Graph Representation Learning in Document Wikification

Mozhgan Saeidi(✉), Evangelos Milios, and Norbert Zeh

Dalhousie University, Halifax, Canada
mozhgan.saeidi@dal.ca, {eem,nzeh}@cs.dal.ca

Abstract. Wikification (entity annotation) is a challenging task in Natural Language Processing (NLP). It is a method to automatically enrich a text with links to Wikipedia as a knowledge base. Wikification starts from detecting ambiguous mentions in the document, and later tries to disambiguate those mentions. In the core of the Wikification task, there is one other important NLP task: word representation. This paper proposes a new word representation for senses of a mention with Graph convolutional networks architecture. Senses are the possible meanings of one mention, based on the knowledge base. In our representation modeling, we used the context document and the first paragraph of each Wikipedia page to enhance our contextual representation. Using the nearest neighbor algorithm for disambiguating the mentions via our sense representations, we show the efficiency of our representations. The results of comparing our method with recent state-of-the-art methods show the efficiency of our solution.

Keywords: Representation learning · Graph convolutional networks · Wikification · Document ambiguity

1 Introduction

The task of "**Entity Recognition and Disambiguation**" (**ERD**) is to identify mentions[1] of entities and link them to a relevant entry in an external knowledge base, which is also known under the names of "**Entity Linking**", "**Wikification**" or more generally "**Text Annotation**" [1]. Given an input document, a Wikifier links entities of the document to the most relevant corresponding Wikipedia pages. Automated document annotation has become an important topic due to the vast growth in Natural Language Processing (NLP) applications [53]. One benefit of document annotation is enhancing text readability and its unambiguousness by inserting connections between the text and an external knowledge base, like Wikipedia, which is the most popular among online encyclopedias [41].

[1] A mention can be one or more tokens.

E. H. Barney Smith and U. Pal (Eds.): ICDAR 2021 Workshops, LNCS 12917, pp. 509–524, 2021.
https://doi.org/10.1007/978-3-030-86159-9_37

The problem of Wikification is closely related to other core NLP tasks, such as Named-Entity Recognition (NER) and Word Sense Disambiguation (WSD) [54]. Wikification, in particular, is the task of associating a word in context with the most suitable meaning from the predefined sense inventory of Wikipedia. Named-Entity Recognition involves identifying certain occurrences of noun phrases as belonging to particular categories of named-entities [34]. These expressions refer to names, including person, organization, and location names, and numeric expressions including time, date, money, and percent expressions [21]. In Word Sense Disambiguation, our knowledge base is not limited to Wikipedia, which is the difference between Wikification and WSD [23]. In WSD, the knowledge base is a treasury like WordNet and Wikipedia [18]. Details and performance of each WSD method are highly dependent on the knowledge base to link to.

The knowledge bases are different in their nature [2]; for example, WordNet is a lexical graph database of semantic relations (e.g. synonyms, hyponyms, and meronyms) between words. Synonyms are grouped into synsets with short definitions and usage examples. WordNet can thus be seen as a combination and extension of a dictionary and thesaurus [4]. Wikipedia is a hyperlink-based graph between encyclopedia entries. So, Wikification, WSD, and NER are closely related but still different because of their underlying knowledge bases. In this paper, our focus is on the Wikification problem, so we did not mention WSD and NER systems in detail [19], and we focus on providing required details of Wikification systems. We compared our proposed method with the Wikifiers by different parameter settings.

The Wikification process involves two steps: *Spotting* or *mention detection*, i.e. identifying the terms that should be wikified, and *Entity Linking* or *Disambiguation to Wikipedia*, i.e. identifying the relevant Wikipedia page among a set of candidate pages [51]. The spotter operates on the text to extract all entity ambiguous mentions and assigns all potential entity candidates for each mention. The entity linker disambiguates the candidate entities by selecting the most probable sense entity for each mention [50]. Our focus is on the second step, and we use the output of the recent spotting system [46] as the input to our algorithm.

A human reader can identify the correct meaning of each word based on the context in which the word is used. Computational methods try to mimic this approach. These methods often represent their output by linking each word occurrence to an explicit representation of the chosen sense [56]. There are two approaches to tackle this problem: The machine learning-based approach and the knowledge-based approach. In the machine learning-based approach, systems are trained to perform the task [45,52]. The knowledge-based approach requires external lexical resources such as Wikipedia, WordNet [30], a dictionary, or a thesaurus. The machine learning-based approaches mainly focus on achieving maximal precision or recall and have their drawbacks of run-time and space requirement at the time of classifier training. So, knowledge-based Wikification methods still have advantages to study. Among different knowledge-based methods, coherence-based has been more effective to explain it [11]. In the coherence-based approach, one

important factor is the coherence of the whole text after disambiguation, while in other approaches, this factor might change to considering the coherence of each sentence or paragraph. It is a significant challenge to perform Wikification accurately but also fast enough to process long text documents [26]. One coherence-based approach models the relatedness between the senses and a *key-entity* in disjoint windows of the text to speed up the approach and disambiguates every word so that the total pairwise relatedness of all chosen word senses and key-entity of the same window is maximized. This method is computationally expensive, and run-time performance is considered as a secondary issue in most of the existing Wikification methods.

