Robust and Collective Entity Disambiguation through Semantic Embeddings

Stefan Zwicklbauer
University of Passau
Passau, Germany
szwicklbauer@acm.org

Christin Seifert
University of Passau
Passau, Germany
christin.seifert@uni-passau.de

Michael Granitzer
University of Passau
Passau, Germany
michael.granitzer@uni-passau.de

ABSTRACT

Entity disambiguation is the task of mapping ambiguous terms in natural-language text to its entities in a knowledge base. It finds its application in the extraction of structured data in RDF (Resource Description Framework) from textual documents, but equally so in facilitating artificial intelligence applications, such as Semantic Search, Reasoning and Question & Answering. We propose a new collective, graph-based disambiguation algorithm utilizing semantic entity and document embeddings for robust entity disambiguation. Robust thereby refers to the property of achieving better than state-of-the-art results over a wide range of very different data sets. Our approach is also able to abstain if no appropriate entity can be found for a specific surface form. Our evaluation shows, that our approach achieves significantly (>5%) better results than all other publicly available disambiguation algorithms on 7 of 9 datasets without data set specific tuning. Moreover, we discuss the influence of the quality of the knowledge base on the disambiguation accuracy and indicate that our algorithm achieves better results than non-publicly available state-of-the-art algorithms.

CCS Concepts

•**Information systems** → **Information extraction;** *Information systems applications;*

Keywords

Entity Disambiguation; Neuronal Networks; Embeddings

1. INTRODUCTION

Entity disambiguation refers to the task of linking phrases in a text, also called surface forms, to a set of candidate meanings, referred to as the knowledge base (KB), by resolving the correct semantic meaning of the surface form. It is an essential task in combining unstructured with structured or formal information; a prerequisite for artificial intelligence applications such as Semantic Search, Reasoning and Question & Answering. While entity disambiguation systems have been well-researched so far [19, 14, 5, 12, 15], most approaches have been optimised to work on a particular type of disambiguation task, like for example on short Twitter messages [2], web pages [12, 28], news documents [10, 15], encyclopedias [16, 11, 5], RSS-Feeds [9] etc. While most authors report to outperform other entity disambiguation algorithms on their domain/data set, they do not achieve comparable accuracy on other domains. So their approaches could be considered as not being very robust against different types of data sets.

In our work, we focus on robust entity disambiguation, where robustness is defined as achieving high accuracy over a large set of different domains. The only assumption we make is to disambiguate entities collectively, i.e. we disambiguate all entities in a given textual document at once. Our approach is based on creating a k-partite relatedness graph between all entity candidates for all given surface forms in a document. The relatedness between entity candidates is determined using semantic embeddings, i.e. real-valued n-dimensional vectors capturing the semantics of entities. We use two types of semantic embeddings, namely embeddings capturing the meaning on the word (entity) level and semantic embeddings capturing the meaning on the document level. As we show, both embeddings are important for our approach.

In particular, we provide the following contributions:

- We present a new state-of-the-art approach for collective entity disambiguation that emphasizes robustness, i.e. achieving high accuracy over different kinds of data sets. Our approach is based on semantic embeddings that capture entity and document contexts and utilizes a sequence of graph algorithms to eliminate wrong candidates.
- We evaluate our algorithm against 5 strong, publicly available entity disambiguation systems on 9 data sets and show that our approach outperforms all other system by a significant margin on nearly all data sets.
- We discuss the influence of the quality of the underlying KB on the entity disambiguation accuracy and indicate that our algorithm achieves better results than non-publicly available current state-of-the-art algorithms.
- We provide our entity disambiguation system as well as the underlying KB as open source solution. These resources allow a fair comparison between future entity disambiguation algorithms and our approach that are not biased by the KB.

The remainder of the paper is structured as follows: In Section 2, we review related work. Section 3 introduces the problem formally and outlines our approach. Sections 4 and 5 explain the process of generating semantic embeddings and our approach. In Section 6 and 7, we describe the experimental setup and the results achieved on 9 data sets. We conclude our paper in Section 8.

