Information Retrieval Research 1 (2025) 137-165 Submitted 11/24; Revised 04/25; Published 06/25

Graph Embeddings to Empower Entity Retrieval

Emma J. Gerritse EMMA.GERRITSE@RU.NL
Radboud University
Nijmegen, The Netherlands

Faegheh Hasibi FAEGHEH.HASIBI@RU.NL
Radboud University
Nijmegen, The Netherlands

Arjen P. de Vries ARJEN.DEVRIES@RU.NL
Radboud University
Nijmegen, The Netherlands

Editor: Negin Rahimi, Makoto Kato

Abstract

In this research, we investigate methods for entity retrieval using graph embeddings. While various methods have been proposed over the years, most utilize a single graph embedding and entity linking approach. This hinders our understanding of how different graph embedding and entity linking methods impact entity retrieval. To address this gap, we investigate the effects of three different categories of graph embedding techniques and five different entity linking methods. We perform a reranking of entities using the distance between the embeddings of annotated entities and the entities we wish to rerank. We conclude that the selection of both graph embeddings and entity linkers significantly impacts the effectiveness of entity retrieval. For graph embeddings, methods that incorporate both graph structure and textual descriptions of entities are the most effective. For entity linking, both precision and recall concerning concepts are important for optimal retrieval performance. Additionally, it is essential for the graph to encompass as many entities as possible.

Keywords: Entity retrieval, Knowledge Graph Embeddings, Word Embeddings

1 Introduction

A significant portion of user queries in web search and question answering explicitly or implicitly reference entities, with users either asking for some entity-related facts or looking for the page of a specific entity (Balog, 2018; Meij et al., 2014). Entities are a vital part of information retrieval research, and this has led to the development of datasets (Hasibi et al., 2017b) and methods for effective entity retrieval (Arabzadeh et al., 2024; Gerritse et al., 2022; Chatterjee and Dietz, 2022; Gerritse et al., 2020; Dietz, 2019; Garigliotti et al., 2019). Entities are commonly stored in a Knowledge Graph (KG), which connects them through various relationships. The entire knowledge graph can then be represented using graph embeddings, which capture the rich and human-curated information in low-dimensional vector spaces, representing the similarity and relations between different entities. Popularized by methods such as Trans-E (Bordes et al., 2013) and Wikipedia2Vec (Yamada et al., 2020), graph embeddings have proven to be useful for tasks like link prediction (Bordes et al., 2013) and text classification (Yamada and Shindo, 2019).

 WWW: https://irrj.org DOI: 10.54195/irrj.19877

This study investigates the use of KG embeddings to enhance lexical entity retrieval models. While recent transformer-based entity retrieval models have demonstrated superior performance compared to KG-augmented lexical models (Chatterjee and Dietz, 2022; Gerritse et al., 2022), they still struggle with long-tail entities (Gerritse et al., 2022). Similarly, dense retrievers face challenges in generalizing to the entity retrieval task (Kamalloo et al., 2024), performing worse than or only on par with strong lexical retrieval models such as BM25F or GEEER (Gerritse et al., 2020). These findings are in line with recent studies that show LLMs struggle to answer factual questions about long tail entities, and Retrieval Augmented Generation (RAG) needs to be employed to fill this gap (Mallen et al., 2023; Soudani et al., 2024).

Various methods have been proposed to include graph embeddings for entity retrieval. Gerritse et al. (2020) and Nikolaev and Kotov (2020) introduce methods using the similarity between the graph embedding vectors to compute relevance between queries and entities. These fairly simple methods improve over the previously established state of the art. However, existing approaches for utilizing graph embeddings for entity retrieval typically employ a single entity linker and embedding methods. Therefore, the effect of different entity linking and graph embedding methods on overall entity retrieval performance has remained unexplored.

In this study, we aim to fill this gap and explore various aspects of utilizing knowledge graph embeddings for entity retrieval. We base our approach on the method proposed by Gerritse et al. (2020), an early work on the use of graph embeddings for entity retrieval that has been reproduced and extended in various setups (Oza and Dietz, 2023; Chatterjee and Dietz, 2022; Jafarzadeh et al., 2022; Daza et al., 2021). Additionally, it provides a simple framework for the fusion of graph embeddings and retrieval models, enabling us to understand the effect of different entity linking and embedding methods on the entity retrieval task.

We tackle the following research questions in this work:

RQ 1. *How do different entity linking methods and their specific properties affect graph embedding–empowered entity retrieval?* Different entity-linking methods have been introduced throughout the years. Some take into account all different scenarios that might be meant by a query, for example, considering all other meanings of homonyms and polysemes, while others consider the most probable interpretation of a query and map each entity mention to a single entity. Additionally, entity linking methods such as REL (Van Hulst et al., 2020) and Nordlys (Hasibi et al., 2017a) provide high precision of named entities like people, location, and organizations. In contrast, methods like SMAPH (Cornolti et al., 2018) and TagMe (Ferragina and Scaiella, 2010) are designed to annotate both named entities and general concepts, such as Democracy. We hypothesize that when using graph embeddings for entity retrieval, it is beneficial to include as many entity embeddings relevant to the query as possible, encompassing both concepts and name entities. To assess this hypothesis, we manually annotate queries of the DBpedia-Entity collection (Hasibi et al., 2017b), and compare different entity linkers using this ground truth. We find that, indeed, the best entity retrieval performance is achieved with entity linkers that annotate both concepts and name entities.

RQ 2. *Which graph embedding techniques work best for entity retrieval methods empowered by graph embeddings?* In addition to diverse approaches proposed for entity linking, numerous algorithms have been introduced for graph embeddings. This work classifies them into three distinct groups, namely skip-gram-based (Yamada et al., 2016), transition-based (Trouillon et al., 2016), and random-walk-based (Ristoski and Paulheim, 2016). Often, research in the field of information retrieval only utilizes one type of graph embedding without considering the other classes of methods. In this work, we compare these various methods. We find that Wikipedia2Vec, as a skip-gram-based method, has the highest performance of all methods, provided that much effort is made to correct missing entities. We also see a positive impact from including as many entities as possible in the graph embedding for entity retrieval, even if it significantly increases the graph size during embedding training.

RQ 3. *How does the structural information captured by the skip-gram–based graph embedding approach, Wikipedia2Vec, contribute to entity retrieval effectiveness?* To address this research question, we train two versions of Wikipedia2Vec embeddings, with and without link graph, and compare the obtained embeddings and retrieval results. Utilizing the cluster hypothesis (Rijsbergen, 1979), we show that a representation of the graph structure in the embeddings leads to better clusters and higher effectiveness of retrieval results. We further see that queries with correctly linked entities (by an entity linker) are helped the most, while queries with wrongly linked entities are helped the least.

This work is an extension of (Gerritse et al., 2020), which makes the following new contributions: (i) We provide a comprehensive investigation of different entity linking and graph embedding methods for entity retrieval, (ii) We provide a new set of entity annotations of the widely-used DBpedia-Entity collection that includes both concepts and named entities. All the resources developed in the course of this study are made publicly available at: `https://github.com/informagi/GEEER`.

2 Related Work

2.1 Word and Entity Embeddings

Distributional representations of language have been the object of study for many years in natural language processing (NLP), because of their promise to represent words not in isolation, but semantically, with their immediate context. Algorithms like Word2Vec (Mikolov et al., 2013b) and Glove (Pennington et al., 2014) construct a vector space of word domains where similar words are mapped together (based on their linguistic context). Word2Vec embeddings are extracted from neural networks that predict words based on their context (continuous bag of words) or that predict the context for a given word (skip-gram). These word-embedding representations have proven to be highly effective in various Information Retrieval (IR) and NLP tasks.

Word embeddings have been shown to improve effectiveness in document retrieval (Dehghani et al., 2017; Diaz et al., 2016). In (Diaz et al., 2016), locally trained word embeddings are used for query expansion. Here, queries are expanded with terms highly similar to the query, and it is shown that this method beats several other neural methods. In (Dehghani et al., 2017), embeddings are used to train a neural ranking model using weak supervision.

The authors use query embeddings and document embeddings to predict relevance between queries and documents when given BM25 scores as labels, outperforming BM25.

Word embeddings capture the immediate linguistic context of word occurrences. Going beyond the text itself, researchers across various research communities have proposed a vast number of Knowledge Graph (KG) embedding methods. A KG is a graph where the nodes represent entities, and the edges represent the relations between them. One of the earliest and most well-known KG embedding methods is TransE (Bordes et al., 2013), which falls under the transition-based category. The TransE method considers graph edges as *(head, label, tail)* triples, where *label* is the value of the edge. Under the objective that adding graph embedding vectors of the *head* and the *label* should result in the vector of the *tail*, these embeddings are learned by gradient descent. TransE has been influential but has proved to be ineffective for anti-symmetric relations present in the knowledge graph. This resulted in various proposals for KG embeddings that extend this method with additional objectives, such as DistMult (Yang et al., 2015), ComplEx (Trouillon et al., 2016), and SimplE (Kazemi and Poole, 2018).

ComplEx (Trouillon et al., 2016) is a robust and widely used graph embedding, which utilizes complex vectors instead of real vectors and is better suited to represent anti-symmetric relations. Several studies (Ruffinelli et al., 2020; Chekalina et al., 2022; Kochsiek et al., 2023) have demonstrated that ComplEx achieves competitive performance on the Wikidata5M dataset(Wang et al., 2021). In particular, Ruffinelli et al. (2020) performed parameter tuning of ComplEx on the Wikidata5M dataset[1], and showed that when training conditions are standardized, models such as TransE, ComplEx, and DistMult perform similarly.

