, where the expanded query model θ_q is estimated by utilizing the set of query entities $\mathcal{E}_q$.

As a general framework, we follow the method proposed by Meij et al. [40], where query expansion is formulated as a double translation process: first, translating the query to a set of relevant entities, then considering the vocabulary of terms associated with those entities as possible expansion terms to estimate the

[3] This formula can be derived by replacing the query term count $c(t;q)$ in Eq. (3.8) with the probability of the term given the query language model, $P(t|\theta_q)$. Note that this scoring is rank-equivalent to measuring the Kullback–Leibler divergence between the document and the query term distributions ($KL(\theta_q||\theta_d)$) [1].

expanded query model. Formally:

$$P(t|\hat{\theta}_q) \propto \sum_{e \in \mathcal{E}_q} P(t|e,q) P(e|q) \ . \tag{8.5}$$

Most existing approaches can be instantiated into Eq. (8.5). The first component, $P(t|e,q)$, expresses how strongly term t is associated with entity e given q. Meij et al. [40] further impose a conditional independence assumption between the term and the query. That is, once an entity is selected for a query, the probability of the expansion term depends only on that entity: $P(t|e,q) \cong P(t|e)$. This distribution governs how terms are sampled from the description of e. We shall refer to it as *term selection*. The second component, $P(e|q)$, identifies entities to be used for query expansion and corresponds to the relevance of e given the query. This latter component we have already discussed in Sect. 8.1. Therefore, we shall now focus on the estimation of $P(t|e,q)$, considering both unsupervised and supervised methods.

8.2.3 Unsupervised Term Selection

The probability distribution $P(t|e)$ is estimated by selecting expansion terms from the description or surface forms of a given query entity ($e \in \mathcal{E}_q$). (Notice that term selection here depends only on the entity, and not on the original query.)

Entity Description One of the simplest ways to perform term selection is to pick the most important terms from the entity's term-based representation, which we refer to as the entity description. Following our notation from before and assuming a single-field entity representation, we write $c(t;e)$ to denote the count (raw frequency) of term t in the description of e. Further, we introduce the shorthand notation $w(t,e)$ for the importance of term t given e. Commonly, this is estimated using the TF-IDF (here: TF-IEF) weighting scheme:

$$w(t,e) = TF(t,e) \times IEF(t) \ ,$$

where term frequency TF and inverse entity frequency IEF have been defined in Eqs. (3.1) and (3.2), respectively. Another popular choice is to use entity language models, i.e., $w(t,e) = P(t|\theta_e)$. (We refer back to Sect. 3.3.1.1 for the construction of entity language models.) Once term scores are computed, $P(t|e)$ is formed by taking the top-k terms and re-normalizing their scores:

$$P(t|e) = \begin{cases} \frac{1}{Z} \, w(t,e) \ , & t \in \mathcal{V}(k) \\ 0 \ , & \text{otherwise} \ , \end{cases}$$

where $\mathcal{V}(k)$ is the set of top-k terms with the highest score, and Z is a normalization coefficient. Xu et al. [56] further exploit the structured nature of entity descriptions, by taking a linear combination of term scores across multiple entity fields. When

the entity does not have a ready-made description available in the knowledge repository, terms may be sampled from documents that mention the entity [40]. Instead of considering entity-term co-occurrences on the document level, they may be restricted to smaller contexts, such as the sentence mentioning the entity or a fixed sized window of words around the entity mention [18]. The key difference between the above methods is how the entity description is obtained; we have covered all these variants earlier in the book, in Chap. 3.

Surface Forms Another approach is to use the various surface forms (aliases) of the query entities as expansion terms [18, 34]. The main intuition behind doing so is that by including the different ways an entity can be referred to (i.e., its "synonyms"), documents relevant to that entity can be retrieved more effectively. Name variants of an entity can be conveniently enumerated as a separate field in its description. Then, the estimation of $P(t|e)$ is performed as before, except that the term importance score is based on $c(t; f_n)$, instead of $c(t; e)$, where f_n is the field holding the name variants of e.

8.2.4 Supervised Term Selection

So far, it has been implicitly assumed that all expansion terms are useful and benefit retrieval performance, when added to the original query. This assumption was challenged by Cao et al. [8], who showed that not all expansion terms are actually valuable. Some terms are neutral (do not affect performance), while others are in fact harmful. They further demonstrated that it is difficult to tell apart good expansion terms from bad ones based solely on term distributions. One needs to incorporate additional signals to be able to select useful expansion terms. Therefore, Cao et al. [8] propose to combine multiple features using supervised learning to predict the usefulness of expansion terms. This task may be formulated as a binary classification problem (separating good expansion terms from bad ones) [8] or cast as a ranking problem (ranking expansion terms based on their predicted utility) [5]. We follow the latter approach and specifically focus on ranking expansion terms given a particular query entity e and the query q. The resulting term importance score $score(t; e, q)$ may be plugged into Eq. (8.5) as an estimate of $P(t|e, q)$. Note that the dependence of the expansion term on the original query is kept.

8.2.4.1 Features

Brandão et al. [5] present five specific feature functions for entity-based query expansion. The order in which we discuss them corresponds to their usefulness (from most to least useful).

The first two features are simple statistical measures of term frequency, which rely on fielded entity descriptions. *Term frequency* is the total number of times t occurs across the set $\mathcal{F}_e$ fields of the entity:

$$TF(t,e) = \sum_{f_e \in \mathcal{F}_e} c(t; f_e) \ .$$

The *term spread* feature measures the spread of a term across multiple fields, by counting in how many different fields the term occurs:

$$TS(t,e) = \sum_{f_e \in \mathcal{F}_e} \mathbb{1}(c(t; f_e) > 0) \ ,$$

where $\mathbb{1}()$ is a binary indicator function.

Term proximity accounts for the proximity between an expansion term and the original query terms in the description of the entity:

$$TP(t,q,e) = \sum_{i=1}^{n} \sum_{w=1}^{m} \frac{c_w(t,q_i;e)}{2^{w-1}} \ ,$$

where $c_w(t,q_i;e)$ is the total number of co-occurrences of terms t and q_i, within an unordered window size of w, in the description of e. In [5], entities are represented by their Wikipedia pages, and windows are measured in terms of sentences, up to $m = 5$.

The last two features are taxonomic, utilizing the types of entities. Let $\mathcal{E}_t$ denote the set of entities that contain the term t: $\mathcal{E}_t = \{e' \in \mathcal{E} : c(t;e') > 0\}$. Further, let $\mathcal{E}_e$ be the set of entities that share at least one type with the query entity e: $\mathcal{E}_e = \{e' \in \mathcal{E} : \mathcal{T}_e \cap \mathcal{T}_{e'} \neq \emptyset\}$. The similarity between the sets of related entities $\mathcal{E}_t$ and $\mathcal{E}_e$ may be measured using *Dice's coefficient*:

$$DC(t,e) = 2\,\frac{|\mathcal{E}_t \cap \mathcal{E}_e|}{|\mathcal{E}_t| + |\mathcal{E}_e|} \ .$$

Another option is to use *mutual information*:

$$MI(t,e) = \begin{cases} |\mathcal{E}_t \cap \mathcal{E}_e| \ \log \frac{|\mathcal{E}_t \cap \mathcal{E}_e|}{|\mathcal{E}_t| \times |\mathcal{E}_e|}, & |\mathcal{E}_t \cap \mathcal{E}_e| > 0 \\ 0, & \text{otherwise} \ . \end{cases}$$

All the above features score candidate expansion terms with respect to a given query entity. It is also possible to leverage information associated with entities in a knowledge base, without utilizing query entities directly. A specific example of such an approach is given by Xiong and Callan [52], who select expansion terms that have similar type distributions with that of the query.

