ENT Rank: Retrieving Entities for Topical Information Needs through Entity-Neighbor-Text Relations

Laura Dietz
University of New Hampshire
Durham, NH, USA
dietz@cs.unh.edu

ABSTRACT

Related work has demonstrated the helpfulness of utilizing information about entities in text retrieval; here we explore the converse: Utilizing information about text in entity retrieval. We model the relevance of Entity-Neighbor-Text (ENT) relations to derive a learning-to-rank-entities model.

We focus on the task of retrieving (multiple) relevant entities in response to a topical information need such as "Zika fever". The ENT Rank model is designed to exploit semi-structured knowledge resources such as Wikipedia for entity retrieval. The ENT Rank model combines (1) established features of entity-relevance, with (2) information from neighboring entities (co-mentioned or mentioned-on-page) through (3) relevance scores of textual contexts through traditional retrieval models such as BM25 and RM3.

ACM Reference Format:
Laura Dietz. 2019. ENT Rank: Retrieving Entities for Topical Information Needs through Entity-Neighbor-Text Relations. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '19), July 21–25, 2019, Paris, France.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3331184.3331257

1 INTRODUCTION

Entity retrieval is important in many different applications where entities are sought in response to a textual description, type definition, or set of related entities. Information needs in natural language, structured SPARQL queries, or hybrids have been explored [6]. Often only a single entity is requested, such as in factoid question answering, conversational retrieval, or quizzes. In contrast, this work[1] studies entity retrieval where, in response to a short information need, all topically related entities are to be retrieved. The motivation is to support authors in writing comprehensive articles about (yet) unfamiliar topics. While the information need is only expressed in a short keyword query, the topic is expected to have several interesting facets which should all be covered. We anticipate that knowing the set of relevant entities, ordered from central to side-topic is helpful for the author. Results also can inform conversational agents with background information on this topic. As our approach is modeling the context of relevant entities, this work also constitutes a first step towards fully automatic article composition approaches or even a new way to find *information* rather than documents through search engines [1].

An example topic is "Zika fever". Despite being a short unambiguous keyword query, several facets need to be covered such as "Signs and Symptoms", "Causes", or "Epidemiology in Americas". Several entities must be mentioned for this topic, such as "Aedes Mosquitoes" which are the vector for transmission, the "2015–2016 Zika fever epidemic" and other outbreaks, that Zika fever is a "Flavivirus", it causes muscle weakness due to nerve damage also called "Guillain-Barré syndrome", the "Neonatal infection" which is the most serious concern, and that "Dengue Fever" is a similar disease with confusable symptoms, and that "Lethal ovitraps" are used to trap adult Aedes mosquitos. Many, but not all of these relevant entities are mentioned on the Wikipedia page for Zika fever[2].

Topical entity retrieval task: Given an article title as query, retrieve a ranking of relevant entities. Relevance is defined based on whether the entity must, should, or could be mentioned in an article on this topic.

The entity retrieval task of the TREC Complex Answer Retrieval track [14] (CAR) offers a suitable benchmark to study this task. The benchmark includes a large amount of training data that is synthetically derived from entities mentioned on Wikipedia pages. This benchmark is complemented by manual assessments. While the official CAR queries provide titles with an outline of facets (e.g., "Zika fever/Causes"), this work focuses on retrieving entities in response to the title queries alone.

The CAR benchmark includes an easily parsable dump of English Wikipedia pages (December 2016). The structure of each Wikipedia page, i.e., headings, paragraphs, and entity links are provided for each page, as well as meta data such as redirect names, categories, and inlinks. Query pages are excluded from the dump. In this work we make use of this format, which could also be derived for other text-centric knowledge resources in bio-medical (NCBI/pubmed), finance (Bloomberg), and news (Washington Post) domains or websites such as www.howstuffworks.com.

Current entity retrieval approaches focus on the development of relevance features. One set of features is derived from a knowledge graph (or Wikipedia), such as names, types, linked entities, and free-form descriptions [3, 36]. Another set of features is derived from entity links in queries and unstructured text documents [18, 31]. Neighbor relations are derived from knowledge graph links [3] or co-mentions in text (i.e, two entities being mentioned in near proximity) [21]. So far, related work would consider all neighbor

[1] Code and data available at https://www.cs.unh.edu/~dietz/appendix/ent-rank/

[2] https://en.wikipedia.org/wiki/Zika_fever

SIGIR '19, July 21–25, 2019, Paris, France

ACM ISBN 978-1-4503-6172-9/19/07…$15.00
https://doi.org/10.1145/3331184.3331257

relations from the same source as equally important for the query. In contrast, this work models the relevance of neighbor relations through textual contexts with different measures of relevance.

Contributions. We introduce the ENT Rank framework for integrating relevance information from the entity, its neighbors, and context. We provide a versatile learning-to-rank-entities algorithm that can be optimized for any rank evaluation metric, such as mean-average precision. The ENT Rank framework can incorporate any existing entity relevance feature and can be easily customized. We demonstrate that even with simple features derived from unfielded unigram models, such as BM25 and RM3, ENT Rank provides a competitive retrieval method. In a comparison between ENT Rank and established methods on TREC Complex Answer Retrieval [14] and DBpedia-entity v2 [20], ENT Rank places best or second-best.

The idea behind ENT Rank is to use text fragments with entity links, so-called contexts, to define neighbor relations between entities. This allows us to derive a hypergraph, where entities are represented as nodes, and context-neighbor relations are represented as edges. Preserving the association between each context and neighbor relation, allows us to use text-retrieval models to predict the relevance of a neighbor relation for the query. Furthermore, relevance information from context-neighbor relations is used to complement traditional entity relevance features.

Outline. In Section 2 we provide an overview of the state-of-the-art on this task. Section 3 introduces the ENT Rank framework and motivates different special cases through random walks. Section 4 discusses the entity, neighbor, and context features used in the experimental evaluation using entity retrieval benchmarks from Complex Answer Retrieval in Section 5 and DBpedia-entity v2 in Section 6.

2 RELATED WORK

Entity retrieval was introduced to integrate information retrieval and semantic search [3]. It is often motivated by the large number of named entities mentioned in search requests [28]. Different flavors of this task are to retrieve related entities through a set of entities, type descriptions, or topics of expertise [4, 5, 12]. Entity retrieval can provide answers to questions, such as "Who invented the paper clip?" In the context of Complex Answer Retrieval, entity retrieval offers entities that should to be discussed in different parts of the answer [14].

Entity retrieval from knowledge base documents. Successful approaches to all these variants of entity retrieval center around a representation of each entity as a fielded document. After full-text indexing, entity retrieval can be addressed by traditional retrieval models for ad hoc document retrieval [28]. Tonon et al. combine full text search with structured queries [33]. Balog et al. [2] suggests a term-based and category-based entity representation, where term statistics are derived from documents representing the entity (such as a Wikipedia page). For queries consisting of terms, categories, and related entities, Balog et al. use a generative retrieval model based on Kullback-Leibler divergence of entity and category language model. Meij et al. [24] further include information from search histories. Garigliotti et al. focus on category or type information [16].