Another class of KG embeddings that has been widely applied in downstream tasks (Oza and Dietz, 2023; Gerritse et al., 2022), without direct comparison with translation-based embeddings in the literature, is skip-gram-based approaches, such as Deepwalk (Perozzi et al., 2014) and RDF2Vec (Ristoski and Paulheim, 2016). Deepwalk (Perozzi et al., 2014) is a graph embedding method that expects non-labeled edges. It first randomly samples vertices from the graph as starting points, and then performs a random walk from each of these starting points. The vertices walked in these random walks can then be represented as sentences. After having created these sentences from the graph, Deepwalk uses these as input for a Word2Vec-based approach using the skip-gram method. RDF2Vec (Ristoski and Paulheim, 2016) extends Deepwalk to work on knowledge graphs, by not only using the vertices, but also the labels of the edges for creating the random walks.

Wikipedia2Vec (Yamada et al., 2016) is a skip-gram-based approach that applies graph embeddings to Wikipedia, creating embeddings that jointly capture link structure and text. The Wikipedia knowledge graph is a natural resource for training graph embeddings, considering that it represents entities in a graph of interlinked Wikipedia pages and their text. The method proposed in (Yamada et al., 2016) embeds words and entities in the same vector space using word and graph contexts. The word-word context is modeled using the Word2Vec approach, entity-entity context considers neighboring entities in the link graph, and word-entity context takes the words in the context of the anchor text that links to an

1. highlighted in the GitHub repository `https://github.com/uma-pi1/kge`, last accessed 20-05-2025

entity. The authors of Wikipedia2Vec demonstrate performance improvements on various NLP tasks, although they did not consider entity retrieval in their work.

Recent transformer-based methods utilize transformers to create vector representations of entities. KGT5 (Saxena et al., 2022) treats link prediction as a sequence to sequence (seq2seq) task using a T5 architecture. It uses the prediction of sequences in the form of "predict tail: `subject mention` | `relation mention` | ", and is trained on facts in the KG with the objective of generating the true answer using teacher forcing. The method achieves competitive scores on link prediction on large datasets like Wikidata5M. This work is improved upon in KGT5-context (Kochsiek et al., 2023), by adding extra context in the sequences, yielding even higher scores on link prediction on Wikidata5M. Li et al. (2024) introduces MoCoKGC, which utilizes three different transformer-based encoders to create its embedding. It separately trains an entity-relation encoder, an entity encoder, and a momentum-entity encoder, which provides more negative samples and allows the gradual updating of entity encodings. In this work, we study three widely used and competitive KG embedding models with reasonable computational demands: ComplEx, RDF2Vec, and Wikipedia2Vec.

2.2 Entity Linking

Methods incorporating entity information in NLP or information retrieval rely on entity linkers to identify entity mentions in the query or document. Entity linking refers to the process of detecting all possible mentions of entities and linking them to the corresponding identifier in a certain knowledge base (Balog, 2018). In this study, we employ the following five entity linking methods to annotate queries:

In Hasibi et al. (2015), the difference between entity linking in queries compared to full texts is discussed. Entity linking for queries is divided into two different tasks: The first is semantic mapping, which is finding a list of ranked entities similar to the queries. The second is interpretation finding, which finds sets of linked entities, representing possible interpretations, which can then be used for machine-understanding of the query. This work then also discusses evaluation methods for both tasks.

TagMe (Ferragina and Scaiella, 2010) is an on-the-fly entity linker specializing in short texts. It works in a three-step pipeline, parsing the query to make a candidate list of entities, then disambiguating. Afterward, it prunes entities with a low link probability/coherence to the other candidates. The REL entity linker (Van Hulst et al., 2020; Joko and Hasibi, 2022) is a popular open-source scalable entity linking toolkit (Kamphuis et al., 2022) for annotating various types of texts (e.g., documents and conversations) with entities. REL detects mentions using Flair, an NLP library that supports Named Entity Recognition (NER). REL performs candidate selection based on Wikipedia2Vec embeddings, and entity disambiguation based on latent relations between entity mentions in the text.

A number of entity linkers are designed for annotating queries. Nordlys (Hasibi et al., 2017a) uses the method created by Hasibi et al. (2015), which employs a learning-to-rank (LTR) model with various textual and semantic similarity features. SMAPH (Cornolti et al., 2018) utilizes a so-called piggybacking approach that uses information from a web search engine to link entities. It does this by first using the text, which needs to be linked as a query, and building a set of candidate entities, which are then filtered by linking the candidate entities to the terms in the input query. ELQ (Li et al., 2020) is a fast end-to-end

entity linking model specialized for questions using bi-encoders. It performs both mention detection and entity linking in one pass. Its architecture encodes queries and entities, then scores candidate entities through the inner product with the entity vectors.

2.3 Entity Retrieval

Knowledge Graphs like Wikipedia enrich the representation of entities by modeling the relations between them. Methods for ad-hoc document retrieval, such as BM25, have been applied successfully to retrieve entity information from knowledge graphs. However, since knowledge bases are semi-structured resources, this structural information may be used as well, for example, by viewing entities as fielded documents extracted from the knowledge graph. BM25F (Robertson et al., 2009) is a fielded retrieval model, where term frequencies between different fields in documents are normalized to the length of each field. Another effective model for entity retrieval is the Fielded Sequential Dependence Model (FSDM) (Zhiltsov et al., 2015), which estimates the probability of relevance using information from single terms and bi-grams, normalized per field.

Linking entities mentioned in the query to the knowledge graph allows relationships encoded in the knowledge graph to improve the estimation of the relevance of candidate entities. Previous work has shown that entity linking can indeed help increase effectiveness of entity retrieval. In (Hasibi et al., 2016), for example, entity retrieval has been combined with entity linking to improve retrieval effectiveness over state-of-the-art methods like FSDM.

Liu et al. (2019) is one of the early works applying graph embeddings to entity retrieval, which demonstrates consistent, albeit modest, improvements. KEWER (Nikolaev and Kotov, 2020) is another work that introduces a method for retrieving entities based on graph embeddings. It learns embeddings for entities and words based on TransE and annotates queries using SMAPH to re-rank entities, which improved on the previous state of the art. Similarly, Gerritse et al. (2020) first annotate queries using TagMe, and then use Wikipedia2Vec embeddings to compute the similarity between queries and documents. This method also improves on the previous state of the art.

Following the popularity of transformer-based methods, various works have introduced combinations of transformers with graph embeddings. In (Oza and Dietz, 2023), the work of Gerritse et al. (2020) is extended to include transformer-based entity embeddings. Daza et al. (2021) introduces a BERT architecture combined with TransE graph embeddings to re-rank entities. After encoding the queries and entities, it uses their similarity as a query score. In (Gerritse et al., 2022), a cross-encoder method is introduced, based on E-BERT (Poerner et al., 2020). It introduces entity tokens in the input layer, of which the encodings are based on Wikipedia2Vec embeddings. In (Tran and Yates, 2022), the BERT encoding of a query is combined with the Wikipedia2Vec embeddings of the annotated entities, which are aggregated using clusters. Last, Chatterjee and Dietz (2022) construct a method for entities without entity embeddings, by identifying the most relevant top-level sections from a Wikipedia page, depending on the query. These sections are then used to train a BERT model to represent the entities, which, in turn, are used as features in an LTR model for entity retrieval. While existing approaches typically examine only a single graph embedding and entity linking method, this work explores the effect of various graph embeddings and entity linkers on entity retrieval.

3 Embedding Based Entity Retrieval

In this section, we describe the graph embedding and entity retrieval approaches used in this paper. We utilize three different graph embedding methods that are representative of the three types of graph embeddings. The first is Wikipedia2Vec, which combines graph-based and text-based structures and learns embeddings using skip-gram algorithms. Next is RDF2Vec, which is based on random walks. Finally, we use ComplEx, which represents transition-based graph embeddings. We conclude by describing the methodology of our graph embedding based on the re-ranking algorithm.

3.1 Wikipedia2Vec

Taking a knowledge graph as the input, Wikipedia2Vec (Yamada et al., 2016, 2020) extends the skip-gram variant of Word2Vec (Mikolov et al., 2013b,a) and learns word and entity embeddings jointly for the Wikipedia knowledge graph. The objective function of this model is composed of three components. The first component infers optimal embeddings for words W in the corpus. Given a sequence of words $w_1 w_2 ... w_T$ and a context window of size c, the word-based objective function is:

$$\mathcal{L}_w = \sum_{t=1}^{T} \sum_{-c \leq j \leq c, j \neq 0} \log \frac{\exp(\mathbf{V}_{w_t}^{\intercal} \mathbf{U}_{w_{t+j}})}{\sum_{w \in W} \exp(\mathbf{V}_{w_t}^{\intercal} \mathbf{U}_{w})}, \tag{1}$$

where matrices $\mathbf{U}$ and $\mathbf{V}$ represent the input and output vector representations, deriving the final embeddings from matrix $\mathbf{V}$.

The two other components of the objective function take the knowledge graph into account. The first considers a link-based measure estimated from the knowledge graph (i.e., Wikipedia). This measure captures the relatedness between entities in the knowledge graph, based on the similarity between their incoming links:

$$\mathcal{L}_e = \sum_{e_i \in \mathcal{E}} \sum_{e_o \in C_{e_i}, e_i \neq e_o} \log \frac{\exp(\mathbf{V}_{e_i}^{\intercal} \mathbf{U}_{e_o})}{\sum_{e \in \mathcal{E}} \exp(\mathbf{V}_{e_i}^{\intercal} \mathbf{U}_{e})}, \tag{2}$$

where C_e denotes entities linked to an entity e, and $\mathcal{E}$ represents all entities in the knowledge graph. The last addition to the objective function places similar entities and words near each other by considering the context of the anchor text. The intuition is the same as in Word2Vec, but here, words in the vicinity of the anchor text are used to predict the entity. Considering a knowledge graph with hyperlinks A and an entity e, the goal is to predict the context words of the entity:

$$\mathcal{L}_a = \sum_{e_i \in A} \sum_{w_o \in a(e_i)} \log \frac{\exp(\mathbf{V}_{e_i}^{\intercal} \mathbf{U}_{w_o})}{\sum_{w \in W} \exp(\mathbf{V}_{e_i}^{\intercal} \mathbf{U}_{w})}, \tag{3}$$

where $a(e)$ gives the previous and next c words of the referent entity e.

