E-BELA: Enhanced Embedding-Based Entity Linking Approach

Ítalo M. Pereira
italo.pereira@ifmg.edu.br
Instituto Federal de Minas Gerais - SJE
São João Evangelista, MG, Brasil

Anderson A. Ferreira
anderson.ferreira@ufop.edu.br
Universidade Federal de Ouro Preto
Ouro Preto, MG, Brasil

ABSTRACT

Entity linking is the process of connecting mentions of entities in natural language texts, such as references to people or places, to specific entities in knowledge graphs, such as DBpedia or Wikidata. This process is crucial in the natural language processing tasks since it facilitates disambiguating entities in unstructured data, enhancing understanding and semantic processing. However, entity linking faces challenges due to the complexity and ambiguity of natural languages, as well as the discrepancy between the form of textual entity mentions and entity representations. Considering that entity mentions are in natural language and entity representations in knowledge graphs have object nodes that describe them in the same way, in this work we propose E-BELA, an effective approach based on literal embeddings. We aim to put close vector representations of mentions and entities in a vector space, allowing linking of mentions and entities by using a similarity or distance metric. The results demonstrate that our approach outperforms previous ones, contributing to the field of natural language processing.

KEYWORDS

Natural Language Processing, Entity Linking, Linked Open Data, Entity Similarity, Embedding, Disambiguation, DBpedia

1 INTRODUCTION

The increasing volume of data published on the web has led to an era of information overload. Persons generate more information than they can actually process and consume [3, 5, 6, 10, 15, 25]. Additionally, natural language, one of the most important forms of data on the web, is inherently complex and ambiguous. To address this scenario, many efforts have been made. Among these efforts, the creation of the Web of Data stands out, achieved through the development of Linked Open Data (LOD) datasets. These datasets are massive Knowledge Bases (KBs), also referred to as Knowledge Graphs (KGs), containing millions of entities and billions of factual relationships [16]. Moreover, a key characteristic is their machine-readable nature [9]. Examples of such datasets include DBpedia[1], Wikidata[2], YAGO[3], among others.

Relating data published on the web to data in such KGs may mitigate the complexity and inherent ambiguity of natural language, as well as enrich these KGs. This purpose may be achieved by aligning entity mentions obtained from the data published on the web, such as references to people or places, and entity representations in knowledge bases, hereinafter referred to as *entities*. In addition, this alignment aid a wide variety of problems, such as populating knowledge bases, content analysis, relation extraction, and question-answering systems [23]. This alignment is one of the main tasks in Natural Language Processing (NLP) [7, 9] and is defined as Entity Linking (EL), also known as Named Entity Linking, Named Entity Disambiguation, or Entity Disambiguation.

Formally, given a document D containing a set of named entity mentions $M = \{m_1, m_2, ..., m_{|M|}\}$ along with its context, and a KG containing a set of named entities $E = \{e_1, e_2, ..., e_{|E|}\}$, the goal is to define a function f that maps each entity mention $m_i \in M$ to its corresponding entity $e_j \in E$ [23].

However, the entity mentions present in sentences are in natural language, whereas the KG data in LOD is represented by graphs. Specifically, these graphs adhere to the *Resource Description Framework* (RDF[4]), in which information or facts are represented as interconnected nodes linked by edges. The connection between two nodes, via an edge, constitutes a triple, composed of a subject (the first node), a predicate or property (the edge), and an object (the second node) ($subject \rightarrow predicate \rightarrow object$). The edge denotes the relationship between the two nodes, which can represent entities or literals (entity data). A collection of these triples constitutes a directed and labeled graph known as an RDF graph, which also serves as a Knowledge Graph [8, 17].

Figure 1 illustrates the Entity Linking process. The rectangle on the left contains a snippet of text in natural language. There are two mentions of entities (people). In this case, the mentions are *Barack Obama* and *Lolo Soetero*. On the right, we illustrate a part of a KG with several RDF triples. In this example, the entities are *Q4115068* (*Lolo Soetero*) and *Q76* (*Barack Obama*) and their properties are represented by the *Label* and *Description* edges. The connections between the mentions of entities and their corresponding entities in the KG are symbolized by dashed arrows. EL employs the contexts of the mentions, represented by the full text in the rectangle, and the contexts of the entities in the KG, indicated by the literal nodes linked to the entities, in its processing

Many current NLP tasks rely on vector representations of their data. In this context, each element (term, phrase, object, node, ...) needs to be represented by feature vectors $\langle f_1, f_2, ..., f_n \rangle$. These vectors usually contain binary ($f_i \in \{True, False\}$), numerical ($F_i \in \mathbb{R}$) or nominal ($f_i \in S$, where S is a finite set of symbols) values [20, 21]. The task of mapping this data to vectors can be accomplished through embedding. The main idea is to represent the meaning of a piece of natural language (such as text, images, or audio) using dense, low-dimensional vectors with real-valued

[1]https://www.dbpedia.org/
[2]https://www.wikidata.org/
[3]https://yago-knowledge.org/

[4]https://www.w3.org/RDF/

In: Proceedings of the Brazilian Symposium on Multimedia and the Web (WebMedia'2024). Juiz de Fora, Brazil. Porto Alegre: Brazilian Computer Society, 2024.

ISSN 2966-2753

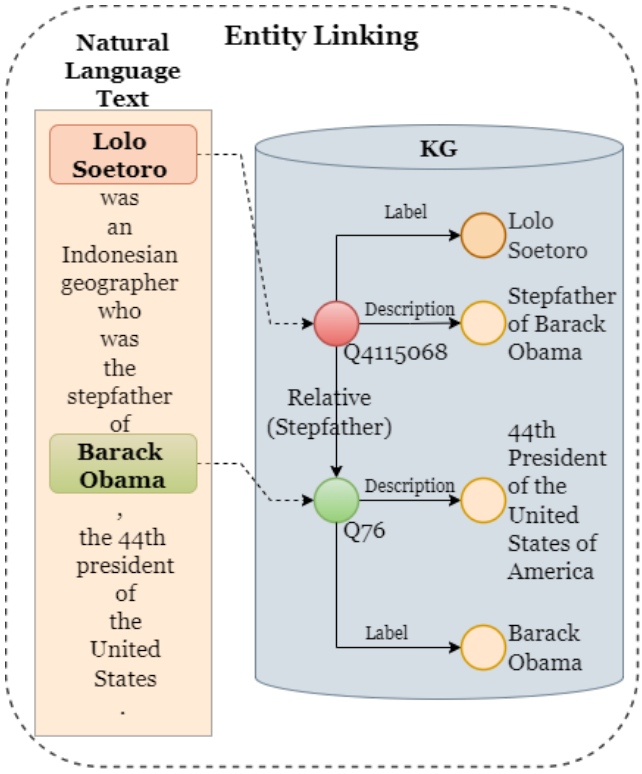

Figure 1: Illustration of Entity Linking. The rectangle on the left contains a snippet of text in natural language (mention context) and highlights two entity mentions in rounded rectangles. The cylinder on the right represents a portion of a KG. The dashed arrows represent the linkages from the mentions to the entities in the KG.

components. These vectors are also referred to as latent representations [23].

