Journal of the Brazilian Computer Society, 2025, 31:1, doi: 10.5753/jbcs.2025.5427

A sentence similarity-based approach for enhancing entity linking

Ítalo M. Pereira [**Federal University of Ouro Preto; Federal Institute of Minas Gerais** | ***italo.pereira@ifmg.edu.br***]
Anderson A. Ferreira [**Federal University of Ouro Preto** | ***anderson.ferreira@ufop.edu.br***]

Federal Institute of Minas Gerais, Av. Primeiro de Junho, 1043, Centro, São João Evangelista, MG, 39705-000, Brazil.

Received: 15 January 2025 • **Accepted:** 07 June 2025 • **Published:** 08 August 2025

Abstract Entity linking involves associating mentions of entities in natural language texts, such as references to people or locations, with specific entity representations in knowledge graphs like DBpedia or Wikidata. This process is essential in natural language processing tasks, as it aids in disambiguating entities in unstructured data, thereby improving comprehension and semantic processing. However, entity linking faces challenges due to the complexity and ambiguity of natural languages, as well as discrepancies between the forms of textual entity mentions and entity representations. Building upon our previous work, this study extends the E-BELA –Enhanced Embedding-Based Entity Linking Approach, which is based on literal embeddings. We extend our previous work by evaluating E-BELA using a new dataset, conducting a comprehensive analysis of failure cases and limitations, and providing further discussion of our results. E-BELA associates mentions and entity representations using a similarity or distance metric between vector representations of them in a shared vector space. The results suggest that our approach achieves comparable performance to other state-of-the-art methods, while employing a much simpler model, contributing to the field of natural language processing.

Keywords: Natural Language Processing, Entity Linking, Linked Open Data, Sentence Encoder, Sentence Similarity, Entity Similarity, Embedding, Disambiguation, Knowledge Graph

1 Introduction

The exponential growth of data published on the Web has ushered in an era of information overload. Persons generate more information than they can actually process and consume [Mello *et al.*, 2024; Naseem *et al.*, 2021; Srinivasan and Mani, 2018; Leng *et al.*, 2018; Colucci *et al.*, 2016; Di Noia and Ostuni, 2015; Di Noia *et al.*, 2012; Mirizzi *et al.*, 2012]. Additionally, natural language, one of the most important forms of data on the Web, is inherently complex and ambiguous. In this context, the term "natural" refers to languages used by humans for communication, writing or speaking [Caseli *et al.*, 2024].

To address this scenario, many efforts have been made. Among them, 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), often organized as Knowledge Graphs (KGs), containing millions of entities and billions of factual relationships [Ngomo *et al.*, 2020]. Moreover, a key characteristic is their machine-readable nature [Jia *et al.*, 2021b]. Examples of such datasets include DBpedia[1], Wikidata[2], YAGO[3], among others.

Associating 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 KGs, hereinafter referred to as *entities*. In addition, this alignment aid a wide variety of tasks, such as populating knowledge bases, content analysis, relation extraction, and question-answering [Shen *et al.*, 2023]. This alignment is one of the main tasks in Natural Language Processing (NLP) [Dubey *et al.*, 2018; Jia *et al.*, 2021b] and is defined as Entity Linking (EL), also known as Named Entity Linking, Named Entity Disambiguation, or Entity Disambiguation.

However, 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 (e.g., 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 US president, "George Washington" [Shen *et al.*, 2023; Yamada *et al.*, 2016; Dubey *et al.*, 2018; Jia *et al.*, 2021b; Li *et al.*, 2022].

Many NLP tasks are based on vector representations of the data. Such vectors, encompassing both feature vectors and embeddings, represent each element (such as term, phrase, object, node, ...) as a sequence of features $\langle f_1, f_2, ..., f_n \rangle$. These vectors usually contain binary values ($f_i \in \{True, False\}$), numerical values ($F_i \in \mathbb{R}$), or nominal val-

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

ues ($f_i \in S$, where S is a finite set of symbols) values. These representations enable models to process diverse structured data, making them essential in computational learning tasks tasks [Ristoski and Paulheim, 2016; Ristoski *et al.*, 2019].

Feature vectors typically encode discrete attributes like word frequency, part-of-speech tags, syntactic dependencies, or named entity categories. In contrast, embeddings play a complementary role by capturing semantic relationships. 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 components. These vectors are also referred to as latent representations [Shen *et al.*, 2023].

Furthermore, these embedding vectors are designed to capture semantic similarity, specifically distributive semantics. In the context of word representations, similar words should be close to each other in the vector space [Yamada *et al.*, 2016]. 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 [Mikolov *et al.*, 2013b].

In addition to the challenge of obtaining vector representations of mentions and entities, 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 and facts are represented as interconnected nodes linked by edges. A directed and labeled graph known as an RDF graph, which also serves as a Knowledge Graph [Heath and Bizer, 2011; Pereira and Ferreira, 2019; Gomes *et al.*, 2022].

For the reasons mentioned above, Pereira and Ferreira [2024] proposed a simple and effective approach for EL, called E-BELA, an acronym for "Enhanced Embedding-Based Entity Linking Approach". E-BELA associates entity mentions from a text with their corresponding entities in a KG. In that work, we used embedding representations to put mention and its entity close in a common vector space, considering their semantic context. Our hypothesis was that this approach would allow for more precise and coherent linking between mentions and entities by calculating their similarity or distance. In short, based on the similarity or distance between the representations of mentions and entities, E-BELA will be capable of linking the mention to the closest entity, using similarity or distance metric. It is essential to emphasize that the context surrounding the mention must be taken into account.

This article expands upon our preliminary findings presented in [Pereira and Ferreira, 2024]. Building on our prior work, we enhance our experimental scope by incorporating a new evaluation dataset, conducting a comprehensive analysis of failure cases, acknowledging the limitations of our approach, and providing a more detailed discussion of our results.

Given that entity mentions are expressed in the form of natural language and the entities in the KG are described using object nodes with natural language descriptions, we hypothesize that obtaining entity embeddings based on their literal data ensures that their vector representations are in the same vector space of the mentions, and, we hope, are also close within that space.

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

Based on the context, problem, and objectives presented, we pose the following research questions:

- RQ 1: Does adopting entity embeddings derived from the embeddings of their associated literals provide advantages in terms of effectiveness for the EL task compared to other approaches documented in the literature?
- RQ 2: Do the vector representations of KG literals and entities retain the semantic contexts of their corresponding literals?
- RQ 3: Does the number of predicates and, consequently, literals associated with KG entities impacts EL accuracy?

Thus, our main contributions include: (1) the proposal of an EL approach that in our experimental study achieves comparable performance to the existing approaches. Furthermore, our approach effectively addresses several challenges of natural language by leveraging the semantics present in entity literals within a KG, as well as in mentions and their contexts. By integrating this rich semantic information, we mitigate the complexity and ambiguity inherent in natural language, thereby establishing more precise and coherent links between mentions and their corresponding entities; (2) additionally, we represent DBpedia through embeddings, which are applicable to tasks involving semantic similarity, such as clustering, textual similarity, and semantic search. Moreover, this representation enables the recovery of entities semantically similar to a given mention within the KG; (3) Additionally, as previously mentioned, we extend our previous work [Pereira and Ferreira, 2024] by incorporating a new evaluation dataset, conducting a comprehensive analysis of failure cases, and identifying the limitations of our approach. These refinements enable a more detailed discussion of our findings, offering deeper insights into the strengths and challenges of our methodology.

The remainder of this article is organized as follows. Section 2 provides the theoretical background by formally defining KG, EL and discussing key aspects of Sentence Encoders. Section 3 presents a comprehensive overview of related work in this field. Section 4 outlines the details of our proposed approach. Section 5 describes the experimental evaluation and analyses the results. Finally, Section 6 summarizes the key findings, discusses limitations, and outlines potential avenues for future research.