The type distribution of a term is estimated according to:

$$P(y|\theta_t) = \frac{P(t|y)}{\sum_{y' \in \mathcal{T}} P(t|y')} \ ,$$

where $\mathcal{T}$ is the type taxonomy and the term probability of a type $P(t|y)$ is approximated based on the term's frequency in the descriptions of all entities with that type:

$$P(t|y) = \frac{\sum_{e \in y} c(t;e)}{\sum_{e \in y} l_e} .$$

Similarly, the type distribution of the query is estimated according to:

$$P(y|\theta_q) = \frac{P(q|y)}{\sum_{y' \in \mathcal{T}} P(q|y')} ,$$

where $P(q|y)$ is the product of the query terms' likelihood given type y:

$$P(q|y) = \prod_{q_i} P(q_i|y) .$$

Then, expansion terms are selected based on the similarity of their type distributions θ_t to the type distribution of the query θ_q, measured by negative *Jensen–Shannon divergence*:

$$score_{JSD}(t;q) = -\frac{1}{2} KL(\theta_q || \theta_{q,t}) - \frac{1}{2} KL(\theta_t || \theta_{q,t}) , \tag{8.6}$$

where

$$P(y|\theta_{q,t}) = \frac{1}{2}\big(P(y|\theta_q) + P(y|\theta_t)\big) .$$

Notice that the estimate in Eq. (8.6) depends only on the query and not on the query entities. Further note that all unsupervised term importance estimates from Sect. 8.2.3 can also be used as features in supervised term selection.

8.2.4.2 Training

To be able to apply supervised learning, target labels are required. The question is: How to measure if a term is a good expansion term? Cao et al. [8] propose to identify the ground truth labels of terms according to their direct impact on retrieval effectiveness. Formally, the *gain* attained by appending the candidate expansion term t to the original query q (denoted by the $\oplus$ operator) is measured as:

$$\delta(t) = \frac{\zeta(q \oplus t) - \zeta(q)}{\zeta(q)} ,$$

where ζ can be any standard IR evaluation measure, such as MAP or NDCG. Then, terms above a certain threshold (0.005 in [8]) may be considered as good expansion terms (target label $+1$), while the rest being bad terms (target label -1).

Instead of measuring the direct impact of terms with respect to some retrieval measure, Xiong and Callan [52] use their influence on ranking scores. We write

$\mathcal{R}^+$ and $\mathcal{R}^-$ to denote the set of relevant and irrelevant documents for query q, respectively. Further, $score(d;q)$ denotes the retrieval score of document d for q. The gain from a term over the retrieved documents is then calculated as:

$$\delta(t) = \frac{1}{|\mathcal{R}^+|} \sum_{d \in \mathcal{R}^+} \Big(score(d; q \oplus t) - score(d;q) \Big) - \frac{1}{|\mathcal{R}^-|} \sum_{d \in \mathcal{R}^-} \Big(score(d; q \oplus t) - score(d;q) \Big) .$$

Xiong and Callan claim that this latter formulation "reflects an expansion term's effectiveness more directly" [52][4] and performs better experimentally.

8.3 Projection-Based Methods

Traditional keyword-based IR models have an inherent limitation of not being able to retrieve (relevant) documents that have no explicit term matches with the query. While query expansion can remedy this to some extent, the limitation still remains. Concept-based retrieval methods attempt to tackle this challenge by relying on auxiliary structures to obtain semantic representations of queries and documents in a higher-level concept space. Such structures include controlled vocabularies (dictionaries and thesauri) [28, 47], ontologies [9], and entities from a knowledge repository [23]. Our interest here is in the latter group.

The overall idea is "to construct a high-dimensional latent entity space, in which each dimension corresponds to one entity, and map both queries and documents to the latent space accordingly" [35]. The relevance between a query and a document is then estimated based on their projections to this latent entity space. This approach allows to uncover hidden (latent) semantic relationships between queries and documents. See Fig. 8.1b for an illustration.

This idea is related to that of topic modeling, as developed in *latent semantic indexing* [19] and *latent Dirichlet allocation* [3]. While topic models can now be computed on web-scale [32], their utility to improve retrieval effectiveness is limited. For example, Yi and Allan [57] have demonstrated that relevance models (cf. Sect. 8.2.1) consistently outperform more elaborate topic modeling methods. Latent entity representations, on the other hand, may be obtained at a relatively low cost, are easy to interpret, and have been clearly shown to improve retrieval effectiveness.

In this section, we present three specific approaches for ranking documents using latent entity representations.

[4] It is more direct in the sense that changes in a given evaluation metric only happen if a given expansion term manages to affect the ranking scores to an extent that documents exchange positions.

8.3.1 Explicit Semantic Analysis

Explicit semantic analysis (ESA) is an influential concept-based retrieval method by Gabrilovich and Markovitch [26], where "the semantics of a given word are described by a vector storing the word's association strengths to Wikipedia-derived concepts" [23]. Unlike in latent semantic analysis (LSA) [21], the use of a knowledge repository gives meaningful interpretation to each element (concept) in the vector representation, hence the name "explicit." Work on ESA has primarily focused on using Wikipedia as the underlying knowledge repository [22, 23, 25, 26]. Nevertheless, it may be used with any other knowledge repository, provided that it has a sufficient coverage of concepts and concepts have textual descriptions associated with them. For terminological consistency, we will continue to use the term "entity" instead of "concept," when referring to entries of a knowledge repository,[5] but follow the terminology "concept vector" and "concept space" from the original work.

8.3.1.1 ESA Concept-Based Indexing

The semantic representation of a given term t is a *concept vector* of length $|\mathcal{E}|$:

$$\mathbf{t} = \left\langle w(e_1,t), \ldots, w(e_{|\mathcal{E}|},t) \right\rangle ,$$

where each element of the vector corresponds to an entity in the knowledge repository and its value quantifies the strength of the association between term t and the given entity. For a given term-entity pair, $w(e,t)$ is computed by taking the TF-IDF weight of t in the description of e (in ESA, the Wikipedia article of e). Further, cosine normalization is applied to disregard differences in entity representation length:

$$w(e,t) = \frac{TFIDF(t,e)}{\sqrt{\sum_{t' \in \mathcal{V}} TFIDF(t',e)^2}} .$$

The semantic representation of a given piece of text (bag of terms) is computed by taking the centroid of the individual terms' concept vectors. Formally, the concept vector corresponding to input text z is given by $\mathbf{z} = \langle w(e_1,z), \ldots, w(e_{|\mathcal{E}|},z) \rangle$. Each element of this vector represents the relatedness of the corresponding entity to the input text. The value of the jth vector element is calculated as:

$$w(e_j,z) = \frac{1}{l_z} \sum_{t \in z} c(t;z)\, w(e_j,t) ,$$

where l_z is the length of z and $c(t;z)$ is the number of times term t appears in z. See Fig. 8.3 for an illustration.

[5] We refer back to Sect. 1.1.1 for the difference between concepts and entities.

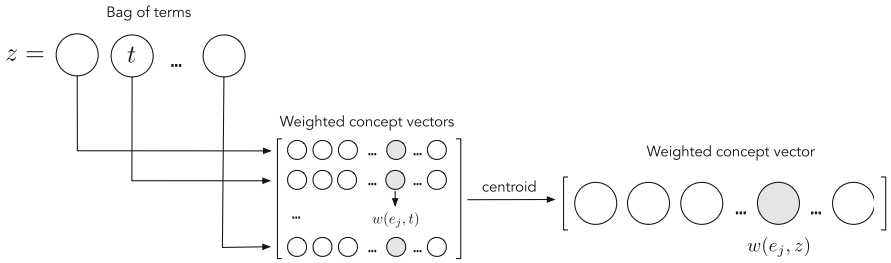


Fig. 8.3 Semantic representation of a piece of text (z) using explicit semantic analysis (ESA) [26]

Given that these concept-based vectors are sparse, with most weights being zero, they can be efficiently represented using an inverted index. These inverted index representations of concept vectors are further pruned by retaining only the top-k entities with the highest weights. In [23], this cutoff is applied to both term and text concept vectors ($\mathbf{t}$ and $\mathbf{z}$, respectively), with k set to 50. This helps to eliminate spurious and insignificant entity associations and also reduces index size.