Raviv et al. [29] extend the sequential dependence model (SDM) [26] to different entity fields, name, document, and type. Zhiltsov [27, 36] suggests the parametrized fielded sequential dependence model (PFSDM), which assigns different weights to matches of different fields, query term types, and bigrams. The retrieval approach is based on the weighted sequential dependence model [8], which combines unigram, bigram, window bigrams with additional information using a Markov random field. The weights for these features are trained with coordinate ascent. Chen et al. [9] demonstrates that learning-to-rank frameworks offer further improvements.

Entity linking tools annotate unstructured text with mentions of entities, providing a new avenue for entity retrieval. Hasibi et al. [18] applies entity linking to queries, to extend the SDM approach with another dependency. Schuhmacher et al. uses entity links in web documents for entity retrieval in a pseudo-relevance feedback approach: Inspecting retrieved web documents, entities are ranked high if they are mentioned in (many) high-ranked documents.

Ad hoc document retrieval with entities. By approaching entity retrieval as retrieval of fielded documents, combinations of ad hoc entity retrieval and document retrieval explored. Raviv et al. [30] suggests to represent queries and documents as bag-of-words and bag-of-entity-links for ad hoc document retrieval tasks. Liu et al. [23] rank documents through relevant entities. While the relevance of entities is latent, indicators of entity relevance are derived from entity links and Freebase abstracts. Xiong et al. suggests a discriminative machine learning approach to incorporate different meta information about entities into the document ranking model. Dalton et al. [11] compute an entity-term-category expansion model based on a feedback run of retrieved documents and sources of entity information: entity links in the query, a ranking of Wikipedia pages (i.e., an entity ranking), and entity link information in documents. Recently, neural network approaches for joint entity-document ranking are further leveraging this connection [35].

Entity linking. Entity linking methods annotate unstructured text with hyperlink-like positional references to Wikipedia. Fast and reliably entity linking toolkits, such as TagMe and Nordlys [15, 19], are readily available. Entity linking combines spotting of possible entity mentions with the disambiguation among similarly named entities. Several information retrieval approaches to entity linking use the fielded entity representations discussed above [17]. Text surrounding the spot can be cast as a search query for entity retrieval [10]. Special features of short text such as tweets [25] can be incorporated.

Graphs, Relations, and Neighbors. Knowledge graphs contain information about how entities are related through RDF triples with relation types. Alternatively, relations with "cheap semantics" [3] can be derived from hyperlinks on Wikipedia, or entities that are mentioned in the same document. Kotov et al. [21] combines both explicit relations available from ConceptNet together with information which entities are mentioned near one another (cf. HAL [21]). With application to question answering, Bast et al. [7] learn weights on different relations by matching corpus-based templates to demanded relation types. This approach is based on the idea of weakly supervised relation extraction to generate training data for relevant relations.

3 ENT-RANK APPROACH

The difficulty of using neighbor relations for entity retrieval lies in the presence of many connections of which the majority are typically not relevant for the query. One example is the entity "South America" which is relevant for the query "Zika fever" as a location of a major outbreak. However, many contexts about South America are unrelated to the Zika fever, such as political incidents or environmental issues due to the loss of rainforest. In fact, there are so many interesting topics to discuss about South America that there is no room to mention the Zika fever outbreak on South America's Wikipedia page.

We notice an asymmetry of relevance: just because South America is relevant for a discussion of the Zika fever, it does not mean that the Zika fever is equally relevant for a discussion about South America. Therefore, short entity descriptions, such as the introductory Wikipedia paragraph, are often not mentioning relevant connections. We compensate this lack with a text-oriented approach. We hypothesize that whenever two entities are mentioned in a relevant context, it is a strong indicator that both entities are topically relevant. The relevance of the context is predicted through text-based retrieval models. We define the relevance of *context-neighbor relations* based on the relevance of contexts and entities. We use entity links in context to estimate (1) the relevance of an entity and (2) the relevance of the neighbor relations. This has the advantage that we can access a wide range of facts about entities, including those with vague semantics that do not fit easily in a relation schema. Our approach is intended to be combined with established entity relevance models discussed in the related work.

The remainder of this section introduces the construction of the ENT Rank framework. In response to a query, (1) a ENT hypergraph is defined. (2) Preserving entities as nodes, the hypergraph is converted into the binary *ENT multi-graph* G, which (3) is associated with query-specific node and edge feature vectors to obtain the *ENT feature vector graph* $\mathcal{G}$. This graph is used in (4) the ENT learning-to-rank-entities model for training and ranking prediction.

3.1 ENT Hypergraph and Binary Multi-Graph

All Wikipedia pages (or alternatively, documents from any corpus) are annotated with entity links and segmented into contexts, such as paragraphs, sections, and pages. Each context induces a hyperedge between all entities that are linked therein. The association between hyperedge and the context is preserved. The construction is depicted in Figure 1.

This approach offers the option of a full-text search index from which hyperedges can be retrieved with different retrieval models such as BM25. We suggest to create the graph from several input rankings of entities e and contexts t. The hypergraph forms the basis for reasoning about the relevance of edges.

Given a search query, the ENT Rank approach formalizes the connections between entities e_i, contexts t_k, and neighboring entities e_j as a binary multi-graph $G = (V, R)$, where nodes V represent entities e_i and edges R represent directed context-neighbor

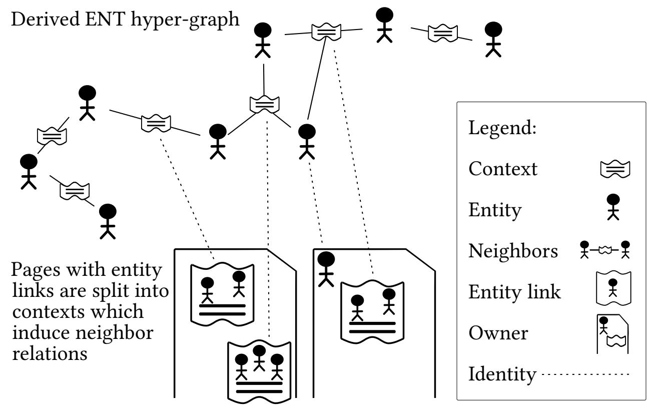

Figure 1: ENT Hypergraph is created from contexts with entity links. Example contexts are paragraphs, pages, and sections on Wikipedia pages.

relations $r = (e_i, t_k, e_j)$. Figure 2 depicts an example with two contexts t_1, t_2 that mentions three entities, e_1, e_2, e_3. They induce a graph G with $V = \{e_1, e_2, e_3\}$, and multi-edges R for each of the six directed neighbor relations (e_1, t_1, e_2), (e_2, t_1, e_1), (e_2, t_1, e_3), ... from t_1 and two directed neighbor relations (e_2, t_2, e_3), (e_3, t_2, e_2) from context t_2. As both contexts mention entity e_2 and e_3, these induce two edges from e_1 to e_2 depicted in black and gray (hence a multi-graph).