2 Background

In this section, we provide an overview of the theoretical background that guided the development of E-BELA. We begin by formally defining what constitutes a KG, then present a definition of EL. Subsequently, we discuss two sentence encoder models that were employed in E-BELA.

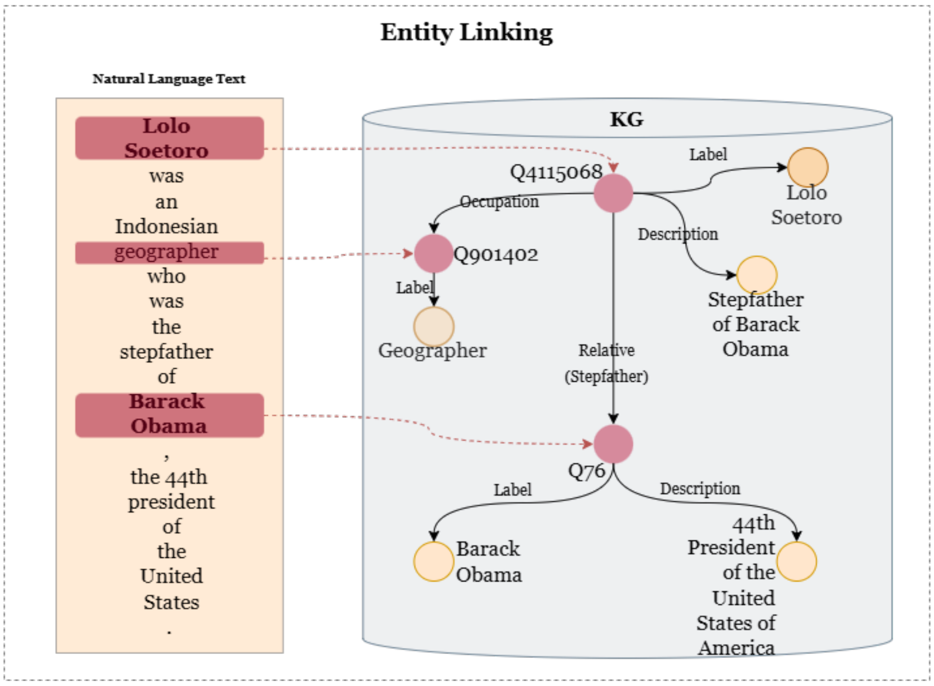
Entity Linking
Natural Language Text
Lolo Soetoro was an Indonesian geographer who was the stepfather of Barack Obama , the 44th president of the United States .
KG
Q4115068
Label
Lolo Soetoro
Occupation
Description
Q901402
Label
Geographer
Stepfather of Barack Obama
Relative (Stepfather)
Q76
Label
Description
Barack Obama
44th President of the United States of America

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.

2.1 Knowledge Graph

A KG is a graph-based data structure where nodes represent entities or literals, and edges represent the relationships between them. This structure allows for the storage and retrieval of interconnected information in a way that is both human-readable and machine-interpretable, enabling various applications such as semantic search, question answering, and recommendation systems. Formally, a KG is defined as a collection of subject–predicate–object triples, denoted as $\langle s,\ r,\ o\rangle$. In this formulation: E represents the set of entities, R represents the set of relations, and L represents the set of literals. Within each triple, $s \in E$ denotes an entity serving as the subject, $r \in R$ represents the relation or predicate connecting elements, and $o \in (E \cup L)$ indicates that the object can be either another entity or a literal value [Li *et al.*, 2023].

2.2 Entity Linking

As previously discussed, EL involves associating mentions within textual data to corresponding entities within KGs, creating a foundation for utilizing the knowledge stored in structured data. This provides semantic information that enables a better understanding of texts. EL typically involves a three-step process: mention recognition: identifying the span of text that constitutes an entity mention; candidate generation: for each entity mention $m \in M$, the EL system filters candidate entities; disambiguation: selecting the most likely entity to be linked to the mention [Li *et al.*, 2023].

EL can be formally defined as follows. 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 $e_j \in E$ [Shen *et al.*, 2023].

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 persons and one mention of professional occupation. In this case, the mentions are *Barack Obama*, *Lolo Soetero* and *geographer*. On the right, we illustrate a part of a KG with several RDF triples. In this example, the entities are *Q4115068* (*Lolo Soetero*), *Q76* (*Barack Obama*) and *Q901402* (geographer) and their properties are represented by the *Label*, *Description* and *Occupation* edges. The connections between the mentions of entities and their corresponding entities in the KG are symbolized by dashed arrows. The EL task can leverage the context surrounding the mentions, such as the surrounding text in the rectangle in Figure 1, and the contextual information associated with entities within the KG, such as their descriptions and related information available in the literal nodes linked to the entities.

2.3 Sentence Encoders

In this section, we discuss models that have long been recognized as state-of-the-art in sentence encoding, focusing on key works that have significantly contributed to the field. These works are employed in this study to enhance the accuracy and effectiveness of EL.

A sentence embedding is a function that maps a sentence—an ordered sequence of words—to a fixed-dimensional vector capturing its semantic content. Formally, let S be the set of all sentences over a vocabulary V and consider a sentence $s \in S$ represented as $s = (w_1, w_2, ..., w_n)$ with each $w_i \in V$. A sentence embedding is defined as a function $f : S \to R^d$, where d is the dimensionality of the embedding space [Stankevičius and Lukoševičius, 2024].

In [Cer *et al.*, 2018], the authors propose the Universal Sentence Encoder (USE) for encoding sentences into embedding vectors. They introduce two models: one based on Transformers [Vaswani *et al.*, 2017] and another based on the Deep Averaging Network [Iyyer *et al.*, 2015]. These models are designed to facilitate transfer learning for various tasks. USE accepts English text of varying lengths as input and produces fixed-dimensional 512-dimensional embedding vectors as output. The training of USE involves both unsupervised data from Wikipedia and other sources, as well as supervised data from the Stanford Natural Language Inference (SNLI) corpus.

In [Reimers and Gurevych, 2019], the authors introduce Sentence-BERT (SBERT), a modification of the pre-trained BERT network. SBERT employs siamese and triplet network structures to generate semantically meaningful sentence embeddings, which can be compared using cosine similarity. To achieve this, SBERT incorporates a pooling operation to the output of BERT, resulting in fixed-size sentence embeddings. The authors explored various training objectives, including classification with softmax, regression with mean-squared error, and triplet loss. Initially, SBERT models were trained on two prominent Natural Language Inference (NLI) datasets: SNLI [Bowman *et al.*, 2015] and Multi-Genre NLI [Williams *et al.*, 2018]. Building upon SBERT, numerous models have been developed. In this work, we utilize the SBERT-based model *all-mpnet-base-v2*, trained on over one billion sentence pairs, which maps sentences and paragraphs to 768-dimensional vectors.

3 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 [Shen *et al.*, 2023]. Therefore, over time, this challenge has attracted a wide variety of solutions [Li *et al.*, 2022].

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 [Shen *et al.*, 2015]. However, the rapid development of deep learning techniques has led to new approaches that outperform the results of previous ones [Li *et al.*, 2022]. As said before, the EL process involves three subtasks: (1) mention recognition, (2) candidate entity generation, and (3) entity disambiguation. Various approaches have been employed to address these subtasks.

Table 1 provides an overview of the works related to the EL task, listing the methods utilized for each subtask of EL. The second column presents the methods employed for mention recognition, while the third column outlines the methods used for candidate generation, highlighting that all works utilize external sources for candidate generation. The fourth column lists the methods applied for disambiguating. The final column enumerates the datasets used in those works for experimental evaluation. Note that, the first three works perform EL and RL jointly. Below, we detail those works.

3.1 Joint Linking of Entities and Relations

The following works share the common characteristic of jointly performing entity and relation linking. Additionally, they obtain the candidate entities using ElasticSearch[5].