8.3.1.2 ESA Concept-Based Retrieval

The semantic similarity between query q and document d may be computed by mapping both to the ESA concept space, and taking the cosine similarity of their concept vectors. That is, $score(q; d) = \cos(\mathbf{q}, \mathbf{d})$, where $\mathbf{q}$ and $\mathbf{d}$ denote the concept vectors corresponding to q and d, respectively.

One issue that arises here is that long documents are difficult to map to the concept space. As Egozi et al. [23] explain, "a small part of a long document might be relevant to the current query, but the semantics of this part may be underrepresented in the concepts vector for the full document." The proposed solution, motivated by prior work on passage-based retrieval [7], is to break up the document into shorter passages. In [23], passages are of fixed length and overlapping. Each passage, $s \in d$, is represented by its own concept vector, $\mathbf{s}$, and matched against the query. The final retrieval score combines the full document's similarity score with that of the best performing passage:

$$score_{ESA}(q; d) = \cos(\mathbf{q}, \mathbf{d}) + \max_{s \in d} \cos(\mathbf{q}, \mathbf{s}) \ .$$

Due to the fact that queries are short and noisy, the initially generated query concept vector needs further refinement. Egozi et al. [23] propose to utilize the idea of pseudo relevance feedback for this purpose. First, keyword-based retrieval (using q) is performed on the passage level. Then, the top-k passages are treated as pseudo-relevant (i.e., positive) examples, while the bottom-k passages are taken to be pseudo-non-relevant (i.e., negative) examples. Then, a subset of the initial query entities is selected, based on the positive and negative example passages, resulting

in a modified query $\mathbf{q}'$. Finally, documents are ranked using the refined concept-based query $\mathbf{q}'$. We refer to [23] for further details on how to select which entities to retain in the modified query concept vector.

8.3.2 Latent Entity Space Model

Liu and Fang [35] present the *latent entity space* (LES) model, which is based on a generative probabilistic framework. The document's retrieval score is taken to be a linear combination of the latent entity space score and the original query likelihood score:[6]

$$score_{LES}(q;d) = \alpha \underbrace{\sum_{e \in \mathcal{E}} P(q|e)P(e|d)}_{\text{LES score}} + (1-\alpha)\, P(q|d)\ . \tag{8.7}$$

In the absence of labeled training data, Liu and Fang [35] suggest to set interpolation parameter α to a value between 0.5 and 0.7. The latent entity space score is calculated as a linear combination over latent entities ($e \in \mathcal{E}$) and involves the estimation of two components: *query projection*, $P(q|e)$, and *document projection*, $P(e|d)$. Next, we detail the estimation of these two probabilities.

Query Projection The probability $P(q|e)$ may be interpreted as the likelihood of the query q being generated by entity e. A straightforward option is to base this estimate on the language model of the entity, θ_e. This language model may be constructed from the entity's description (cf. Sect. 3.3.1.1). The query projection probability is then computed by taking a product over the query terms (which is essentially the query likelihood score of the entity):

$$P(q|e) = \prod_{t \in q} P(t|\theta_e)^{c(t;q)}\ .$$

Another approach to estimating this probability is to leverage the set of query entities $\mathcal{E}_q$ (which we have obtained in Sect. 8.1) in a pairwise manner:

$$P(q|e) \propto \sum_{e' \in \mathcal{E}_q} sim(e,e')\ P(e'|q)\ , \tag{8.8}$$

where $sim(e,e')$ may be any symmetric pairwise similarity measure (in [35] the cosine similarity between θ_e and $\theta_{e'}$ is used; see Sect. 4.5.1 for other possibilities), and $P(e'|q)$ is the query association probability for e'.

Liu and Fang [35] find that the latter, entity-similarity-based method works better than the former, unigram-based approach. Using Eq. (8.8) implies that the summation in Eq. (8.7) can be restricted to the set of query entities $\mathcal{E}_q$ as opposed to

[6] In [35], the scores of the two components are re-normalized to make them compatible.

Table 8.2 Features used in EsdRank [51]

Group	Description
Query-entity features	
$P(e\|q)$	Query-entity association probability (cf. Sect. 8.1)
$score_{ER}(e;q)$	Entity retrieval score (relevance of e given q)
$sim(\mathcal{T}_e, \mathcal{T}_q)$	Type-based similarity between the entity ($\mathcal{T}_e$) and the query ($\mathcal{T}_q$)
$maxSim(e, \mathcal{E}_q)$	Max. pairwise similarity between e and other query entities $e' \in \mathcal{E}_q$
$avgSim(e, \mathcal{E}_q)$	Avg. pairwise similarity between e and other query entities $e' \in \mathcal{E}_q$
Entity-document features	
$sim(e, d)$	Textual similarity between e and d
$sim(\mathcal{T}_e, \mathcal{T}_d)$	Type-based similarity between the entity ($\mathcal{T}_e$) and the document ($\mathcal{T}_d$)
$numMentioned(\mathcal{E}_e^i, d)$	Number of related entities (at most i hops from e) mentioned in d
Other features	
$IDF(e)$	IDF score of the entity (based on the number of documents that are annotated with e)
$quality(d)$	Quality score of d (document length, SPAM score, PageRank, etc.)

Note that many of these features are instantiated multiple times, using different similarity methods or parameter configurations

the entire entity catalog $\mathcal{E}$, which will have a significant positive effect on efficiency. It is further shown in [35] that the quality of entity language models θ_e can have a significant impact on end-to-end retrieval performance.

Document Projection The probability $P(e|d)$ may be interpreted as the projection of document d to the latent entity space. It may be estimated using existing document retrieval models, e.g., by computing entity likelihood (i.e., the probability of e generated from the language model of d). Liu and Fang [35] estimate $P(e|d)$ based on the negative cross-entropy between the document and entity language models:

$$P(e|d) = \exp\left(-CE(\theta_e \parallel \theta_d)\right) = \exp\left(\sum_{t \in \mathcal{V}} P(t|\theta_e) \log P(t|\theta_d)\right).$$

8.3.3 EsdRank

The idea of using entities as a bridge between documents and queries may also be expressed in a discriminative learning framework. Xiong and Callan [51] introduce *EsdRank* for ranking documents, using a combination of query-entity and entity-document features. These correspond to the notions of query projection and document projection components of LES, respectively, from before. Using a discriminative learning framework, additional signals can also be incorporated easily, such as entity popularity or document quality. Next, we present these main groups of features, which are summarized in Table 8.2. We then continue by briefly discussing the learning-to-rank algorithm used in EsdRank.

8.3.3.1 Features

Query-entity features include the following:

- *Query entity probability*, which can be computed by different methods in the query-to-entity mapping step (cf. Sect. 8.1).
- *Entity retrieval score*, which may be computed by using any standard retrieval model (such as LM, BM25, SDM) to score the query against the entity's description (in the case of unstructured entity representations) or a given entity field (in the case of structured entity representations).
- *Type-based similarity* between the target types of the query, $\mathcal{T}_q$, and the types of the entity, $\mathcal{T}_e$. The former may be computed using the target type detection methods we presented in Sect. 7.2, while the latter is provided in the knowledge base. Specifically, Xiong and Callan [51] consider the top three types identified for the query.
- *Entity similarity* considers the similarity between the candidate entity and other query entities in a pairwise manner. Then, these pairwise similarities are aggregated by taking their maximum or average.