3.2 ENT Feature Vector Graphs

We endow nodes $e_i \in V$ and context-neighbor edges (e_i, t_k, e_j) in G with feature vectors as follows, deriving the *ENT feature vector graph* $\mathcal{G}$ for the search query. Node feature vectors $\vec{f}_{e_i}$ are comprised of features that indicate the (direct) relevance of the entity e_i for the query. Many established entity relevance features have been discussed in the related work—these are directly applicable to the ENT-Rank model as node features. The novel contribution of ENT rank lies in the incorporation of relevance indicators from context-based neighbor relations. Every multi-edge (e_i, t_k, e_j) is endowed with an edge feature vector $\vec{f}_{(e_i, t_k, e_j)}$ that is comprised of features that indicate how relevant the context-based neighbor relation is for the query. A wide range of features can be included such as the relevance of the context measured by a BM25 score, the saliency of the entity in the context, the role of the neighbor relationship, the similarity of neighbors, and other entity features of the neighbor. The concrete list of features used in this work is given in Section 4.

3.3 ENT Learning-to-Rank-Entities Model

The major challenge in using the ENT Rank model for entity ranking is the vast amount of heterogeneous feature choices: Offering multiple sources for contexts, different neighbor roles, different entity (node) relevance features, different context (edge) relevance features can result in hundreds of combinations to explore. We suggest the following learning-to-rank approach to choose the ideal

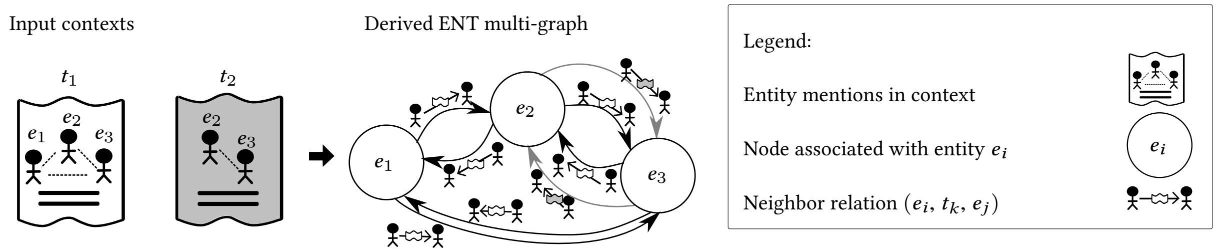

Figure 2: The binarized ENT-multi graph is derived from contexts, where each context t_1 and t_2 induces a (directed) neighbor relation between all entity pairs that are mentioned in it. Multiple edges are induced between entities that co-occur in two contexts, here e_2 and e_3.

weighted combination of these choices, given sufficient training data.

The entity ranking is derived from the ENT feature vector graph $\mathcal{G}$ using a learning-to-rank model with weight parameters $\vec{\psi}$ and $\vec{\theta}$. As customary in learning-to-rank, the weight parameters are trained across many queries; dependence on the query is expressed through the features. We first discuss the prediction of a ranking, second how to train the weight parameters, and finally give a motivation that is based on random walks. We define features $\vec{g}$ of an entity pair as,

$$\vec{g}(e_i, e_j) = \sum_{\forall k: e_i, e_j \in t_k} \vec{f}_{(e_i, t_k, e_j)} \tag{1}$$

Ranking prediction. Given trained node and multi-edge parameters $\vec{\psi}$ and $\vec{\theta}$ and a query-specific feature vector graph $\mathcal{G}$. We define the rank score of an entity (i.e., node) e_j as the sum of both linear models for nodes and multi-edges as follows. We will give a detailed motivation for this equation below.

$$\begin{aligned} \text{score}(e_j) &= \vec{\psi}\,\vec{f}_{e_j} + \frac{1}{|V|}\sum_i \vec{\theta}\,\vec{g}(e_i, e_j) \qquad (2) \\ &= \begin{pmatrix} \vec{\psi} \\ \vec{\theta} \end{pmatrix} \begin{pmatrix} \vec{f}_{e_j} \\ \frac{1}{|V|}\sum_i \vec{g}(e_i, e_j) \end{pmatrix} \end{aligned}$$

Here $|V|$ is the number of nodes in the graph. The second line follows after rearranging inner vector products and stacking weight parameters $\vec{\psi}$ and $\vec{\theta}$ into a single weight parameter vector which contains all entries in $\vec{\psi}$ followed by all entries in $\vec{\theta}$. Likewise, node and multi-edge feature vectors are stacked, after summing vectors across all multi-edges (e_i, t_k, e_j) that connect to e_j. The summing here refers to a component-wise vector addition.

Training. The weight parameters are trained to achieve optimal entity ranking performance on the training set. In this work, we use mini-batched coordinate-ascent as a training algorithm, but other training algorithms are equally applicable. Coordinate-ascent is an iterative algorithm that optimizes the weight of one feature at at time in a round-robin fashion until no further improvement in ranking performance can be achieved. The ranking performance is evaluated with mean-average precision (MAP). We use a variant called mini-batch stochastic gradient ascent, which performs each iteration on a different random subset of 150 training queries. The algorithm is stopped when the relative change in MAP is less than 1%. This convergence is usually achieved within 5 iterations, since our features are all positively correlated with relevance.

Motivation. The ENT learning-to-rank-entities model is inspired by random walks with restarts [32], where $P(e_j)$ is the probability of chosing node e_j during restart. Due to space constraints, this work only discusses the simple case of weighted degree centrality, i.e., random walks with only one step, for which an analytic solution to the optimization problem is available.

Nodes are initialized uniformly at random (i.e., $\frac{1}{|V|}$). The transition from node e_i to e_j is given by the transition probability $P(e_i \to e_j | e_i)$ given that the random surfer is on the start node e_i. Using teleportation probability $\alpha \in (0, 1)$, transition probabilities are denoted as matrix $\mathbf{T}$, where

$$T_{ij} = \alpha P(e_j) + (1 - \alpha) P(e_i \to e_j | e_i).$$

Under degree centrality (i.e., one random walk step) the score of the receiving node e_j is

$$\text{score}(e_j) = \frac{1}{|V|}\sum_i T_{ij} = \alpha P(e_j) + (1-\alpha)\frac{1}{|V|}\left(\sum_i P(e_i \to e_j | e_i)\right)$$

The fraction of $|V|$ vanishes from the first term, when the teleport is summed over all sending nodes.

For learning-to-rank-entities we model the restart probabilities and transition probabilities as linear models of node feature vectors $\vec{f}$ and edge feature vectors $\vec{g}$. The ratio of teleportation versus transition ($\frac{\alpha}{1-\alpha}$) is absorbed into parameters and estimated as part of the training process. Since only a rank-equivalent rank score is necessary, we let

$$\alpha P(e_j) \overset{\text{rank}}{=} \vec{\psi}\,\vec{f}_{e_j} \tag{3}$$

$$(1 - \alpha) P(e_i \to e_j | e_i) \overset{\text{rank}}{=} \vec{\theta}\,\vec{g}(e_i, e_j) \tag{4}$$

$$\text{score}(e_j) \overset{\text{rank}}{=} \vec{\psi}\,\vec{f}_{e_j} + \frac{1}{|V|}\sum_i \vec{\theta}\,\vec{g}(e_i, e_j) \tag{5}$$

These are combined into Equation 5. Thereby, we arrive at the formulation which is given in Equation 2.