Furthermore, these vectors are designed to capture semantic similarity. In the context of words, this means that similar words should be close to each other in the vector space [27]. Additionally, distributed representations of words in dense vectors help learning algorithms achieve better results in NLP tasks, due, for example, to the proximity of similar words, considering the distance between them in the vector space [14].

In addition to the challenge of obtaining vector representations of mentions and entities, EL must address the complexity and ambiguity of natural language, since entity mentions may be written in many ways. For instance, a name can be written in full, partial, abbreviated, or even nickname forms (such as the abbreviation "LA" and the nickname "City of Angels" for the city of "Los Angeles"). Moreover, mentions can ambiguously refer to multiple entities. For instance, the mention "Washington" could refer to the American state of "Washington", the capital city of the United States, "Washington DC", the actor "Denzel Washington", or even the first U.S. president, "George Washington" [7, 9, 12, 23, 27].

For the reasons mentioned above, we propose in this work a simple and effective approach for EL, called E-BELA, an acronym for "Enhanced Embedding-Based Entity Linking Approach". E-BELA integrates entity mentions from a text with the entities present in the KG. We use embedding representations to put mention and its entity close in a common vector space, considering their semantic context. Our hypothesis is that this approach will allow for more precise and coherent linking between mentions and entities by calculating a similarity or distance metric. In simplified terms, based on the calculated similarity or distance between the representation of a mention and the representations of entities, it will be capable of linking the mention to the closest entity, whether considering similarity or distance metrics. It is essential to emphasize that the context surrounding the mention must be taken into account.

Considering that entity mentions are in the form of natural language and the entities present in the KG have object nodes that describe them in the same way, our hypothesis is that obtaining representations of these entities based on their literals can guarantee that their vector representations are in the same vector space of the mentions, and, we hope, are also close in that space.

Based on the context, problem, and objectives presented, the following research questions have been raised:

- Question 1: Does adopting embeddings of entities from a KG in a vector space, based on the embeddings of their literals, provide advantages in terms of effectiveness for the EL task compared with other approaches documented in the literature?
- Question 2: Will the vector representations of KG literals and KG entities retain the semantic contexts of their corresponding literals?
- Question 3: Does the number of properties and, consequently, literals associated with KG entities impacts EL accuracy?

Thus, our main contributions include: the proposal of an EL approach that in an experimental study outperforms existing approaches; the representation of DBpedia through embeddings, applicable to tasks that include semantic similarity, such as clustering, textual similarity, semantic search, even allows recovering entities semantically similar to a mention in this KG.

The rest of this paper is organized as follows. Section 2 describes related work. Section 3 describes our approach. Section 4 describes the experimental evaluation and discusses its results. And Section 5 presents the conclusion and future work.

2 RELATED WORK

EL is an essential task in Natural Language Processing, widely used in several downstream applications, such as question-answering systems, relation extraction, knowledge base population, content analysis, and so on [23]. Therefore, over time, this challenge has attracted a wide variety of solutions [12].

Initially, many works in this field were based on traditional machine learning techniques, relying on local context compatibility, global coherence, manually designed features (such as entity popularity), and rule-based methods [24]. However, the rapid development of deep learning techniques has led to new approaches that outperform the results of previous ones [12]. In general, the EL process involves two subtasks: (1) candidate entity generation, and (2) entity disambiguation. Different solutions have been employed to address these subtasks.

The following works share the common characteristic of jointly performing entity and relation linking. Additionally, they obtain the candidate entities using indices generated by ElasticSearch[5], with data from various sources, including Wikidata and DBpedia. In [11], a method called JLEAR is proposed. JLEAR aims to explore both independent and joint features of the candidates for disambiguation. Independent features include Entity Popularity, Literal Similarity (Levenshtein distance), and Semantic Similarity. Joint features leverage the correlation between candidate entities and relations. In [22], the authors propose Falcon 2.0, which employs a disambiguation method based on two modules: the first one called "Matching and Classification", and the second called "Relevant Rule Selection". "Matching and Classification" module combines entities and relations from the candidate list into RDF triples and verifies their existence in the knowledge graph (KG). Relations and entities related to existing triples receive higher scores. The "Relevant Rule Selection" module interacts with the previous module, suggesting score adjustments for some candidates based on a pre-built rule catalog. Finally, Dubey et al. [7] propose a framework called EARL. In EARL, disambiguation relies on two strategies: Generalized Travelling Salesman Problem (GTSP) and Connection Density. GTSP evaluates the shortest path between combinations of entities and relations in the candidate lists, while Connection Density is based on features such as the number of connections and hops in the knowledge graph.

Other works employ a variety of strategies. In [2], the authors propose modeling the EL task using reinforcement learning. They introduce high-level and low-level procedures and policies for mention detection and disambiguation. In [12], an approach called KGEL (Knowledge Graph-based Entity Linking) is proposed to enrich the EL process by incorporating structural information from the knowledge graph. KGEL utilizes both local and global features to evaluate the mapping between mentions and entities. In [28], the authors propose entity disambiguation using a model based on BERT [4]. This model is similar to the BERT Masked Language Model but is trained to predict masked entities. Finally, in [27], a method based on embeddings is proposed. It jointly embeds words and entities into the same vector space using three Skip-gram models[13, 14]. Its disambiguation relies on two contexts: textual context similarity that is based on the similarity between entity and word vectors, and coherence that is based on a target entity and other related entities. That method utilizes candidate lists created in [18].