In [Li *et al.*, 2023], a method called JLEAR is proposed. JLEAR performs mention recognition by jointly labeling entity and relation mentions through end-to-end neural networks, specifically utilizing BERT, BiLSTM, and CRF. The subtask of candidate generation is executed using Elasticsearch to create and index lists of pairs consisting of entity mention-URIs and relation mention-URIs. 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 [Sakor *et al.*, 2020], the authors introduce Falcon 2.0. The mention recognition phase of Falcon 2.0 consists of three modules: POS tagging, tokenization & compounding, and N-Gram tiling. For candidate generation, they used indexes of mention-URI pairs built with Elasticsearch. Falcon 2.0 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 combines entities and relations from the candidate list into RDF triples and verifies their existence in the KG. Relations and entities related to existing triples receive higher scores. Relevant Rule Selection interacts with the previous module, suggesting score adjustments for some candidates based on a pre-built rule catalog.

Finally, Dubey *et al.* [2018] propose a framework called EARL. EARL performs mention recognition through shallow parsing, which involves extracting keywords and classifies them as entity or relation mentions. As in previously cited works, Dubey *et al.* [2018] used Elasticsearch indexes to generate candidates. In EARL, disambiguation relies on two strategies: Generalized Traveling 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 KG.

3.2 Entity Linking

The following works perform EL in an isolated manner, employing a variety of strategies.

In [Chen *et al.*, 2023], 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. The authors detect mentions at a high level using a transformer followed by a feed-forward network. In this work, they consider a pre-computed set of candidates for entity disambiguation. For disambiguation, they employ a state-of-the-art generative entity retrieval method proposed in [Cao *et al.*, 2021].

In [Luo *et al.*, 2023], the authors propose a model for entity linking through a deep semantic modeling of sentence

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

Table 1. Overview of related works.

Authors	Mention Recognition	Candidate Generation	Linking	Datasets
Li *et al.* [2023] (JLEAR)	BERT+BiLSTM+CRF	Elasticsearch	Independent and Joint Features	LC-QuAD QALD-7
Sakor *et al.* [2020] (Falcon 2.0)	Catalog of rules	Elasticsearch	"Matching Ranking" and "Relevant Rule Selection"	LC-QuAD 2.0 SimpleQuestion
Dubey *et al.* [2018] (EARL)	Keywords and LSTM	Elasticsearch	GTSP and Connection Density	LC-QuAD QALD-7
Chen *et al.* [2023] (Reinforcement learning)	Transformer + FNN (Softmax)	Generative Entity Retrieval	Generative Entity Retrieval	AIDA-CoNLLR500, OKE15, OKE16, Der
Li *et al.* [2022] (KGEL)	-	Use local and global models (Wikidata and Wikipedia)	Use local and global models (Wikidata and Wikipedia)	AIDA-CoNLL, MSNBC, AQUAINT, ACE2004
Yamada *et al.* [2022]	-	PPRforNED dataset, Elasticsearch	BERT Based Masked Language Model (Softmax over candidates entities)	AIDA-CoNLL, MSNBC, AQUAINT, ACE2004
Luo *et al.* [2023]	-	Local and Global Models	Local and Global Models	AIDA-CoNLL, MSNBC, AQUAINT, ACE2004
Jia *et al.* [2021a]	-	Elasticsearch	Siamese Network (BERT based)	AIDA, KBP2017
Yamada *et al.* [2016] Extended Skip-gram Model (Wikipedia Link-based Measure, Anchor Context Model)	-	PPRforNED dataset, Elasticsearch	Text Similarity Textual context Modeling Coherence	AIDA-CoNLL, TAC 2010

representations. The model first leverages unsupervised contrastive learning to optimize a BERT-based semantic space, thereby producing more expressive sentence embeddings. These enriched embeddings are further refined via an attention mechanism that facilitates semantic interactions among them, ensuring a robust capture of the contextual nuances present in the text. This strategy is similar to the one proposed in [Jia *et al.*, 2021a]. Additionally, the model incorporates supplementary sentence-level similarity features, extracted from both the surrounding context and entity descriptions, which are integrated with the local similarity signals inherent in traditional entity linking frameworks, this strategy is based on [Le and Titov, 2018].

In [Li *et al.*, 2022], the authors propose KGEL (Knowledge Graph-based Entity Linking) to enrich the EL process by incorporating structural information from the knowledge graph. To generate candidates, the authors employed a method proposed in [Le and Titov, 2018]. KGEL utilizes both local and global models to evaluate the mapping between mentions and entities. The local model calculates a score based on the similarity between context embedding and the entity embedding from candidate set. In the global model, the scores between the candidate entities of all mentions in the document are taken as the global score.

In [Yamada *et al.*, 2022], the authors propose using two sets of candidate entities, one for each specific set of evaluation datasets. They also propose entity disambiguation using a BERT-based model [Devlin *et al.*, 2019], which is similar to the BERT Masked Language Model but trained to predict masked entities.

In [Jia *et al.*, 2021a], the authors proposed a sentence representation-based model for Entity Linking, utilizing a BERT-based Siamese Neural Network to effectively capture sentence similarity. Its input layer processes two sentences: one containing the entity mention and its surrounding context, and the other providing a description of the selected entity from Freebase. The architecture incorporates a BERT encoder, followed by an attention layer and a Multi-Layer Neural Network. Similar to other studies, Elasticsearch was utilized to index candidate generation.

Finally, in [Yamada *et al.*, 2016], the authors propose an embedding-based method. As in [Yamada *et al.*, 2022], the authors use two sets of candidate entities, one for each specific set of evaluation datasets. For disambiguating, they jointly embeds words and entities into the same vector space using three Skip-gram models[Mikolov *et al.*, 2013b,a]. The 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 [Pershina *et al.*, 2015].

Among the mentioned works, a variety of resources are employed for the EL task, including: use of local and global features [Li *et al.*, 2022; Jia *et al.*, 2021b; Dubey *et al.*, 2018; Yamada *et al.*, 2022, 2016]; embeddings [Yamada *et al.*, 2016, 2022; Li *et al.*, 2022; Chen *et al.*, 2023]; algorithms based on Neural Networks and Reinforcement Learning [Dubey *et al.*, 2018; Jia *et al.*, 2021b; Chen *et al.*, 2023].

Although the approaches detailed in [Luo *et al.*, 2023] and [Jia *et al.*, 2021a] also employ sentence representations for entity linking—similar to our proposed model—E-BELA offers several advantages. Unlike these referenced methods, E-BELA utilizes pretrained models trained on extensive corpora without any fine-tuning. Moreover, candidate generation is directly derived from vector representations produced by encoders, enhancing efficiency. Another key advantage of E-BELA is its ability to handle multiple entity mentions within natural language queries, making it a robust and scalable solution for EL tasks.

In E-BELA, we obtain literal embeddings by leveraging transfer learning, a simplified process that eliminates the need for additional training or fine tuning. This allows E-BELA to obtain the list of candidate entities to be linked to a mention directly from the KG. Many existing approaches rely on external data sources. Furthermore, we disambiguate by 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.

4 E-BELA

4.1 Overview

As mentioned, we proposed E-BELA in our previous work [Pereira and Ferreira, 2024]. E-BELA aims to put representations of mentions and their corresponding entities close together within a vector space, enabling to perform EL by applying a similarity/distance 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 KG data, followed by the actual embedding process that obtains the entity vector representation. Next, E-BELA stores these vector representations in a vector database. The lower lane illustrates the entity linking process, where E-BELA retrieves candidate entities for mentions based on their vector representations. Finally, it uses context information of the mentions to disambiguate and associate 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. We explain the processes in the upper and lower lanes in the following subsections.

4.2 Embedding Process

In this subsection, we discuss the process of embedding, from data selection and preprocessing to the generation of vector representations for entities.