3.4 Options for Multi-edge Feature Vectors

We envision features $\vec{\mathrm{f}}$ and $\vec{\mathrm{g}}$ in the ENT Rank feature graph $\mathcal{G}$ to be tailored to the application domain. Before providing details on the set of features used in this work, we discuss how we envision information about contexts, neighbors, and relation types to be integrated into the ENT Rank framework. Our suggestions are based on probabilistic random walks.

3.4.1 Neighbor features. When entity features of the sending neighbor e_i are available, the feature vector of the multi-edge (e_i, t_k, e_j) can be derived by letting

$$\vec{f}_{(e_i,t_k,e_j)} = \vec{\mathrm{f}}(e_i)$$

Following Equation 2, this results in a rank score for e_j that is

$$\mathrm{score}(e_j) \overset{\mathrm{rank}}{=} \vec{\psi}\,\vec{\mathrm{f}}e_j + \frac{1}{|V|}\sum_i \sum_{\forall k: e_i, e_j \in t_k} \vec{\theta}\,\vec{\mathrm{f}}e_i$$

where $\vec{\psi}$ and $\vec{\theta}$ control the importance of different neighbor features versus entity features. This formulation naturally incorporates the multiplicity of multi-edges between e_i and e_j. In the context of semi-supervised classification, this model is also known as linear neighborhood propagation [34].

3.4.2 Relation-typed neighbor features. As mentioned earlier, different entities can play different roles in the context, such as being the owner of the context versus being mentioned in the context. These roles can define a type of the neighbor relation (e.g., owner-link). Furthermore, different types of contexts can be considered, such as paragraph, section, or page. We suggest to incorporate different context and neighbor types of (e_i, t_k, e_j) as relation types r, by reserving separate blocks in the feature vector $\vec{\mathrm{f}}$ for different relation types. These blocks can be stacked to obtain the feature vector $\vec{\mathrm{f}}(e_j, r)$.

$$\vec{\mathrm{f}}(e_j, r) = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ \vec{\mathrm{f}}(e_j) \\ 0 \\ \vdots \\ 0 \end{pmatrix} \leftarrow \text{block for relation type } r$$

If across multiple contexts, entity e_j and e_i have different roles, we suggest to copy the entity feature vector of neighbor e_i into all corresponding relation type blocks. The consequence is that the training algorithm would then not only learn to balance entity features of e_j versus neighbor e_i, but also assign different importance weights depending on neighbor relation type and context type.

3.4.3 Context-Relevance Features. When contexts are retrieved from a full-text index, contexts t_k are naturally associated with features from text retrieval models, such as the retrieval score of the context under a BM25 model or the RM3 expansion models with different hyperparameters. Each of these retrieval models would contribute a separate relevance feature for the context t_k (or a reasonable default value if the context is not included in the ranking).

We incorporate the case of context feature vectors based on related work [37] on random walks for hypergraphs. Zhou et al. suggest the following random surfer process: A random surfer on node e_i first surfs to an adjacent hyper edge t_k proportionally to its edge weight $\omega(t_k)$, next the surfer chooses one node adjacent to the hyperedge uniformly at random to surf to. This process includes the possibility of surfing back to the starting node e_i.

Under this model, the transition probability from node e_i to e_j via t_k is proportional to $\frac{1}{|t_k|}\omega(t_k)$. When multiple hyperedges connect e_i to e_j, the marginal transition probability from node e_i to e_j is given by $P(e_i \to e_j | e_i) \propto \sum_{\forall k: \exists (e_i, t_k, e_j)} \frac{1}{|t_k|}\omega(t_k)$.

This corresponds to Equation 4 in our feature graph formulation where the feature vector for transition from e_i to e_j can be expressed as features of connecting hyperedges t_k:

$$\vec{\mathrm{f}}(e_i, e_j) = \sum_{\forall k: e_i, e_j \in t_k} \frac{1}{|t_k|}\vec{\mathrm{f}}(t_k)$$

Here $|t_k|$ denotes the number of entities mentioned in the context. In relation to Equation 1, it follows that

$$\vec{f}_{(e_i,t_k,e_j)} = \frac{1}{|t_k|}\vec{\mathrm{f}}(t_k) \tag{6}$$

Our experiments empirically confirm that dividing hyperedge feature vectors by the number of neighbors provides slightly better results than the unnormalized alternative, $\vec{f}_{(e_i,t_k,e_j)} = \vec{\mathrm{f}}(t_k)$.

3.4.4 Combinations of Multi-edge Features Vectors. We envision that multi-edge feature vectors $\vec{\mathrm{f}}$ are composed of both relation-typed neighbor features, context features, and many other feature sources by stacking feature vectors into one combined feature vector,

$$\vec{f}_{(e_i,t_k,e_j)} = \begin{pmatrix} \vec{\mathrm{f}}(e_j, r) \\ \frac{1}{|t_k|}\vec{\mathrm{f}}(t_k) \\ \ldots \end{pmatrix}$$

We use this representation to construct the edge feature vector for ENT Learning-to-rank-entities for use in Equation 1.

4 WIKIPEDIA FEATURES FOR ENT RANK

In this study we use the following set of retrieval-based features for entities feature vectors $\vec{\mathrm{f}}(e_j)$ and multi-edge feature vectors $\vec{f}(e_i, t_k, e_j)$ as described in Section 3.4.4. The features used in the evaluation are derived from a 2016 Wikipedia dump and a corpus of paragraphs (as provided with the TREC CAR data)—other datasets of knowledge graphs and text are equally applicable.

From the Wikipedia dump and a text corpus we extract the following types of information which are used as a source of contexts and/or entity relevance, from which we derive ENT feature vector graphs.

- **Page:** Full-text of Wikipedia pages, including all visible text including title, headings, and content paragraphs. For the graph only bi-directional entity links are included as neighbors (i.e., links to pages that link back).
- **Entity:** Knowledge graph representation of entities using only head information such as title, lead text, and name variations

derived from anchor text of incoming links, redirects, and disambiguations. This is the typical representation commonly used by entity linking methods such as TagMe [15]. The graph structure is derived only from bi-directional entity links.
- **Section:** Sections (top-level) of a Wikipedia pages as a representation of topical entity aspects, which include heading and section content, as well as page title and lead text. For the graph, all outgoing entity links are used.
- **Paragraph:** Paragraphs from the corpus with full text and entity links preserved. The graph structure is derived from entity links.

In this work use the TREC CAR benchmark. We derive page, entity, and section from the allButBenchmark data (omitting query pages) and derive paragraph data from the paragraphCorpus.