Embeddings have been shown to play an important role in different NLP tasks [33,45], especially in disambiguation tasks [48]. Embeddings based on pre-trained deep language models have attracted much interest recently as they have proved to be superior to classical embeddings for several NLP tasks, including WSD. These models, e.g., ELMO [37], BERT [10], XLNET [58], encode several pieces of linguistic information in their word representations. These representations differ from static neural word embeddings [36] in that they are dependent on the surrounding context of the word. This difference makes these vector representations especially interesting for disambiguation, where effective contextual representations can be highly beneficial for resolving lexical ambiguity. These representations enabled sense-annotated corpora to be exploited more efficiently [24].

Here, we face the problem of link ambiguity, meaning a phrase can be usually linked to more than one Wikipedia page in which the correct link depends on the context where it occurs. For example, the word "bar" can be linked to different articles, depending on whether it was used in a business or musical context.

In this study, we overview different current approaches for text embedding with focusing on the contextualized sense representation. We also provide an overview of the disambiguation methods, and the most used ones in the literature. Our novel contribution provides a new representation learning using the graph deep learning approach and uses the nearest neighbor heuristic algorithm for disambiguating the document. We finally compare the performance of our proposed approach with our representations with the most recent approaches in the disambiguation task.

2 Background

In this section, First we provide an overview for wikification systems, and second, we go through a general background of the embedding approaches. Our focus is on the pre-trained deep language embedding models.

2.1 Document Disambiguation with Wikipedia

A large group of NLP systems for various word disambiguation tasks rely on Lexical Knowledge Bases. These knowledge bases vary significantly in their

structure, size, and subject, making them more appropriate for certain domains. For instance, WordNet was used for *synonym exploration* [13,32], and the sophisticated Unified Medical Language System (UMLS) ontology was used for *medical text dissambiguation* [17,25]. Disambiguation based on Wikipedia has been demonstrated to be comparable in terms of coverage to domain-specific ontology [55] since it has broad coverage, with documents about entities in a variety of domains [26]. Moreover, Wikipedia has unique advantages over the majority of other knowledge bases [61]. One advantage is the text in Wikipedia is primarily factual and available in a variety of languages. The other advantage is about the articles which can be directly linked to the entities they describe in other knowledge bases. Also, mentions of entities in Wikipedia articles often provide a link to the relevant Wikipedia pages, thus providing labeled examples of entity mentions and associated anchor texts in various contexts, which could be used for supervised learning in Wikification.

Knowledge-Based Approaches. In this type of approach, WSD is considered a graph-based problem. They use the semantic network structure, e.g., Wikipedia, WordNet, BabelNet, to find the correct meaning based on its context for each input word [33]. The latest work in this series is SensEmBERT [47] which shows the power of language models combined with a vast amount of knowledge in a semantic network to produce latent semantic representations of senses in multiple languages. ARES followed this model and created sense embeddings for the lexical meanings within a lexical knowledge base. These embeddings lie in a space that is comparable to that of contextualized word vectors [48].

As mentioned in the previous section, using any knowledge base for text disambiguation requires an "entity linker". When UMLS is used as the knowledge base, MetaMap is widely accepted as the entity linker [3,15]. In the case of Wikipedia, the entity linker is referred to as **Wikifier**. In most studies, a Wikifier uses two groups of features, local and global [57]. Local features include the context around the entity mention and some data-driven statistics regarding the mention-entity relation, such as *commonness* or *prior probabilities* [43]. The most famous example of global features is the **semantic coherence measure**. This feature is established based on the assumption that words in a given *neighbourhood* (i.e., a segment of the text) will tend to share a common topic. Examples of widely accepted Wikifiers include Wikify! [28], Wikipedia Miner [31], TagME [12], and GLOW [43].

The system of Wikify! is a wikification method that disambiguates and ranks candidates to indicate the most valuable ones to the user in terms of the meaning. Wikify! uses the Lesk algorithm [20] to choose the most appropriate sense. The Lesk algorithm identifies the most likely meaning for an ambiguous word based on the *contextual overlap* between the content of the Wikipedia pages corresponding to the candidate senses and the local context of the ambiguous word.

Wikify! has been the first widely accepted Wikifier. However, it has already been outperformed by more recent Wikifiers, such as large-scale named entity disambiguation [9]. In large-scale named entity disambiguation [9], the process is

based on maximizing the agreement between the contextual information corresponding to each candidate sense and the context of the anchor text. The information for each candidate sense is a combination of various features extracted from its Wikipedia page. One of these features is the set of incoming Wikipedia links for each candidate sense, including irrelevant entities that happen to have significant overlap, in terms of the number of common words, with the anchor text. As a result, an irrelevant candidate entity with such incoming Wikipedia links could end up getting selected.