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

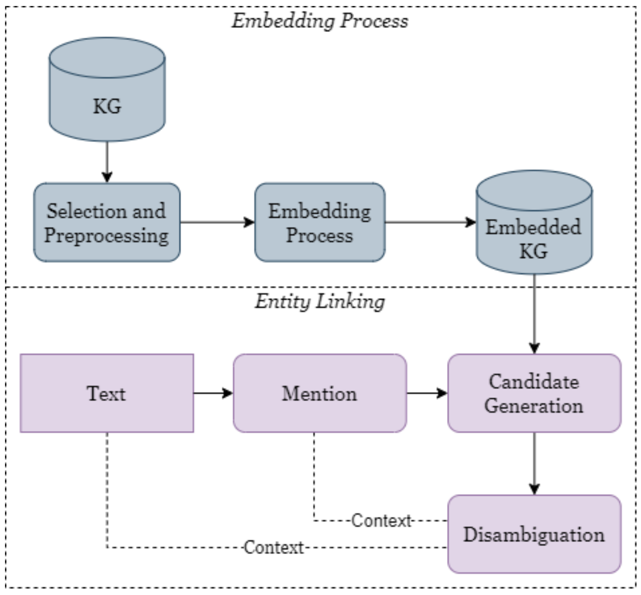

Figure 2. Overview of E-BELA. The top lane illustrates the process of embedding the KG entities. The bottom lane illustrates the entity linking process, from generating candidate entities to subsequent ambiguity resolution.

4.2.1 Data Selection and Preprocessing

The KG datasets, such as DBpedia and Wikidata, are available for download through specific web pages. In our work, we used DBpedia data, which includes ontology information and links to other datasets, as well as the data. We gathered files from "`https://downloads.dbpedia.org/2016-10/`" that contain literal data describing the entities, as well as files 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 due to its ability to handle large volumes of data and its efficiency in performing queries, similar to a relational database management systems.

4.2.2 Embedding Process

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

These representations can be obtained by training a language model or through transfer learning from pre-trained models. In our work, we chose transfer learning from pre-trained models, which means that we did not perform any type of training or fine-tuning of the models. These models were trained on large data corpora, ensuring higher accuracy and decreasing the computational cost and time.

We directly obtain vector representations for entity mentions using these models. For KG entities, we employ 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.

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

[8]https://spark.apache.org/

Literal Embeddings

In the first step, we evaluated the USE [Cer *et al.*, 2018][9] and *all-mpnet-base-v2*[10] models. For more details, refer to subsection 2.3. The latter is one of the original models from SBERT [Reimers and Gurevych, 2019] for sentence encoding. Both USE and SBERT encode text into low-dimensional vectors, and can be utilized for text classification, semantic similarity, clustering, and other natural language processing tasks.

We chose these models because they have long been recognized as state-of-the-art methods for encoding sentences into embedding vectors, as demonstrated in [Cer *et al.*, 2018] and [Reimers and Gurevych, 2019]. Although large language models (LLMs) can sometimes deliver superior performance, they typically incur much higher computational costs. Furthermore, these models are specifically designed for transfer learning applications in NLP tasks.

The strategy for obtaining embeddings involves inputting literals into the model, which processes them and generates representative multidimensional vectors. Equation 1 expresses this operation, where $embed(l_i)$ 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.

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

To conduct the experiments, we first sorted the data previously assigned to the Spark DataFrame (see subsection 4.2.1) by the 'subject' attribute. This sorting allows us to iterate on the data frame entity by entity. Next, E-BELA selects triples with 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 the following characters: $\{(,),', '', ., ,, :, <, >, ?, !, @, \$, \%, \&\}$.

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

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. We obtain literal vector representations for *"Barack Obama", "44th president of the United States of America", "Stepfather of Barack Obama"* using Equation 1. In the figure, these representations are depicted as squares with an orange background.

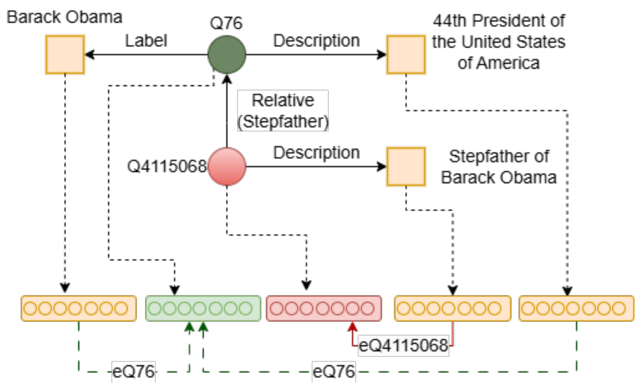

Figure 3. The figure illustrates the acquisition of vector representations for entities. In this example, we have Q76 and Q4115068, along with their literals. The rectangles containing multiple circles represent the vectors.

Subsequently, E-BELA obtains the entity vectors by averaging the vector representations of their respective literals. We hypothesize that constructing an entity's representation by averaging the semantic vectors of its associated literals provides an efficient and effective method for positioning the entity at a central point within the semantic space. In Figure 3, the vector representation of entity Q76 is calculated by averaging the vectors of its literals. The vectors involved in this calculation are highlighted by dashed green arrows whose labels are $eQ76$. We also obtain the vector for entity Q4115068 by averaging its vectors, in this case, only the literal vector of *"Stepfather of Barack Obama"*.

For obtaining entity vector representation close to vector representation of its corresponding mention, we chose to use the vector representations from its literals to infer its vector representation. This approach differs from methods like RDF2Vec [Ristoski and Paulheim, 2016], which utilize graph-specific embedding techniques.

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,

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

where L_{e_j} represents the set of literal vectors associated with that entity, and $\vec{l_i}$ is the i-th vector of the i-th literal in the set.

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

[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

4.3 Entity Linking

Generally, after mention recognition, we carry out EL 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 distributed within the vector space, these tasks can be accomplished using similarity or distance metrics.

4.3.1 Candidate Generation

We generate a candidate list for a given mention by comparing the cosine similarity between its vector representation with both entity and literal vectors from the KG. The candidate list contains the n most similar vectors, encompassing both entity and literal vectors, compared with the mention vector. This process is formally defined by Equation 3,

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

where the function $embed(m_i)$ 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, the function $sort$ orders the similarity results, and $[: n]$ obtains the top n sorted results. In our experiments, we assigned the value $n = 100$.

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 representation of the mention "Japan" tends to be more similar with entities whose properties $name$ or $label$ contain the term "Japan". (ii) the cosine similarity function enables us to retrieval candidates, that are distant using the euclidean distance metric, but have small angular difference from the mention vector representation. 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.

4.3.2 Disambiguation

The disambiguation process aims to associate a given mention with a single entity selected from a list of candidates. As mentioned, this candidate list is generated by comparing the cosine similarity between the vector representation of the target mention and the vectors representing the entities in the KG, which comprise both literal and entity vectors. However, it is anticipated that some literal vectors associated with these entities may not be included in the initial list. To address this, the list is augmented by incorporating all literal vectors corresponding to the entities already present. Consequently, the final candidate list encompasses all literal vectors associated with these entities, alongside the entity vectors themselves. The disambiguation process is then performed on this comprehensive list.

Furthermore, 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.

Disambiguating a mention solely based on its individual vector representation can be problematic. For example, in the case of Asian countries, only entities representing those countries should exhibit higher similarity scores. Therefore, incorporating the context of the mention is crucial. However, in sentences containing multiple mentions, generating distinct candidate lists for each mention is a challenge. If we utilize a single vector representation for the entire context, we may retrieve entities representing the soccer teams of the mentioned countries alongside teams from other countries. Alternatively, we could isolate the target mention by removing other mentions from the sentence. However, this approach can significantly diminish the informativeness of the context.

Thus, we combine the mention vector of a target mention with a context vector derived from the entire sentence. For this, we employ a simple averaging strategy, similar to that used for obtaining entity vectors.