4.1 Entity and Context-Relevance Features

For each representation of **page**, **entity**, **section**, **paragraph**, we create an full-text search index with a single text field. Using this index and the keyword query (e.g., the page title or concatenation of headings), we use the following retrieval models to produce a ranking.

- **BM25:** The Lucene-BM25 model with default parameters without expansion.
- **BM25-RM:** A BM25 ranking with RM3-style query expansion on a BM25 feedback run.
- **QL:** The Querylikelihood model with Dirichlet smoothing ($\mu = 1500$).
- **QL-RM:** QL ranking with RM3-style query expansion on a QL feedback run.

We use a fixed interpolation for RM variations for input runs: query terms weighted by 1.0; expansion terms weighted by expansion probability. We learn a refined interpolation between QL and QL-RM as part of an larger learning-to-rank-entities model.

Drawing inspiration from the entity context model described by Dalton et al. [11], we further include the following entity-expansion model: We represent a pseudo-relevance feedback run of contexts d as a bags-of-entities e. Using entity links instead of words, the relevance model [22] is used to compute expansion entities as in Equation 7.

$$p_{\mathrm{EcmX}}(e|q) = \sum_{d} p(d|q)p(e|d) \tag{7}$$

We use entity-expansion model in two variations:

- **EcmX:** A ranking of expansion entities ranked by their expansion probability $p(e|q)$.
- **EcmPsg:** Expanding BM25 or QL with top 20 expansion entities under $p(e|q)$ to retrieve a new ranking of contexts via an RM3-like combination of query term matches in text field and expansion entity matches in the entity link field.

When multiple rankings are to be combined, an effective alternative to learning to rank is unsupervised rank aggregation. All distinct items d across all rankings R are assigned a new aggregated rank score from reciprocal ranks $\sum_R \frac{1}{\mathrm{rank}(d)}$. We include aggregated rank features for entities and each context type:

- Entity feature **Aggr:** Rank aggregation across all entity rankings (i.e., rankings from page and entity index, and rankings with the EcmX expansion model).
- For each context type, **Aggr:** Rank aggregation across all context rankings of this type (paragraph, page, or section).

Feature vectors are derived from all of these rankings:

- **Entity relevance**: Features $\vec{f}$ are derived from rank scores. We use BM25, QL, BM25-RM, QL-RM, BM25-EcmPsg and QL-EcmPsg scores when retrieving from page and entity indexes in addition to the scores of the EcmX model on all representations.
- **Context relevance**: Features $\vec{g}$ use rank scores of BM25, QL, BM25-RM, QL-RM, BM25-EcmPsg, and QL-EcmPsg retrieval from the context representation (paragraph, section, and page).

The ENT Hypergraph is created from the top 1000 of all entity rankings and context rankings. As we use retrieval models that only assign positive retrieval scores, missing features are set to zero. Finally, Z-score normalization is applied.

4.2 Relation-typed Neighbor Features

We include neighbor features as described in Section 3.4.1 based on entity features described above. The relation type is based on the context-type (paragraph, page, or section) and the roles two entities play the context. Here we only use two roles, **Link** if the entity is mentioned in the context or **Owner** if the context is derived from the Wikipedia page of the entity. For relation types of a multi-edge (e_i, t_k, e_j) we include all combinations of context type following neighbor-relation types

- **Link-Link:** when both entities are mentioned in the same context (i.e., co-coupled nodes).
- **Owner-Link:** when entity e_j is linked in a context owned by entity e_i (and vice versa, **Link-Owner**).
- **Owner-Self:** modelling loops of an entity with itself through the context.

Owner roles are not available for paragraph contexts, as these are derived from the paragraphCorpus of the CAR data set.

5 EVALUATION ON COMPLEX ANSWER RETRIEVAL

The TREC Complex Answer Retrieval track (CAR), hosted by NIST, aims to support users who seek a comprehensive answer in response to a topical keyword query. The track targets a scenario where a suitable answer needs to cover a range of backgrounds and context of the answer. In this light, a short "yes", a single sentence, or a single entity are not desired answers. While a short answer can often be effectively found through keyword matches, it is rather difficult to identify a large set of entities that are sufficiently relevant to be mentioned in the populated outline.

The CAR dataset [13] comes with a large collection of about 5.41 million (permitted) Wikipedia pages in easily accessible format, which we use as a collection of Wikipedia pages.[3] Additionally, a large corpus of paragraphs with hyperlinks to Wikipedia pages is provided.

[3]Resource "unprocessedAllButBenchmark", available at http://trec-car.cs.unh.edu

CAR queries are hierarchical page outlines that consist of a page title and headings. These outlines are to be populated with passages from a paragraph corpus and/or entities from a provided Wikipedia dump (pages of test queries are removed). We focus on the entity retrieval task. Two kinds of relevance data are provided, automatic and manual. *Automatic relevance data* is created synthetically from the Wikipedia page that corresponds to the query (those pages are not included in the dump). For the entity retrieval task, an entity is defined as relevant if the corresponding Wikipedia page contains a hyperlink to the entity. Synthetic relevance data is complemented by *manual assessments* conducted by NIST using pool-based evaluation. We make use of the following subsets provided in the TREC CAR v2.1 data release.[4]

- BenchmarkY1train-auto: 117 title queries, 1,816 title-heading queries, and 13,031 automatic entity assessments.
- BenchmarkY2test-manual: 271 title-heading queries, and 8,415 manual entity assessments.
- BenchmarkY2test-auto: 976 title-heading queries and 17,044 automatic entity assessments.

List of experiments. While the goal of this paper is to retrieve entities in response to short title queries, we evaluate our ENT Rank model both on title queries and title-heading queries for which official baselines are available. Following the track guidelines, we always train on the benchmarkY1train queries. In the **page-level experiment** we train/test on title queries from benchmarkY1train-auto using 5-fold cross validation. To compare to the state-of-the-art in CAR, we conduct a **section-level experiment** trained title-heading queries and qrels from benchmarkY1train-auto and evaluated on benchmarkY2test-manual and benchmarkY2test-auto. The keyword query in the section-level experiment is formed by concatenating the title, the heading, and parent headings of the section. We complement the experiments with a **study** of the running example "Zika fever", before continuing with experiments on DBpedia-Entity in Section 6.

We evaluate resulting entity rankings by metrics R-Precision (Rprec), Mean-average Precision (MAP), Normalized Discounted Cumulative Gains (ndcg@10 and ndcg@100); conducting significance testing with paired-t-tests. More results available in the online appendix.

Experimental Setup. To carry out these experiments, full-text indexes and rankings were created with Lucene 7, using the English analyzer for tokenization of text and whitespace tokenization for entity ids. (Using the standard analyzer on text suffers from a 50% performance loss.)

We apply our mini-batched coordinate ascent learning-to-rank-entities algorithm (Section 3.3) to optimize parameter vectors $\vec{\psi}$ and $\vec{\theta}$ across all queries to achieve the best mean-avg precision ranking performance. We use a mini-batch size of 150 queries. We use five random restarts of which we choose the model with the best evaluation score on the training folds. Rankings are predicted on the remaining data for each fold, then concatenated for evaluation.