Wikipedia Miner [31] outperforms the Wikify! and the large-scale named entity methods [9,28] introduced above. The entity disambiguation approach of Wikipedia Miner relies on the graph structure of Wikipedia. This structure is used to perform disambiguation based on two concepts: *commonness* and *relatedness*. The commonness of a sense is defined by the number of times it is used as a destination in Wikipedia. Hence, commonness is sometimes referred to as prior probability. The relatedness of a candidate sense is its similarity to the context. Their approach aims to balance the commonness of a sense with its relatedness to the surrounding context. They use machine learning to combine these features so that the balance can be adjusted from document to document.

The following two methods, namely TagMe and GLOW, while still using commonness and relatedness, are very close to our solution. One of them is based on a voting system, and the other is using a ranking approach, which are two aspects of our proposed solution.

In the system of TagMe [12] a voting scheme is proposed, where the candidates for each mention can vote for all candidate entities of other mentions based on relatedness; Candidates with higher prior probabilities have stronger votes. In our proposed solution, the voting power of each candidate sense depends on its rank based on the previous voting round. Moreover, in TagMe, candidates with a prior probability below a fixed threshold are pruned. Then TagMe uses two algorithms to decide the chosen sense for each mention: disambiguation by classifier (DC), which uses a probabilistic approach based on prior probability and relatedness to select the correct candidate; and disambiguation by threshold (DT), which makes a shortlist of the top candidates with relatedness above a predefined threshold, and then chooses the candidate with the highest prior probability among them. Later, they released a more efficient version of TagMe as WAT algorithm [40].

The system of GLOW [43] uses two sub-systems; a ranker and a linker. The goal of the ranker is to select the best candidate, and the linker decides if the recommended sense by the ranker is good or not. The ranker sub-system uses two sets of features, namely local and global. Local features calculate the similarities between mentions and their candidate entities, incorporating the term frequency inverse document frequency (TF-IDF) vectors for Wikipedia pages and the anchor text [14]. The global features measure the coherence among all candidate senses in terms of the sum of pairwise normalized Google distance (NGD) [8], and Point-wise Mutual Information (PMI) [5] across all mentions in the whole disambiguation text. The linker sub-system is trained as a

linear support vector machine (LSVM) to separate correct and incorrect linker outputs based on data collected from Wikipedia, which provides positive and negative examples for each mention according to Wikipedia's gold standard.

The state-of-the-art Wikifier RedW [49] is a run-time oriented Wikification solution. RedW is based on Wikipedia redirects and can wikify massive corpora with good performance. This approach is based on mapping the *longest sub-string match* between the mention and the Wikipedia entity titles. RedW assumes that a term often matches its Wikipedia title or a corresponding Wikipedia redirect page. They have this assumption because one advantage of redirects over anchor dictionaries is their dynamic nature. Redirects are updated both automatically and by Wikipedia editors. Hence they are expected to contain less noise compared to automatically created dictionaries. RedW creates a table of all Wikipedia titles, including the redirect titles. For every text, RedW tries to match its n-grams to the table, preferring longer matches. Unlike TagMe and Glow, RedW does not consider global features such as coherence. However, TagMe is designed for short text Wikification and is inferior to RedW compared to long text, and Glow has not been compared against this scheme. However, experiments show our proposed solution outperforms RedW, as we consider the coherence of the text. There is one recent work which is an application of Wikification task by using RedW method. In this work [60], authors try to navigate through information pollution by capturing the provenance of every claim. They define a provenance graph for a given natural language claim, aiming to understand where the claim may come from and how it has evolved. Specifically, to wikify a source mention for each claim, they adapt the redirect-based wikification method of RedW since RedW is efficient and context-free.

Supervised Approaches. The supervised approaches use sense-annotated data for their training. These types of methods have traditionally gained state-of-the-art results in terms of accuracy. Even before introducing pre-trained language models, supervised WSD methods have been shown to outperform the knowledge-based models. At the same time, their defect is the *knowledge acquisition bottleneck*, which makes it challenging to construct broad manually curated corpora. It limits the ability of these methods to scale to new words [35].

Neural sequencing models are trained for end-to-end word sense disambiguation task [42]. They re-framed WSD as a translation task that sequences of words are translated into sequences of senses. Later, some works showed the potential of contextual representation for WSD [27,37]. Sense embeddings initialization using glosses and adapted the skip-gram objective of word2vec is done by [7] to learn and improve the sense embeddings jointly with word embeddings. Later, by the appearance of NASARI vectors [6], sense embeddings were created using structural knowledge from a large multilingual semantic network. These methods represent sense embeddings in the same space as the pre-trained word embeddings, while they suffer from fixed embedding spaces. The LMMS representation considers creating sense-level embeddings with complete coverage of WordNet and shows the power of this representation for WSD by applying a

simple Nearest Neighbors (k-NN) method [24]. ARES used this 1-NN method with its representations and showed improved results in disambiguating. In the following, we briefly introduce the recent state-of-the-art embeddings and then analyze their results.

2.2 Language Modelling Representation

Most NLP tasks now use semantic representations derived from language models. There are static word embeddings and contextual embeddings. In this section, we cover aspects of the word and contextual embeddings that are especially important to our work.