Entity-document features comprise the following:

- *Text-based similarity* is measured between the document and the entity description (or fields thereof). These may be computed, e.g., by using cosine similarity or by applying standard retrieval models (LM, BM25, SDM, etc.) to score documents by treating the entity description as the search query.
- *Type-based similarity* may be computed between documents and entities, similarly to how it is done for queries and entities. Assigning types to the document may be approached as a multiclass classification problem or as a ranking problem (as it was done for queries in Sect. 7.2). Ultimately, the top-k types, i.e., with the highest confidence score, are considered for the document ($k = 3$ in [51]).
- *Graph-based similarity* considers the relationships of the entity. Let $\mathcal{E}_e^i$ denote the set of entities that are reachable from entity e in i hops, where $i \in [0..2]$ and $\mathcal{E}_e^0 = \{e\}$. Then, graph-based similarity in [51] is measured by the number of entities in $\mathcal{E}_e^i$ that are mentioned in d.

Other features may include, among others:

- *Entity frequency*, which reflects the popularity of the entity within the corpus and can be measured, e.g., using IDF.
- *Document quality indicators*, such as document length, URL length, SPAM score, PageRank score, number of inlinks, etc.

8.3.3.2 Learning-to-Rank Model

Xiong and Callan [51] introduce Latent-ListMLE, which extends the ListMLE [50] method. ListMLE is a listwise learning-to-rank algorithm that uses a parametric

model to estimate the probability of a document ranking being generated by a query. Then, it employs maximum likelihood estimation (MLE) to find the parameters that maximize the likelihood of the best ranking. One important assumption that ListMLE makes, to keep the optimization problem tractable, is that the probability of a document being ranked at a given position i is independent from all those documents that are ranked at earlier positions, $1 \ldots i-1$.

Latent-ListMLE extends ListMLE by adding a latent layer of candidate entities in the generation process. Similarly to ListMLE, it is assumed that the probabilities of selecting entities and documents at each position are independent of those that have been selected at earlier positions. However, instead of conditioning the document ranking probability directly on the query, first entities are sampled based on the query, then the probability of a document ranking is conditioned on the sampled entities. The probability of a document ranking $\mathbf{d} = \langle d_1, \ldots, d_k \rangle$ given q is:

$$P(\mathbf{d}|q;w,\theta) = \prod_{i=1}^{k} \sum_{e \in \mathcal{E}_q} P\left(d_i|e,\mathcal{D}_q(i,k)\right) P(e|q) \,,$$

where $\mathcal{D}_q(i,k) = \{d_i, \ldots, d_k\}$ is the set of documents that were not ranked in positions $1, \ldots, i-1$. The parameters of the model, w and θ, are learned using MLE and the EM algorithm. We refer to Xiong and Callan [51] for the details.

8.4 Entity-Based Representations

The main difference between the approaches in the previous section and those that will follow below is that instead of projecting documents to a *latent* entity layer, we will make use of *explicit* entity annotations of documents. We shall assume that the document has been annotated by some entity linking tool. The resulting set $\mathcal{E}_d$ of entities will be referred to as *document entities*. Entities may be blended with terms in a single representation layer, such as it is done in entity-based language models (Sect. 8.4.1). Alternatively, a separate bag-of-entities representation may be introduced and combined with the traditional bag-of-terms representation (Sect. 8.4.2).

8.4.1 Entity-Based Document Language Models

Raviv et al. [42] introduce entity-based language models (ELM), which consider individual terms as well as term sequences that have been annotated as entities (both in documents and in queries). They implement this idea by extending the vocabulary of terms ($\mathcal{V}$) with entities ($\mathcal{E}$). We shall write x to denote a vocabulary token, which here may be a term or an entity, $x \in \mathcal{V} \cup \mathcal{E}$. Further, we write $c(x;d)$ to denote

the (pseudo) count of x in document d. The representation length of the document is then given by $l_d = \sum_{x \in d} c(x;d)$. The maximum likelihood estimate of token x given d is defined as:

$$P(x|d) = \frac{c(x;d)}{l_d} . \tag{8.9}$$

This maximum likelihood estimate is then smoothed with a background (collection-level) language model analogously to how it is done for unigram language models, e.g., using Dirichlet smoothing:

$$P(x|\theta_d) = \frac{c(x;d) + \mu P(x|\mathcal{D})}{l_d + \mu} , \tag{8.10}$$

where μ is a smoothing parameter, and the collection language model is also a maximum likelihood estimate, computed over the set $\mathcal{D}$ of documents:

$$P(x|\mathcal{D}) = \frac{\sum_{d \in \mathcal{D}} c(x;d)}{\sum_{d \in \mathcal{D}} l_d} .$$

What remains to be defined is how the token pseudo-counts are computed. Raviv et al. [42] propose two alternatives:

- **Hard confidence-level thresholding** Only those entity annotations are considered in the document that are above a given (pre-defined) threshold $\tau \in [0,1]$. That is, the pseudo-count of token x is (1) the raw frequency of the term in the document, if the token is a term, and (2) the total number of mentions of the entity in the document with a minimum annotation confidence of τ, if x is an entity:

$$\tilde{c}(x;d) = \begin{cases} \lambda\, c(x;d) , & x \in \mathcal{V} \\ (1-\lambda) \sum_{i=1}^{l_d} \mathbb{1}(x_i = x, score_{EL}(x_i;d) \geq \tau) , & x \in \mathcal{E} , \end{cases}$$

  where x_i refers to the token at position i in the document and $score_{EL}(x_i;d)$ is the entity linking confidence associated with that token. The binary indicator function $\mathbb{1}()$ returns 1 if its argument evaluates to true, otherwise returns 0. The λ parameter controls the relative importance given to term vs. entity tokens.
- **Soft confidence-level thresholding** Instead of considering only entity annotations above a given threshold and treating them uniformly, the second method recognizes all entities that are linked in the document and weighs them by their corresponding confidence levels:

$$\tilde{c}(x;d) = \begin{cases} \lambda\, c(x;d) , & x \in \mathcal{V} \\ (1-\lambda) \sum_{i=1}^{l_d} \mathbb{1}(x_i = x)\, score_{EL}(x_i;d) , & x \in \mathcal{E} . \end{cases}$$

The ranking of documents is based on the negative cross-entropy (CE) between the query and document language models:[7]

$$score_{ELM}(d;q) = -CE(\theta_q||\theta_d) = \sum_{x \in \mathcal{V} \cup \mathcal{E}} P(x|\theta_q) \log P(x|\theta_d) \,,$$

where the query language model θ_q is a maximum-likelihood estimate (as in Eq. (8.9), but by replacing q with d). The document language model θ_d is instantiated by Eq. (8.10).

8.4.2 Bag-of-Entities Representation

Entity-based language models use a single representation layer, in which terms and entities are mixed together. We shall now discuss a line of work by Xiong et al. [53–55], where the term-based and entity-based representations are kept apart and are used in "duet." That is, queries and documents are represented in the term space as well as in the entity space. The latter is referred to as the *bag-of-entities* representation. Recall that we have already discussed this idea in the context of the ad hoc entity retrieval task in Sect. 4.2.2.[8]

8.4.2.1 Basic Ranking Models

Xiong et al. [53] present two basic ranking models based on bag-of-entities representations.