Among the mentioned works, a variety of resources are employed for the EL task. Highlights include: use of local and global features [7, 9, 12, 27, 28]; embeddings [2, 12, 27, 28]; algorithms based on Neural Networks and Reinforcement Learning [2, 7, 9].

In E-BELA, we obtain the embeddings of literals by transfer learning, a simplified process that eliminates the need for additional training or fine tuning. This approach allows E-BELA to obtain the list of candidate entities to be linked to a mention directly from the KG. Many works depend on external data. Furthermore, disambiguation is performed using the contexts of mentions and entities. This enables E-BELA to achieve effective results when dealing with the dynamic contexts of KGs, demonstrating the model's robustness in adapting to different scenarios without the need for complex training interventions.

[5]https://www.elastic.co/products/elasticsearch

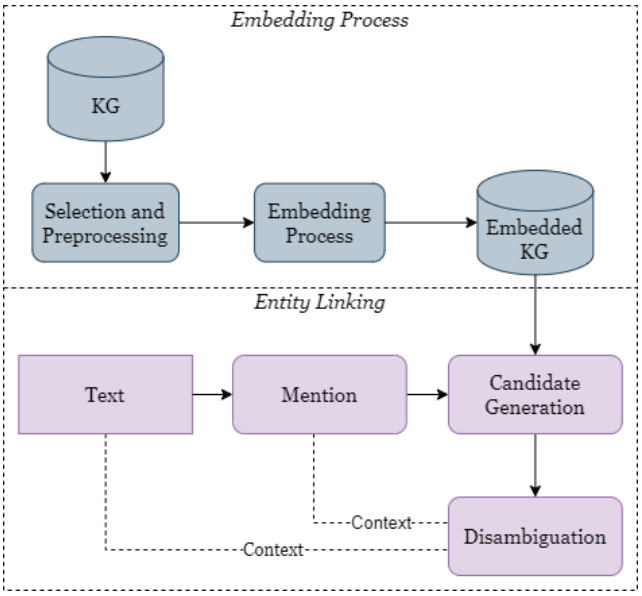

Figure 2: Overview of E-BELA. The top lane illustrates the process of embedding the KG entities. The bottom lane illustrates the entity linking process, from generating candidate entities to subsequent ambiguity resolution.

3 E-BELA

3.1 Overview

Our proposed approach, E-BELA, aims to put representations of mentions and their corresponding entities close together in a vector space, enabling to perform EL by applying a similarity metric. All artifacts comprising E-BELA are available for download at the following address: https://github.com/italompereira/E-BELA.

Figure 2 provides a general overview of E-BELA. In the upper lane, the embedding process takes place, which includes the selection and preprocessing of a KG data, followed by the actual embedding process that obtains the entity vector representation. Next, E-BELA stores these vector representation into a vector database. The lower lane illustrates the entity linking process. E-BELA obtains candidate entities for mentions, as well as their vector representation. And, finally, It uses the context information to disambiguate and provide the corresponding entities to the mentions. It is worth highlighting that it is not part of the scope of our work to identify mentions in a natural language text; tools such as spaCy[6] or NLTK[7] can be used for this task.

3.2 Embedding Process

This section discusses the process of embedding, from data selection and preprocessing to the acquisition of vector representations for entities.

[6]https://spacy.io/
[7]https://www.nltk.org/

Since entity mentions in texts are in natural language and the entities in KGs have object nodes that describe them also in natural language, as said before, it is possible to obtain their representations based on their literals and thus ensure that their vector representations are in the same vector space. Additionally, we hope that semantically similar mentions and entities would have close vector representations.

3.2.1 Data Selection and Preprocessing. The KG datasets, such as DBpedia and Wikidata, are available for download through specific web pages. In this work, we used the DBpedia data containing ontology information, and links to other datasets, beyond to the data. We collected files from "https://downloads.dbpedia.org/2016-10/" that contain literal data describing the entities, as well as those containing the necessary references for disambiguating the entities.

We structure those files into an extensive three-column Apache Spark[8] DataFrame: $\langle subject \rangle$ $\langle predicate \rangle$ $\langle object \rangle$. Apache Spark was chosen for its ability to handle large volumes of data and its efficiency in performing queries, similar to a relational database management systems.

3.2.2 Embedding Process. In this subsection, we discuss the process of obtaining vector representations for literals and entities by E-BELA.

These vector representations can be obtained by training a language model or by transferring learning from pre-trained models. In this work, we choose transfer learning from pre-trained models. Those models are trained on large data corpora, ensuring higher accuracy, and making possible to reduce the computational cost and time.

We straightly obtain the vector representation for the entity mentions by using those models, and, for the KG entities, we apply a two-step process: In the first step, E-BELA obtains vector representations for the literal nodes. In the second step, it obtains a vector representation for each entity by aggregating the vector representations from its literals.

Literal Embeddings. For the first step, we evaluated the USE [1][9] and *all-mpnet-base-v2*[10] models. The latter is one of the original models from SBERT [19] for sentence encoding. Both USE and SBERT encode text into low-dimensional vectors, which can be utilized for tasks such as text classification, semantic similarity, clustering, and other natural language processing tasks.

We chose these models because they are considered state-of-the-art in encoding sentences into embedding vectors, according to the results presented [1] and [19]. Furthermore, they specifically aim to transfer learning to other NLP tasks.

The strategy for obtaining embeddings involves providing literals as input to the model, which processes them and generates representative multidimensional vectors. Equation 1 expresses this operation, where *embedding* is the function that takes a literal l_i (e.g., a word or sentence) as input and returns its vector representation $\vec{l_i}$ as output. These vectors encapsulate the semantics contained within the literals.

$$\vec{l_i} = embedding(l_i) \tag{1}$$

To conduct the experiments in this work, we sort the data previously assigned to the Spark DataFrame (see Subsection 3.2.1) by the subject attribute. This sorting allows us iterate on the data frame entity by entity. Next, E-BELA selects the literals in English language. DBpedia identifies such literals by using the '@en' language tag , such as "*story and dialogues*"@en, for instance. Additionally, E-BELA preprocesses the literals by removing characters belonging to $\{(,), ', '', ., ,, :, <, >, ?, !, @, \$, \%, \&\}$.