These three components (word context, link structure, and anchor context) are then combined linearly into the following objective function:

$$\mathcal{L} = \mathcal{L}_w + \mathcal{L}_e + \mathcal{L}_a. \tag{4}$$

3.2 RDF2Vec

RDF2Vec (Ristoski and Paulheim, 2016) is a graph embedding method that takes a collection of triples represented in the Resource Description Framework (RDF) format and generates entity-relation sentences using random walks throughout the graph. These sentences are then used to compute Word2Vec-based embeddings. Suppose we have an RDF graph $G = (R, \mathcal{E})$ where R is a set of relations and $\mathcal{E}$ is a set of entities. RDF2Vec generates all paths P_e in the RDF of depth d for all entities $e \in \mathcal{E}$. These walks are generated using a breadth-first algorithm. First, for a starting entity e_s, the algorithm explores the direct outgoing relations $R(e_s)$. Of these edges, a path $e_s \rightarrow r_i$ is randomly selected, where $r_i \in R(e_s)$. Then, for each previously explored node, the algorithm will visit the connected vertices. This generates a path $e_s \rightarrow r_i \rightarrow e_i$. This will continue until d iterations are reached, in which d is a hyperparameter. After this, all paths $\cup_{e \in \mathcal{E}} P_e$ are entered as sentences into the SkipGram model, as is seen in equation 1. This will result in an embedding for all $e \in \mathcal{E}$, thus leading to an embedding space for G.

3.3 ComplEx

The TransE triple-based graph embeddings have been introduced in (Bordes et al., 2013). The intuition behind TransE is to construct embeddings for head-relation-tail triples. Given triples $< e_h, r, e_t >$, where $e_h, e_t \in \mathcal{E}$, $r \in R$, it minimalizes the vectors $\overrightarrow{e_h}, \overrightarrow{r}, \overrightarrow{e_t} \in \mathbb{R}^d$ with respect to the function $s(e_h, r, e_t) = ||\overrightarrow{e_h} + \overrightarrow{r} - \overrightarrow{e_t}||$. This results in an embedding space where the tail of the triple can be found using vector addition, i.e., $\overrightarrow{e_h} + \overrightarrow{r} = \overrightarrow{e_t}$. However, TransE cannot embed one-to-many and many-to-many relations properly. Multiple algorithms expanding on TransE have been introduced, one of which is ComplEx (Trouillon et al., 2016), where entities are embedded in the complex field. The intuition for utilizing complex vectors is to capture the many anti-symmetric relations in knowledge graphs more effectively than TransE, and this makes the computations arguably simpler since they use only the Hermitian dot product.

ComplEx first initializes random embeddings $\overrightarrow{e_h}, \overrightarrow{r}, \overrightarrow{e_t} \in \mathbb{C}^d$ for $e_h, e_t \in \mathcal{E}$, $r \in R$. Then, for all training triples, the following scoring function $S(\overrightarrow{e_h}, \overrightarrow{r}, \overrightarrow{e_t})$ is minimized:

$$S(e_h, r, e_t) = \text{Re}(\overrightarrow{e_h}^T \text{ diag}(\overrightarrow{r}) \overrightarrow{e_t})$$

where Re() is the function that maps a complex vector to its real part, and diag() the function that maps a vector of size d to a matrix with dimensions $d \times d$, that has the input vector as diagonal and all other elements are 0.

We use the setup for sampling and loss computation described in (Ruffinelli et al., 2020). During training time, both positive and negative examples are presented, where the negative samples are obtained by randomly perturbing one of the head, relation, or tail entities. The loss over these positive and negative samples is computed as follows: first, apply the sigmoid function on $S(e_h, r, e_t)$, then compute cross-entropy loss over the resulting value and the label of that triple.

3.4 Re-ranking Entities

In this section, we discuss the method of using these graph embeddings in the setting of entity retrieval. We propose a two-stage ranking model, where we first produce a ranking

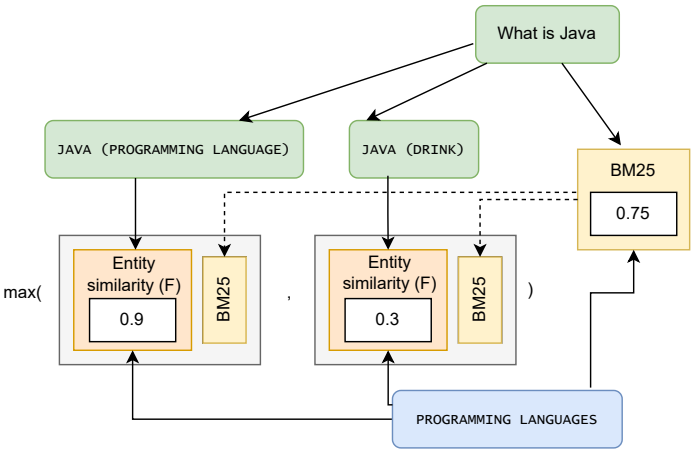


Figure 1: A diagram illustrating our re-ranking approach for an example query with multiple entity linking interpretations. The re-ranking method computes the similarity score (Eq. 5) between target entity (Programming Language) and linked entities in each query interpretation. These scores will then be combined with the score of the initial retriever (BM25 in this example). The maximum of these scores is considered as the final re-ranking score.

of candidate entities using a high-performing baseline entity retrieval model (see Section 2.3), and then use the graph embeddings to reorder these entities based on their similarity to the entities mentioned in the query, as measured in the derived graph embedding space.

Following the related work discussed in Section 2.2, we use the selected entity linkers to identify the entities mentioned in the query. Given input query q, we obtain a set of linked entities E_q and a confidence score $s(e_q)$ for each entity $e_q \in E_q$, which represents the strength of the relationship between the query and the linked entity. We then compute an embedding-based score for the linked entities of query q and entity e:

$$F(e, E_q) = \sum_{e_q \in E_q} s(e_q) \cdot \cos(\overrightarrow{e}, \overrightarrow{e_q}), \tag{5}$$

where $\overrightarrow{e}, \overrightarrow{e_q}$ denote the embeddings vectors for entities e and e_q, respectively.

The rationale for the scoring in Equation (5) is the hypothesis that relevant entities for a given query are situated close (in graph embedding space) to the query entities identified by the entity linker. Consider, for example, the query *"Who is the daughter of Bill Clinton married to"*, which is linked to three entities by entity linker: Bill Clinton, Daughter, and Same-sex marriage with a confidence of scores of 0.66, 0.13, and 0.21, respectively; i.e., $E_q = \{$Bill Clinton, Daughter, Same-sex marriage$\}$. The relevant entities for this query (according to the DBpedia-Entity V2 collection) are Chelsea Clinton, who is Bill Clinton's daughter, and Clinton Family. Highly ranked entities thus exhibit strong similarity to these linked entities, with similarity to Bill Clinton contributing more to the score than similarity to Daughter or Same-sex marriage, due to the higher confidence score associated with Bill Clinton. Given their close semantic and relational ties, it is

reasonable to expect relevant entities to be located near the linked entities in the embedding space, which confirms our intuition.

To produce our final score, we interpolate the embedding-based score computed using Equation (5) with the score of the baseline entity retrieval model $score_{other}$ used to produce the candidate entities in stage one:

$$score_{total}(e, q) = (1 - \lambda) \cdot score_{other}(e, q) + \lambda \cdot F(e, E_q) \qquad \lambda \in [0, 1]. \tag{6}$$

When considering entity linking methods that return multiple interpretations of the query (i.e., multiple sets of E_q) , we compute Equation (6) for every entity linking interpretation E_q^i and choose the interpretation that generates the highest score:

$$score_{total}(e, q) = \max_{E_q^i} \Big((1 - \lambda) \cdot score_{other}(e, q) + \lambda \cdot F(e, E_q^i) \Big). \tag{7}$$

Equation 7 extends Equation 6 in the following way: It first computes $score_{total}(e, q)$ for every entity linking interpretation of the query and then chooses the highest score as the final score. We select the highest similarity score, rather than other aggregation methods, because a specific entity is typically relevant to only one interpretation of a query. For example, consider the query *"What is Java?"*, which has three entity linking interpretations: $E_q^1 = \{$Java (island)$\}$, $E_q^2 = \{$Java (drink)$\}$, and $E_q^3 = \{$Java (programming language)$\}$. The similarity between the entity Programming Language and the third interpretation $E_q^3 = \{$Java (programming language)$\}$ would be high, while its similarity to the other two interpretations would be low. Since Programming Language is highly relevant to one specific sense of the query, taking the maximum score rather than an average is more appropriate. A visual representation of this can be seen in Figure 1.

4 Experimental Setup

4.1 Test collection

In our experiments, we use the standard entity retrieval collection, DBpedia-Entity V2 (Hasibi et al., 2017b). The collection consists of 467 queries and relevance assessments for 49280 query-entity pairs, where the entities are drawn from the DBpedia 2015-10 dump. The relevance assessments are graded values of 2, 1, and 0 for highly relevant, relevant, and non-relevant entities, respectively. The queries are categorized into 4 different groups: **SemSearch ES** consisting of short and ambiguous keyword queries (e.g., *"Nokia E73"*), **INEX-LD** containing IR-Style keyword queries (e.g., *"guitar chord minor"*), **ListSearch** consisting of queries seeking for a list of entities (e.g., *"States that border Oklahoma"*), and **QALD-2** containing entity-bearing natural language queries (e.g., *"Which country does the creator of Miffy come from"*). Following the baseline runs curated with the DBpedia-Entity V2 collection, we use the stopped version of queries provided by the dateset maintainers, where stop patterns like "which" and "who" are removed from the queries.