- **Coordinate Match** ranks documents based on the number of query entities they mention:

$$score_{CM}(d;q) = \sum_{e \in \mathcal{E}_q} \mathbb{1}\big(c(e;d) > 0\big) \,. \tag{8.11}$$

- **Entity Frequency** also considers the frequency of query entities in documents:

$$score_{EF}(d;q) = \sum_{e \in \mathcal{E}_q} c(e;q) \log c(e;d) \,. \tag{8.12}$$

[7] Note that scoring based on cross-entropy $CE(\theta_q||\theta_d)$ is rank-equivalent to scoring based on Kullback–Leibler divergence $KL(\theta_q||\theta_d)$ [58].

[8] Interestingly, the idea of a bag-of-entities representation was proposed independently and published at the same conference by Hasibi et al. [30] and Xiong et al. [53] for entity retrieval and document retrieval, respectively.

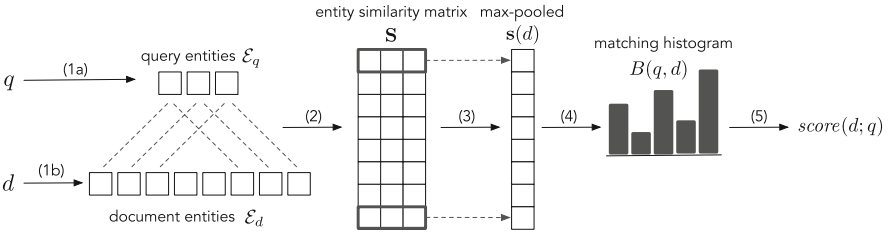


Fig. 8.4 Overview of the explicit semantic ranking (ESR) model [55]. The steps are: (1) entity linking in queries (1a) and in documents (1b); (2) computing pairwise entity similarities; (3) max pooling along the query dimension; (4) bin-pooling; (5) ranking documents using the histogram counts as features

These ranking functions are used to re-rank the top-k documents retrieved by a standard term-based retrieval model ($k = 100$ in [53]). Despite their simplicity, both models were shown to significantly outperform conventional term-based retrieval models [53].

8.4.2.2 Explicit Semantic Ranking

The *explicit semantic ranking* (ESR) [55] model incorporates relationship information from a knowledge graph to enable "soft matching" in the entity space. Figure 8.4 depicts an overview of the approach.

ESR first creates a query-document *entity similarity matrix* $\mathbf{S}$. Each element $\mathbf{S}(e, e')$ in this matrix represents the similarity between a query entity $e \in \mathcal{E}_q$ and a document entity $e' \in \mathcal{E}_d$:

$$\mathbf{S}(e, e') = \cos(\mathbf{e}, \mathbf{e}') \,,$$

where $\mathbf{e}$ is the embedding vector of entity e. In [55], entity embeddings are trained based on neighboring entities (i.e., entity relationships) in a knowledge graph.

ESR performs two pooling steps. The first one is max-pooling along the query dimension:

$$\mathbf{s}(d) = \max_{e \in \mathcal{E}_q} \mathbf{S}(e, \mathcal{E}_d) \,.$$

The second step is bin-pooling (introduced as *matching histogram mapping* in [29]), to group and count the number of document entities according to the strength of their matches to the query:

$$B_i(q, d) = \log \sum_j \mathbb{1}(st_i \leq \mathbf{s}_j(d) < ed_i) \,, \tag{8.13}$$

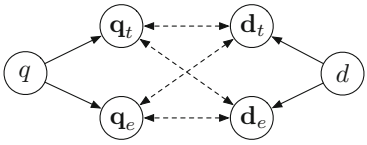


Fig. 8.5 Query-document matching in the word-entity duet framework [54]

where $[st_i, ed_i)$ is the score range for the ith bin, and $B_i(q,d)$ is the number of entities that fall into that bin. The bin ranges in [55] are $[0, 0.25)$, $[0.25, 0.5)$, $[0.5, 0.75)$, $[0.75, 1)$, $[1, 1]$. Negative bins are discarded. The rightmost bin with range $[1, 1]$ counts the *exact matches* in the entity space, while the other bins correspond to various degrees of *soft matches*. The resulting bin scores B_i are fed as features to a standard learning-to-rank model.

8.4.2.3 Word-Entity Duet Framework

Most recently, Xiong et al. [54] present the *word-entity duet framework*, which also incorporates cross-space interactions between term-based and entity-based representations, leading to four types of matches. The idea is illustrated on Fig. 8.5, where $\mathbf{q}_t$ and $\mathbf{d}_t$ denote the bag-of-words, while $\mathbf{q}_e$ and $\mathbf{d}_e$ denote the bag-of-entities representations of the query and the document, respectively. Each element in these vectors corresponds to the frequency of a given term/entity in the query/document. Based on these representations, query-document matching may be computed in four different ways:

- **Query terms to document terms** ($match(\mathbf{q}_t, \mathbf{d}_t)$): This corresponds to traditional term-based matching between a query and a document, and can be computed using standard retrieval models (e.g., LM or BM25) on various document fields (title and body in [54]).
- **Query entities to document terms** ($match(\mathbf{q}_e, \mathbf{d}_t)$): Relevance matching is performed by using the names or descriptions of query entities as (pseudo-)queries, and employing standard retrieval models to score them against the document (title or body fields).
- **Query terms to document entities** ($match(\mathbf{q}_t, \mathbf{d}_e)$): Similar in spirit to the previous kind of matching, the relevance between the query text and document entities is estimated by considering the names and descriptions of those entities. However, since the document may mention numerous entities, only the top-k ones with the highest relevance to the query are considered. Specifically, Xiong et al. [54] consider the top three entities from the document's title field and the top five entities from the document's body.
- **Query entities to document entities** ($match(\mathbf{q}_e, \mathbf{d}_e)$): Matches in the entity space can be measured using the coordinate match and entity frequency methods, cf. Eqs. (8.11) and (8.12). Additionally, matches can also be considered by using entity embeddings from a knowledge graph. In particular, Xiong et al. [54] learn

entity embeddings using the TransE model [4] and then use the ESR matching histogram scores (cf. Eq. (8.13)) as query-document ranking features.

The four-way matching scores from above are combined in a feature-based ranking framework.

8.4.2.4 Attention-Based Ranking Model

A main challenge with entity-based representations is the inherent uncertainty of automatic query entity annotations. It is inevitable that some entity mentions will be mistakenly linked, especially in short queries. Consequently, documents that match these (erroneous) entities would end up being promoted in the ranking. Xiong et al. [54] address this problem by developing an attention mechanism that can effectively demote noisy query entities.

A total of four attention features are designed, which are extracted for each query entity. *Entity ambiguity* features are meant to characterize the risk associated with an entity annotation. These are: (1) the entropy of the probability of the surface form being linked to different entities (e.g., in Wikipedia), (2) whether the annotated entity is the most popular sense of the surface form (i.e., has the highest commonness score, cf. Eq. (5.3))), and (3) the difference in commonness scores between the most likely and second most likely candidates for the given surface form. The fourth feature is *closeness*, which is defined as the cosine similarity between the query entity and the query in an embedding space. Specifically, a joint entity-term embedding is trained using the skip-gram model [41] on a corpus, where entity mentions are replaced with the corresponding entity identifiers. The query's embedding is taken to be the centroid of the query terms' embeddings.

We write $\Phi_{\mathbf{q}_t,\mathbf{d}_t}$, $\Phi_{\mathbf{q}_e,\mathbf{d}_t}$, $\Phi_{\mathbf{q}_t,\mathbf{d}_e}$, and $\Phi_{\mathbf{q}_e,\mathbf{d}_e}$ to refer to the four-way query-document features in the word-entity duet framework (cf. Sect. 8.4.2.3). Attention features are denoted as Φ_{Attn}. Using these five groups of features, the *AttR-Duet* model aims to learn a ranking function $score(d;q)$ that will be used for re-ranking an initial set of candidate documents.