According to the USE documentation, there is no limitation on the input size. However, as longer as the input text more "diluted" is its embedding. The SBERT model, specifically *all-mpnet-base-v2*, restricts input size to 384 characters. The USE model produces real-valued vectors with 512 dimensions, whereas the *all-mpnet-base-v2* model generates real-valued vectors with 768 dimensions. These vectors are primarily useful for semantic search tasks and semantic textual similarity.

Entity Embeddings. Aiming to get an entity vector representation close to its corresponding mention, we chose to obtain it from vector representations from its literals, instead of using a graph-specific embedding technique such as RDF2Vec[20], for instance. E-BELA obtains the entity vector representation by averaging its literal associated vectors. We also evaluated other strategies for obtaining the entity vector representation, including vector summation and weighting based on literal frequencies in the KG. But, averaging yielded the best results. Equation 2 expresses this operation for obtaining the vector representation of the j-th entity e_j, where L_{e_j} represents the set of literal vectors associated with that entity, and $\vec{l_i}$ is the i-th vector of a literal in that set.

$$\vec{e}_j = \frac{1}{|L_{e_j}|} \sum_{i=1}^{|L_{e_j}|} \vec{l_i}, \forall l \in L_{e_j} \tag{2}$$

Figure 3 shows a simplified illustration of our proposed approach. In this figure, entities Q76 and Q4115068 from Wikidata represent *"Barack Obama"* and his stepfather *"Lolo Soetoro"*, respectively, along with their literals. Initially, literal vector representations for *"Barack Obama", "44th president of the United States of America", "Stepfather of Barack Obama"* are obtained using Equation 1. In the figure, these representations appear as inner circles within orange background. After obtaining the vector representations of the literals, E-BELA obtains the entity vectors by using Equation 2. In Figure 3, the vector representation of entity Q76 is calculated based on the average of the vectors of its literals. The vectors involved in this calculation are highlighted by green arrows whose labels are $eQ76$. The vector for entity Q4115068 is also obtained based on the average of its vectors, in this case, only the literal vector of *"Stepfather of Barack Obama"*.

Considering the amount of vector generated using a dataset such as DBpedia, which implies in a vast search space during the entity linking process, we adopted PostgreSQL 14.12 and along with the pgvector[11] plugin as our vector database management system. This plugin enables PostgreSQL to perform exact and approximate

[8]https://spark.apache.org/
[9]https://tfhub.dev/google/universal-sentence-encoder/4
[10]https://huggingface.co/sentence-transformers/all-mpnet-base-v2
[11]https://github.com/pgvector/pgvector

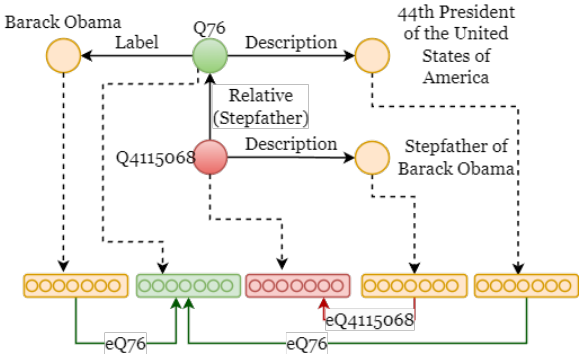

Figure 3: The figure illustrates the acquisition of vector representations for entities. In this example, entities Q76 and Q4115068 are displayed, along with their literals. The obtained vectors are represented by rectangles containing multiple circles.

searches (indexing the data) for nearest neighbors, using metrics such as L2 distance (Euclidean), inner product, cosine distance, L1 distance (Manhattan), Hamming distance, and Jaccard distance. PostgreSQL efficiently indexes and retrieves close vectors using a distance function.

3.3 Entity Linking

Generally, the EL process is carried out in two steps: (i) obtaining the candidate list and (ii) performing the disambiguation process. Since the vector representations of the entities and their literals are arranged in a vector space, these tasks can be accomplished using similarity or distance metrics.

3.3.1 Candidate Generation. We gather the candidate list for a mention comparing mention vector with entity vectors, by using cosine similarity. We also compare a mention vector with the literal vector to compose the candidate list, i.e., the candidate list contains the n most similar vectors, among entity and literal vectors, compared with the mention vector. Formally, Equation 3 obtains such a list.

$$candidates_{m_i} = sort(sim(embedding(m_i), E \cup L))[:n], \quad (3)$$

where the function $embedding(m_i)$ returns the mention vector of the mention m_i, the function sim returns the similarity values between the mention vector and all entity and literal vectors, E is the set of entity vectors, L is the set of literal vectors, $sort$ orders the similarity results, and $[:n]$ obtains the top n sorted results.

We chose the cosine similarity function based on: (i) usually entities from a KG have properties, as *name* and *label*, whose vectors are close to the corresponding mention vector. For instance, the vector mention of "Japan" tends to be more similar with entities whose properties *name* or *label* contain the term "Japan". (ii) the cosine similarity function allows us retrieval distant candidates considering Euclidean distance but the angles to the mention vector are small. The usefulness of this method is related to the fact that the embeddings carry relevant semantic information, and some models position these distant entities with a small angle.

To illustrate, consider the following sentence: "Japan scored two goals against China". In this sentence, the mentions "Japan" and "China" individually describe Asian countries. However, within the context of the sentence, their meaning is related to soccer teams. Cosine similarity enables to retrieve soccer teams as candidate entities even when they are more distant, considering Euclidean distance.

Equation 4 calculates the cosine similarity,

$$sim_{cos}(\vec{v}_e, \vec{v}_m) = \frac{\vec{v}_e \cdot \vec{v}_m}{\|\vec{v}_e\|\|\vec{v}_m\|} \quad (4)$$

where $\vec{v}_e$ represents the vector of an entity and $\vec{v}_m$ represents the vector of a mention, $\cdot$ (dot) represents the scalar product between the vectors, and $\|\vec{v}\|$ represents the Euclidean norm of a vector.

3.3.2 Disambiguation. To perform disambiguation, it is necessary to consider the context of the mentions. In the previous example, "Japan scored two goals against China", "Japan" and "China" are the mentions, and the whole sentence constitutes the context. Although the mentions seem to refer to Asian countries, the semantic context reveals that they correspond, in fact, to soccer teams representing those countries.