SIGIR '16, July 17 - 21, 2016, Pisa, Italy

ISBN 978-1-4503-4069-4/16/07. . . $15.00

DOI: http://dx.doi.org/10.1145/2911451.2911535

2. RELATED WORK

Entity disambiguation has been studied extensively in the past 10 years. One of the first works defined a similarity measure to compute the cosine similarity between the text around the surface form and the referent entity candidate's Wikipedia page [4]. The publicly available framework DBpedia Spotlight [19] for disambiguating Linked Data Resources is also based on the vector space model and cosine similarity. Cucerzan et al. introduced topical coherence for entity disambiguation [7]. The authors used the referent entity candidate and other entities within the same context to compute topical coherence by analyzing the overlap of categories and incoming links in Wikipedia. Milne and Witten [22] improved the exploitation of topical coherence using Normalized Google Distance and unambiguous entities in the context only. A well-known publicly available system is Wikifier [25, 5] from 2013. It incorporates, along with statistical methods, richer relational analysis of the text. In 2014, the authors Guo et al. [10] proposed the use of a probability distribution resulting from a random walk with restart over a suitable entity graph to represent the semantics of entities and documents in a unified way. Their algorithm updates the semantic signature of the document as surface forms are disambiguated.

Other works explicitly make use of topic models to link surface forms to a KB. For instance, Kataria et al. [16] proposed a topic model that uses all words of Wikipedia to learn entity-word associations and the Wikipedia category hierarchy to capture co-occurrence patterns among entities. The authors of [11] also propose a generative approach which jointly models context compatibility, topic coherence and its correlation.

Several other work focus on graph-based algorithms. For instance, publicly available, graph-based disambiguation approaches are AIDA [14], Babelfy [23], WAT [24] and AGDISTIS [28]. AIDA is based on the YAGO2 KB and relies on sophisticated graph algorithms. This approach uses dense sub-graphs of the underlying KB to identify coherent surface forms using a greedy algorithm. In 2014, Babelfy [23] has been presented to the community. It is based on random walks and densest subgraph algorithms. In contrast to our work, Babelfy differentiates between word sense disambiguation, i.e., resolution of polysemous lexicographic entities like play, and entity disambiguation. The WAT [24] system is a redesign of TagMe [8] components and introduces two disambiguation families: graph-based algorithms for collective entity disambiguation and vote-based algorithms for local entity disambiguation [29]. Finally, AGDISTIS [28] is based on string similarity measures and the graph-based Hypertext-Induced Topic Search algorithm. AGDISTIS disambiguates named entities only and exclusively relies on RDF-KBs like DBpedia or YAGO2. All these collective disambiguation approaches rely on graph algorithms but mostly compute the coherence measure with the help of relations between entities within KBs (i.e. DBpedia, YAGO2).

Another graph-based approach was presented by Alhelbawy et al. [1], who applied the PageRank algorithm to a disambiguation graph. To compute the edge weights between entity candidates the authors either used a boolean relation whether two entities refer to each other or estimated a probability of both entities appearing in the same sentence. The authors Han et al. [12] proposed the graph-based representation called Referent Graph, which models the global interdependence between different disambiguation decisions. Then, they proposed a collective inference algorithm, which can jointly infer the referent entities of all name surface forms by exploiting the interdependence captured in the Referent Graph.

Semantic embeddings have also been used for entity disambiguation. In 2013, He et al. [13] proposed an entity disambiguation model, based on Deep Neural Networks. The model learns a context-entity similarity measure for entity disambiguation. The intermediate representations are learned leveraging large-scale an- notations of Wikipedia. The most recent approach of semantic embeddings is presented by Huang et al. [15] in 2015. The authors present a new entity semantic relatedness model for topical coherence modeling. Similar to our approach the model can be directly trained on large-scale knowledge graphs. It maps heterogeneous types of knowledge of an entity from knowledge graphs to numerical feature vectors in a feature space such that the distance between two semantically-related entities is minimized. Unfortunately, the approach is only evaluated on two data sets and, thus, the robustness properties of this approach are debatable.

3. PROBLEM STATEMENT & APPROACH

The goal of entity disambiguation is to find the correct semantic mapping between surface forms in a document and entities in a KB. More formally, let $M =< m_1, ..., m_K >$ be a tuple of K surface forms in a document D, and $\Omega = \{e_1, ..., e_{|\Omega|}\}$ be a set of target entities in a KB. Let Γ be a possible entity configuration $< t_1, ..., t_K >$ with $t_i \in \Omega$, where t_i is the target entity for surface form m_i. Here, we assume that each entity in Ω is a candidate for surface form m_i. The goal of collective entity disambiguation can then be formalized as finding the optimal configuration Γ^* [25]:

$$\Gamma^* = \underset{\Gamma}{\operatorname{argmin}} \sum_{i=1}^{K} \phi(m_i, t_i) + \Psi(\Gamma) \tag{1}$$

Different to [25] we do **not** pose the optimization problem as maximizing the sum of the scores of a locality function ϕ and a coherence function Ψ (cf. Equation 1), which has been proven to be NP-hard [7]. We approximate the solution using the PageRank (PR) algorithm with priors [3, 30] on specially constructed graphs which encompass the locality and the coherence function (still NP-hard). Our locality function reflects the likelihood that a target entity t_i is the correct disambiguation for m_i, whereas the coherence function computes a score describing the coherence between entities in Γ^*.