The architecture of *AttR-Duet* is shown in Fig. 8.6. The model takes four matrices as input: $\mathbf{R}_t$, $\mathbf{R}_e$, $\mathbf{A}_t$, and $\mathbf{A}_e$. In the following, we will suppose that the query contains n words $q = \langle q_1, \ldots, q_n \rangle$ and there are m query entities $\mathcal{E}_q = \{e_1, \ldots, e_m\}$. $\mathbf{R}_t$ and $\mathbf{R}_e$ are ranking features for terms and entities, respectively. The rows of these matrices are made up of the word-duet feature vectors corresponding to each query term/entity:

$$\mathbf{R}_t(q_i, :) = \Phi_{\mathbf{q}_t,\mathbf{d}_t}(q_i) \sqcup \Phi_{\mathbf{q}_t,\mathbf{d}_e}(q_i)$$
$$\mathbf{R}_e(e_j, :) = \Phi_{\mathbf{q}_e,\mathbf{d}_t}(e_j) \sqcup \Phi_{\mathbf{q}_e,\mathbf{d}_e}(e_j) ,$$

where $\sqcup$ is a vector concatenation operator. $\mathbf{A}_t$ and $\mathbf{A}_e$ are attention features for terms and entities, respectively. Recall that the main objective is to handle the

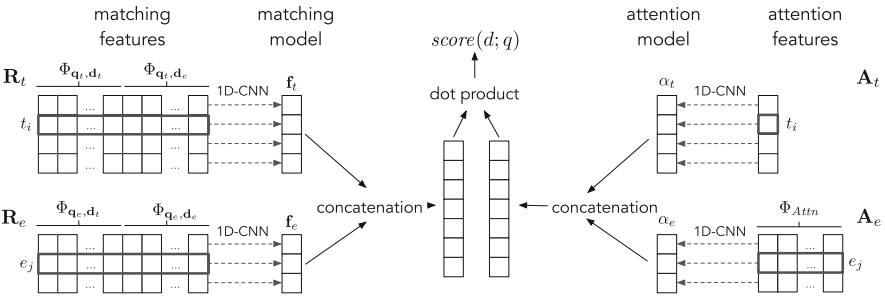


Fig. 8.6 Architecture of the AttR-Duet model [54]. The left side models query-document matching using the four-way term-entity features. The right side models the importance of query entities via attention features. The combination of these two yields the final document score

uncertainty of query entity annotations. Therefore, for terms, uniform attention is used; for entities, we employ the attention features introduced above:

$$\mathbf{A}_t(q_i, :) = 1$$
$$\mathbf{A}_e(e_j, :) = \Phi_{Attn}(e_j) \, .$$

The matching part (left side of Fig. 8.6) consists of two *convolutional neural networks* (CNNs), one for matching query terms ($\mathbf{R}_t$) and another for matching query entities ($\mathbf{R}_e$) against the document d. The convolution is applied on the term/entity dimension, "assuming that the ranking evidence from different query words [terms] or entities should be treated the same" [54]. Using a single CNN layer, one filter, and a linear activation function, the matching scores of terms and entities can be written as the following linear models:

$$\mathbf{f}_t(q_i) = \mathbf{w}_t^m \cdot \mathbf{R}_t(q_i, :) + b_t^m$$
$$\mathbf{f}_e(e_j) = \mathbf{w}_e^m \cdot \mathbf{R}_e(e_j, :) + b_e^m \, ,$$

where $\cdot$ is the dot product; $\mathbf{f}_t$ and $\mathbf{f}_e$ are n- and m-dimensional vectors, respectively; $\{\mathbf{w}_t^m, \mathbf{w}_e^m, b_t^m, b_e^m\}$ are the matching parameters to learn.

The attention part (right side of Fig. 8.6) also contains two CNNs, one for query terms ($\mathbf{A}_t$) and one for query entities ($\mathbf{A}_e$), using the same convolution idea as before. Using a single CNN layer and ReLU activation (to ensure non-negative attention weights), the attention weights on terms and entities can be written as:

$$\alpha_t(t) = ReLU\left(\mathbf{w}_t^a \cdot \mathbf{A}_t(t, :) + b_t^a\right)$$
$$\alpha_e(e) = ReLU\left(\mathbf{w}_e^a \cdot \mathbf{A}_e(e, :) + b_e^a\right) \, ,$$

where $\{\mathbf{w}_t^a, \mathbf{w}_e^a, b_t^a, b_e^a\}$ are the attention parameters to learn.

The final model, AttR-Duet, combines the matching and attention scores as:

$$score_{AD}(d; q) = \mathbf{f}_t \cdot \alpha_t + \mathbf{f}_e \cdot \alpha_e \, .$$

The matching part and attention parts of the model are learned simultaneously, by optimizing pairwise hinge loss:

$$L(q, \mathcal{R}^+, \mathcal{R}^-) = \sum_{d \in \mathcal{R}^+} \sum_{d' \in \mathcal{R}^-} [1 - score(d; q) + score(d'; q)]_+ ,$$

where $\mathcal{R}^+$ and $\mathcal{R}^-$ denote the set of relevant and non-relevant documents, respectively, and $[\,]_+$ is the hinge loss.

8.5 Practical Considerations

Efficiency is a key concern when serving search results. Compared to traditional term-based approaches, the computational overhead involved with the presented approaches stems from two components: (1) identifying the query entities, and (2) leveraging query entities in document scoring. Modern entity linking tools can already handle (1) with low latency, cf. Sect. 7.3. As for (2), document scoring is typically implemented as a re-ranking mechanism. That is, an initial retrieval is performed using the original query and a standard retrieval method, such as BM25 or LM. Then, the top-k scoring documents are re-ranked using the more advanced retrieval method. This is the same standard practice as in learning-to-rank [38]. Using a smaller k can result in markedly improved efficiency compared to a larger k. At the same time, using lower k values limits the scope, and hence potential, of the advanced method. A typical k value used in published work is around 100 [35, 51].

The efficiency of query expansion methods (Sect. 8.2) can be strongly affected by the number of expansion terms used (e.g., Meij et al. [40] consider maximum ten expansion terms). For approaches that operate with entity-based representations of documents (Sects. 8.4 and 8.3) entity annotations of documents can be performed offline and the linked entities can be stored in an inverted index structure. Similarly, entity descriptions can be constructed and indexed offline.

8.6 Resources and Test Collections

Experimental evaluation is commonly conducted using the test suites of the TREC 2009–2014 Web track [11–16], which employ the ClueWeb09 and ClueWeb12 collections. Additionally, the Robust04 newswire collection has also been used, with a set of topics from the ad hoc task in TREC 6–8 (#301–450) and topics developed for the TREC 2003–2004 Robust track (#601–700) [48]. See Table 8.3 for an overview. The reference knowledge base is typically Freebase, due to the availability of Freebase-annotated versions of the ClueWeb corpora, released by Google, referred to as the FACC1 dataset [27]; see Sect. 5.9.2. In addition to document annotations, the FACC1 dataset also

Table 8.3 Test collections for evaluating document retrieval methods that leverage entities

Document collection	#Documents	TREC topics	#Queries
Robust04	528k	Ad hoc #301–450, Robust #601–700	250
ClueWeb09-B	50M	Web #1–#200	200
ClueWeb12-B13	52M	Web #201–#300	100

contains manual entity annotations for the TREC 2009–2012 Web track queries. These annotations are limited to explicitly mentioned entities; 94 out of the 200 queries contain an entity. Dalton et al. [18] provide manually revised query annotations to improve recall, resulting in 191 of the 200 queries containing an entity.[9] For obtaining automatic entity annotations, TAGME [24] is a popular choice.