As seen, performing the disambiguation of a mention using its representation in isolation is not a good alternative, since entities representing their Asian countries would be considered more similar. Thus, we must use the vector representation of their context. But, in such a sentence, we have two mentions and we must generate different candidate lists for both mentions. If we use only a vector representation of the context, entities representing the soccer teams of these countries could be retrieved as well as soccer teams of other countries. Alternatively, we could retain the mention target and remove other mentions from the sentence, but removing a mention within this sentence implies in a context less informative.

Thus, we choose to combine the mention vector of a mention target with a context vector based on the whole sentence. We average such vectors for combining them, similar to how we obtain the entity vectors.

Furthermore, in the disambiguation process, we use Euclidean distance as distance function, instead of the cosine similarity function. Mean vectors represent central points among vectors involved in the operation. Thus, applying Euclidean distance seems more promising than using the cosine similarity function. Nonetheless, we evaluated both functions.

In this scenario, the closest entity, considering the Euclidean distance, is the entity to be linked to the mention. Equation 5 performs the Euclidean distance,

$$d(\vec{v}_e, \vec{v}_m) = \sqrt{\sum_{i=1}^{n} (\vec{v}_{e_i} - \vec{v}_{m_i})^2} \quad (5)$$

where n is the number of dimensions of the vectors $\vec{v}$ and i is the index of each dimension.

Table 1: The DBpedia files used in this work

Files
infobox_properties_en.ttl.bz2
instance_types_en.ttl.bz2
labels_en.ttl.bz2
long_abstracts_en.ttl.bz2
mappingbased_literals_en.ttl.bz2
mappingbased_objects_en.ttl.bz2
persondata_en.ttl.bz2
disambiguations_en.ttl.bz2
infobox_property_definitions_en.ttl.bz2
specific_mappingbased_properties_en.ttl.bz2

4 EVALUATION

4.1 Experimental Setup

Datasets

For evaluating our proposal, we use DBpedia (version 2016-10)[12], a robust knowledge graph composed of millions of RDF triples. Specifically, we focus on files containing literal data that describes entities, including the *disambiguations_en.ttl.bz2* file, which disambiguates different Uniform Resource Identifiers (URIs) about the same entities. Table 1 details the files (downloaded from https://downloads.dbpedia.org/2016-10/core/) used in our experiments. Those files contain 119, 157, 509 RDF triples, about 35, 318, 483 entities and 27, 370, 487 literals.

We evaluated E-BELA using the LC-QuAD dataset [26]. The LC-QuAD is a dataset of complex questions available for evaluating Question Answering Systems over KGs. It contains 5,000 questions and their respective SPARQL queries over the DBpedia dataset. In [7], the authors adapted LC-QuAD as a benchmark dataset for entity and relation linking. Each question contains the mapping between mentions and URIs of entities and relations from the KG, along with the corresponding part of the text in the question. According to the authors, the annotation process was carried out semi-automatically and then manually reviewed. The annotated dataset is publicly available at https://figshare.com/articles/dataset/Full_Annotated._LC_QuAD_dataset/5782197.

LC-QuAD has 6,612 links between mentions and entities, with 3,963 unique entities. We did not find 64 entities in the KG, reducing the number to 3,899 unique entities. We randomly selected 370 of the aforementioned links as a sample of the population for evaluating E-BELA. Adopting a confidence level of 95% , the estimated margin of error for this sample is 5%.

Evaluation Metric

To quantify the efficiency of E-BELA, we employed accuracy as the performance metric, aligning with methodological guidelines from previous studies. Accuracy is defined as the ratio between the total number of correctly identified entities and the total number of entities. In general, it represents the proportion of correct predictions (both true positives and true negatives) relative to the total evaluated observations, as expressed by Equation 6.

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Predictions}} \tag{6}$$

Baselines

We selected recognized state-of-the-art works as baselines for the EL task: EARL[7], Falcon 2.0[22], and JLEAR[11]. A common feature of these works is performing EL alongside relation linking, an approach we intend to explore in the future.

In these works, EL involves processing a list of candidates followed by disambiguation. Notably, none of them directly performed semantic entity candidate search in the KG. However, E-BELA does perform this search. For this purpose, they used ElasticSearch to create an index of mention-URI pairs with data from external sources and other KGs.

EARL and JLEAR evaluated EL performance using accuracy on the LC-QuAD evaluation set. Falcon 2.0 adopted precision, recall, and *f-score* as evaluation metrics. The evaluation was conducted on the LC-QuAD 2.0 test set. However, the authors did not describe how mention-entity alignment was performed, as this set is not annotated for the EL task. Their performance, based on LC-QuAD accuracy, was obtained from the results reported in [11].

In [11], the authors used entity disambiguation information from DBpedia. This information is available through a file containing mappings of disambiguated URIs. Such ambiguity exists in the KG and can impact model results. Consider the following RDF triples as an example:

dbr:David_Bowen, disambiguates, dbr:David_Bowen_(cricketer),
dbr:David_Bowen, disambiguates, dbr:David_Bowens

The URIs dbr:David_Bowen, dbr:David_Bowen_(cricketer) e dbr:David_Bowens ambiguously represent the same entity.

Computational Environment

We conducted the experimental evaluation of this work on a Dell Alienware R15 computer, with Windows 11 Home Edition operating system, equipped with an Intel i9-13900K processor, an NVIDIA GeForce RTX 4070Ti graphics card, and 32GB of RAM. For programming, we used Python 3.7.16 as the programming language. The libraries we used include Apache Spark 3.3.4 with Hadoop 3, TensorFlow 2.10.1, Torch 1.10.1+cu113, and Numpy 1.21.6. Spark depends on the availability of the installed Java Virtual Machine, in this case, JVM 17.0.9 was available. As the Database Management System, we utilized PostgreSQL 14.12 along with the pgvector plugin.

4.2 Results and Discussion

In this study, we conducted experiments structured around the established research questions, as outlined in Section 1. The method employed aimed to obtain empirical data to systematically address these inquiries. We designed each experiment specifically to test the related hypotheses and collect relevant information, ensuring the validity and reliability of the obtained results.