DBpedia-Entity V2 comprises queries from six benchmark evaluation campaigns, providing a diverse and heterogeneous set of queries. We point the reader to Oza and Dietz (2023), which shows that our entity ranking method also generalizes to the TREC CAR dataset (Dietz and Foley, 2019) expanded with automatic entity annotations.

4.2 Entity Annotation

We use two sets of human-annotated queries as ground truth, Webis annotations provided by (Kasturia et al., 2022), and Radboud annotations, a set created in this work.

Webis annotations. The annotations created in (Kasturia et al., 2022) present multiple scenario entity linking interpretations of the queries. For example, a query like *"java"* may have the island, the coffee, or the programming language as relevant documents, resulting in a set of interpretations where each interpretation relates to a different set of entities. Following (Hasibi et al., 2015), Webis annotations include only named entities (NEs).

Radboud annotations. To compare the influence of concepts to that of named entities (NE), we construct a new set of annotations of DBpedia-Entity queries, which we refer to as Radboud annotations. We ask expert annotators to annotate all queries with any entity that would help solve the information needs of the queries. This leads to annotations of *Named entities*, such as Nokia and *Concepts* such as movie producer in query *"who produced the film the ritual."* Annotators use the INCEpTION annotation platform (Klie et al., 2018) to link mention spans to WikiData instances. We let one expert annotate the entire dataset and let 2 additional users annotate 50 randomly selected queries, similar to the setup in (Kasturia et al., 2022).

We compute the F-measure and Cohen's Kappa for agreement, following (Deleger et al., 2012; Cui et al., 2022). Cohen's Kappa score is computed on the token level (i.e., each word in a query is treated as a separate data point), a common strategy for handling multiple annotations per query. We found Kappa scores of 0.54 and 0.59, and F-scores of 0.51 and 0.57, which is sufficient for agreement (Cohen, 1960). An explanation for not having higher scores is that annotators might interpret a query differently. Since only one interpretation is asked, these annotations will then not agree. For example, the query *"sri lanka government gazette"* could be linked to the two entities SriLanka and The Sri Lanka Gazette, but also to the non-overlapping entities Government of Sri Lanka and Government Gazette, which have high similarity in the embedding space (and are thus likely to only have small differences when using the differences to these entity embeddings), but still have an overlap of 0.

4.3 Entity Linking

We employ the following entity linking methods: TagMe (Ferragina and Scaiella, 2010), REL (Van Hulst et al., 2020), Nordlys (Hasibi et al., 2017a), SMAPH (Cornolti et al., 2018), and ELQ (Li et al., 2020). To make a fair comparison, we use the default settings of all tools available, using the API if available.

- *TagME and REL:* We use their APIs with the default settings.
- *Nordlys:* Following (Nikolaev and Kotov, 2020), we use the Nordlys toolkit API with the Learning to Rank option.
- *SMAPH:* Following (Nikolaev and Kotov, 2020), we annotate using the *d4science* API, with the Google API as search engine.

- *ELQ:* We utilize the example script as listed on the official GitHub repository, using the default settings and preprocessing, most importantly casting all input to lowercase and using the *elq wiki large* model.

4.4 Embedding Training

Wikipedia2Vec. Wikipedia2Vec provides pre-trained embeddings. These embeddings, however, are not available for all entities in Wikipedia; e.g., 25% of the assessed entities in DBpedia-Entity V2 collection have no pre-trained embedding. The reasons for these missing embeddings are two-fold: (i) "rare" entities were excluded from the training data, and, (ii) entity identifiers evolve over time, resulting in entity mismatches with those in the DBpedia-Entity collection.

For training new graph embeddings, we used the Wikipedia 2019-07 dump, which is also compatible with recent entity linkers. We address the entity mismatch problem by identifying the entities that have been renamed in the new Wikipedia dump. Some of these entities were obtained using the redirect API of Wikipedia.[2] Others were found by matching the Wikipedia page IDs of the two Wikipedia dumps. The page IDs of Wikipedia 2019-07 were available on the Wikipedia website. For the dump where DBpedia-Entity is based on, however, these IDs are not available anymore; we obtained them from the Nordlys package Hasibi et al. (2017a).

To avoid excluding rare entities and generate embeddings for a wide range of entities, we changed several Wikipedia2Vec settings. The two settings that resulted in the highest coverage of entities are: (i) minimum number of times an entity appears as a link in Wikipedia, (ii) whether to include or exclude disambiguation pages. Table 1 shows the effect of these settings on the number of missing entities; specifically the number of entities that are assessed in the DBpedia-Entity collection, but have missing embeddings. We categorize these missing entities into two groups:

- *No-page*: Entities without any pages, i.e., although they have an identifier, there is no actual Wikipedia page with that exact identifier name. These entities were neither found by the Wikipedia redirect API nor could they be matched by their page IDs.

- *No-emb*: Entities that could be found by their identifiers, but were not included in the Wikipedia2Vec embeddings.

The first line in Table 1 corresponds to the default setting of Wikipedia2Vec, which covers only 75% of assessed entities in the DBpedia-Entity collection. When considering all entities in the knowledge graph, this setting discards an even larger number of entities, which is not an ideal setup for entity ranking. By choosing the right settings (the last line of Table 1), we increased the coverage of entities to 97.6%.

To make the comparison fair to the other graph embedding methods, we also use the Wikipedia 2015-10 dump, the version on which DBpedia 2015-10 was based.

RDF2vec. RDF2Vec and ComplEx are trained using the DBpedia link graph as input. DBpedia consists of over 20 possible input files, some containing more relevant information

2. `https://wikipedia.readthedocs.io/en/latest/`

Settings	No-emb	No-page	Total
min-entity-count = 5, disambiguation = False	9640	608	10248
min-entity-count = 1, disambiguation = False	1220	398	1618
min-entity-count = 1, disambiguation = True	1220	377	1597
min-entity-count = 0, disambiguation = False	**724**	380	1104
min-entity-count = 0, disambiguation = True	**724**	**333**	**1057**

Table 1: Missing entities with different settings.

for retrieval-oriented graph embeddings than others. For training RDF2vec and ComplEx, we use the following files from DBpedia:

- Disambiguations: Links extracted from Wikipedia Disambiguation pages, which are the Wikipedia pages that redirect a user when there are multiple entities with the same name.
- Infobox properties: Information which was extracted from the Wikipedia Infoboxes.
- Instance types: Triples of the form object, RDF type, and class from the mapping-based extraction.
- Instance types transitive: Transitive RDF type-class based on the ontology.
- Mapping-based objects: Mapping-based statements with object values.
- Transitive redirects: Transitively resolved redirects between articles in Wikipedia.
- Pagelinks: Contains internal links between entities on DBpedia, created based on the internal links between Wikipedia pages.

In the original RDF2Vec paper, the files used for finetuning are as listed above, except for the 'Pagelinks' file. Including this file increases the graph size from 11GB to 35GB, which in turn increases the training time and the final embedding output size. However, excluding this file results in missing a substantial number of entities, as seen in Table 2. Since several triples in the Pagelinks files appear to include alternatives for semantically similar entities, incorporating these nodes and edges could potentially dilute the effect of fine-tuning. We hypothesize that this occurs because entities seen sparsely during fine-tuning tend to have lower-quality representations in the embedding space. Distributing such an entity across multiple semantically similar entities may further degrade the quality of embeddings. As a resolution, we include results with and without the Pagelinks file.

For finetuning RDF2Vec, we use the jRDF2VEC package (Ristoski and Paulheim, 2016). We finetune using the default settings of the package, which is *walk depth* of 4, *numberOfWalks* of 100 per entity, *walkGenerationMode* as Random, *Dimension size* of 200 and *Number of Epochs* of 5.

Embedding	#Missing entities
Wikipedia2Vec 2019	36326
Wikipedia2Vec 2015	14124
ComplEX	25512
ComplEX Pagelinks	58
RDF2Vec	31782
RDF2Vec Pagelinks	75

Table 2: Number of missing entities with different settings of graph embeddings.

ComplEx. For ComplEx, we use the same training files as for RDF2Vec, reusing the same setup with and without the Pagelinks file. We use the LibKGE (Broscheit et al., 2020) package and the same configuration as used with the Wikipedia5M dataset, which has similar properties to DBpedia.

Table 2 shows the number of missing entity embeddings per embedding type. This is after using the DBpedia redirect file to solve redirects. We can see that, even when using DBpedia 2015-10 and Wikipedia 2015-10, many files are still missing in the eventual embedding space, which is bound to hurt results for re-ranking.

4.5 Evaluation metrics

Entity Linking. Given that Webis annotations contain multiple entity linking scenarios and Radboud annotations contain only one scenario, we need an evaluation method to accommodate both formats. One could choose, for example, evaluating exclusively the best possible scenario, the average across all scenarios, or the union of all scenarios as a ground truth. However, using these strategies, comparing queries with a difference in the number of scenarios presents a challenging task. We, therefore, use the lean evaluation metrics introduced by Hasibi et al. (2015).