8.7 Summary

In this chapter, we have focused on leveraging entities for ad hoc document retrieval. The guiding principle behind these approaches is to obtain a semantically richer representation of the user's information need by identifying entities that are related to the query. This knowledge can then be utilized in the document retrieval process in various ways. In particular, we have discussed three families approaches: (1) *expansion-based*, which uses entities as a source for expanding the query with additional terms; (2) *projection-based*, where the relevance matching between a query and a document is performed by projecting them to a latent space of entities; and (3) *entity-based*, where explicit semantic representations of queries and documents are obtained in the entity space to augment the term-based representations. Moving from (1) to (2) and then from (2) to (3) corresponds to making increasingly more explicit use of entities, which, as it turns out, also translates to increasingly higher retrieval effectiveness. Entity-based representations, according to the current state of the art, can outperform a language modeling baseline by over 80% and a strong learning-to-rank baseline by over 20% in terms of NDCG@20, measured on the ClueWeb09-B collection [54].

8.8 Further Reading

It is also possible to combine the different perspectives of the discussed methods in a hybrid approach. For example, the EQFE method by Dalton et al. [18] uses explicit entity annotations of documents and performs query expansion based on entities and their properties (types and categories). Thereby, it bears some characteristics of

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

both entity-based representations and expansion-based methods. However, they do not use query expansion in the conventional sense (i.e., creating an expanded query model), but rather expand ranking features which are combined in a learning-to-rank approach.

Entity-based text representation may be utilized in many other tasks, e.g., computing document similarity [44], text classification [10], or question answering [6, 45]. Medical search is another prominent example for the use of controlled vocabulary representations, with a lot of work conducted in the context of the TREC Genomics track [31, 36, 46].

References

1. Balog, K., Weerkamp, W., de Rijke, M.: A few examples go a long way: Constructing query models from elaborate query formulations. In: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '08, pp. 371–378. ACM (2008). doi: 10.1145/1390334.1390399
2. Bendersky, M., Metzler, D., Croft, W.B.: Effective query formulation with multiple information sources. In: Proceedings of the Fifth ACM International Conference on Web Search and Data Mining, WSDM '12, pp. 443–452 (2012). doi: 10.1145/2124295.2124349
3. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent Dirichlet allocation. J. Mach. Learn. Res. **3**, 993–1022 (2003)
4. Bordes, A., Usunier, N., Garcia-Durán, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: Proceedings of the 26th International Conference on Neural Information Processing Systems, NIPS'13, pp. 2787–2795. Curran Associates Inc. (2013)
5. Brandão, W.C., Santos, R.L.T., Ziviani, N., de Moura, E.S., da Silva, A.S.: Learning to expand queries using entities. J. Am. Soc. Inf. Sci. Technol. pp. 1870–1883 (2014)
6. Cai, L., Zhou, G., Liu, K., Zhao, J.: Large-scale question classification in cQA by leveraging Wikipedia semantic knowledge. In: Proceedings of the 20th ACM International Conference on Information and Knowledge Management, CIKM '11, pp. 1321–1330. ACM (2011). doi: 10.1145/2063576.2063768
7. Callan, J.P.: Passage-level evidence in document retrieval. In: Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '94, pp. 302–310. Springer (1994)
8. Cao, G., Nie, J.Y., Gao, J., Robertson, S.: 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 '08, pp. 243–250. ACM (2008). doi: 10.1145/1390334.1390377
9. Castells, P., Fernandez, M., Vallet, D.: An adaptation of the vector-space model for ontology-based information retrieval. IEEE Trans. on Knowl. and Data Eng. **19**(2), 261–272 (2007). doi: 10.1109/TKDE.2007.22
10. Chang, M.W., Ratinov, L., Roth, D., Srikumar, V.: Importance of semantic representation: Dataless classification. In: Proceedings of the 23rd National Conference on Artificial Intelligence - Volume 2, AAAI'08, pp. 830–835. AAAI Press (2008)
11. Clarke, C.L.A., Craswell, N., Soboroff, I.: Overview of the TREC 2009 Web track. In: The Eighteenth Text REtrieval Conference Proceedings, TREC '09. NIST Special Publication 500-278 (2010)
12. Clarke, C.L.A., Craswell, N., Soboroff, I., V. Cormack, G.: Overview of the TREC 2010 Web track. In: The Nineteenth Text REtrieval Conference Proceedings, TREC '10. NIST Special Publication 500-294 (2011)

13. Clarke, C.L.A., Craswell, N., Soboroff, I., Voorhees, E.M.: Overview of the TREC 2011 Web track. In: The Twentieth Text REtrieval Conference Proceedings, TREC '11. NIST Special Publication 500-296 (2012)
14. Clarke, C.L.A., Craswell, N., Voorhees, E.M.: Overview of the TREC 2012 Web track. In: The Twenty-First Text REtrieval Conference Proceedings, TREC '12. NIST Special Publication 500-298 (2013)
15. Collins-Thompson, K., Bennett, P., Diaz, F., Clarke, C.L.A., Voorhees, E.M.: TREC 2013 Web track overview. In: The Twenty-Second Text REtrieval Conference Proceedings, TREC '13. NIST Special Publication 500-302 (2014)
16. Collins-Thompson, K., Macdonald, C., Bennett, P., Diaz, F., Voorhees, E.M.: TREC 2014 Web track overview. In: The Twenty-Third Text REtrieval Conference Proceedings, TREC '14. NIST Special Publication 500-308 (2015)
17. Croft, B., Metzler, D., Strohman, T.: Search Engines: Information Retrieval in Practice. 1st edn. Addison-Wesley Publishing Co. (2009)
18. Dalton, J., Dietz, L., Allan, J.: Entity query feature expansion using knowledge base links. In: Proceedings of the 37th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '14, pp. 365–374. ACM (2014). doi: 10.1145/2600428.2609628
19. Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K., Harshman, R.: Indexing by latent semantic analysis. J. Am. Soc. Inf. Sci. Technol. **41**(6), 391–407 (1990)
20. Diaz, F., Metzler, D.: Improving the estimation of relevance models using large external corpora. In: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '06, pp. 154–161. ACM (2006). doi: 10.1145/1148170.1148200
21. Dumais, S.T.: Latent semantic analysis. Ann. Rev. Info. Sci. Tech. **38**(1), 188–230 (2004). doi: 10.1002/aris.1440380105
22. Egozi, O., Gabrilovich, E., Markovitch, S.: Concept-based feature generation and selection for information retrieval. In: Proceedings of the 23rd National Conference on Artificial Intelligence - Volume 2, AAAI'08, pp. 1132–1137. AAAI Press (2008)
23. Egozi, O., Markovitch, S., Gabrilovich, E.: Concept-based information retrieval using explicit semantic analysis. ACM Trans. Inf. Syst. **29**(2), 8:1–8:34 (2011)
24. Ferragina, P., Scaiella, U.: TAGME: On-the-fly annotation of short text fragments (by Wikipedia entities). In: Proceedings of the 19th ACM International Conference on Information and Knowledge Management, CIKM '10, pp. 1625–1628. ACM (2010). doi: 10.1145/1871437.1871689
25. Gabrilovich, E., Markovitch, S.: Overcoming the brittleness bottleneck using Wikipedia: Enhancing text categorization with encyclopedic knowledge. In: Proceedings of the 21st National Conference on Artificial Intelligence - Volume 2, AAAI'06, pp. 1301–1306. AAAI Press (2006)
26. Gabrilovich, E., Markovitch, S.: Wikipedia-based semantic interpretation for natural language processing. J. Artif. Int. Res. **34**(1), 443–498 (2009)
27. Gabrilovich, E., Ringgaard, M., Subramanya, A.: FACC1: Freebase annotation of Clueweb corpora, version 1. Tech. rep., Google, Inc. (2013)
28. Gonzalo, J., Verdejo, F., Chugur, I., Cigarrin, J.: Indexing with WordNet synsets can improve text retrieval. In: Proceedings of the COLING/ACL'98 Workshop on Usage of WordNet for NLP, pp. 38–44 (1998)
29. Guo, J., Fan, Y., Ai, Q., Croft, W.B.: A deep relevance matching model for ad-hoc retrieval. In: Proceedings of the 25th ACM International on Conference on Information and Knowledge Management, CIKM '16, pp. 55–64. ACM (2016). doi: 10.1145/2983323.2983769
30. Hasibi, F., Balog, K., Bratsberg, S.E.: Exploiting entity linking in queries for entity retrieval. In: Proceedings of the 2016 ACM on International Conference on the Theory of Information Retrieval, ICTIR '16, pp. 209–218. ACM (2016). doi: 10.1145/2970398.2970406