The PR algorithm is a well-researched, link-based ranking algorithm simulating a random walk on graphs and reflecting the importance of each node. It has been shown to provide good performance in many applications [31], also in disambiguation tasks [12, 1].

The graphs we construct consist of nodes for all entity candidates per surface form and one node, the topic node, that represents the current predominant topic of already disambiguated entities. This topic node allows us to include a-priori information from previous steps into the structure of the graph and, hence, influence the PR algorithm. The edge weights are based on similarities between **entity embeddings** as well as similarities between **entity-context embeddings** and the surface forms' surrounding context. In our work, an entity embedding is a trained vector that is used to compute the semantic similarity between entities. Moreover, an entity-context embedding is a trained context vector of an entity to compute a matching how good this entity fits to the context of a surface form.

Abstaining is an important task if no appropriate entity can be found in the entity set Ω. In this case, our disambiguation algorithm returns the pseudo-entity *NIL*. In our work, a surface form is linked to *NIL* if one of the following cases occurs: 1. when the surface form has no candidate entities, or 2. the algorithm is uncertain about the relevant entity after the last PR application.

4. SEMANTIC EMBEDDINGS

Embeddings are n-dimensional vectors of concepts which describe the similarities between these concepts using the cosine similarity. This has already been well researched for words [20, 21] and documents [18] in literature. In this work, we make use of both embedding types in form of entity embeddings (Word2Vec) and entity-context embeddings (Doc2Vec) to improve entity disambiguation. First, we briefly introduce Word2Vec, a set of models that are used to produce word embeddings, and Doc2Vec, a modification of Word2Vec to generate document embeddings, in Section 4.1. Second, we describe how we create corpora that serve as input for the Word2Vec and Doc2Vec algorithms in Section 4.2.

4.1 Learning Semantic Embeddings

Word2Vec is a group of state-of-the-art, unsupervised algorithms for creating word embeddings from (textual) documents [20]. To train these embeddings, Word2Vec uses a two-layer neural network to process non-labeled documents. The neuronal network architecture is based either on the continuous bag-of-words (CBOW) or the skip-gram architecture. Using CBOW, the input to the model for a word w_i are the words preceding and succeeding this word, e.g. $w_{i-2}, w_{i-1}, w_{i+1}, w_{i+2}$ when using two words before and after the current word. The output of the network is the probability of w_i being the correct word. The task can be described as predicting a word given its context. The skip-gram model works vice-versa: the input to the model is a word w_i and Word2Vec predicts the surrounding context words $w_{i-2}, w_{i-1}, w_{i+1}, w_{i+2}$. In contrast to other natural language neuronal network models, Word2Vec models can be trained very fast and can be further significantly improved by using parallel training [26].

An important property of Word2Vec is that it groups the vectors of similar words together in the vector space. If sufficient data is used for training, Word2Vec makes highly accurate guesses about a word's meaning based on its context words in the training corpus. The resulting word embeddings capture linguistic regularities, for instance the vector operation $vec(\text{“President”}) - vec(\text{“Power”}) \approx vec(\text{“Prime Minister”})$. The semantic similarity between two words, which is important in the context of our work, denotes the cosine similarity between the words' Word2Vec vectors.

Since our approach considers only the semantic similarity between entities (and not words), we treat entities similar to words. This means, we build entity embeddings with the help of an entity corpus instead of a textual corpus containing sentences and paragraphs (cf. Section 4.2). Our evaluation of the influence of the specific architecture in Section 7.3 shows that the skip-gram model outperforms CBOW in our disambiguation setting. Thus, for all other experiments in this paper, the skip-gram model is used.

Doc2Vec, a modification of Word2Vec, learns fixed-length embeddings from variable-length pieces of texts like documents [18]. It addresses some of the key weaknesses of bag-of-word models by incorporating more semantics and considering the word order within a small context. As an example for the semantic embedding, the Doc2Vec model embeds the word “powerful” closer to “strong” than to “Paris”, which is not the case in bag-of-word models.