Suppose $\hat{I} = \{\hat{E}_1, ..., \hat{E}_m\}$ denote the query interpretations for the ground truth, and $I = \{E_1, ..., E_n\}$ the interpretation returned by the system. Here, each $\hat{E}_i$, E_i is a set of entities, and thus $\hat{I}, I$ are sets of sets of entities. Let $\hat{E} = \bigcup \hat{E}_i$ and $E = \bigcup E_i$. We first define the precision and recall based on the different interpretations, which we call P_{int} and R_{int}:

$$P_{int} = \begin{cases} \frac{|\hat{I} \cap I|}{|I|}, & I \neq \emptyset \\ 1, & I = \emptyset, \hat{I} = \emptyset \\ 0, & I = \emptyset, \hat{I} \neq \emptyset \end{cases} \qquad R_{int} = \begin{cases} \frac{|\hat{I} \cap I|}{|\hat{I}|}, & \hat{I} \neq \emptyset \\ 1, & \hat{I} = \emptyset, I = \emptyset \\ 0, & \hat{I} = \emptyset, I \neq \emptyset \end{cases}$$

We then define the precision and recall based on all the entities linked, which we refer to as P_{ent} and R_{ent}:

$$P_{ent} = \begin{cases} \frac{|\hat{E} \cap E|}{|E|}, & E \neq \emptyset \\ 1, & E = \emptyset, \hat{E} = \emptyset \\ 0, & E = \emptyset, \hat{E} \neq \emptyset \end{cases} \qquad R_{ent} = \begin{cases} \frac{|\hat{E} \cap E|}{|\hat{E}|}, & \hat{E} \neq \emptyset \\ 1, & \hat{E} = \emptyset, E = \emptyset \\ 0, & \hat{E} = \emptyset, E \neq \emptyset. \end{cases}$$

Lean precision and recall are then the combination between these two scores, being:

$$P_{lean} = \frac{P_{int} + P_{ent}}{2}, \quad R_{lean} = \frac{R_{int} + R_{ent}}{2}.$$

When given only one scenario for both system annotations and ground truth, we see that $P = P_{lean} = P_{int} = P_{ent}$ and $R = R_{lean} = R_{int} = R_{ent}$. Thus, lean evaluation can be utilized for Webis annotations, which encompass multiple scenarios, as well as Radboud annotations, which do not include multiple scenarios. Lean evaluation can be interpreted as the standard precision and recall for the latter.

Entity retrieval. We evaluate each combination of all methods and queries using the method used in (Hasibi et al., 2017b), which is the Normalized Discounted Cumulative Gain (NDCG) at ranks 10 and 100. We report on statistical significance for NDCG@10 and NDCG@100 using a two-sided t-test with p-value < 0.05.

5 Results and Analysis

5.1 Entity Linking Results

To compare all different entity linking methods, we compute precision, recall, and F-measure for all entity linkers in Table 3, using lean evaluation described in Section 4.5. The line 'combined' indicates the score for the combination of all linked entities, which is the union of TagMe, REL, Nordlys, SMAPH, and ELQ. With Webis as ground truth, Nordlys has the highest recall and precision. Using our concept annotations as ground truth, ELQ has the highest precision, and TagMe has the highest recall. As seen in Table 3, there is a difference in what each of these entity linkers excels in, resulting in no single best method.

5.2 Entity Retrieval Results

Table 4 depicts entity retrieval results using different automatic entity linkers and human annotations; the breakdown of results by query type can be found in Table 8 of the Appendix. We use the Wikipedia2Vec embeddings from (Gerritse et al., 2020) and perform re-ranking with BM25F-CA. When using human annotations, Radboud annotations receive better results than the multiple scenarios used in the Webis annotations. Especially in the SemSearch, INEX, and ListSearch categories, we see an increase in both NDCG@10 and NDCG@100 using Radboud annotations. Since the Radboud annotations focuses on adding concept annotations, this indicates that concept entities are, in fact, an essential factor for entity retrieval using entity embeddings.

	#Linked entities	Webis (NEs)			Radboud (Concepts+NEs)		
		P_{lean}	R_{lean}	F_{lean}	P	R	F
TagMe	1186	0.479	0.598	0.532	0.704	**0.814**	**0.755**
REL	363	0.699	0.618	0.656	0.534	0.416	0.467
Nordlys	450	**0.722**	**0.642**	**0.680**	0.553	0.439	0.490
SMAPH	797	0.526	0.550	0.538	0.758	0.720	0.739
ELQ	647	0.606	0.587	0.597	**0.787**	0.701	0.741
Combined	1795	0.437	0.686	0.534	0.621	0.911	0.738

Table 3: Performance of different entity linkers against Webis (only NE) and Radboud (both NEs and concepts) annotations. Lean evaluation is used for multiple interpretations (Webis), and regular precision/recall is used for single interpretations (Radboud). 'Combined' refers to the union of all entities returned by all linkers.

	NDCG@10	NDCG@100
BM25F-CA	0.461	0.551
+ Wikipedia2Vec (w/ TagMe)	0.484^{b}	0.570^{b}
+ Wikipedia2Vec (w/ SMAPH)	0.483^{b}	0.570^{b}
+ Wikipedia2Vec (w/ Nordlys)	0.477^{b}	0.563^{bts}
+ Wikipedia2Vec (w/ REL)	0.472^{bts}	0.561^{bts}
+ Wikipedia2Vec (w/ ELQ)	$\mathbf{0.489}^{bnr}$	$\mathbf{0.573}^{bnr}$
+ Wikipedia2Vec (w/ Combined)	0.498^{btsnre}	0.58^{btsnre}
+ Wikipedia2Vec (w/ Webis)	0.475^{bea}	0.563^{btsea}
+ Wikipedia2Vec (w/ Radboud)	0.492^{bsnrw}	0.577^{bnrw}

Table 4: Entity retrieval results on DBpedia-Entity V2 collection using different entity linkers and gold annotations. Wikipedia2Vec 2019 is used for re-ranking of BM25F-CA results. Superscripts denote statistically significant differences (better or worse) corresponding to the beginning letter of entity linkers' names, BM25F-CA, and Webis.

The entity retrieval methods using TagMe, SMAPH and ELQ as entity linkers obtain the best results, with no overall significant differences between each other. These three entity linkers obtain the highest F-measure when using the Radboud annotations as ground truth. This answers our first research question **RQ1**: Entity-linking methods with a high F-measure with respect to both concepts and named entities are best suited for the entity retrieval method discussed in this paper.

Table 5 shows the results for different embedding algorithms using TagMe and Radboud annotations; the full table can be found in Table 9 of the Appendix. In the first block, denoted as *'base'*, we only rank using the entity similarity score per embedding type; i.e., using Equation 5 only, using TagMe as an entity linker. Next, we re-rank the BM25F-CA results, both using TagMe (as used in (Gerritse et al., 2020)) and Radboud annotations. We see that Wikipedia2Vec embeddings outperform ComplEx and RDF2Vec in base form, but not in the base form where many entities are missing. We also see that RDF2Vec and

	NDCG@10	NDCG@100
Base		
Wikipedia2Vec 2019	**0.262**[1]	**0.335**[1]
Wikipedia2Vec 2015	0.184[12]	0.278[12]
ComplEx	0.182[12]	0.216[123]
ComplEx Pagelinks	0.204[1234]	0.255[1234]
RDF2Vec	0.188[125]	0.23[12345]
RDF2Vec Pagelinks	0.195[12]	0.262[12346]
TagMe		
BM25F-CA	0.461	0.551
+ Wikipedia2Vec 2019	0.484[1]	0.57[1]
+ Wikipedia2Vec 2015	0.466[2]	0.559[12]
+ ComplEx	0.468[12]	0.558[12]
+ ComplEx Pagelinks	0.474[1234]	0.564[1234]
+ RDF2Vec	0.461[245]	0.554[235]
+ RDF2Vec Pagelinks	0.467[256]	0.56[126]
Radboud annotations		
BM25F-CA	0.461	0.551
+ Wikipedia2Vec 2019	**0.492**[1]	**0.576**[1]
+ Wikipedia2Vec 2015	0.473[12]	0.564[12]
+ ComplEx	0.467[12]	0.558[123]
+ ComplEx Pagelinks	0.476[124]	0.564[124]
+ RDF2Vec	0.463[235]	0.556[1235]
+ RDF2Vec Pagelinks	0.471[126]	0.566[1246]

Table 5: Entity retrieval results on DBpedia-Entity V2 collection using different graph embeddings. Superscripts denote statistically significant differences (better or worse) corresponding to that line of the table.

ComplEx perform significantly better when using the Pagelink pages, indicating that they are essential when re-ranking with entity retrieval. Last, we see that Radboud annotations yield higher scores than the TagMe annotations.

5.3 Embedding Analysis

To judge how suitable each graph embedding is for retrieval, we compare the embeddings of documents relevant to the same query, based on the cluster hypothesis (Rijsbergen, 1979). This states that documents relevant to the same query should cluster together in a higher dimensional space. Here, we consider the entity embedding of a document as its representation. Using this hypothesis, we compute each query's coherence score as defined in (He, 2011), which measures the similarity between all pairs of documents relevant to the same query and returns the percentage of items with a similarity score higher than a

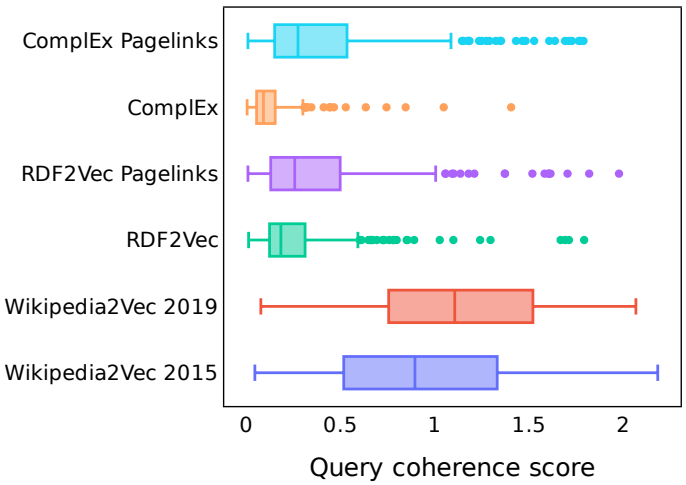


Figure 2: Query coherence score of three different graph embedding algorithms. Higher coherence score is better. Wikipedia2Vec has higher coherence scores compared to RDF2Vec and ComplEx.

threshold. Formally, given a document set D, the coherence score is computed as:

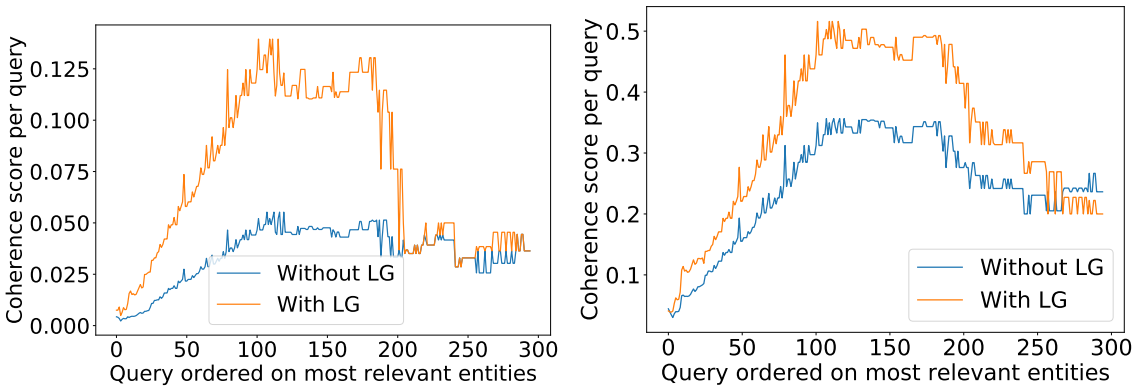
$$Co(D) = \frac{\sum_{i \neq j \in 1,\ldots,M} \delta(d_i, d_j)}{\frac{1}{2}M(M-1)}, \tag{8}$$

where M is total number of documents and the δ function for each document pair d_i and d_j is defined as:

$$\delta(d_i, d_j) = \begin{cases} 1, & \text{if } sim(d_i, d_j) \geq \tau \\ 0, & \text{otherwise.} \end{cases} \tag{9}$$

In Figure 2, we see coherence scores for the different entity embedding methods used in this paper, with a threshold of $\tau = 0.7$. This threshold is selected via grid search as the highest value at which none of the box plots had a first quartile equal to 0. For RDF2Vec and ComplEX, we include both versions with and without pagelinks. We compute the coherence score on the 295 queries with at least ten relevant entities. The higher the query coherence scores are, the better the distance between embeddings should be able to represent relevance for documents to the same query. We see that Wikipedia2Vec leads to the highest coherence scores. Interestingly enough, we see that the 2019 version of Wikipedia has a higher average coherence score than the 2015 version, even though more entities seem to be missing. A reason for this could be that the quality of Wikipedia has improved over the years, with more new pages and additional text added, resulting in a better coherence score. Besides, we see that for both ComplEX and RDF2Vec, the additional page links improve the coherence scores.

We now answer our second research question **RQ2**: Wikipedia2Vec leads to the best results for the entity retrieval method discussed in this paper. However, it is essential to invest in solving the missing entities for optimal performance. For ComplEx and RDF2Vec, including a sufficiently large number of entities in the graph is considerably important.



(a) threshold $\tau = 0.9$

(b) threshold $\tau = 0.8$

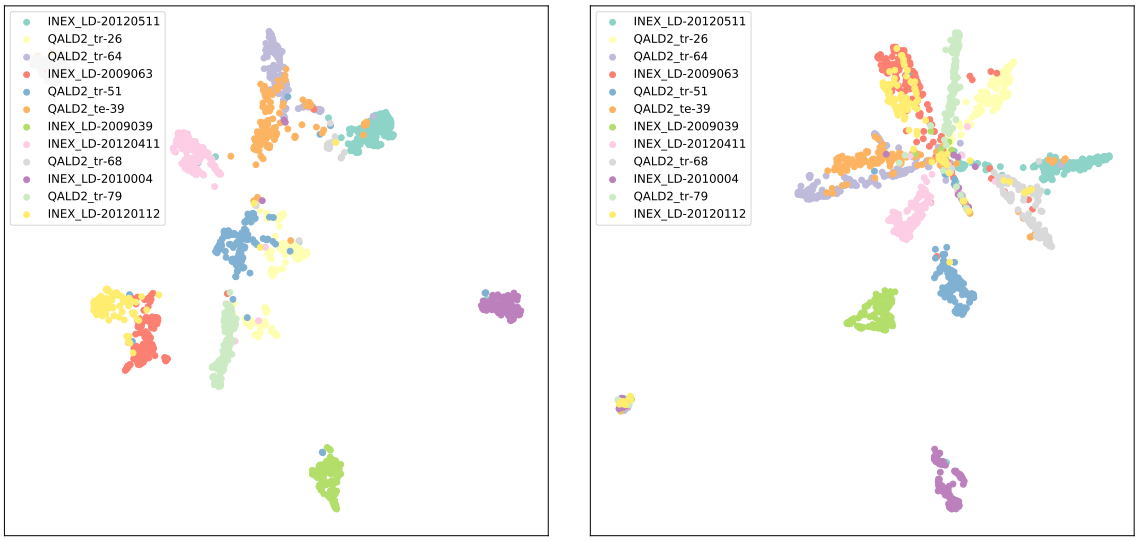
Figure 3: Coherence score of all relevant entities per query, computed for the Wikipedia2Vec embeddings without and with link graph. The queries are ordered by the number of their relevant entities in the x-axis.

5.4 Wikipedia2Vec Embedding Analysis

We empirically showed that Wikipedia2Vec graph embeddings yield better performance compared to other embeddings. To analyze why Wikipedia2Vec graph embeddings are beneficial for entity retrieval models, we conduct a set of experiments and investigate how the graph structure captured by Wikipedia2Vec embeddings improves effectiveness. Specifically, we trained two versions of Wikipedia2Vec embeddings: with and without link graph; i.e., using Eq. (4) with and without the $\mathcal{L}_e$ component.

Figure 3 shows the coherence scores for all queries in our collection. Each point represents the coherence score of all relevant entities (according to the qrels) for a query. We considered only queries with more than 10 relevant entities, ensuring the clusters were sufficiently large to yield meaningful scores. Queries are sorted on the x-axis by the number of relevant entities. The plots clearly show that the coherence score for graph-based entity embeddings are higher than for context-only ones. Based on these performance improvements, we conclude that adding the graph structure results in embeddings that are more suitable for entity-oriented tasks.

Figure 4 helps to visually understand how clusters of entities differ for the two embedding variations. The data points correspond to the entities with a relevance grade higher than 0, for 12 queries with 100–200 relevant entities in the ground truth data. We use Uniform Manifold Approximation and Projection (UMAP) (McInnes et al., 2018) to reduce the dimensions of the embeddings from 100 to two and plot the projected entities for each query. In Figure 4b, most of the clusters are overlapping in a star-like shape, while in Figure 4a, the clusters are more separated, and the ones with similar search intents are close to each other; e.g., queries `QALD2_te-39` and `QALD2_tr-64` (which are both about companies), or `INEX_LD-20120112` and `INEX_LD-2009063` (which are both about war) are situated next to each other. These analyses support the answer that we formulate for our third research question **RQ 3:** The graph structure, combined with the textual representation of entities incorporated in Wikipedia2Vec graph embeddings, plays an important role in improving the



(a) Embeddings with link graph (b) Embeddings without link graph

Figure 4: UMAP visualization of entity embeddings for a subset of queries. Color codes correspond to the relevant entities per query. Queries per code are listed in Table 10 of the Appendix. Default settings of UMAP in python were used.

cluster quality of the representation of the entities and explains the enhanced effectiveness of retrieval results.

5.5 Query Analysis

Next, we analyze queries that are helped and hurt the most by our embedding-based method. Table 6 shows six queries that are affected the most by BM25F-CA+ Wikipedia2Vec compared to BM25F-CA, with respect to NDCG@100. Each of the three queries with the highest gains is linked to at least one relevant entity (according to the assessments). The losses can be attributed to various sources of errors. For the query *"spring shoe canada"*, the only relevant entity belongs to the 2.4% of entities that have no embedding (cf. §4.4). Query *"vietnam war movie"* is linked to entities Vietnam War and War film, with confidence scores of 0.7 and 0.2, respectively. This emphasizes Vietnam war facts instead of its movies, and could be resolved by improving the accuracy of the entity linker and/or employing a re-ranking approach that is more robust to linking errors. The query *"mr rourke fantasy island"* is linked to a wrong entity due to a spelling mistake, which emphasizes the importance of the quality of the entity linker.

To further understand the difference between the two versions of the embeddings at the query-level, we selected the queries with the highest and lowest gain in NDCG@100 (i.e., comparing BM25F-CA + Wikipedia2Vec and BM25F-CA + Wikipedia2Vec (no graph)). For the query *"Which instruments did John Lennon play?"*, the two linked entities (with the highest confidence score) are John Lennon and Musical Instruments. Their closest entity in graph embedding space is John Lennon's musical instruments, relevant to

Query	Gain in NDCG	
	@10	@100
st paul saints	0.716	0.482
continents in the world	0.319	0.362
What did Bruce Carver die from?	0.307	0.307
spring shoes canada	-0.286	-0.286
vietnam war movie	-0.470	-0.240
mr rourke fantasy island	-0.300	-0.307

Table 6: Top queries with the highest gains and losses in NDCG at cut-offs 10 and 100, BM25F-CA + Wikipedia2Vec vs. BM25F-CA.

Query	Gain in NDCG	
	@10	@100
What did Bruce Carver die from?	0.307	0.307
Which other weapons did the designer of the Uzi develop?	0.236	0.248
Which instruments did John Lennon play?	0.154	0.200
Companies that John Hennessey serves on the board of	-0.173	-0.173
Which European countries have a constitutional monarchy?	-0.101	-0.197
vietnam war movie	-0.276	-0.222

Table 7: Top queries with the highest gains and losses in NDCG at cut-offs 10 and 100, BM25F-CA + Wikipedia2Vec vs. BM25F-CA + Wikipedia2Vec (no graph).

the query. This entity, however, is not among the most similar entities when we consider the context-only case. For the other queries in Table 7, the effect is similar but less prominent than in the BM25F-CA and BM25F-CA + Wikipedia2Vec case, probably due to the lower value of λ.