[12]https://downloads.dbpedia.org/2016-10/

Table 2: E-BELA compared with baselines

Approach	Accuracy
EARL	0.65
Falcon	0.74
JLEAR	0.83
E-BELA (USE) without disambiguation	0.68
E-BELA (USE) with disambiguation	0.74
E-BELA (SBERT) without disambiguation	0.78
E-BELA (SBERT) with disambiguation	**0.84**

Table 3: The E-BELA performance in different scenarios

Approach	Accuracy
E-BELA (USE) (literals only)	0.742
E-BELA (USE) (entities only)	0.711
E-BELA (USE) (both)	0.745
E-BELA (SBERT) (literals only)	0.836
E-BELA (SBERT) (entities only)	0.829
E-BELA (SBERT) (both)	0.840

4.2.1 Experiment 1: Question 1: Does adopting embeddings of entities from a KG in a vector space, based on the embeddings of their literals, provide advantages in terms of effectiveness for the EL task compared with other approaches documented in the literature?

We assessed the effectiveness of E-BELA by conducting a comparative analysis with previously established baselines. We used accuracy as the performance metric to quantify the efficiency of E-BELA. Table 2 shows the results.

Although we cannot make a direct comparison due to differences in the sample space used in this work and the baselines, the data shown in Table 2 highlight the validity of the proposed approach, considering the higher accuracy achieved by E-BELA. It is worth noting that the transfer learning model whose accuracy outperformed the others was the *all-mpnet-base-v2*, associated with SBERT. We theorize that the performance of the USE model may have been compromised during the embedding process because we did not limit the size of the input sentences. Despite the lower accuracy achieved by the USE model, its result is comparable to that obtained in Falcon[22]. Furthermore, the results of E-BELA, using both models, are substantially superior to those presented in EARL[7].

It is also important to emphasize that unlike the JLEAR approach, proposed in [11], no training process or fine-tuning of the model was adopted. However, we have the hypothesis that the process of fine-tuning the models can improve the results obtained by E-BELA, we intend to evaluate this hypothesis in future work.

Thus, as in [11], we utilized the disambiguation information provided by DBpedia. This information allows us validating whether a prediction made by the model, initially incorrect, is an ambiguous reference.

However, due to the volume of data or the constant updates in DBpedia, this file cannot resolve all the ambiguous references present. For instance, our model linked the mention "Us congress" to the entity "http://dbpedia.org/resource/Us_congress", but the correct entity labeled by LC-QuAD is "http://dbpedia.org/resource/United_States_Congress". Upon manual verification, we note that both URIs refer to the same entity, and the disambiguation file does not contain information about this ambiguity. This suggests that the accuracy of the model may be even higher than reported in Table 2.

Furthermore, it is worth mentioning that due to the nature of the approach proposed by E-BELA, we can search for candidate entities directly in DBpedia, unlike the baselines, which use a list of mention-URI pairs constructed externally. The baselines depend on data external to KG.

4.2.2 Experiment 2: Question 2: Will the vector representations of KG literals and KG entities retain the semantic contexts of their corresponding literals?

To investigate this question, we conducted experiments to compare the accuracy of E-BELA on the EL task in different scenarios. Since we obtain vector representations of entities from their literals and store both entity and literal representations, we can evaluate the performance of E-BELA using vector representations in exclusive or integrated way. The analyzed scenarios include: (i) using only literal vectors; (ii) using only entity vectors; and (iii) using both literal and entity vectors. This experiment aims to clarify how representative the aggregation approach used by E-BELA is compared with isolated literal representations and using both literal and entity representations. Table 3 shows the results of the experiments conducted, considering the different proposed scenarios.

Importantly, we can obtain candidate entities for mentions from literal representations because each vector representation of a literal is associated with the literal itself and the corresponding KG entity. This allows us to retrieve lists of candidate entities directly from literal representations.

Table 3 shows that entity vectors, when used in isolation, are capable of maintaining much of the context obtained from their literals, despite the slightly inferior result. It is worth noting that the best result was obtained through the use of both vector representations, including literals and entities. Considering the USE model, the percentage difference between the best result and the isolated representation of entities was approximately 4.55%. Considering the model used by SBERT, the difference was approximately 1.31%.

These results demonstrate that vector representations, both of literals and KG entities, are capable of retaining the semantic context of the literals. Furthermore, the use of both representations allowed the model to achieve a better result.

4.2.3 Experiment 3: Question 3: Does the number of properties and, consequently, literals associated with KG entities impacts EL accuracy?

To explore this question, we collected quantitative data related to the literal values associated with KG entities. We conducted the analysis from three perspectives: First, we considered descriptive metrics (see Table 4). Second, we examine the performance variation of E-BELA based on the number of literals per entity (see Table 5). Lastly, we analyzed the relationship between the number of literals in correct and predicted entities within the set of incorrect predictions (see Figure 4).

Table 4 summarizes the statistical metrics of the literals of the entities in the KG, present in the LC-QuAD set. The metrics include

Table 4: Descriptive Statistics of Literals per Entity in the KG.

Set	Entities	Mean	Std Dev
Population	3899	4.691	3.348
Sample	358	4.782	2.464
Errors	59	4.500	2.760

Table 5: Analysis of EL Performance Based on the Number of Literals per Entity

Number of Literals	Number of Entities	%
1	3	0.666
2	59	0.813
3	107	0.850
4	30	0.700
5	24	0.791
6	25	0.840
7+	122	0.893

the total number of entities, the average, and the standard deviation of the number of literals per entity.

The data presented in Table 4 reveal that the distribution of the literals associated with the entities shows a limited variation, fluctuating between 4.50 and 4.78 literals per entity. We observed the largest deviation in the total population. Moreover, the deviation in the set of incorrect predictions slightly exceeded that in the sample set.

Table 5 presents the variation in E-BELA's EL performance based on the number of literals per entity. Each row contains: the number of literals under analysis in the first column; the number of entities containing exactly that quantity in the second column; and the percentage of correct predictions in the third column. The last row considers entities that have 7 or more literals.

The data in Table 5 does not allow us to infer a direct relationship between the number of literals per entity and the percentage of correct predictions. We notice that entities with three literals have a slightly higher percentage of correct predictions than those with only two literals. However, this value decreases for entities containing four or five literals. The accuracy rate increases again for entities with six or more literals, but the numbers remain statistically insignificant.

Figure 4 presents a matrix, analogous to a confusion matrix, which instead of displaying classes, shows the actual and predicted quantities of literals per entity in the set of incorrect predictions made during the experiments.