The architecture is either based on the distributed memory model (PV-DM), which is similar to the CBOW model of Word2Vec, or on the distributed bag-of-words model (PV-DBOW), which is similar to the skip-gram model. Using PV-DM, the context words' corresponding document vector d_i (vector of the document which contains the context words) is added as an input in the neural net. Thus, the input becomes d_i, w_{j-1}, w_{j+1}, meaning that the document vector with the two context vectors is used to predict the word w_j. For more details on calculating the document vector see [18].

Similar to the skip-gram model in Word2Vec, the intention of PV-DBOW is to ignore the context words in the input, but force the model to predict words randomly sampled from the document in the output. A disadvantage is, that it ignores the word sequence.

The authors of Doc2Vec report consistently better results with the PV-DM architecture. PV-DM also outperforms PV-DBOW in the context of our disambiguation algorithm (cf. Section 7.3). Since we want to compute the similarity between an entity-context embedding and the surrounding context of a surface form, one needs to perform an inference step to compute the surrounding context vector. The Doc2Vec model is trained on the entity-context corpus yielding the entity-context embeddings (see next section), and the same model is later used to generate the surface-form-context embeddings. For a detailed introduction to Word2Vec and Doc2Vec, we refer the reader to the original works [20, 21, 18].

4.2 Corpus Creation

Word2Vec typically accepts a set of corpora containing natural language text as input and trains its word vectors according to the words' order in the corpora. Since we want to learn **entity representations** (entity embeddings) only, we have to create an appropriate Word2Vec input corpus file that exclusively comprises entities. The entities' order in the corpus file should be retained as given in the entity-annotated document KBs. For this purpose, we iterate over all documents in these corpora and replace all available, linked surface forms with its respective target entity identifier. Further, all non-entity identifiers like words and punctuations are removed so that all documents consist of entity identifiers separated by whitespaces only. However, the collocation of entities is maintained as given by the original document, but the distance between the annotations is ignored. All resulting documents are concatenated to create a single Word2Vec corpus file. Details are provided in [32].

To generate **entity-context embeddings** with Doc2Vec, any natural language source can be used that offers sufficient descriptions of the entities. To generate embeddings, the source needs to be either already a single document, or needs to be aggregated to a single document. These documents describe the respective entities as detailed as possible. A well-known example for entity describing documents are Wikipedia pages, which are used in our experiments.

5. DISAMBIGUATION ALGORITHM

Our disambiguation algorithm accepts documents that contain one or multiple surface forms that should be linked to entities. It disambiguates all surface forms within a document using a graph-based collective approach. Algorithm 1 gives an overview of the disambiguation process, which is explained in the following. The first step in the disambiguation chain is the *Candidate Generation*. The goal is to reduce the number of possible entity candidates for each input surface form m_i by determining a set of relevant target entities, the candidate set CS_{m_i}. Details of our candidate generation process are described in Section 5.1. Given these candidates we disambiguate surface forms with none or one candidate and initialize the entity set E_d with the entities of unambiguous surface forms or already disambiguated surface forms (Lines 2-7).

Our second step *Semantic Embedding Candidate Filter* filters entity candidates that fit to the general topic described by the already disambiguated entities (Lines 8-17) requiring at least 3 already assigned entities. The underlying assumption is, that all entities in a paragraph are somehow topically related. To infer this general topic, we create a topic vector $tv = \sum_{e \in E_d} v(e)$, with E_d being the set of already disambiguated entities, and $v(e)$ the entity embedding of entity e (Word2Vec vector). Next, we compute the semantic similarity (cosine similarity) between the general topic vector tv and the entity candidates of all not yet disambiguated surface forms. If the similarity exceeds the a-priori given *CandidateFilter* threshold, the entity candidate remains in the candidate list of the respective surface form. If no candidate of a specific surface form