Static Word Embeddings. Word embeddings are distributional semantic representations usually with one of two goals: predict context words given a target word (Skip-Gram), or the inverse (CBOW) [29]. In both, the target word is at the center, and the context is considered as a fixed-length window that slides over tokenized text. These models produce dense word representations. One limit for word embeddings, as mentioned before, is meaning conflict around word types. This limitation affects the capability of these word embeddings for the ones that are sensitive to their context [44].

Contextual Word Embeddings. The problem mentioned as a limitation for the static word embeddings is solved in this type of embeddings. The critical difference is that the contextual embeddings are sensitive to the context. It allows the same word types to have different representations according to their context. The first work in contextual embeddings is ELMO [37], which is followed by BERT [10], as the state-of-the-art model. The critical feature of BERT, which makes it different, is the quality of its representations. Its results are task-specific fine-tuning of pre-trained neural language models. The recent representations which we analyze their effectiveness are based on these two models [38,39].

3 Graph Representation Modelling

This section presents our sense embedding approach, which is a novel method based on deep graph convolutional networks. This representation is a context-aware representation model of Wikipedia senses by combining the semantic and textual information derived from the document context of the mention and the first paragraph of the mention's Wikipedia page. In this representation, we used the power of neural language models, i.e., BERT [10]. We divide our approach into the following subdivisions.

3.1 Concept Embedding

We use concept which refers to word or mention, as well as s sense. We use the long heuristic search and the table of Wikipedia titles for extracting mentions in the document. when we match them together, the mentions of the document which are corresponding with a Wikipedia page. For each mention m in the document, we generate BERT representation of m with $R(m)$. In our experiments, we used BERT-base-cased[2], since it has been shown the performance of the BERT-base-cased model, in comparison with uncased- is better in the task of sense disambiguation. We repeat the same procedure to produce the representations of senses for each mention, with $R(s)$. The length of each one of these representations is 300, as the dimension of BERT representations.

3.2 Context Representation

When we work with Wikipedia page of each mention, we consider the first paragraph of each Wikipedia page. The first paragraph is the main part of each Wikipedia page with the most noticeable information about the mention (title) of the page. Using the same pre-trained language model, we generate the representation of the first paragraphs of each mention in the Wikipedia page. In our implementations, if the length of the first paragraph is more than 512 words, the algorithm does not include the rest of the words. While our experiments show it does not happen, since the first paragraph of all the Wikipedia pages of the mentions in the used dataset are not including more than 512 words. We show this representation by $R(P)$. On the other hand, we have the input document from where we extracted our mentions. For each mention, our algorithm considers the paragraph which includes the mention. We use the representation of this document paragraph in our settings by $R(D)$. Starting from a Wikipedia page of mention m, we collect the set of its senses in the Wikipedia knowledge base, which are the redirect pages, i.e., all the redirects that are connected to m. We use these senses as the nodes of the graph that we are going to build and connect the redirect ones, which builds the graph's edges.

3.3 Sense Representation

In this part of building our representations, we merge the contextual information computed in the two previous steps to enrich the representation with additional information of the document and the semantic network, which is the first paragraph of the Wikipedia page. For each sense s of a mention m, now we have four variables of $R(m)$, $R(s)$, $R(P)$, and $R(D)$, each with dimension 300.

3.4 Graph Convolutional Network

Graph Convolutional Networks (GCN) are a very powerful multilayer neural network architecture for machine learning on graphs [16]. GCN operates directly

[2] https://huggingface.co/bert-base-cased.

on a graph and induces embedding vectors of nodes based on the properties of their neighborhoods. In fact, they are so powerful that even a randomly initiated 2−layer GCN can produce useful feature representations of nodes in networks[3]. Formally, consider a graph $G = (V, E)$, where $V(|V| = n)$ and E are sets of nodes and edges, respectively. Every node is assumed to be connected to itself, i.e., $(v, v) \in E$ for any v which the reason for this assumption is mentioned at the end of this paragraph. Let $X \in R^{n \times m}$ be a matrix containing all n nodes with their features, where m is the dimension of the feature vectors, each row $x_v \in R_m$ is the feature vector for v. We introduce an adjacency matrix A of G and its degree matrix D, where $D_{ii} = \sum_j A_{ij}$. Because of self-loops, the diagonal elements of A are all 1. We now have a graph, its adjacency matrix A, and a set of input feature X. After applying the propagation rule $f(X, A) = AX$ and $X = I$, the representation of each node (each row) is now a sum of its neighbor's features. In other words, the graph convolutional layer represents each node as an aggregate of its neighborhood. The reason for considering the self-loops in the graph is the aggregated representation of a node to include its own features.