Furthermore, during the disambiguation process, we employ Euclidean distance as distance function instead of the cosine similarity. 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 most probable entity to be linked to the mention.

5 Evaluation

5.1 Experimental Setup

Datasets

We evaluated E-BELA using DBpedia (version 2016-10)[12] as our KG, a robust knowledge graph composed of millions of RDF triples. We specifically focus on files containing literal data that describes entities, including the 'disambiguations_en.ttl.bz2' file, which provides mappings between different URIs that refer to the same real-world entity. Table 2 details the files used in our experiments (downloaded from `https://downloads.dbpedia.org/2016-10/core/`). These files contain $119,157,509$ RDF triples, representing $35,318,483$ entities and $27,370,487$ literals.

To evaluate E-BELA, we utilized two widely-used benchmark datasets: QALD-7 [Usbeck *et al.*, 2017] and LC-QuAD [Trivedi *et al.*, 2017]. Originally designed for the Question Answering over Linked Data task, the QALD-7

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

Table 2. 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

dataset contains natural language questions with entity mentions and SPARQL queries that include entity URIs. However, despite its extensive use in previous research detailed annotations mapping entity mentions to their corresponding entities were not available.

The QALD-7 dataset comprises 215 training questions and 43 test questions. Given that our methodology does not involve model training or fine-tuning, and in alignment with established evaluation practices—where the test subset is typically used exclusively—we opted to exclude the training data from our analysis. To ensure a robust evaluation, we carefully annotated the 43 test questions using available SPARQL queries, establishing 54 links between mentions and entities.

To promote reproducibility, we have made both the annotated dataset and our source code publicly available on GitHub, as detailed in subsection 4.1. Our evaluation specifically focuses on the 'qald-7-test-multilingual' subset.

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 [Dubey *et al.*, 2018], the authors adapted LC-QuAD as a benchmark dataset for entity and relation linking. Each question incorporates mappings between mentions and URIs of entities and relations from the KG, along with the corresponding part of the text in the question. The authors employed a semi-automated annotation process followed by manual review to ensure data quality. The annotated dataset is publicly available on `https://figshare.com/articles/dataset/Full_Annotated._LC_QuAD_dataset/5782197`.

For evaluating E-BELA using the LC-QuAD dataset, we randomly selected a sample of 370 links between mentions and entities out of the 6,612 available from LC-QuAD. The LC-QuAD dataset encompasses 3,899 unique entities, as we excluded 64 entities not found within the KG. Adopting a confidence level of 95% , the estimated margin of error for this sample is 5%.

Evaluation Metric

To evaluate E-BELA's performance, we employed accuracy as the primary metric. This choice aligns with established methodological practices in prior research. Furthermore, since EL is a single-label multiclass problem where each mention is linked to one unique entity, accuracy, micro precision, micro recall, and micro F1 yield identical values. The aggregation of true positives, false positives, and false negatives across all classes ensures that these metrics all reflect the same performance outcome [Shen *et al.*, 2015].

Accuracy is defined as the ratio correctly linked entity-mention pairs to the total number of evaluated pairs. In general, it represents the proportion of correct predictions (both true positives and true negatives) relative to the total observations evaluated, as expressed by Equation 4.

$$Accuracy = \frac{Number\ of\ Correct\ Predictions}{Total\ Predictions} \tag{4}$$

Baselines

To compare our work, we selected recognized EL state-of-the-art as baselines: EARL [Dubey *et al.*, 2018], Falcon 2.0 [Sakor *et al.*, 2020], and JLEAR [Li *et al.*, 2023]. A common feature of these works is performing EL alongside relation linking, an approach we intend to explore in the future.

EL typically involves two key steps: generating a candidate list followed by disambiguation. Notably, none of the baselines directly performed semantic entity candidate search in the KG. However, E-BELA does perform this search. To retrieve candidates, some of our baselines use ElasticSearch to create an index of mention-URI pairs with data from external sources and other KGs.

EARL and JLEAR evaluated the 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 dataset does not have annotated data for the EL task. We obtained its performance, based on LC-QuAD accuracy, from the results reported in [Li *et al.*, 2023].

In JLEAR [Li *et al.*, 2023], 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, for 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.

In our experiments, the DBpedia disambiguation file was utilized during the E-BELA evaluation process to verify the alignment of the model's predictions with previously mapped ambiguities. Specifically, when the entity predicted by the model deviated from the entity labeled in the evaluation datasets, the disambiguation file was queried to validate whether the prediction corresponded to an ambiguity identified in the mappings.

Language Considerations

We evaluate E-BELA using English-language datasets in conjunction with the USE and all-mpnet-base-v2 models, the latter being specifically optimized for English tasks. However, since E-BELA relies on embedding representations, we expect it to extend effectively to other linguistic contexts when appropriate datasets and pre-trained models in those languages are available.

Computational Costs

We performed EL with E-BELA by generating candidates followed by disambiguation. For this research, we utilized the PostgreSQL database alongside the pgvector plugin, which supports the use of indexes to accelerate the comparison process. However, since these indexes produce approximate rather than exact results, we decided not to employ them during E-BELA's evaluation to ensure the accuracy of our results.

The candidate generation phase imposes the highest computational cost. Without leveraging database indexes, E-BELA systematically traverses the entire database, comparing the vector representation of each mention to the vector representations of all literals and their respective entities.

Assuming there are M mentions, N literals/entities, and each vector is represented in d dimensions, the similarity (or distance) computation for a single comparison has a complexity of $O(d)$. As E-BELA repeats this operation for all M mentions, comparing them to all N literals/entities, the overall computational complexity of this phase amounts to $O(M \cdot N \cdot d)$.

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. We used Python 3.7.16 as the programming language. We used Apache Spark 3.3.4 with Hadoop 3, TensorFlow 2.10.1, Torch 1.10.1+cu113, and Numpy 1.21.6 libraries. Spark depends on the availability of a Java Virtual Machine. We use JVM 17.0.9. As the Database Management System, we utilized PostgreSQL 14.12 along with the pgvector plugin.

5.2 Results and Discussion

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

5.2.1 Experiment 1

Question 1: Does adopting entity embeddings derived from the embeddings of their associated literals provide advantages in terms of effectiveness for the EL task compared to other approaches documented in the literature?

Table 3. Comparing the accuracy of E-BELA with baselines on QALD-7 and LC-QuAD

Approach	QALD-7	LC-QuAD
EARL	0.57	0.65
Falcon	0.38	0.74
JLEAR	**0.73**	0.83
E-BELA (USE) - disambiguation	0.63	0.68
E-BELA (USE) + disambiguation	0.67	0.74
E-BELA (SBERT) - disambiguation	0.64	0.78
E-BELA (SBERT) + disambiguation	0.72	**0.84**

To address this research question, we compared the effectiveness of E-BELA against established baseline methods. As mentioned, the evaluation was carried out on two datasets and employed accuracy as the primary metric to quantify the performance of E-BELA.

Table 3 presents the results, with the best performance highlighted in bold. We present the results of E-BELA in the last four lines of the table, utilizing both USE and SBERT models. Furthermore, the descriptions '- disambiguation' and '+ disambiguation' indicate whether we used the DBpedia disambiguation file in the evaluation.

Although we cannot make a direct comparison due to differences in the sample space used in this work and the baselines, as discussed in subsection 5.1, the data shown in Table 3 highlight the validity of the proposed approach, considering the comparable accuracy achieved by E-BELA in both evaluation datasets. Specifically, when considering the sample error margin of the LC-QuaD dataset, E-BELA's best results demonstrate performance equivalent to JLEAR. It is worth noting that the transfer learning model that achieved our best results was the SBERT model *all-mpnet-base-v2*.