Based on the data presented in this matrix, we can observe a significantly higher number of incorrectly predicted entities in the upper left corner, corresponding to those with fewer available literals. However, the number of entities with a smaller number of associated literals (2 and 3 literals) is significantly larger, and the number of errors is proportional.

After analyzing the data from Tables 4 and 5, as well as Figure 4, we noticed that although entities with 7 or more literals exhibit a slightly higher accuracy, we cannot definitively assert that the number of literals significantly impacts the effectiveness of EL.

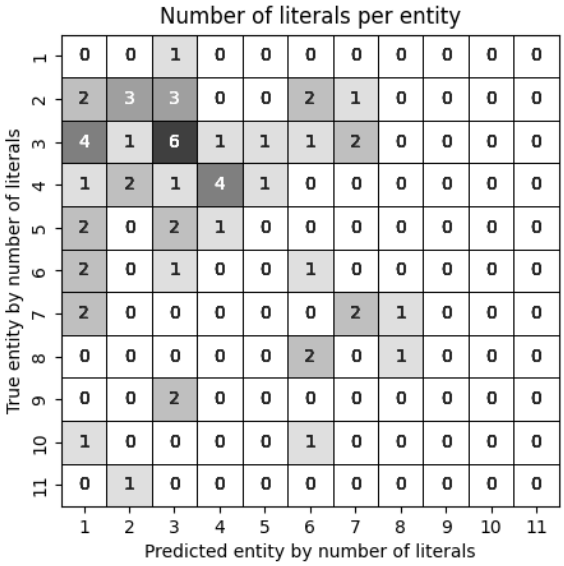

Figure 4: This figure illustrates a matrix showing the relationship between the number of literals in the correct and predicted entities in the set of incorrect predictions.

5 CONCLUSIONS AND FUTURE WORK

In this work, we introduce E-BELA (Enhanced Embedding-Based Entity Linking Approach), a straightforward and effective approach for EL. E-BELA obtains entity embeddings from a KG using its literal data presented in the form of natural language, the same form used in texts and entity mentions, and stores them in a vector database management system. These embeddings aim to position mentions and entities close in the vector space, enabling their linkage through some similarity metric. The EL process occurs through the search for a list of candidate entities, from the embedding of a mention, followed by disambiguation. For disambiguation, context information from the mention and candidate entities is used. Our results demonstrate that this methodology outperforms previous approaches, making a significant contribution to the field of NLP.

In future work, we intend to perform the task of Relation Linking in an integrated manner through the embeddings of the entities and relations of the KG. Additionally, we plan to evaluate the EL task performance after a fine-tuning the all-mpnet-base-v2 model with mention and entity contexts. We also intend to apply E-BELA to other problems that involve EL in their pipeline. Furthermore, we aim to enhace the evaluation process by using a sample with a balanced quantity of literal data.

ACKNOWLEDGMENTS

This research is partially sponsored by the Universidade Federal de Ouro Preto - UFOP, the Instituto Federal de Minas Gerais - IFMG, CNPq, FAPEMIG and CAPES.

REFERENCES

[1] Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2018. Universal Sentence Encoder. *CoRR* abs/1803.11175 (2018). arXiv:1803.11175 http://arxiv.org/abs/1803.11175

[2] Lihan Chen, Tinghui Zhu, Jingping Liu, Jiaqing Liang, and Yanghua Xiao. 2023. End-to-End Entity Linking with Hierarchical Reinforcement Learning. *Proceedings of the AAAI Conference on Artificial Intelligence* 37, 4 (Jun. 2023), 4173–4181. https://doi.org/10.1609/aaai.v37i4.25534

[3] Lucas Colucci, Prachi Doshi, Kun-Lin Lee, Jiajie Liang, Yin Lin, Ishan Vashishtha, Jia Zhang, and Alvin Jude. 2016. Evaluating Item-Item Similarity Algorithms for Movies. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems* (San Jose, California, USA) *(CHI EA '16)*. Association for Computing Machinery, New York, NY, USA, 2141–2147. https://doi.org/10.1145/2851581.2892362

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, Jill Burstein, Christy Doran, and Thamar Solorio (Eds.). Association for Computational Linguistics, Minneapolis, MN, USA, 4171–4186. https://doi.org/10.18653/V1/N19-1423

[5] Tommaso Di Noia, Roberto Mirizzi, Vito Claudio Ostuni, Davide Romito, and Markus Zanker. 2012. Linked Open Data to Support Content-Based Recommender Systems. In *Proceedings of the 8th International Conference on Semantic Systems* (Graz, Austria) *(I-SEMANTICS '12)*. Association for Computing Machinery, New York, NY, USA, 1–8. https://doi.org/10.1145/2362499.2362501

[6] Tommaso Di Noia and Vito Claudio Ostuni. 2015. *Recommender Systems and Linked Open Data*. Springer International Publishing, Cham, 88–113.

[7] Mohnish Dubey, Debayan Banerjee, Debanjan Chaudhuri, and Jens Lehmann. 2018. EARL: Joint Entity and Relation Linking for Question Answering over Knowledge Graphs. In *The Semantic Web – ISWC 2018*, Denny Vrandečić, Kalina Bontcheva, Mari Carmen Suárez-Figueroa, Valentina Presutti, Irene Celino, Marta Sabou, Lucie-Aimée Kaffee, and Elena Simperl (Eds.). Springer International Publishing, Cham, 108–126.

[8] Jorão Gomes, Rômulo Chrispim de Mello, Victor Ströele, and Jairo Francisco de Souza. 2022. A Hereditary Attentive Template-based Approach for Complex Knowledge Base Question Answering Systems. *Expert Systems with Applications* 205 (2022), 117725. https://doi.org/10.1016/j.eswa.2022.117725

[9] Ningning Jia, Xiang Cheng, Sen Su, and Liyuan Ding. 2021. CoGCN: Combining co-attention with graph convolutional network for entity linking with knowledge graphs. *Expert Systems* 38, 1 (2021), e12606. https://doi.org/10.1111/exsy.12606 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/exsy.12606

[10] Hongkun Leng, Caleb De La Cruz Paulino, Momina Haider, Rui Lu, Zhehui Zhou, Ole Mengshoel, Per-Erik Brodin, Julien Forgeat, and Alvin Jude. 2018. Finding similar movies: dataset, tools, and methods. In *Proceedings of the 8th International Conference on Semantic Systems (WSCG'2018)*. Václav Skala-UNION Agency, Plzen, Czech Republic, 115–124.