[4] http://trec-car.cs.unh.edu/datareleases/v2.1/

Table 1: Page-level results on benchmarkY1train title queries measured in MAP. Comparison of feature subsets and context types: paragraph, section, and page. Significantly higher$^{\triangle}$or lower$^{\triangledown}$than AllExp (*) according to 1% (5%) paired-t-test.

Run	Paragraph	Section	Page
*AllExp	0.311*	**0.291***	**0.287***
JustAggr	0.274$^{\triangledown}$	0.158$^{\triangledown}$	0.211$^{\triangledown}$
No Entity	0.280$^{\triangledown}$	0.156$^{\triangledown}$	0.235$^{\triangledown}$
No Neighbor	**0.318**	0.278$^{\triangledown}$	0.274
No Context	0.299	0.280$^{\triangledown}$	0.275$^{\triangledown}$
Only Entity	0.287	0.287	0.282
ExpEcm	0.226$^{\triangledown}$	0.220$^{\triangledown}$	0.260$^{\triangledown}$
No Expansion	0.227$^{\triangledown}$	0.148$^{\triangledown}$	0.113$^{\triangledown}$

Next, feature vectors $\vec{f}$ for nodes and $\vec{g}$ for binarized edges are constructed using retrieval models as detailed in Section 4. We apply Z-score normalization to all feature vectors during training, which is inverted to obtain graph visualizations.

5.1 Page-level Experiment on TREC CAR

We study the advantage of the ENT learning-to-rank-entities model with respect to the features described in Section 4. We analyze the retrieval performance achieved on the following subsets of features described:

(1) **AllExp:** Use all described features.
(2) **JustAggr:** Combining multiple entity and context rankings with unsupervised rank aggregation. The relative weight between different context rankings and entity features needs to be trained.
(3) **No Entity**: All entity features were excluded.
(4) **No Neighbor**: All neighbor features were excluded.
(5) **No Context**: All context-relevance features were excluded (neighbor features are not affected).
(6) **Only Entity:** Only entity features are included.
(7) **ExpEcmX:** Like AllExp, but only rankings from EcmX are included.
(8) **No Expansion:** Like AllExp, but only BM25 and QL rankings without expansion are included.

For comparison, the strongest input entity retrieval feature is QL EcmX on the Paragraph index, with MAP of 0.21. Inspecting the trained model parameters, we find that this feature also receives one of the highest weights among EcmX entity features. One of the strongest edge features is QL without expansion. This is interesting, because the EcmX entity feature is equivalent to using QL edge features in the ENT Rank framework with Owner-Self roles. This can be seen when Equation 6 is inserted into Equations 1 and 2. Of course, ENT Rank is a more flexible and powerful model as will be demontrated in the evaluation on the DBpedia v2 dataset in Section 6.

Table 1 displays the ranking performance in MAP across different feature subsets and context types. The best performance is achieved for paragraph contexts with all features included. Neither

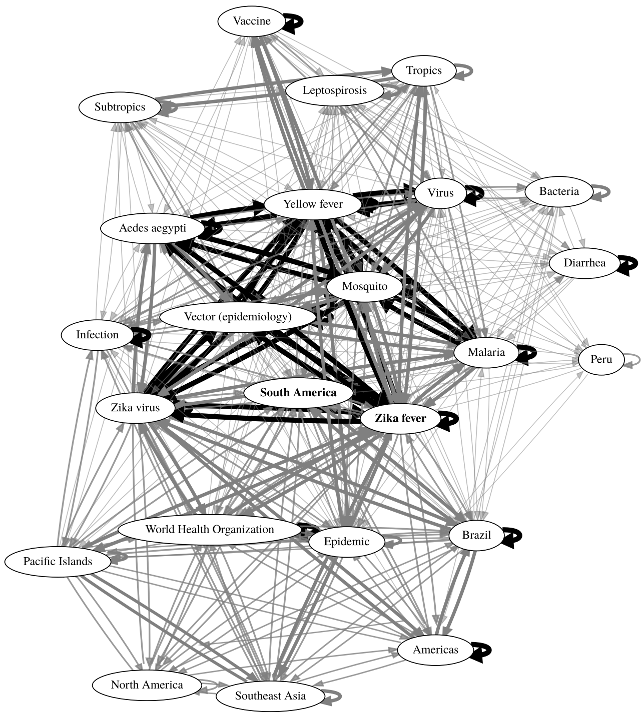

Figure 3: The 2-hop neighbor relation graph for example query "Zika fever" and entity South America. Edge weights are predicted with the ENT Rank model using paragraph contexts. The graph was not manually cleaned.

section nor page contexts are not significantly improving over entity features alone. We conclude that large contexts, such as pages, are not effective to model neighbor relations. Generally the inclusion of more features, as in AllExp, does not hurt. For paragraph contexts, the lower values in No Entity, No Neighbor, and No Context demonstrate that all components of the ENT Rank model provide value. Lower score of JustAggr demonstrates the benefits of machine learning.

The best variant of ENT Rank, AllExp on paragraph contexts achieves Rprec of 0.356 and ndcg@100 of 0.674.

While excluded for brevity, similar results are also obtained for title-queries in benchmarkY1test and benchmarkY2test.

5.2 Case Study: Zika fever

We demonstrate the algorithm by analyzing the results for our motivating example query "Zika fever".

Figure 3 displays the 2-hop neighborhood relation graph for the example query entity "Zika virus" and entity South America. Edge weights are predicted with the ENT Rank model on paragraph contexts with the AllExp subset. The resulting graph includes many topical connections between South America and Infection, World Health Organization, and Mosquito.

We want to remark that the knowledge base provided with TREC CAR v2.1 does not include the page Zika fever (since test queries are held out). Furthermore the Wikipedia page of South America

Table 2: Rank at which the target entity South America is found for example query "Zika fever". As the query terms are not mentioned on the target's Wikipedia page, it is not in runs with RM, EcmPsg, or no expansion.

Input ranking	rank	ENT Rank	rank
Paragraph BM25 EcmX	55	ExpEcmX	**49**
Page BM25 EcmX	102	Only Entity	66
Section BM25 EcmX	119	AllExp	86
Entity BM25 EcmX	146	JustAggr	109

does not mention the connection to Zika fever. Therefore, all connections in the ENT graph in Figure 3 are identified through contexts, neighbor relations, and EcmX features.

Table 2 displays the rank at which South America can be found in different input rankings (Table 2, left). The ENT Rank models (right) place South America at an even higher rank than any of the input rankings, demonstrating that the model can successfully incorporate different information sources even in challenging cases.

5.3 Section-level Experiment on TREC CAR

To compare ENT Rank to baseline systems from the TREC CAR challenge, we train ENT Rank models on title-heading queries from benchmarkY1train, then predict entities for benchmarkY2test outlines. Table 3 compares the two best ENT Rank variants with the two best entity retrieval systems from TREC CAR, "UNH-e-L2R" and "UNH-e-mixed".[5] ENT Rank either outperforms or is equivalent to the CAR baseline systems. We want to remark, that ENT Rank runs did not contribute to the pool for manual assessments, giving baseline systems a slight advantage.