31. Hersh, W., Voorhees, E.: TREC genomics special issue overview. Inf. Retr. **12**(1), 1–15 (2009). doi: 10.1007/s10791-008-9076-6
32. Jagerman, R., Eickhoff, C., de Rijke, M.: Computing web-scale topic models using an asynchronous parameter server. In: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '17, pp. 1337–1340. ACM (2017). doi: 10.1145/3077136.3084135
33. Lavrenko, V., Croft, W.B.: Relevance based language models. In: Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '01, pp. 120–127. ACM (2001). doi: 10.1145/383952.383972
34. Liu, X., Chen, F., Fang, H., Wang, M.: Exploiting entity relationship for query expansion in enterprise search. Inf. Retr. **17**(3), 265–294 (2014). doi: 10.1007/s10791-013-9237-0
35. Liu, X., Fang, H.: Latent entity space: A novel retrieval approach for entity-bearing queries. Inf. Retr. **18**(6), 473–503 (2015). doi: 10.1007/s10791-015-9267-x
36. Lu, Z., Kim, W., Wilbur, W.J.: Evaluation of query expansion using mesh in pubmed. Inf. Retr. **12**(1), 69–80 (2009). doi: 10.1007/s10791-008-9074-8
37. Lv, Y., Zhai, C.: A comparative study of methods for estimating query language models with pseudo feedback. In: Proceedings of the 18th ACM Conference on Information and Knowledge Management, CIKM '09, pp. 1895–1898. ACM (2009). doi: 10.1145/1645953.1646259
38. Macdonald, C., Santos, R.L., Ounis, I.: The whens and hows of learning to rank for web search. Inf. Retr. **16**(5), 584–628 (2013). doi: 10.1007/s10791-012-9209-9
39. Manning, C.D., Raghavan, P., Schütze, H.: Introduction to Information Retrieval. Cambridge University Press (2008)
40. Meij, E., Trieschnigg, D., de Rijke, M., Kraaij, W.: Conceptual language models for domain-specific retrieval. Inf. Process. Manage. **46**(4), 448–469 (2010). doi: http://dx.doi.org/10.1016/j.ipm.2009.09.005
41. Mikolov, T., Sutskever, I., Chen, K., Corrado, G., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Proceedings of the 26th International Conference on Neural Information Processing Systems, NIPS'13, pp. 3111–3119. Curran Associates Inc. (2013)
42. Raviv, H., Kurland, O., Carmel, D.: Document retrieval using entity-based language models. In: Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '16, pp. 65–74. ACM (2016). doi: 10.1145/2911451.2911508
43. Rocchio, J.: Relevance feedback in information retrieval. In: Salton, G. (ed.) The SMART Retrieval System—Experiments in Automatic Document Processing. Prentice-Hall, Inc. (1971)
44. Schuhmacher, M., Ponzetto, S.P.: Knowledge-based graph document modeling. In: Proceedings of the 7th ACM International Conference on Web Search and Data Mining, WSDM '14, pp. 543–552. ACM (2014). doi: 10.1145/2556195.2556250
45. Srba, I., Bielikova, M.: A comprehensive survey and classification of approaches for community question answering. ACM Trans. Web **10**(3), 18:1–18:63 (2016). doi: 10.1145/2934687
46. Stokes, N., Li, Y., Cavedon, L., Zobel, J.: Exploring criteria for successful query expansion in the genomic domain. Inf. Retr. **12**(1), 17–50 (2009). doi: 10.1007/s10791-008-9073-9
47. Voorhees, E.M.: Using wordnet to disambiguate word senses for text retrieval. In: Proceedings of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '93, pp. 171–180. ACM (1993). doi: 10.1145/160688.160715
48. Voorhees, E.M.: The TREC Robust retrieval track. SIGIR Forum **39**(1), 11–20 (2005). doi: 10.1145/1067268.1067272
49. Weerkamp, W., Balog, K., de Rijke, M.: Exploiting external collections for query expansion. ACM Trans. Web **6**(4), 18:1–18:29 (2012). doi: 10.1145/2382616.2382621
50. Xia, F., Liu, T.Y., Wang, J., Zhang, W., Li, H.: Listwise approach to learning to rank: Theory and algorithm. In: Proceedings of the 25th International Conference on Machine Learning, ICML '08, pp. 1192–1199. ACM (2008). doi: 10.1145/1390156.1390306
51. Xiong, C., Callan, J.: Esdrank: Connecting query and documents through external semi-structured data. In: Proceedings of the 24th ACM International on Conference on

Information and Knowledge Management, CIKM '15, pp. 951–960. ACM (2015a). doi: 10.1145/2806416.2806456
52. Xiong, C., Callan, J.: Query expansion with freebase. In: Proceedings of the 2015 International Conference on The Theory of Information Retrieval, ICTIR '15, pp. 111–120. ACM (2015b). doi: 10.1145/2808194.2809446
53. Xiong, C., Callan, J., Liu, T.Y.: Bag-of-entities representation for ranking. In: Proceedings of the 2016 ACM on International Conference on the Theory of Information Retrieval, ICTIR '16, pp. 181–184. ACM (2016). doi: 10.1145/2970398.2970423
54. Xiong, C., Callan, J., Liu, T.Y.: Word-entity duet representations for document ranking. In: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '17, pp. 763–772. ACM (2017a). doi: 10.1145/3077136.3080768
55. Xiong, C., Power, R., Callan, J.: Explicit semantic ranking for academic search via knowledge graph embedding. In: Proceedings of the 26th International Conference on World Wide Web, WWW '17, pp. 1271–1279. International World Wide Web Conferences Steering Committee (2017b). doi: 10.1145/3038912.3052558
56. Xu, Y., Jones, G.J.F., Wang, B.: Query dependent pseudo-relevance feedback based on Wikipedia. In: Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '09, pp. 59–66 (2009). doi: 10.1145/1571941.1571954
57. Yi, X., Allan, J.: A comparative study of utilizing topic models for information retrieval. In: Proceedings of the 31th European Conference on IR Research on Advances in Information Retrieval, ECIR '09, pp. 29–41. Springer-Verlag (2009). doi: 10.1007/978-3-642-00958-7_6
58. Zhai, C.: Statistical language models for information retrieval A critical review. Found. Trends Inf. Retr. **2**(3), 137–213 (2008)
59. Zhai, C., Lafferty, J.: Model-based feedback in the language modeling approach to information retrieval. In: Proceedings of the 10th international conference on Information and knowledge management, CIKM '01, pp. 403–410. ACM (2001). doi: 10.1145/502585.502654