Algorithm 1: Our graph-based disambiguation algorithm

```
   input : $M =< m_1, ..., m_K >$, Threshold $\lambda$, $margin_1$, $margin_2$
   output: Assignment $\Gamma^* =< t_1, ..., t_K >$, with $t_i$ being the entity of $m_i$
1  configuration $\Gamma^* = tuple()$; dis. entities $E_d = \emptyset$; candidate set $CS_{m_i} = \emptyset$
   // Candidate Generation
2  for $m_i \in M$ do
3      $CS_{m_i} =$ generateCandidates($m_i$)
4      if $|CS_{m_i}| = 0$ then
5          $\Gamma(i) = NIL$
6      else if $|CS_{m_i}| = 1$ then
7          $\Gamma^*(i) = e \in CS_{m_i}; E_d = E_d \cup CS_{m_i}$
   // Semantic Embedding Candidate Filter
8  if $|E_d| > 2$ then
9      for $m_i \in M$ and $|CS_{m_i}| > 1$ do
10         $set = \emptyset$
11         for $e_j \in CS_{m_i}$ do
12             if $cosineSim(sumEmbeddings(E_d), e) > \lambda$ then
13                 $set = set \cup e$
14         if $set \neq \emptyset$ then
15             $CS_{m_i} = set$
16         if $|set| = 1$ then
17             $\Gamma^*(i) = CS_{m_i}; E_d = E_d \cup CS_{m_i}$
   // High Probability Candidate Disambiguation
18 CreateDisambiguationGraphAndSolvePageRank($CS_{m_i}, E_d$)
19 Rank candidates.
20 Select highest PR score $h$, second highest PR score $s$, average PR score $avg$.
21 for $m_i \in M$ and $|CS_{m_i}| > 1$ do
22     $dynThreshold = h - margin_2 \cdot (h - avg)$
23     if $s < dynThreshold$ then
24         $\Gamma^*(i) = h; E_d = E_d \cup h; C_{m_i} = h$
25     else
26         $CS_{m_i} = selectTop4RankedCandidates$
   // Final Disambiguation and Abstaining
27 CreateDisambiguationGraph($CS_{m_i}, E_d$)
28 for $m_i \in M$ and $|CS_{m_i}| > 1$ do
29     Perform PR and rank candidates.
30     Select highest PR score $h$, second highest PR score $s$, average PR score $avg$.
31     $abstainingThreshold = margin_2 \cdot (h - avg)$
32     if $s < abstainingThreshold$ then
33         $\Gamma^*(i) = h; E_d = E_d \cup h; C_{m_i} = h$
34     else
35         $\Gamma^*(i) = NIL; C_{m_i} = \emptyset$
36     updateGraph($CS_{m_i}, E_d$)
```

exceeds the threshold, the candidate set for this surface forms remains unchanged. We note that this filter is a crucial step towards fast and accurate entity disambiguation. Omitting this step results in a significantly lower performance combined with decreasing results ($\approx$ 2 to 5 percentage points F1, depending on the data set).

The third step *High Probability Candidate Disambiguation* comprises the PageRank (PR) application on a disambiguation graph to disambiguate high probability candidates (Lines 18-26). Detailed information for graph construction and PR can be found in Section 5.2. Next, we rank the entity candidates for each surface form according to their relevance score given by PR in descending order. Additionally, we select the highest PR score h, second-highest PR score s and average PR score avg across all entities that belong to the same surface form. Given these parameters, we define a threshold $dynT$ for determining the certainty in the ranking based on the differences between the first and the second ranked candidate:

$$dynT = h - margin_1 \cdot (h - avg) \tag{2}$$

whereas details on the parameter $margin_1$ are discussed in Section 6.2. We use this threshold as a certainty criterion, indicating whether the top-ranked entity candidate of a surface form is the correct disambiguation target. More specific, if the PR score s of the second ranked candidate does not exceed the threshold $dynT$, the highest ranked entity denotes the target entity of its surface form. In other words, if the relevance score margin between the highest ranked candidate and the other candidates is large, then the likelihood of the top-ranked candidate being the correct target entity is also high. If the threshold is exceeded, we reduce the candidate set of the respective surface form to the top 4 ranked entity candidates.

The last step *Final Disambiguation and Abstaining* disambiguates the remaining entities or abstains if the algorithm is uncertain about the correct target entity (Lines 27-36). We first create a disambiguation graph (cf. Section 5.2) and, then, iteratively disambiguate the entities of the remaining surface forms. For this purpose, every iteration applies the PR to the underlying graph and ranks the candidate entities of each surface form in descending order. The scores h, s, and avg are calculated as in the previous step. The abstaining threshold *abstainingThreshold* is calculated using formula 2 with a different margin parameter ($margin_2$). If the second ranked entity candidate exceeds the abstaining threshold *abstainingThreshold*, the algorithm returns the *NIL* identifier for the respective surface form. Otherwise, the top ranked entity candidate denotes the target entity. After every iteration, we update the graph according to the changes in candidates and disambiguated entities and proceed until all surface form have been processed.