For a one-layer GCN, the new k-dimensional node feature matrix $L^{(1)} \in R^{n \times k}$ is computed as:

$$L^{(1)} = \rho(\hat{A}XW_0) \tag{1}$$

where $\hat{A}$ is $D^{-0.5}AD^{-0.5}$, the normalized symmetric adjacency matrix and $W_0 \in R^{m \times k}$ is the weight matrix. The ρ is the activation function (RELU); $\rho(x) = max(0, x)$. GCN can capture information only about immediate neighbors with one layer of convolution. When multiple GCN layers are stacked, information about larger neighborhoods are integrated;

$$L^{(j+1)} = \rho(\hat{A}L^j W_j) \tag{2}$$

which j is the layer number and $L^0 = X$. In other words, the size of the second dimension of the weight matrix determines the number of features at the next layer. The feature representations can be normalized by node degree with transforming the adjacency matrix A by multiplying it with the inverse degree matrix D. First we used the simple propagation rule $f(X, A) = D^{-1}AX$, while then improved it. The improved version is inspired by a recent work [16] that proposes a fast approximate spectral graph convolutions using a spectral propagation rule $f(X, A) = \sigma(D^{-0.5}\hat{A}D^{-0.5}XW)$. They showed this property is very useful, that connected nodes tend to be similar (e.g. have the same label).

We consider each mention of the document as one node of the graph, and a new added node (redirect link) will connect with its nearest neighbor by using cosine similarity, which makes the edges of the graph. The cosine similarity between two nodes on the edges makes the weight matrix. The number of nodes in the text graph $|V|$ is the number of mentions. For each sense s, we use an integrated representation of its mention m with its own representation, i.e.,. $R(m, s)$. We set the feature matrix X as extracted representation of BERT as

[3] The notation we used for GCN in this paper are the same as notations in [59].

input to GCN. The dimension of the feature matrix here is 600, as it is the representation length of two BERT embeddings, one for the mention and the other for the sense. We name our representation **msBERT**.

As mentioned, formally, the weights of edge between node i and node j defines as:

$$W_{ij} = \text{cosine sim}(R(i), R(j)) = \frac{R(i).R(j)}{||R(i)||||R(j)||} \tag{3}$$

which $R(i)$ is representation of node i.

After building the graph, we feed it into a simple 2−layers GCN as [16], the second layer node (mention,sense) embeddings are fed into a softmax classifier:

$$Z = softmax(\hat{A}RELU(\hat{A}XW_0)W_1) \tag{4}$$

where

$$\hat{A} = D^{-0.5}AD^{-0.5}$$

and

$$softmax(x_i) = \frac{1}{Z}exp(x_i)$$

with $S = \sum_i exp(x_i)$. The loss function is the one defined in [59] as:

$$L = -\sum_{d \in Y}\sum_{f=1}^{F} Y_{df} ln Z_{df} \tag{5}$$

where Y_D is the set of mention indices that have labels and F is the dimension of the output feature. Y is the label indicator matrix. Similar to [59], the weight parameters W_0 and W_1 can be trained via gradient descent. The $\hat{A}XW_0$ contains the first layer (mention, sense) and embeddings, and $\hat{A}RELU(\hat{A}XW_0)W_1$ contains the second layer (mention, sense) and embeddings. This two-layer GCN performs message passing between nodes to two steps away, maximum. Therefore, the two-layer GCN allows the exchange of information between pairs of nodes. This GCN model on our experimental datasets (next section) shows better performance than a one-layer model and models with more than two layers. This shows the validity of our model, based on similar results in other recent works [16,22].

4 Disambiguation Model

We used a 1-nearest neighbor approach to test **msBERT** on the disambiguation task. For each target mention m in test set, we computed its contextual embedding by means of BERT and compared it against the embeddings of msBERT associated with the senses of m, via training our GCN. Hence, we took as prediction for the target word the sense corresponding to its nearest neighbour. We note that the embeddings produced by msBERT are created by concatenating two BERT representations, i.e., context and sense (see Sect. 3.4), hence we repeated the BERT embedding of the target instance to match the number of

dimensions. In contrast to most supervised systems, this approach does not rely on the Most Frequent Sense (MFS) backoff strategy, i.e., predicting the most frequent sense of a redirect link in Wikipedia for instances unseen at training time, as msBERT ensures full coverage for the English nominal senses. At the time of cosine similarity calculation, we include considering the distances of the $R(m, s)$ with $R(D, P)$. It integrates more context at the time of finding the nearest neighbor.

4.1 Evaluation Datasets

We have two main distinct categories of datasets that we used in our experiment; First, we carried out the evaluation on the English all-words WSD framework by Raganato et al. [42], comprising five standard test sets, namely, Senseval-2 (Edmonds and Cotton, 2001), Senseval-3 (Snyder and Palmer, 2004), SemEval-07 (Pradhan et al., 2007), SemEval-13 (Navigli et al., 2013), SemEval-15 (Moro and Navigli, 2015) along with ALL, i.e., the concatenation of all the test sets. Second, we used the five Wikification datasets of KORE, MSNBC, AQUAINT, Wiki5000, and Wiki30000. The first three in this category are all from news, and the last two are the Wikipedia pages.[4]

4.2 msBERT Configuration

We used Wikipedia as input corpus since it is the largest general-domain resource currently available. We varied the number of senses (redirect pages) to give as input to the GCN between 5 and 25 with a 5 step and selected the value n = 5 by manually assessing the quality of a sample of the disambiguation output.