6 Conclusion

In this paper, we investigated the use of different types of entity embeddings and different types of entity linkers for entity retrieval. We trained entity embeddings using Wikipedia2Vec, ComplEx, and RDF2Vec, combined these with state-of-the-art entity ranking models, and found empirically that using graph embeddings leads to increased effectiveness of entity retrieval.

We analyzed the effect of different entity linkers and concluded that the most suitable entity linkers are SMAPH, TagMe, and ELQ, annotating both named entities and concepts. When evaluating with the baselines of annotated entities with two different philosophies, multiple scenarios compared to named entities and concepts, we found that SMAPH, TagMe, and ELQ align the most with the annotations focused on named entities and concepts, thus confirming our conclusion that annotated concept are important for retrieval.

We then compared three classes of graph embedding methods, Wikipedia2Vec, RDF2Vec, and ComplEx, and found that first, having as many different entities in the graph embedding will give the best performance, even if they might be redundant. Second, Wikipedia2Vec performs best in all categories, provided that effort is put into solving as many entities linked to an entity without embedding as possible. Wikipedia2Vec has the highest cluster similarity score, confirming that Wikipedia2Vec is a highly suitable method for performing entity retrieval.

We conclude that enriching entity retrieval methods with entity embeddings is valuable, efficient, and effective. The choice of entity linker, graph embedding method, and effort to find missing entities are integral to the method's performance. For future work, we would like to evaluate how these different graph embedding methods influence more modern Transformer-based entity retrieval methods, as well as how well these methods can be adapted to work on domain-specific entities or sparser knowledge graphs.

Acknowledgments and Disclosure of Funding

This research is supported by the Dutch Research Council (NWO) under project numbers NWA.1389.20.183 (LESSEN) and the EU's Horizon Europe program under grant No. 101070014 (OpenWebSearch.EU, https://doi.org/10.3030/101070014). All content represents the opinion of the authors, which is not necessarily shared or endorsed by their respective employers and/or sponsors.

References

Negar Arabzadeh, Amin Bigdeli, and Ebrahim Bagheri. Laque: Enabling entity search at scale. In *Advances in Information Retrieval*, pages 270–285, 2024.

Krisztian Balog. Entity-oriented search, *Springer*, 2018.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *Proceedings of the 26th International Conference on Neural Information Processing Systems*, pages 2787–2795. ACM, 2013.

Samuel Broscheit, Daniel Ruffinelli, Adrian Kochsiek, Patrick Betz, and Rainer Gemulla. LibKGE - A knowledge graph embedding library for reproducible research. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 165–174, 2020.

Shubham Chatterjee and Laura Dietz. BERT-ER: Query-specific BERT entity representations for entity ranking. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '22, page 1466–1477. Association for Computing Machinery, 2022.

Viktoriia Chekalina, Anton Razzhigaev, Albert Sayapin, Evgeny Frolov, and Alexander Panchenko. MEKER: Memory efficient knowledge embedding representation for link

prediction and question answering. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 355–365. Association for Computational Linguistics, 2022.

Jacob Cohen. A coefficient of agreement for nominal scales. *Educational and psychological measurement*, 20(1):37–46, 1960.

Marco Cornolti, Paolo Ferragina, Massimiliano Ciaramita, Stefan Rüd, and Hinrich Schütze. Smaph: A piggyback approach for entity-linking in web queries. *ACM Transactions on Information Systems*, 37(1), 2018. ISSN 1046-8188.

Wen Cui, Leanne Rolston, Marilyn Walker, and Beth Ann Hockey. Openel: An annotated corpus for entity linking and discourse in open domain dialogue. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 2245–2256, 2022.

Daniel Daza, Michael Cochez, and Paul Groth. Inductive entity representations from text via link prediction. In *Proceedings of the Web Conference*, WWW'21, page 798–808, 2021.

Mostafa Dehghani, Hamed Zamani, Aliaksei Severyn, Jaap Kamps, and W Bruce Croft. Neural ranking models with weak supervision. *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 65–74, 2017.

Louise Deleger, Qi Li, Todd Lingren, Megan Kaiser, Katalin Molnar, Laura Stoutenborough, Michal Kouril, Keith Marsolo, Imre Solti, et al. Building gold standard corpora for medical natural language processing tasks. In *AMIA Annual Symposium Proceedings*, volume 2012, page 144. American Medical Informatics Association, 2012.

Fernando Diaz, Bhaskar Mitra, and Nick Craswell. Query expansion with locally-trained word embeddings. *Procdeedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 367–377, 2016.

Laura Dietz. 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, page 215–224, 2019.

Laura Dietz and John Foley. Trec car y3: Complex answer retrieval overview. In *Proceedings of Text REtrieval Conference (TREC)*, 2019.

Paolo Ferragina and Ugo Scaiella. Tagme: on-the-fly annotation of short text fragments (by Wikipedia entities). In *Proceedings of the 19th ACM international IW3C2 on Information and Knowledge Management*, pages 1625–1628. ACM, 2010.

Darío Garigliotti, Faegheh Hasibi, and Krisztian Balog. Identifying and exploiting target entity type information for ad hoc entity retrieval. *Information Retrieval Journal*, 22(3): 285–323, 2019.

Emma Gerritse, Faegheh Hasibi, and Arjen De Vries. Graph-embedding empowered entity retrieval. In *Proceedings of the European Conference on Information Retrieval*, ECIR '20, pages 97–110, 2020.

Emma Gerritse, Faegheh Hasibi, and Arjen De Vries. Entity-aware Transformers for Entity Search. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '22, 2022.

Faegheh Hasibi, Krisztian Balog, and Svein Erik Bratsberg. Entity linking in queries: Tasks and evaluation. In *Proceedings of ACM SIGIR International Conference on the Theory of Information Retrieval*, ICTIR '15, pages 171–180, 2015.

Faegheh Hasibi, Krisztian Balog, and Svein Erik Bratsberg. Exploiting entity linking in queries for entity retrieval. In *Proceedings of the 2016 ACM International Conference on the Theory of Information Retrieval*, pages 209–218. ACM, 2016.

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

Faegheh Hasibi, Fedor Nikolaev, Chenyan Xiong, Krisztian Balog, Svein Erik Bratsberg, Alexander Kotov, and Jamie Callan. 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*, pages 1265–1268. ACM, 2017b.

Jiyin He. *Exploring topic structure: Coherence, diversity and relatedness.* PhD thesis, University of Amsterdam, 2011.

Parastoo Jafarzadeh, Zahra Amirmahani, and Faezeh Ensan. Learning to rank knowledge subgraph nodes for entity retrieval. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '22, page 2519–2523, 2022.

Hideaki Joko and Faegheh Hasibi. Personal entity, concept, and named entity linking in conversations. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, CIKM '22, page 4099–4103. Association for Computing Machinery, 2022.

Ehsan Kamalloo, Nandan Thakur, Carlos Lassance, Xueguang Ma, Jheng-Hong Yang, and Jimmy Lin. Resources for brewing beir: Reproducible reference models and statistical analyses. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '24, page 1431–1440. Association for Computing Machinery, 2024. ISBN 9798400704314.

Chris Kamphuis, Faegheh Hasibi, Jimmy Lin, and Arjen P. de Vries. REBL: entity linking at scale (prototype). In Omar Alonso, Ricardo Baeza-Yates, Tracy Holloway King, and Gianmaria Silvello, editors, *Proceedings of the third International Conference on Design of Experimental Search & Information REtrieval Systems (DESIRES)*, volume 3480, pages 68–75, 2022.

Vaibhav Kasturia, Marcel Gohsen, and Matthias Hagen. Query Interpretations from Entity-Linked Segmentations. In *15th ACM International Conference on Web Search and Data Mining (WSDM 2022)*, pages 449–457, 2022.

Seyed Mehran Kazemi and David Poole. Simple embedding for link prediction in knowledge graphs. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, page 4289–4300, 2018.

Jan-Christoph Klie, Michael Bugert, Beto Boullosa, Richard Eckart de Castilho, and Iryna Gurevych. The INCEpTION platform: Machine-assisted and knowledge-oriented interactive annotation. In *Proceedings of the 27th International Conference on Computational Linguistics: System Demonstrations*, pages 5–9, 2018.

Adrian Kochsiek, Apoorv Saxena, Inderjeet Nair, and Rainer Gemulla. Friendly neighbors: Contextualized sequence-to-sequence link prediction. In *Proceedings of the 8th Workshop on Representation Learning for NLP (RepL4NLP 2023)*, pages 131–138. Association for Computational Linguistics, 2023.

Belinda Z. Li, Sewon Min, Srinivasan Iyer, Yashar Mehdad, and Wen-tau Yih. Efficient one-pass end-to-end entity linking for questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 6433–6441. Association for Computational Linguistics, 2020.

Qingyang Li, Yanru Zhong, and Yuchu Qin. MoCoKGC: Momentum contrast entity encoding for knowledge graph completion. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 14940–14952. Association for Computational Linguistics, 2024.

Zhenghao Liu, Chenyan Xiong, Maosong Sun, and Zhiyuan Liu. Explore entity embedding effectiveness in entity retrieval. *arXiv preprint arXiv:1908.10554*, 2019.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*, pages 9802–9822. Association for Computational Linguistics, 2023.

Leland McInnes, John Healy, Nathaniel Saul, and Lukas Grossberger. Umap: Uniform manifold approximation and projection. *The Journal of Open Source Software*, 3(29): 861, 2018.

Edgar Meij, Krisztian Balog, and Daan Odijk. Entity linking and retrieval for semantic search. In *Proceedings of the 7th ACM International Conference on Web Search and Data Mining*, WSDM '14, page 683–684. Association for Computing Machinery, 2014.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In *1st International Conference on Learning Representations, ICLR*, pages 1–12, 2013a.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems (NIPS)*, pages 3111–3119, 2013b.

Fedor Nikolaev and Alexander Kotov. Joint word and entity embeddings for entity retrieval from a knowledge graph. In *Proceedings of the European Conference on Information Retrieval*, ECIR '20, pages 141–155, 2020.