We note, that we apply the PR only once in step 3 due to performance reasons. The disambiguation graph in step 4 usually does not include many entity candidates and, thus, we apply the PR in every iteration, also to provide the maximum accuracy in the abstaining task. The $margin$ parameter to compute the *high probability threshold* and *abstaining threshold* varies in both steps. Information about the parameter choice is presented in Section 6.2.

5.1 Candidate Generation

In the first step, the goal is to reduce the number of possible entity candidates for each input surface form m_i by determining a set of relevant target entities. We proceed as follows:

First, we search for all those entities that have already been annotated with m_i in a corpus. All entities that provide an exact surface form matching serve as entity candidate. To perform this task our algorithm relies on a pre-built index, which has to be created before entities can be disambiguated (cf. Section 6.1). If the candidate set is empty, we additionally use the candidate generation approach proposed by Usbeck et al. for AGDISTIS [28], which includes String normalization and String comparison via trigram similarity. The corresponding parameters are adopted from the default settings in the AGDISTIS framework[1].

Gathering all relevant entity candidates might result in a long list of candidates. To keep the list short and to improve the efficiency, we prune noisy candidates according to the following three criteria:

1. **Prior probability:** Some entities (e.g. Influenza) occur more frequently than others in documents (e.g. the IIV3-011L gene). Thus, these popular entities have a higher probability to reoccur in other documents. In our work, the prior probability [19] $p(e_i|m)$ estimates the probability of seeing an entity with a given surface form. We select the top x entities as the candidates to keep the popular candidates.

2. **Context similarity:** We select the top x entities ranked by their context matching. To this end, we compute the cosine similarity between the entity-context embeddings and the Doc2Vec inferred context vector of the surface form.

[1] http://aksw.org/Projects/AGDISTIS.html

3. **Entity-topic similarity:** If a document contains at least two surface forms that have already been disambiguated ($|E_d| > 1$), we create a topic vector $tv = \sum_{e \in E_d} v(e)$, with $v(e)$ denoting the entity embedding of e, and E_d is the set of already disambiguated entities. In the following, we select those remaining candidates of each surface form where the cosine similarity between the entity candidate embedding and tv exceeds the *CandidateFilter* threshold (cf. Section 1).

For all criteria we use $x = 8$, which is enough to capture the relevant entity candidates. An experimental increase of x, results in a negligibly higher recall of the candidate generation task, but significantly decreases disambiguation performance.

5.2 Disambiguation Graph and PageRank

In our approach, we generate a disambiguation graph twice in order to disambiguate high probability candidate entities first and to perform abstaining afterwards. On this graph we perform a random walk and determine the entity relevance which can be seen as the average amount of its visits. The random walk is simulated by a PR algorithm that permits edge weights and non-uniformly-distributed random jumps [3, 30].

First, we create a **complete, directed** K-partite graph whose set of nodes V is divided in K disjoint subsets $V_0, V_1, ..., V_K$. K denotes the amount of surface forms and V_i is the set of generated entity candidates $\{e^i_1, ..., e^i_{|V_i|}\}$ for surface form m_i. We define m_0 as pseudo surface form and use the subset $V_0 = \{e^0_1\}$ to contain the topic node. The topic node represents the average topic of all already disambiguated entities in E_d. Hence, the edge weight between an entity e^i_a and the topic node e^0_1 represents the relatedness between e^i_a and all already disambiguated entities. Since our graph is K-partite, there are only directed, weighted edges between entity candidates that belong to different surface forms. Connecting the entities that belong to the same surface form would be wrong since the correct target entities of surface forms are determined by the other surface forms' entity candidates (coherence).

The edge weights in our graph represent entity transition probabilities (ETP) which describe the likelihood to walk from a node to the adjacent node. We compute these probabilities by first computing the *Transition Harmonic Mean* (THM) between two nodes. The THM is the harmonic mean between two nodes' **semantic similarity** and the **context similarity** of the target entity (cf. Equation 3).