Table 1. F-Measure performance of WSD evaluation framework on the test sets of the all-words English datasets.

Model	Senseval-2	Senseval-3	Semeval-7	Semeval-13	Semeval-15	All
BERT	77.1 ± 0.3	73.2 ± 0.4	66.1 ± 0.3	71.5 ± 0.2	74.4 ± 0.3	73.8 ± 0.3
LMMS	76.1 ± 0.6	75.5 ± 0.2	68.2 ± 0.4	75.2 ± 0.3	77.1 ± 0.4	75.3 ± 0.2
SensEmBERT	72.4 ± 0.1	69.8 ± 0.2	60.1 ± 0.4	78.8 ± 0.1	75.1 ± 0.2	72.6 ± 0.3
ARES	78.2 ± 0.3	77.2 ± 0.1	71.1 ± 0.2	77.2 ± 0.2	83.1 ± 0.2	77.8 ± 0.1
msBERT	79.6 ± 0.5	79.3 ± 0.2	74.6 ± 0.1	78.1 ± 0.4	81.5 ± 0.4	76.5 ± 0.7

4.3 Comparison Systems

We compared msBERT against the best-performing knowledge-based systems evaluated on the disambiguation framework. These systems include the mentioned ones in the background section. **Wikisim** [46] which is the most recent

[4] We used this dataset of the second category from: https://github.com/asajadi/wikisim/tree/master/datasets.

key-entity Wikifier, and **TagME** [12] which is available as a web service[5]. We also compare with **GLOW** [43] and **Wikipedia Miner** [31] since we have their $F1$ measures on three common datasets. Lastly, we compare our approach with **RedW** [49], which is a context-free run-time oriented Wikifier. The latest knowledge base approach in this task is SensEmBERT, which is included in the compared systems.

5 Experimental Results

We now report the results of the evaluation we carried out on the English disambiguation task. In Table 1 we report the results of the underlying systems as well as our proposed system. This comparison shows the effectiveness of different approaches in finding the correct sense for each ambiguous mention, based on its context. This results show our msBERT representation is as effective as the recent state-of-the-art contextualized embeddings in lexical ambiguity. Our experiments aim to explore the effectiveness of the proposed approach for the lexical ambiguity problem. The idea of engaging more context to disambiguate entity concepts more related to their context is supported by the results in Table 1. In our other experiment, we compared the results of disambiguating the text using our approach with other knowledge-based approaches. The results of this experiment are shown in Table 2. Using proposed embedding msBERT for disambiguating the text, we are more confident about considering all relevant information in the document for disambiguating the ambiguous mentions. We compare the senses of each mention with the knowledge base information and document content, which was missing in the previous methods. The improvement in our results over the baselines is statistically significant (χ^2 test with $p < 0.05$). For this aim, one comparison we run is measuring precision, recall, and F1 measure of our algorithm with the baseline approaches.

Table 2. F-measure on the Wikification task of unsupervised knowledge-based approaches of GLOW, Wikipedia Miner, TAGME, Wikisim, and RedW in comparison with our proposed algorithm using the msBERT representation.

Method	Kore	AQUAINT	MSNBC	Wiki5000	Wiki30000
GLOW	0.68	0.72	0.73	0.65	0.66
Wikipedia Miner	0.70	0.73	0.68	0.67	0.70
TAGME	0.68	0.64	0.55	0.60	0.61
Wikisim	0.64	0.62	0.58	0.57	0.57
RedW	–	0.80	–	0.62	–
msBERT	0.74	0.79	0.67	0.73	0.75

[5] https://tagme.d4science.org/tagme/.

6 Conclusion

In this paper, we presented msBERT, an approach for producing embeddings of senses in English. The msBERT can couple the information within the knowledge base with contextual information from the document mentioned in it. This feature results in high-quality latent representations for the concepts within a lexical knowledge base. Our experiments showed that despite relying on English data only msBERT is comparable with all its alternatives on the English disambiguation task. Our graph modeling produces the integrated representation of each ambiguous mention with all its possible senses in the Wikipedia knowledge base. The other novel idea of integrating the document representation along with the first paragraph of the Wikipedia page for each mention and each sense improved the efficiency of our representations, specifically for the lexical ambiguity task. As future work, we plan to exploit the information brought by our embeddings to other related tasks, including word sense disambiguation using other knowledge bases and experiments on different datasets.