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. As discussed in Subsection 4.2.2, longer input sequences can lead to "diluted" embeddings. The *all-mpnet-base-v2*, on the other hand, incorporates a parameter defining the maximum sequence length, which is set to 384 characters. Despite the lower accuracy achieved by the USE model, its results are comparable to those obtained by Falcon [Sakor *et al.*, 2020]. Furthermore, the results of E-BELA, using both models, are substantially superior to those presented in EARL [Dubey *et al.*, 2018].

It is also important to emphasize that unlike the JLEAR approach [Li *et al.*, 2023], E-BELA does not train or fine-tune any model. Moreover, it obtains the candidate list directly from the representations of the KG entities. However, we hypothesize that the fine-tuning process of the models could potentially improve the results obtained by E-BELA, and we intend to evaluate this hypothesis in future work.

Similar to the approach in [Li *et al.*, 2023], 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 example, our model linked the mention "Us congress" to the entity "`http://dbpedia.org/resource/Us_congress`", while 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 our model might be even higher than reported in Table 3.

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

5.2.2 Experiment 2

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

To investigate this question, we conducted experiments comparing 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.

It is worth noting that each literal is associated with its own entity within our system. This implies that multiple instances of one literal and its representation can be stored in our database, each one associated with a different entity. This aspect is crucial as we need to accurately identify the specific entity associated with each literal.

To avoid ambiguity errors, we obtained the following results by utilizing the information from the disambiguation file, which provides mappings between different URI's that refer to the same entity.

The analyzed scenarios include:

1. Using only literal vectors: EL is performed solely based on the vector representations of the associated literals.
2. Using only entity vectors: EL is performed exclusively using the vector representations of the entities themselves.
3. Using both literal and entity vectors: is performed by combining the vector representations of both literals and entities.

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.

Importantly, since each literal representation is inherently linked to its corresponding KG entity, we can effectively leverage these representations to generate candidate entities for a given mention. This allows us directly retrieval potential entity matches from the space of literal vectors.

Table 4 presents the results of these experiments. Focusing on the LC-QuAD dataset, a notable observation is that entity vectors, when used in isolation, demonstrate a high degree of semantic context preservation derived from their corresponding literals. This is evidenced by its competitive accuracy despite the lower performance compared to the combined approach. The highest accuracy was achieved when utilizing both literal and entity vectors, indicating that the combined representation effectively leverages the semantic information captured by both sources.

Table 4. Accuracy of E-BELA on the EL task using different vector representation scenarios

Approach	QALD-7	LC-QuAD
E-BELA (USE) (literals only)	0.666	0.742
E-BELA (USE) (entities only)	0.696	0.711
E-BELA (USE) (both)	0.666	0.745
E-BELA (SBERT) (literals only)	0.717	0.836
E-BELA (SBERT) (entities only)	0.717	0.829
E-BELA (SBERT) (both)	0.717	0.840

When using the USE model, E-BELA combining both vector representations improves accuracy by at least 4.55% compared to using only literal or entity vector representations. Using the SBERT model, the improvement was approximately 1.31%. However, when considering the sample's margin of error, this difference suggests a statistical tie between the approaches.

In contrast, for the QALD-7 dataset, using only entity vectors or in combination with literal vectors yielded comparable results. We speculate that this limited performance variation might be attributed to the significantly smaller size of the QALD-7 test set, which may not provide sufficient data to reliably differentiate between the performance of the different vector representation strategies.

These results strongly suggest that both literal and entity vector representations are capable of retaining the semantic context of the literals. Moreover, the combination of both representations significantly enhances the overall performance of the E-BELA system.

5.2.3 Experiment 3

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

To investigate this question, we collected quantitative data related to the total number of literals associated with KG entities. We focused our analysis on the LC-QuAD dataset, utilizing SBERT as the encoder. We conducted the analysis from three perspectives: First, we considered descriptive metrics (see Table 5). Second, we examine the E-BELA performance based on the number of literals per entity (see Table 6). Finally, we analyzed the relationship between the number of literals in correct and predicted entities within the set of incorrect predictions (see Figure 4).

Table 5 summarizes descriptive statistical metrics of the literals of the entities in the KG, present in the LC-QuAD set. The metrics include the total number of entities, the average, and standard deviation about the number of literals per entity.

The data in Table 5 reveals that the distribution of the literals associated with the entities exhibits limited variation, fluctuating between 4.50 and 4.78 literals per entity. We observed that the standard deviation of the number of literals of our sample was slightly higher than the ones from the entities related with the incorrect predictions. This suggests that the number of literals associated with an entity might have some influence on the accuracy of the EL process.

Table 6 presents the E-BELA's EL performance based on the number of literals associated with the entities. Each row

Table 5. Descriptive Statistics of Literals per Entity in the KG.

Data subset	# of entities	Mean of literals per entity	Std Dev of literals per entity
Population	3899	4.691	3.348
Sample	358	4.782	2.464
Errors	59	4.500	2.760

Table 6. Analysis of EL Performance Based on the Number of Literals per Entity

# of literals	# of entities	Accuracy
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

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 per number of literals in the third column. The last row of the table considers entities with seven or more literals.

Our analysis of Table 6 did not reveal a strong and consistent correlation between the number of literals per entity and the accuracy of E-BELA. We observed 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 shows a matrix, analogous to a confusion matrix, which instead of displaying classes, shows the real and predicted quantities of literals per entity in the set of incorrect predictions made during the experiments. Based on the data showed in this matrix, we observed 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 5 and 6, as well as Figure 4, we noticed that although entities with seven or more literals exhibit a slightly higher accuracy, we cannot definitively assert that the number of literals significantly impacts the effectiveness of E-BELA.

5.2.4 Failure Cases

In this subsection, we present a detailed analysis of the failure cases observed during the evaluation on the LC-QuAD dataset. Specifically, we employed the SentenceTrans-

Table 7. Error analysis results of evaluation errors.

Failure cases	# number of errors
Disambiguation	21
Insufficient Context	7
Same entity	13
Lack or Incorrectly Written Mention	12
Annotation Error	6

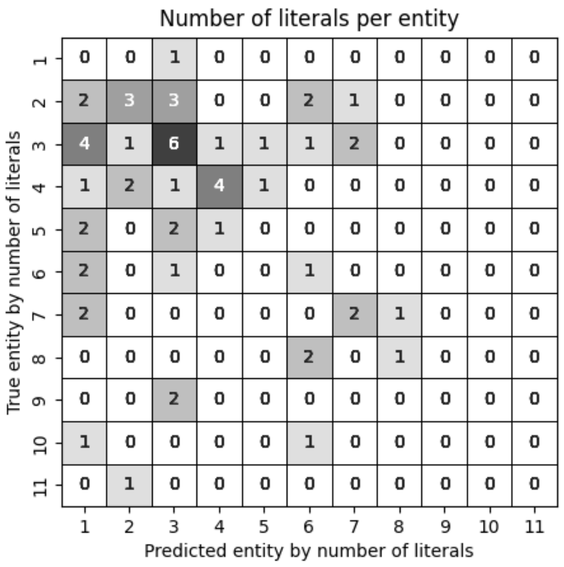

Figure 4. This figure illustrates a matrix showing the relationship between the number of literals in the correct and predicted entities in the set of incorrect predictions.

former as the encoder in this specific scenario, where we observed a total of 59 errors as detailed in Table 5. After analyzing the failures, we categorized them and enumerated the causes in Table 7, which shows the number of errors per category. Our failure cases are:

- Disambiguation: Occurs when E-BELA cannot identify the correct entity among the candidates, either due to the presence of multiple similar entities or because our method generates a representation distant from the correct entity. For example, in the question ”What organizations were founded in Texas and Dallas?”, the model picked ”Episcopal Diocese of Texas” (slightly closer) instead of ”Texas” due to the second reason.
- Insufficient Context: This type of failure occurs when the mention and its context lack enough information to enable E-BELA to retrieve relevant candidates entities or to disambiguate the correct entity from others. For example, the question “Which location country of Ennejma Ezzahra is also the origin of Boga?” lacks context to distinguish between a soft drink (Boga) and a language (also Boga). Even humans might struggle with this ambiguity.
- Same entity: Despite the presence of a disambiguation file (disambiguations_en.ttl.bz2), DBpedia may still contain multiple unmapped entries for the same real-world entity.
- Lack or Incorrectly Written Mention: Related with LC-QuAD annotations, it occurs when mentions are missing, annotations are absent, or the format is incorrect.
- Annotation Error: An incorrect link is annotated between a mention and a entity. Where a mention is mistakenly assigned to a wrong entity.