The semantic similarity between two nodes is the cosine similarity (cos) of the entities' semantic embeddings (vectors) $v(e^i_a)$ and $v(e^j_b)$. The semantic embedding of our topic node e^0_1 is the sum of all entity embeddings in E_d (i.e. $v(e^0_1) = \sum_{e \in E_d} v(e)$). The context similarity between the target entity e^j_b and the surrounding context of its surface form m_j is the cosine similarity of e^j_b's entity-context embedding $cv(e^j_b)$, and the inferred surrounding context vector $cv(m_j)$ of m_j. In case, the target entity is our topic node the context similarity equals 0. The ETB is computed by normalizing the respective THM value (cf. Equation 4).

$$THM(e^i_a, e^j_b) = \frac{2 \cdot cos(v(e^i_a), v(e^j_b)) \cdot cos(cv(e^j_b), cv(m_j))}{cos(v(e^i_a), v(e^j_b)) + cos(cv(e^j_b), cv(m_j))} \tag{3}$$

$$ETP(e^i_a, e^j_b) = \frac{THM(e^i_a, e^j_b)}{\sum_{k \in (V \setminus V_i)} THM(e^i_a, k)} \tag{4}$$

Given the current graph, we additionally integrate a possibility to jump from any node to any other node in the graph during the random walk with probability α. Typical values for α (according to the original paper [30]) are in the range $[0.1, 0.2]$. We compute a probability for each entity candidate being the next jump target. We employ the prior probability as jump probability for each node (entity). The probability to jump to or from the topic node equals 0. Combining the prior probability and context similarity instead leads to decreasing results of ≈ 3 percentage points F1 on average.

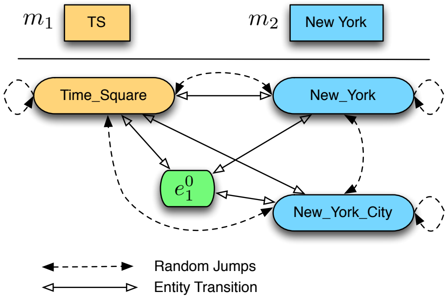

Figure 1: Entity candidate graph with candidates for the surface forms "TS" and "New York".

Figure 1 shows a possible entity candidate graph. The surface form "TS" has only one entity candidate and consequently has already been linked to the entity "Time Square". The surface form "New York" is still ambiguous, providing two candidates. The topic node e^0_1 comprises the already disambiguated surface form "Time Square". We omit the edge weights and jump probabilities in the figure to improve visualization.

After constructing the disambiguation graph, we apply the PR algorithm and compute a relevance score for each entity candidate. Depending on the disambiguation task, our approach decides which entity candidate is the correct target entity or abstains if no appropriate candidate is available (cf. Algorithm 1).

6. EXPERIMENTAL SETUP

Our disambiguation algorithm is fully-implemented in Java and is embedded in our publicly available disambiguation system *DoSeR*[2] (**D**isambiguation **o**f **Se**mantic **R**esources) which is being developed continuously. For the Word2Vec and Doc2Vec algorithms we chose Gensim [26], a robust and efficient framework to realize unsupervised semantic modeling from plain text. Before we report the disambiguation results in detail, we first describe the preprocessing details in Section 6.1, important parameter settings and evaluation measures in Section 6.2 and 9 test data sets in Section 6.3.

6.1 Preprocessing

Before our algorithm is able to disambiguate entities, we first have to perform some preprocessing steps. First, we choose a KB whose entities define our target entity set Ω. In the context of this work, we make use of the current version of DBpedia 2015-04 as entity data base, which reflects information from the last years Wikipedia version. Overall, we extract $\approx 4.1M$ entities (all entities belonging to the *owl:thing* class) out of DBpedia that we would like to disambiguate in our work.

Next, we select Wikipedia ($\approx 81M$ annotations) and the Google Wikilinks Corpus[3] ($\approx 40M$ annotations) as entity-annotated document KB that serve as training data for our semantic entity embeddings (Word2Vec). To create the Doc2Vec entity-context embeddings, we parse the entities' Wikipedia pages and remove all Wikipedia syntax elements as well as tables. The resulting natural-language texts serve as input for the Doc2Vec algorithm.

[2] https://github.com/quhfus/DoSeR

[3] https://code.google.com/p/wiki-links/

In the following, we learn entity embeddings and entity-context embeddings with Word2Vec and Doc2Vec. To train the entity embeddings with Word2Vec, we define a feature space of $d = 400$ dimensions and employ the skip-gram architecture that performs better with infrequent words [20]. In terms of Doc2Vec we use a feature space of $d = 1000$ dimensions and employ the PV-DM learning architecture. An experimental comparison between the architectures and various settings for parameter d is presented in Section 7.3. The Word2Vec training time takes ≈ 90 minutes on our personal computer with a 4x3.4GHz Intel Core i7 processor and 16 GB RAM (1 corpus iteration). The training time for Doc2Vec takes $\approx 11/2$ days since we performed 5 iterations overall.