Pooja Oza and Laura Dietz. Entity embeddings for entity ranking: A replicability study. In *Proceedings of the 45th European Conference on Information Retrieval*, ECIR '23, pages 117–131, 2023.

Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1532–1543. ACL, 2014.

Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international Conference on Knowledge discovery and data mining*, pages 701–710. ACM, 2014.

Nina Poerner, Ulli Waltinger, and Hinrich Schütze. E-BERT: Efficient-yet-effective entity embeddings for BERT. In *Findings of the Association for Computational Linguistics*, ELMNLP '20, pages 803–818, 2020.

C. J. van Rijsbergen. *Information Retrieval, 2nd edition.* Butterworths, 1979.

Petar Ristoski and Heiko Paulheim. RDF2Vec: RDF graph embeddings for data mining. In *International Semantic Web Conference*, pages 498–514. Springer, 2016.

Stephen Robertson, Hugo Zaragoza, et al. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends in Information Retrieval*, 3(4):333–389, 2009.

Daniel Ruffinelli, Samuel Broscheit, and Rainer Gemulla. You CAN teach an old dog new tricks! On training knowledge graph embeddings. In *International Conference on Learning Representations*, 2020.

Apoorv Saxena, Adrian Kochsiek, and Rainer Gemulla. Sequence-to-sequence knowledge graph completion and question answering. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2814–2828. Association for Computational Linguistics, 2022.

Heydar Soudani, Evangelos Kanoulas, and Faegheh Hasibi. Fine tuning vs. retrieval augmented generation for less popular knowledge. In *Proceedings of the 2024 Annual International ACM SIGIR Conference on Research and Development in Information Retrieval in the Asia Pacific Region*, SIGIR-AP 2024, page 12–22. Association for Computing Machinery, 2024.

Hai Dang Tran and Andrew Yates. Dense retrieval with entity views. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, CIKM '22, page 1955–1964. Association for Computing Machinery, 2022.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. Complex embeddings for simple link prediction. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, page 2071–2080, 2016.

Johannes M. van Hulst, Faegheh Hasibi, Koen Dercksen, Krisztian Balog, and Arjen P. de Vries. Rel: An entity linker standing on the shoulders of giants. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '20, 2020.

Xiaozhi Wang, Tianyu Gao, Zhaocheng Zhu, Zhengyan Zhang, Zhiyuan Liu, Juanzi Li, and Jian Tang. KEPLER: A unified model for knowledge embedding and pre-trained language representation. *Transactions of the Association for Computational Linguistics*, 9:176–194, 2021.

Ikuya Yamada and Hiroyuki Shindo. Neural attentive bag-of-entities model for text classification. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 563–573. ACL, 2019.

Ikuya Yamada, Hiroyuki Shindo, Hideaki Takeda, and Yoshiyasu Takefuji. Joint learning of the embedding of words and entities for named entity disambiguation. *The SIGNLL Conference on Computational Natural Language Learning*, 2016.

Ikuya Yamada, Akari Asai, Jin Sakuma, Hiroyuki Shindo, Hideaki Takeda, Yoshiyasu Takefuji, and Yuji Matsumoto. Wikipedia2Vec: An efficient toolkit for learning and visualizing the embeddings of words and entities from Wikipedia. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 23–30. Association for Computational Linguistics, 2020.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding entities and relations for learning and inference in knowledge bases. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, 2015.

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

Appendix A. Extra Results

	SemSearch		INEX-LD		ListSearch		QALD-2	
	@10	@100	@10	@100	@10	@100	@10	@100
TagMe	0.661^{b}	0.738^{b}	0.464^{b}	0.552^{b}	0.447^{b}	0.532^{b}	0.387^{b}	0.479^{b}
SMAPH	0.661^{b}	0.734	0.455	0.546^{b}	0.448^{b}	0.534^{b}	0.389^{b}	0.483^{b}
Nordlys	0.645	0.719^{t}	0.451^{b}	0.537^{bt}	0.444^{b}	0.529^{b}	0.387^{b}	0.481^{b}
REL	0.647^{b}	0.732^{b}	0.451	0.536^{t}	0.436	0.523^{bs}	0.376^{bs}	0.472^{bsn}
ELQ	0.645	0.729	0.474^{bnr}	0.554^{bnr}	0.455^{b}	0.537^{b}	0.402^{btr}	0.488^{br}
Combined	0.667^{be}	0.738^{n}	0.476^{btsnr}	0.56^{btsnr}	0.468^{btsnr}	0.549^{btsnr}	0.402^{btsnr}	0.492^{btsnr}
Webis	0.63^{tsa}	0.711^{tsra}	0.452^{ea}	0.537^{tea}	0.439^{a}	0.532^{bra}	0.396^{bnr}	0.488^{br}
Radboud	0.651	0.727	0.478^{btsnrw}	0.56^{bsnrw}	0.462^{bsnrw}	0.543^{bsnrw}	0.399^{br}	0.495^{btsnr}

Table 8: Breakdown per query type of entity retrieval results using different entity linkers and gold annotations. Wikipedia2Vec 2019 is used for re-ranking of BM25F-CA results. Superscripts denote statistically significant differences corresponding to the beginning letter of entity linkers' names, BM25F-CA, and Webis.

	SemSearch		INEX-LD		ListSearch		QALD-2	
	@10	@100	@10	@100	@10	@100	@10	@100
Base								
Wikipedia2Vec 2019	**0.417**	**0.478**	**0.217**	**0.286**	**0.211**	**0.302**	**0.212**	**0.282**
Wikipedia2Vec 2015	0.249^{1}	0.345^{1}	0.152^{1}	0.24^{1}	0.153^{1}	0.26^{1}	0.181^{1}	0.265
ComplEx	0.309^{12}	0.322^{1}	0.136^{1}	0.17^{12}	0.139^{1}	0.178^{12}	0.148^{12}	0.193^{12}
ComplEx Pagelinks	0.32^{12}	0.371^{13}	0.148^{1}	0.187^{12}	0.183^{13}	0.239^{13}	0.168^{1}	0.224^{123}
RDF2Vec	0.317^{12}	0.346^{1}	0.159^{1}	0.182^{12}	0.128^{14}	0.175^{124}	0.154^{1}	0.216^{12}
RDF2Vec Pagelinks	0.308^{12}	0.376^{135}	0.179^{13}	0.239^{1345}	0.142^{14}	0.216^{1235}	0.158^{1}	0.225^{123}
TagMe								
BM25F-CA	0.628	0.720	0.439	0.530	0.425	0.511	0.369	0.461
+ Wikipedia2Vec 2019	$\mathbf{0.661}^{1}$	$\mathbf{0.738}^{1}$	$\mathbf{0.464}^{1}$	$\mathbf{0.552}^{1}$	$\mathbf{0.447}^{1}$	$\mathbf{0.532}^{1}$	$\mathbf{0.387}^{1}$	$\mathbf{0.479}^{1}$
+ Wikipedia2Vec 2015	0.633^{2}	0.724^{2}	0.443^{2}	0.539^{12}	0.434^{2}	0.525^{12}	0.373^{2}	0.467^{2}
+ ComplEx	0.64	0.727	0.441^{2}	0.53^{2}	0.433^{2}	0.523^{12}	0.376^{2}	0.47^{1}
+ ComplEx Pagelinks	0.653^{13}	0.732^{1}	0.445^{2}	0.537^{2}	0.443^{1}	0.532^{14}	0.377	0.474^{1}
+ RDF2Vec	0.633^{25}	0.722^{25}	0.429^{2345}	0.53^{23}	0.428^{25}	0.519^{125}	0.374^{2}	0.465^{2}
+ RDF2Vec Pagelinks	0.636^{25}	0.727	0.436^{2}	0.539^{126}	0.432^{2}	0.52^{125}	0.381^{1}	0.474^{136}
Concepts								
BM25F-CA	0.628	0.720	0.439	0.530	0.425	0.511	0.369	0.461
+ Wikipedia2Vec 2019	0.652	0.73	0.48^{1}	0.558^{1}	0.46^{1}	0.543^{1}	0.398^{1}	0.491^{1}
+ Wikipedia2Vec 2015	0.632^{2}	0.721	0.449^{2}	0.541^{12}	0.442^{12}	0.531^{12}	0.387^{12}	0.479^{12}
+ ComplEx	0.635	0.723	0.444^{2}	0.535^{2}	0.434^{2}	0.522^{23}	0.375^{23}	0.47^{123}
+ ComplEx Pagelinks	0.65^{14}	0.727	0.449^{2}	0.539^{12}	0.44^{2}	0.53^{124}	0.384^{1}	0.477^{12}
+ RDF2Vec	0.637^{5}	0.723	0.429^{2345}	0.529^{235}	0.43^{23}	0.52^{235}	0.373^{235}	0.47^{1235}
+ RDF2Vec Pagelinks	0.637	0.733^{346}	0.443^{26}	0.543^{1246}	0.437^{2}	0.526^{12}	0.384^{126}	0.482^{146}

Table 9: Breakdown per query type of entity retrieval results using different graph embeddings. Superscripts denote statistically significant differences (better or worse) corresponding to that line of the table.

Appendix B. Queries

Query ID	Query text
INEX_LD-20120511	female rock singers
QALD2_tr-26	Which bridges are of the same type as the Manhattan Bridge?
QALD2_tr-64	Which software has been developed by organizations founded in California?
INEX_LD-2009063	D-Day normandy invasion
QALD2_tr-51	Give me all school types.
QALD2_te-39	Give me all companies in Munich.
INEX_LD-2009039	roman architecture
INEX_LD-20120411	bicycle sport races
QALD2_tr-68	Which actors were born in Germany?
INEX_LD-2010004	Indian food
QALD2_tr-79	Which airports are located in California, USA?
INEX_LD-20120112	vietnam war facts

Table 10: Queries used for Figure 4.