[11] Huiying Li, Wenqi Yu, and Xinbang Dai. 2023. Joint linking of entity and relation for question answering over knowledge graph. *Multimedia Tools and Applications* 82, 29 (01 Dec 2023), 44801–44818. https://doi.org/10.1007/s11042-023-15646-w

[12] Qijia Li, Feng Li, Shuchao Li, Xiaoyu Li, Kang Liu, Qing Liu, and Pengcheng Dong. 2022. Improving Entity Linking by Introducing Knowledge Graph Structure Information. *Applied Sciences* 12, 5 (2022), 44801–44818. https://doi.org/10.3390/app12052702

[13] Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. In *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*, Yoshua Bengio and Yann LeCun (Eds.). Association for Computing Machinery, Scottsdale, Arizona, USA. http://arxiv.org/abs/1301.3781

[14] Tomás Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. *CoRR* abs/1310.4546 (2013). arXiv:1310.4546 http://arxiv.org/abs/1310.4546

[15] Roberto Mirizzi, Tommaso Di Noia, Azzurra Ragone, Vito Ostuni, and Eugenio Di Sciascio. 2012. Movie recommendation with DBpedia, In Movie recommendation with DBpedia. *CEUR Workshop Proceedings* 835, 101–112.

[16] Jean Gabriel Nguema Ngomo, Giseli Rabello Lopes, Maria Luiza Machado Campos, and Maria Claudia Reis Cavalcanti. 2020. An Approach for Improving DBpedia as a Research Data Hub. In *Proceedings of the Brazilian Symposium on Multimedia and the Web* (São Luís, Brazil) *(WebMedia '20)*. Association for Computing Machinery, New York, NY, USA, 65–72. https://doi.org/10.1145/3428658.3431075

[17] Ítalo M. Pereira and Anderson A. Ferreira. 2019. An Item-Item Similarity Approach Based on Linked Open Data Semantic Relationship. In *Proceedings of the 25th Brazillian Symposium on Multimedia and the Web* (Rio de Janeiro, Brazil) *(WebMedia '19)*. Association for Computing Machinery, New York, NY, USA, 425–432. https://doi.org/10.1145/3323503.3349547

[18] Maria Pershina, Yifan He, and Ralph Grishman. 2015. Personalized Page Rank for Named Entity Disambiguation. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Rada Mihalcea, Joyce Chai, and Anoop Sarkar (Eds.). Association for Computational Linguistics, Denver, Colorado, 238–243. https://doi.org/10.3115/v1/N15-1026

[19] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan (Eds.). Association for Computational Linguistics, Hong Kong, China, 3982–3992. https://doi.org/10.18653/v1/D19-1410

[20] Petar Ristoski and Heiko Paulheim. 2016. RDF2Vec: RDF Graph Embeddings for Data Mining. In *The Semantic Web – ISWC 2016*, Paul Groth, Elena Simperl, Alasdair Gray, Marta Sabou, Markus Krötzsch, Freddy Lecue, Fabian Flöck, and Yolanda Gil (Eds.). Springer International Publishing, Cham, 498–514.

[21] Petar Ristoski, Jessica Rosati, Tommaso Di Noia, Renato De Leone, and Heiko Paulheim. 2019. RDF2Vec: RDF graph embeddings and their applications. *Semantic Web* 10, 4 (2019), 721–752.

[22] Ahmad Sakor, Kuldeep Singh, Anery Patel, and Maria-Esther Vidal. 2020. Falcon 2.0: An Entity and Relation Linking Tool over Wikidata. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management* (Virtual Event, Ireland) *(CIKM '20)*. Association for Computing Machinery, New York, NY, USA, 3141–3148. https://doi.org/10.1145/3340531.3412777

[23] W. Shen, Y. Li, Y. Liu, J. Han, J. Wang, and X. Yuan. 2023. Entity Linking Meets Deep Learning: Techniques and Solutions. *IEEE Transactions on Knowledge; Data Engineering* 35, 03 (mar 2023), 2556–2578. https://doi.org/10.1109/TKDE.2021.3117715

[24] Wei Shen, Jianyong Wang, and Jiawei Han. 2015. Entity Linking with a Knowledge Base: Issues, Techniques, and Solutions. *IEEE Transactions on Knowledge and Data Engineering* 27, 2 (Feb 2015), 443–460. https://doi.org/10.1109/TKDE.2014.2327028

[25] Uma Srinivasan and Chidambaram Mani. 2018. Diversity-Ensured Semantic Movie Recommendation by Applying Linked Open Data. *International Journal of Intelligent Engineering and Systems* 11 (04 2018), 275–286.

[26] Priyansh Trivedi, Gaurav Maheshwari, Mohnish Dubey, and Jens Lehmann. 2017. LC-QuAD: A Corpus for Complex Question Answering over Knowledge Graphs. In *The Semantic Web – ISWC 2017*, Claudia d'Amato, Miriam Fernandez, Valentina Tamma, Freddy Lecue, Philippe Cudré-Mauroux, Juan Sequeda, Christoph Lange, and Jeff Heflin (Eds.). Springer International Publishing, Cham, 210–218.

[27] Ikuya Yamada, Hiroyuki Shindo, Hideaki Takeda, and Yoshiyasu Takefuji. 2016. Joint learning of the embedding of words and entities for named entity disambiguation. In *CoNLL 2016 - 20th SIGNLL Conference on Computational Natural Language Learning, Proceedings (CoNLL 2016 - 20th SIGNLL Conference on Computational Natural Language Learning, Proceedings)*. Association for Computational Linguistics (ACL), United States, 250–259. https://doi.org/10.18653/v1/k16-1025 Publisher Copyright: © 2016 Association for Computational Linguistics.; 20th SIGNLL Conference on Computational Natural Language Learning, CoNLL 2016 ; Conference date: 11-08-2016 Through 12-08-2016.

[28] Ikuya Yamada, Koki Washio, Hiroyuki Shindo, and Yuji Matsumoto. 2022. Global Entity Disambiguation with BERT. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Marine Carpuat, Marie-Catherine de Marneffe, and Ivan Vladimir Meza Ruiz (Eds.). Association for Computational Linguistics, Seattle, United States, 3264–3271. https://doi.org/10.18653/v1/2022.naacl-main.238