6 EVALUATION ON DBPEDIA-ENTITY V2

We further evaluate our approach on several established entity retrieval datasets, provided in the DBpedia-Entity v2 benchmark [20]. The benchmark includes the following categories of queries with updated relevance judgments using a pool of methods:

SemSearch ES are short and ambiguous named entity queries. 113 queries such as "brooklyn bridge".

INEX-LD are IR-style keyword queries for linked data. 99 queries such as "electronic music genres".

List Search contain list search queries. 115 queries such as "Professional sports teams in Philadelphia".

QALD-2 is comprised of questions for linked data. 140 queries such as "Who is the mayor of Berlin?". Question-specific stopwords were removed by Hasibi et al.

The benchmark is designed for the English part of DBpedia from October 2015. As our algorithm makes heavy use of the Wikipedia article structure (paragraphs, sections, entity links in addition to meta data), we project the DBpedia-Entity v2 benchmark onto the Wikipedia dataset provided with TREC CAR (English part from December 2016). Only 2% of assessed entities could not be aligned because of page re-organizations. Since our method did not contribute to the assessment pool, it retrieves many unjudged documents. To enable a fair comparison, unjudged entries are removed

[5] Available at http://trec-car.cs.unh.edu/results/trec-car-y2-appendix.html

Table 3: Comparison of section-level retrieval on TREC CAR benchmarkY2test between best performing ENT Rank variants in comparison to two best entity retrieval baseline systems. Significantly higher △or lower ▽according to 5% paired-t-test.

	Automatic				Manual			
	MAP	Rprec	ndcg@10	ndcg@100	MAP	Rprec	ndcg@10	ndcg@100
⋆CAR Rank 1: UNH-e-L2R	0.146⋆	0.181⋆	0.258⋆	0.316⋆	0.310⋆	0.315⋆	**0.453**⋆	0.514⋆
CAR Rank 2: UNH-e-mixed	0.142	0.175	**0.275**△	0.298▽	0.260▽	0.278▽	0.386▽	0.435▽
ENT Rank AllExp	0.136▽	0.161▽	0.239▽	0.391△	0.276▽	0.275▽	0.395▽	0.538△
ENT Rank ExpEcm	0.156△	0.180	0.254▽	0.427△	0.307	0.304▽	0.443▽	0.578△
ENT Rank JustAggr	**0.161**△	**0.186**	0.270△	**0.428**△	0.322	0.312	0.443	**0.592**△
ENT Rank No Expansion	0.152△	0.179	0.255	0.416△	**0.323**△	**0.317**	0.448	0.590△

from the ranking. The benchmark also provides contributed baseline runs. These were also projected onto the 2016 Wikipedia dump, and likewise unjudged entities were removed to obtain a fair comparison (obtaining different evaluation results than described on the benchmark's web page).

Datasets are merged for training with 5-fold cross validation. Table 4 displays the result of our suggested ENT Rank model in comparison to the best-performing baselines BM25F-CA and FSDM-ELR. With the exception of the SemSearch ES subset, our ENT Rank method outperforms all twelve baseline systems. ENT Rank especially improves on recall-oriented measures MAP and ndcg@100.

Inspecting the feature weights reveal that—in comparison to complex answer retrieval–these datasets require that weight is placed on entity features. Restricting the features to the ExpEcmX subset does drastically hurt the performance. In contrast, limiting features to only access un-expanded BM25 and QL runs, obtains relatively good results. When entity features are removed, ENT rank increases the weight of neighbor features, thereby practically recovering a retrieval performance of 0.671 ndcg@100.

7 CONCLUSION

We propose ENT Rank, a framework for modeling entity-neighbor-text relations for entity retrieval. While ENT Rank can incorporate a wide range of context, neighbor, and entity features, here we focus on features that are derived from traditional text retrieval methods, such as BM25, and neighbor relations that are based on co-occuring entity links. We explore different sizes of contexts and find that paragraph-sized contexts work best.

The approach is evaluated through several experiments on the TREC Complex Answer Retrieval and DBpedia-Entity v2 benchmarks which include title-heading queries, semantic search queries, and question answering queries. ENT Rank is consistently the best or second-best method among a set of eleven baseline systems that participated in TREC CAR and twelve systems from DBpedia-Entity. A case study on the running example "Zika fever" demonstrates the ability to detect relevant entities even when their relevance cannot be concluded from their Wikipedia page alone.

In future, we would like to use ENT Rank to not only rank entities, but also to provide a useful order among entities and support them with text. Such a system could support both human article authors and automated conversational agents with background knowledge. One day, such systems might respond to web search requests with automatically written Wikipedia articles that do not exist yet.

Table 4: Results of ENT Rank on the DBpedia-Entity v2 dataset. Baselines BM25F-CA and FSDM+ELR [20]. Significantly higher△or lower▽than BM25F-CA (⋆) baseline according to 5% paired-t-test.

All	MAP	Rprec	ndcg@100	ndcg@10
⋆BM25F-CA	0.454⋆	0.433⋆	0.680⋆	**0.545**⋆
FSDM-ELR	0.440▽	0.416▽	0.663▽	0.537
ENT Rank AllExp	0.465	0.430	0.702△	0.536
ENT Rank JustAggr	**0.476** △	**0.438**	**0.711**△	0.544
SemSearch_ES				
⋆BM25F-CA	0.606⋆	0.549⋆	0.782⋆	0.671⋆
FSDM-ELR	**0.620**	**0.550**	**0.791**	**0.694**
ENT Rank AllExp	0.601	0.532	0.783	0.666
ENT Rank JustAggr	0.590	0.506▽	0.779	0.658
ListSearch				
⋆BM25F-CA	0.441⋆	0.427⋆	0.689⋆	**0.550**⋆
FSDM-ELR	0.422	0.404	0.665▽	0.533
ENT Rank AllExp	0.478△	0.450	0.733△	0.542
ENT Rank JustAggr	**0.493**△	**0.471**△	**0.744**△	0.549
INEX_LD				
⋆BM25F-CA	0.420⋆	0.414⋆	0.666⋆	0.525⋆
FSDM-ELR	0.399	0.395	0.645	0.511
ENT Rank AllExp	0.437	0.412	0.693	0.520
ENT Rank JustAggr	0.439	0.422	0.696	0.519
ENT Rank Only Entity	**0.443**	**0.425**	**0.702**△	**0.532**
QALD2				
⋆BM25F-CA	0.366⋆	0.359⋆	0.600⋆	0.455⋆
FSDM-ELR	0.339	0.332	0.572▽	0.432
ENT Rank AllExp	0.366	0.346	0.618	0.439
ENT Rank JustAggr	**0.396**	**0.366**	**0.639**△	**0.465**

Acknowledgements

This material is based upon work supported by the National Science Foundation under Grant No. 1846017. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

[1] James Allan, Bruce Croft, Alistair Moffat, and Mark Sanderson. 2012. Frontiers, challenges, and opportunities for information retrieval: Report from SWIRL 2012 the second strategic workshop on information retrieval in Lorne. In *ACM SIGIR Forum*, Vol. 46. ACM, 2–32.