Our analysis of failure cases revealed that Disambiguation failure was the most prevalent issue, accounting for 21 out of 59 errors. This highlights the significant challenge of accurately identifying the correct entity among potential candidates, particularly in cases with multiple entities exhibiting similar names or descriptions. Insufficient Context, while representing a smaller fraction of errors, poses a complex problem that requires careful attention. Furthermore, same entity within the DBpedia also significantly contributed to errors, emphasizing the fact of there are a lot of same entities not mapped yet.

Issues related to data quality, such as Empty or Incorrectly Written Mentions and Annotation Errors, significantly impacted the performance of E-BELA despite being inherent to the evaluation dataset. These findings emphasize the need for robust disambiguation strategies, high quality datasets, and careful attention to potential ambiguities within the knowledge base to enhance the accuracy of the EL task.

6 Conclusions and Future Work

In our previous work [Pereira and Ferreira, 2024], we introduced E-BELA (Enhanced Embedding-Based Entity Linking Approach), a straightforward and effective approach for EL. E-BELA obtains embeddings for entities from a KG by pooling the representations of the literals associated with each entity. These literals are in natural language, mirroring the representation of textual mentions. We store both literal and entity embeddings in a vector database management system, facilitating efficient search and retrieval operations. These embeddings aim to put mentions and entities close in the same vector space, enabling their linkage through some similarity metric. We conduct the EL process by searching for a list of candidate entities from the embedding of a mention, followed by disambiguation. For disambiguation, E-BELA use the context information from the mention and candidate entities. Our results demonstrate that this methodology achieves comparable performance to other state-of-the-art methods, while employing a much simpler model, making a significant contribution to the field of NLP.

In this work, we enhance our previous work by evaluating E-BELA on a new dataset (QALD-7) and conducting a comprehensive analysis of failure cases and limitations. Since we did not find publicly available annotations for QALD-7, we annotated the dataset ourselves and made these annotations available along with our code. Furthermore, we evaluated E-BELA on this new dataset and compared its performance against relevant baselines. This evaluation reinforced the value of E-BELA. The analysis of failure cases highlighted the significant challenge of accurately identifying the correct entity among potential candidates.

A limitation of our approach is its inability to handle mentions of entities that do not have a corresponding entry in the KG. Since E-BELA operates within a vector space and does not explicitly incorporate a threshold for entity linking, it may still return a candidate entity even if no true match exists in the KG for a given mention. This can lead to incorrect linkages and impact the overall performance of E-BELA.

In future work, we intend to perform Relation Linking in an integrated manner by leveraging the embeddings of the entities and relations of the KG. Furthermore, we intend to exploit the inherent connections between entities and their relations to generate candidate relation lists for relation mentions that occur in close contextual proximity to the corresponding entity mentions. Additionally, we also plan to evaluate the E-BELA's EL performance after fine-tuning the *all-mpnet-base-v2* model with mention and entity contexts. Furthermore, we intend to apply E-BELA to other problems that involve EL in their pipelines. Finally, we aim to enhance the evaluation process by using a sample with a balanced quantity of literal data.

Declarations

Acknowledgements

We thank the Federal University of Ouro Preto (UFOP), the Federal Institute of Minas Gerais (IFMG), CNPq, FAPEMIG and CAPES.

Funding

This research is partially sponsored by Federal University of Ouro Preto (UFOP), Federal Institute of Minas Gerais (IFMG), CNPq, CAPES, FAPEMIG, and Instituto Nacional de Ciência e Tecnologia em Inteligência Artificial Responsável para Linguística Computacional, Tratamento e Disseminação de Informação (INCT-TILD-IAR) (grant # 408490/2024-1).

Authors' Contributions

All authors contributed to the writing of this article, read and approved the final manuscript.

Competing interests

The authors declare that they have no competing interests.

Availability of data and materials

All artifacts comprising E-BELA are available for download at the following address: https://github.com/italompereira/E-BELA.

References

Bowman, S. R., Angeli, G., Potts, C., and Manning, C. D. (2015). A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics. DOI: 10.18653/v1/D15-1075.

Cao, N. D., Izacard, G., 0001, S. R., and Petroni, F. (2021). Autoregressive entity retrieval. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net. DOI: 10.48550/arXiv.2010.00904.

Caseli, H. d. M., Nunes, M. d. G. V., and Pagano, A. (2024). O que é pln? In Caseli, H. M. and Nunes, M. G. V., editors, *Processamento de Linguagem Natural: Conceitos, Técnicas e Aplicações em Português*, book chapter 1. BPLN, 2 edition. Available at:`https://brasileiraspln.com/livro-pln/2a-edicao/parte-introducao/cap-introducao/cap-introducao.html`.

Cer, D., Yang, Y., Kong, S., Hua, N., Limtiaco, N., John, R. S., Constant, N., Guajardo-Cespedes, M., Yuan, S., Tar, C., Sung, Y., Strope, B., and Kurzweil, R. (2018). Universal sentence encoder. *CoRR*, abs/1803.11175. DOI: 10.48550/arXiv.1803.11175.

Chen, L., Zhu, T., Liu, J., Liang, J., and Xiao, Y. (2023). End-to-end entity linking with hierarchical reinforcement learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(4):4173–4181. DOI: 10.1609/aaai.v37i4.25534.

Colucci, L., Doshi, P., Lee, K.-L., Liang, J., Lin, Y., Vashishtha, I., Zhang, J., and Jude, A. (2016). Evaluating item-item similarity algorithms for movies. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems*, CHI EA '16, page 2141–2147, New York, NY, USA. Association for Computing Machinery. DOI: 10.1145/2851581.2892362.

Devlin, J., Chang, M., Lee, K., and Toutanova, K. (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*, pages 4171–4186, Minneapolis, MN, USA. Association for Computational Linguistics. DOI: 10.18653/V1/N19-1423.

Di Noia, T., Mirizzi, R., Ostuni, V. C., Romito, D., and Zanker, M. (2012). Linked open data to support content-based recommender systems. In *Proceedings of the 8th International Conference on Semantic Systems*, I-SEMANTICS '12, page 1–8, New York, NY, USA. Association for Computing Machinery. DOI: 10.1145/2362499.2362501.

Di Noia, T. and Ostuni, V. C. (2015). *Recommender Systems and Linked Open Data*, pages 88–113. Springer International Publishing, Cham. DOI: 10.1007/978-3-319-21768-0_4.

Dubey, M., Banerjee, D., Chaudhuri, D., and Lehmann, J. (2018). Earl: Joint entity and relation linking for question answering over knowledge graphs. In *The Semantic Web – ISWC 2018*, pages 108–126, Cham. Springer International Publishing. DOI: 10.1007/978-3-030-00671-6_7.

Gomes, J., de Mello, R. C., Ströele, V., and de Souza, J. F. (2022). A hereditary attentive template-based approach for complex knowledge base question answering systems. *Expert Systems with Applications*, 205:117725. DOI: https://doi.org/10.1016/j.eswa.2022.117725.

Heath, T. and Bizer, C. (2011). *Linked Data Design Considerations*, pages 41–68. Springer International Publishing, Cham. DOI: 10.1007/978-3-031-79432-2_4.