References

1. Aghaebrahimian, A., Cieliebak, M.: Named entity disambiguation at scale. In: IAPR Workshop on Artificial Neural Networks in Pattern Recognition Proceeding, pp. 102–110 (2020)
2. Aleksandrova, D., Drouin, P., Lareau, F.C.C.O., Venant, A.: The multilingual automatic detection of 'e nonc é s bias 'e s in wikip é dia. ACL (2020)
3. Amos, L., Anderson, D., Brody, S., Ripple, A., Humphreys, B.L.: UMLS users and uses: a current overview. J. Am. Med. Inform. Assoc. **27**(10), 1606–1611 (2020)
4. Azad, H.K., Deepak, A.: A new approach for query expansion using Wikipedia and wordnet. Inf. Sci. **492**, 147–163 (2019)
5. Bouma, G.: Normalized (pointwise) mutual information in collocation extraction. In: Proceedings of GSCL, pp. 31–40 (2009)
6. Camacho-Collados, J., Pilehvar, M.T.: From word to sense embeddings: a survey on vector representations of meaning. J. Artif. Intell. Res. **63**, 743–788 (2018)
7. Chen, X., Liu, Z., Sun, M.: A unified model for word sense representation and disambiguation. In: EMNLP, pp. 1025–1035 (2014)
8. Cilibrasi, R.L., Vitanyi, P.M.: The google similarity distance. IEEE Trans. Knowl. Data Eng. **19**(3), 370–383 (2007)
9. Cucerzan, S.: Large-scale named entity disambiguation based on Wikipedia data. In: EMNLP, pp. 708–716 (2007). https://www.aclweb.org/anthology/D07-1074.pdf
10. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
11. Dixit, V., Dutta, K., Singh, P.: Word sense disambiguation and its approaches. CPUH Res. J. **1**(2), 54–58 (2015)
12. Ferragina, P., Scaiella, U.: TAGME: on-the-fly annotation of short text fragments (by Wikipedia entities). In: ACM, pp. 1625–1628 (2010)
13. Hajar, E.H., Mohammed, B.: Using synonym and definition wordnet semantic relations for implicit aspect identification in sentiment analysis. In: NISS, pp. 1–5 (2019)

14. Jones, K.S.: A statistical interpretation of term specificity and its application in retrieval. J. Documentation 53–60 (1972)
15. Kim, M.C., Nam, S., Wang, F., Zhu, Y.: Mapping scientific landscapes in UMLs research: a scientometric review. J. Am. Med. Inform. Assoc. **27**(10), 1612–1624 (2020)
16. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)
17. Kraljevic, Z., et al.: MedCAT-medical concept annotation tool. arXiv preprint arXiv:1912.10166 (2019). https://arxiv.org/ftp/arxiv/papers/1912/1912.10166.pdf
18. Kwon, S., Oh, D., Ko, Y.: Word sense disambiguation based on context selection using knowledge-based word similarity. Inf. Process. Manage. **58**(4), 102551 (2021)
19. Lee, J., Fuxman, A., Zhao, B., Lv, Y.: Leveraging knowledge bases for contextual entity exploration. In: Proceedings of ACM, pp. 1949–1958 (2015)
20. Lesk, M.: Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In: Systems Documentation, pp. 24–26 (1986)
21. Li, B.: Named entity recognition in the style of object detection. arXiv preprint arXiv:2101.11122 (2021)
22. Li, Q., Han, Z., Wu, X.M.: Deeper insights into graph convolutional networks for semi-supervised learning. In: AAAI, Proceedings of the AAAI Conference on Artificial Intelligence, New Orleans, vol. 32, pp. 234–242 (2018)
23. Logeswaran, L., Chang, M.W., Lee, K., Toutanova, K., Devlin, J., Lee, H.: Zero-shot entity linking by reading entity descriptions. arXiv preprint arXiv:1906.07348. https://arxiv.org/pdf/1906.07348.pdf (2019)
24. Loureiro, D., Jorge, A.: Language modelling makes sense: propagating representations through wordnet for full-coverage word sense disambiguation. In Proceedings of ACM, pp. 5682–5691 (2019)
25. Mao, Y., Fung, K.W.: Use of word and graph embedding to measure semantic relatedness between unified medical language system concepts. J. Am. Med. Inform. Assoc. **27**(10), 1538–1546 (2020)
26. Martinez-Rodriguez, J.L., Hogan, A., Lopez-Arevalo, I.: Information extraction meets the semantic web: a survey. Semant. Web Preprint **11**, 255–335 (2020)
27. Melamud, O., Goldberger, J., Dagan, I.: context2vec: learning generic context embedding with bidirectional LSTM. In: SIGNL, pp. 51–61 (2016)
28. Mihalcea, R., Csomai, A.: Wikify!: linking documents to encyclopedic knowledge. In: ACM, pp. 233–242 (2007)
29. Mikolov, T., Chen, K., Corrado, G.S., Dean, J.: Efficient estimation of word representations in vector space. In: Proceedings of ICLR, vol. 4, pp. 321–329 (2013)
30. Miller, G.A., Beckwith, R., Fellbaum, C., Gross, D., Miller, K.J.: Introduction to wordnet: an on-line lexical database. Int. J. Lexicography **3**(4), 235–244 (1990)
31. Milne, D., Witten, I.H.: Learning to link with Wikipedia. In: Proceedings of the 17th ACM Conference on Information and Knowledge Management, pp. 509–518 (2008)
32. Munirsyah, M., Bijaksana, M.A., Astuti, W.: Development synonym set for the English wordnet using the method of comutative and agglomerative clustering. Jurnal Sisfokom (Sistem Informasi dan Komputer) **9**(2), 171–176 (2020). http://jurnal.atmaluhur.ac.id/index.php/sisfokom/article/download/855/633
33. Navigli, R.: Word sense disambiguation: a survey. ACM Comput. Surv. (CSUR) **41**(2), 1–69 (2009)