[2] Krisztian Balog, Marc Bron, and Maarten De Rijke. 2011. Query modeling for entity search based on terms, categories, and examples. *ACM Transactions on Information Systems (TOIS)* 29, 4 (2011), 22.

[3] Krisztian Balog, Edgar Meij, and Maarten De Rijke. 2010. Entity search: building bridges between two worlds. In *Proceedings of the 3rd International Semantic Search Workshop*. 9.

[4] Krisztian Balog, Pavel Serdyuko, and Arjen de Vries. 2011. A neighborhood relevance model for entity linking. In *TREC*.

[5] Krisztian Balog, Pavel Serdyuko, Arjen de Vries, Paul Thomas, and Thijs Westerveld. 2009. Overview of the TREC 2009 Entity Track. In *TREC*.

[6] Holger Bast, Alexandru Chitea, Fabian Suchanek, and Ingmar Weber. 2007. Ester: efficient search on text, entities, and relations. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 671–678.

[7] Hannah Bast and Elmar Haussmann. 2015. More accurate question answering on freebase. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. 1431–1440.

[8] Michael Bendersky, Donald Metzler, and W Bruce Croft. 2010. Learning concept importance using a weighted dependence model. In *Proceedings of the third ACM international conference on Web search and data mining*. ACM, 31–40.

[9] Jing Chen, Chenyan Xiong, and Jamie Callan. 2016. An empirical study of learning to rank for entity search. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 737–740.

[10] Jeffrey Dalton and Laura Dietz. 2013. A Neighborhood Relevance Model for Entity Linking. In *Proceedings of the 10th Conference on Open Research Areas in Information Retrieval*. 149–156.

[11] Jeffrey Dalton, Laura Dietz, and James Allan. 2014. Entity Query Feature Expansion Using Knowledge Base Links. In *SIGIR*.

[12] Gianluca Demartini, Tereza Iofciu, and Arjen P De Vries. 2009. Overview of the INEX 2009 entity ranking track. In *International Workshop of the Initiative for the Evaluation of XML Retrieval*. Springer, 254–264.

[13] Laura Dietz and Ben Gamari. 2018. TREC CAR 2.0: A Data Set for Complex Answer Retrieval. http://trec-car.cs.unh.edu. Version 2.0.

[14] Laura Dietz, Ben Gamari, Jeff Dalton, and Nick Craswell. 2018. TREC Complex Answer Retrieval Overview. TREC.

[15] Paolo Ferragina and Ugo Scaiella. 2010. Tagme: on-the-fly annotation of short text fragments (by wikipedia entities). In *Proc. of CIKM*. 1625–1628.

[16] Darío Garigliotti and Krisztian Balog. 2017. On Type-Aware Entity Retrieval. In *SIGIR*. 27–34.

[17] Faegheh Hasibi, Krisztian Balog, and Svein Erik Bratsberg. 2015. Entity Linking in Queries: Tasks and Evaluation. In *ICTIR*. 171–180.

[18] Faegheh Hasibi, Krisztian Balog, and Svein Erik Bratsberg. 2016. Exploiting Entity Linking in Queries for Entity Retrieval. In *ICTIR*. 209–218.

[19] Faegheh Hasibi, Krisztian Balog, Darío Garigliotti, and Shuo Zhang. 2017. Nordlys: A toolkit for entity-oriented and semantic search. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 1289–1292.

[20] Faegheh Hasibi, Fedor Nikolaev, Chenyan Xiong, Krisztian Balog, Svein Erik Bratsberg, Alexander Kotov, and Jamie Callan. 2017. DBpedia-entity v2: a test collection for entity search. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 1265–1268.

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

[22] Victor Lavrenko and W Bruce Croft. 2001. Relevance based language models. In *Proc. of SIGIR-01*. 120–127.

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

[24] Edgar Meij, Marc Bron, Laura Hollink, Bouke Huurnink, and Maarten de Rijke. 2011. Mapping queries to the Linking Open Data cloud: A case study using DBpedia. *Web Semantics: Science, Services and Agents on the World Wide Web* 9, 4 (2011), 418–433.

[25] Edgar Meij, Wouter Weerkamp, and Maarten De Rijke. 2012. Adding semantics to microblog posts. In *Proceedings of the fifth ACM international conference on Web search and data mining*. ACM, 563–572.

[26] Donald Metzler and W Bruce Croft. 2005. A Markov random field model for term dependencies. In *Proc. of SIGIR-05*. 472–479.

[27] Fedor Nikolaev, Alexander Kotov, and Nikita Zhiltsov. 2016. Parameterized Fielded Term Dependence Models for Ad-hoc Entity Retrieval from Knowledge Graph. In *SIGIR*.

[28] Jeffrey Pound, Peter Mika, and Hugo Zaragoza. 2010. Ad-hoc object retrieval in the web of data. In *Proceedings of the 19th international conference on World wide web (WWW2010)*. ACM, 771–780.

[29] Hadas Raviv, David Carmel, and Oren Kurland. 2012. A ranking framework for entity oriented search using Markov random fields. In *Proceedings of the 1st Joint International Workshop on Entity-Oriented and Semantic Search*. ACM, 1.

[30] Hadas Raviv, Oren Kurland, and David Carmel. 2016. Document Retrieval Using Entity-Based Language Models. In *SIGIR*.

[31] Michael Schuhmacher, Laura Dietz, and Simone Paolo Ponzetto. 2015. Ranking Entities for Web Queries Through Text and Knowledge. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management (CIKM 2015)*. ACM, 1461–1470.

[32] Hanghang Tong, Christos Faloutsos, and Jia-Yu Pan. 2006. Fast random walk with restart and its applications. In *Sixth International Conference on Data Mining (ICDM'06)*. IEEE, 613–622.

[33] Alberto Tonon, Gianluca Demartini, and Philippe Cudré-Mauroux. 2012. Combining Inverted Indices and Structured Search for Ad-hoc Object Retrieval. In *SIGIR*.

[34] Fei Wang and Changshui Zhang. 2008. Label Propagation through Linear Neighborhoods. *IEEE Transactions on Knowledge and Data Engineering* 1, 20 (2008), 55–67.

[35] Chenyan Xiong, Jamie Callan, and Tie-Yan Liu. 2017. Word-entity duet representations for document ranking. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 763–772.

[36] Nikita Zhiltsov, Alexander Kotov, and Fedor Nikolaev. 2015. Fielded Sequential Dependence Model for Ad-Hoc Entity Retrieval in the Web of Data. In *SIGIR*. 253–262.

[37] Denny Zhou, Jiayuan Huang, and Bernhard Schölkopf. 2007. Learning with hypergraphs: Clustering, classification, and embedding. In *Advances in neural information processing systems*. 1601–1608.