Iyyer, M., Manjunatha, V., Boyd-Graber, J., and Daumé III, H. (2015). Deep unordered composition rivals syntactic methods for text classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1681–1691, Beijing, China. Association for Computational Linguistics. DOI: 10.3115/v1/P15-1162.

Jia, B., Wu, Z., Zhou, P., and Wu, B. (2021a). Entity linking based on sentence representation. *Complexity*, 2021(1):8895742. DOI: 10.1155/2021/8895742.

Jia, N., Cheng, X., Su, S., and Ding, L. (2021b). Cogcn: Combining co-attention with graph convolutional network for entity linking with knowledge graphs. *Expert Systems*, 38(1):e12606. DOI: 10.1111/exsy.12606.

Le, P. and Titov, I. (2018). Improving entity linking by modeling latent relations between mentions. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1595–1604, Melbourne, Australia. Association for Computational Linguistics. DOI: 10.18653/v1/P18-1148.

Leng, H., De La Cruz Paulino, C., Haider, M., Lu, R., Zhou, Z., Mengshoel, O., Brodin, P.-E., Forgeat, J., and Jude, A. (2018). Finding similar movies: dataset, tools, and methods. In *Proceedings of the 8th International Conference on Semantic Systems*, WSCG'2018, pages 115–124, Plzen, Czech Republic. Václav Skala-UNION Agency. DOI: 10.24132/CSRN.2018.2802.15.

Li, H., Yu, W., and Dai, X. (2023). Joint linking of entity and relation for question answering over knowledge graph. *Multimedia Tools and Applications*, 82(29):44801–44818. DOI: 10.1007/s11042-023-15646-w.

Li, Q., Li, F., Li, S., Li, X., Liu, K., Liu, Q., and Dong, P. (2022). Improving entity linking by introducing knowledge graph structure information. *Applied Sciences*, 12(5):44801–44818. DOI: 10.3390/app12052702.

Luo, Y.-X., Yang, B.-L., Xu, D.-H., Tian, L.-G., and He, J.-Y. (2023). Entity linking improvement model by deep modeling of sentence semantics. *Journal of Network Intelligence*, 8(1):224–236. Available at:`https://bit.nkust.edu.tw/~jni/2023/vol8/s1/16.JNI-0391.pdf`.

Mello, R., Jr., J. G., Souza, J., and Ströele, V. (2024). Constructing a kbqa framework: Design and implementation. In *Proceedings of the 30th Brazilian Symposium on Multimedia and the Web*, pages 89–97, Porto Alegre, RS, Brasil. SBC. DOI: 10.5753/webmedia.2024.243150.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013a). 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*, Scottsdale, Arizona, USA. Association for Computing Machinery. DOI: 10.48550/arXiv.1301.3781.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G., and Dean, J. (2013b). Distributed representations of words and phrases and their compositionality. *CoRR*, abs/1310.4546. DOI: 10.48550/arXiv.1310.4546.

Mirizzi, R., Di Noia, T., Ragone, A., Ostuni, V., and Di Sciascio, E. (2012). Movie recommendation with dbpedia. In *Movie recommendation with DBpedia*, volume 835, pages 101–112, Bari, Italy. Available at::https://ceur-ws.org/Vol-835/paper12.pdf.

Naseem, T., Ravishankar, S., Mihindukulasooriya, N., Abdelaziz, I., Lee, Y.-S., Kapanipathi, P., Roukos, S., Gliozzo, A., and Gray, A. (2021). A semantics-aware transformer model of relation linking for knowledge base question answering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the*

11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers), pages 256–262, Online. Association for Computational Linguistics. DOI: 10.18653/v1/2021.acl-short.34.

Ngomo, J. G. N., Lopes, G. R., Campos, M. L. M., and Cavalcanti, M. C. R. (2020). An approach for improving dbpedia as a research data hub. In *Proceedings of the Brazilian Symposium on Multimedia and the Web*, WebMedia '20, page 65–72, New York, NY, USA. Association for Computing Machinery. DOI: 10.1145/3428658.3431075.

Pereira, I. M. and Ferreira, A. A. (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*, WebMedia '19, page 425–432, New York, NY, USA. Association for Computing Machinery. DOI: 10.1145/3323503.3349547.

Pereira, I. M. and Ferreira, A. A. (2024). E-bela: Enhanced embedding-based entity linking approach. In *Proceedings of the 30th Brazilian Symposium on Multimedia and the Web*, pages 115–123, Porto Alegre, RS, Brasil. SBC. DOI: 10.5753/webmedia.2024.243160.

Pershina, M., He, Y., and Grishman, R. (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*, pages 238–243, Denver, Colorado. Association for Computational Linguistics. DOI: 10.3115/v1/N15-1026.

Reimers, N. and Gurevych, I. (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)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics. DOI: 10.18653/v1/D19-1410.

Ristoski, P. and Paulheim, H. (2016). Rdf2vec: Rdf graph embeddings for data mining. In *The Semantic Web – ISWC 2016*, pages 498–514, Cham. Springer International Publishing. DOI: 10.1007/978-3-319-46523-4_30.

Ristoski, P., Rosati, J., Di Noia, T., De Leone, R., and Paulheim, H. (2019). Rdf2vec: Rdf graph embeddings and their applications. *Semantic Web*, 10(4):721–752. DOI: 10.3233/SW-180317.

Sakor, A., Singh, K., Patel, A., and Vidal, M.-E. (2020). Falcon 2.0: An entity and relation linking tool over wikidata. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, CIKM '20, page 3141–3148, New York, NY, USA. Association for Computing Machinery. DOI: 10.1145/3340531.3412777.

Shen, W., Li, Y., Liu, Y., Han, J., Wang, J., and Yuan, X. (2023). Entity linking meets deep learning: Techniques and solutions. *IEEE Transactions on Knowledge; Data Engineering*, 35(03):2556–2578. DOI: 10.1109/TKDE.2021.3117715.

Shen, W., Wang, J., and Han, J. (2015). Entity linking with a knowledge base: Issues, techniques, and solutions. *IEEE Transactions on Knowledge and Data Engineering*, 27(2):443–460. DOI: 10.1109/TKDE.2014.2327028.

Srinivasan, U. and Mani, C. (2018). Diversity-ensured semantic movie recommendation by applying linked open data. *International Journal of Intelligent Engineering and Systems*, 11:275–286. DOI: 10.22266/ijies2018.0430.30.

Stankevičius, L. and Lukoševičius, M. (2024). Extracting sentence embeddings from pretrained transformer models. *Applied Sciences*, 14(19). DOI: 10.3390/app14198887.

Trivedi, P., Maheshwari, G., Dubey, M., and Lehmann, J. (2017). Lc-quad: A corpus for complex question answering over knowledge graphs. In *The Semantic Web – ISWC 2017*, pages 210–218, Cham. Springer International Publishing. DOI: 10.1007/978-3-319-68204-4_22.

Usbeck, R., Ngomo, A.-C. N., Haarmann, B., Krithara, A., Röder, M., and Napolitano, G. (2017). 7th open challenge on question answering over linked data (qald-7). In *Semantic Web Challenges*, pages 59–69, Cham. Springer International Publishing. DOI: 10.1007/978-3-319-69146-6_6.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.. DOI: 10.48550/arXiv.1706.03762.

Williams, A., Nangia, N., and Bowman, S. (2018). A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics. DOI: 10.18653/v1/N18-1101.

Yamada, I., Shindo, H., Takeda, H., and Takefuji, Y. (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*, pages 250–259, United States. Association for Computational Linguistics (ACL). DOI: 10.18653/v1/k16-1025.

Yamada, I., Washio, K., Shindo, H., and Matsumoto, Y. (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*, pages 3264–3271, Seattle, United States. Association for Computational Linguistics. DOI: 10.18653/v1/2022.naacl-main.238.