34. Nguyen, D.B., Hoffart, J., Theobald, M., Weikum, G.: AIDA-light: high-throughput named-entity disambiguation. In: LDOW, vol. 14, pp. 22–32 (2014)
35. Pasini, T., Elia, F.M., Navigli, R.: Huge automatically extracted training sets for multilingual word sense disambiguation. arXiv preprint arXiv:1805.04685 (2018)
36. Pennington, J., Socher, R., Manning, C.D.: Glove: global vectors for word representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP, Qatar, pp. 1532–1543 (2014)
37. Peters, M., et al.: Deep contextualized word representations. Association for Computational Linguistics, pp. 2227–2237 (2018)
38. Peters, M.E., Logan IV, R.L., Schwartz, R., Joshi, V., Singh, S., Smith, N.A.: Knowledge enhanced contextual word representations. arXiv preprint arXiv:1909.04164 (2019)
39. Peters, M.E., Neumann, M., Zettlemoyer, L., Yih, W.T.: Dissecting contextual word embeddings: architecture and representation. In: EMNLP, pp. 1499–1509 (2018)
40. Piccinno, F., Ferragina, P.: From TagME to WAT: a new entity annotator. In: Proceedings of the First International Workshop on Entity Recognition & Disambiguation, pp. 55–62. ACM (2014)
41. Raganato, A., Bovi, C.D., Navigli, R.: Automatic construction and evaluation of a large semantically enriched Wikipedia. In: IJCAI, pp. 2894–2900 (2016)
42. Raganato, A., Bovi, C.D., Navigli, R.: Neural sequence learning models for word sense disambiguation. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pp. 1156–1167 (2017)
43. Ratinov, L., Roth, D., Downey, D., Anderson, M.: Local and global algorithms for disambiguation to Wikipedia. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1, pp. 1375–1384 (2011)
44. Reisinger, J., Mooney, R.: Multi-prototype vector-space models of word meaning. In: Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, pp. 109–117 (2010)
45. Saeidi, M., Sousa, S.B.d.S., Milios, E., Zeh, N., Berton, L.: Categorizing online harassment on Twitter. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases, pp. 283–297 (2019)
46. Sajadi, A.: Semantic analysis using Wikipedia graph structure. Ph.D. thesis, Dalhousie University (2018)
47. Scarlini, B., Pasini, T., Navigli, R.: SensEmBERT: context-enhanced sense embeddings for multilingual word sense disambiguation. In: AAAI, pp. 8758–8765 (2020)
48. Scarlini, B., Pasini, T., Navigli, R.: With more contexts comes better performance: contextualized sense embeddings for all-round word sense disambiguation. In: EMNLP, pp. 3528–3539 (2020)
49. Shnayderman, I., et al.: Fast end-to-end wikification. arXiv preprint arXiv:1908.06785 (2019)
50. Singh, H., Bhattacharyya, P.: A survey on word sense disambiguation. ACM Comput. Surv. (CSUR) (2019)
51. Song, Y., Roth, D.: Machine learning with world knowledge: the position and survey. arXiv preprint arXiv:1705.02908 (2017)
52. Sysoev, A., Nikishina, I.: Smart context generation for disambiguation to Wikipedia. In: Conference on Artificial Intelligence and Natural Language, pp. 11–22 (2018)

53. Szymański, J., Naruszewicz, M.: Review on wikification methods. AI Commun. **27**(2), 97–111 (2019)
54. Wang, Y., Wang, M., Fujita, H.: Word sense disambiguation: a comprehensive knowledge exploitation framework. Knowl. Based Syst. 105–117 (2019)
55. Weikum, G., Dong, L., Razniewski, S., Suchanek, F.: Machine knowledge: creation and curation of comprehensive knowledge bases. arXiv preprint arXiv:2009.11564 (2020)
56. West, R., Paranjape, A., Leskovec, J.: Mining missing hyperlinks from human navigation traces: a case study of Wikipedia. In: Proceedings of the 24th International Conference on World Wide Web, pp. 1242–1252 (2015)
57. Xin, K., Hua, W., Liu, Y., Zhou, X.: LoG: a locally-global model for entity disambiguation. World Wide Web **24**, 1–23 (2020)
58. Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R.R., Le, Q.V.: XLNet: generalized autoregressive pretraining for language understanding. Adv. Neural Inf. Process. Syst. **32**, 221–229 (2019)
59. Yao, L., Mao, C., Luo, Y.: Graph convolutional networks for text classification. In: Proceedings of the AAAI Conference on Artificial Intelligence, AAAI, Honolulu, vol. 33, pp. 7370–7377 (2019)
60. Zhang, Y., Ives, Z., Roth, D.: "who said it, and why?" provenance for natural language claims. In: ACL, pp. 4416–4426 (2020)
61. Zhao, G., Wu, J., Wang, D., Li, T.: Entity disambiguation to Wikipedia using collective ranking. Inf. Process. Manage. **52**(6), 1247–1257 (2016)