Next, we create a disambiguation index with each entry defining an entity. The index comprises the following three entity describing information which are relevant for our disambiguation algorithm:

1. **Labels:** By default, we extract the *rdfs:label* attributes of DBpedia, which are used by the candidate generation approach proposed by Usbeck et al. [28] (cf. Section 5.1).
2. **Surface Forms:** We gather and generate surface forms from DBpedia and Wikipedia and store them separately in a surface form field. Additionally, we store the amount of occurrences how often a surface form has been annotated with an entity to compute the prior probability (cf. Section 5.1).
3. **Semantic Entity Embeddings:** We store the semantic embeddings created with Word2Vec and Doc2Vec in our index to provide fast access.

Our manually-constructed disambiguation index is publicly available on the GitHub page.

6.2 Parameters and Evaluation Measures

Our approach offers several parameters to tweak the disambiguation results. In the following, we will mention only those that have the most impact on the results. An overview of all parameters can be found on the GitHub page.

- **Surrounding Context:** For Doc2Vec we use a surrounding context of 75 words, which denotes that 75 words before and after the surface forms form the context. Using more context words, results in less meaningful query vectors. However, the best context size also depends on the domain of the input documents and should be adapted accordingly (e.g. tables).
- **Candidate Filter:** The cosine similarity ranges from -1 (unequal) to 1 (equal). A reasonable way to tune λ is to sweep the value between $0.25 < \lambda < 0.8$ (necessary similarity). We select the value $\lambda = 0.57$ according to the best averaged F1 values throughout the experiments.
- **PageRank:** We perform 100 PR iterations since the overall results do not change with more iterations. In terms of the PR jump probability α, we choose $\alpha = 0.1$ in algorithm step 3 (according to the original paper [30]). In algorithm step 4 we choose $\alpha = 0.2$ to increase the prior influence (i.e. a robust baseline) since the correct entity could not been determined with the help of topical coherence in the steps before. In the disambiguation step, *High Probability Candidate Disambiguation*, we determined the parameter $margin_1 = 0.5$ by sweeping the value between $0.2 < margin_1 < 0.6$. Again, the best value is selected according to the best averaged F1 values throughout the experiments.
- **Abstaining:** We note that abstaining is disabled by default using a $margin_2 = -\infty$. To provide the best abstaining results we chose $margin_2 = 0.3$ by sweeping the value between $0.2 < margin_2 < 0.6$ as described above.

Table 1: Statistics of our test corpora.

Data Set	Topic	#Doc.	#Ent.	Ent./Doc.	Annotation
ACE2004	news	57	253	4.44	voting
AIDA TestB	news/web	231	4458	19.40	voting
AQUAINT	news	50	727	14.50	voting
DBpedia Spot.	news	58	330	5.69	domain experts
MSNBC	news	20	658	32.90	domain experts
N3-Reuters	news	128	650	5.08	voting
IITB	news/web	103	11245	109.01	domain experts
Microposts	tweets	1165	1440	1.24	domain experts
N3-RSS 500	RSS-feeds	500	524	1.05	domain experts

In our evaluation, we use the well-known standard measures: *recall* ρ, *precision* π, *F1* and *accuracy* a. Given the ground truth G and the output of an entity disambiguation system O, in which G_{ent} and O_{ent} are the sets of surface forms that link to entities, and G_{nil} and O_{nil} are the sets of surface forms that link to NIL ($G = G_{\text{ent}} \cup G_{\text{nil}}$, $O = O_{\text{ent}} \cup O_{\text{nil}}$, and $|G| = |O|$):

$$\rho = \frac{|G_{ent} \cap O_{ent}|}{|G_{ent}|}, \quad \pi = \frac{|G_{ent} \cap O_{ent}|}{|O_{ent}|}, \quad F1 \quad = \frac{2 * \pi * \rho}{\rho + \pi}$$

$$a = \frac{|G_{ent} \cap O_{ent}| + |G_{nil} \cap O_{nil}|}{|G|}$$

6.3 Data Sets

An often occurring problem in evaluating entity disambiguation systems is that authors often download available data sets and ignore those entity ground truth annotations that are not available in their underlying KB [12, 14, 13]. Thus, results often slightly differ due to different data set queries across literature. To avoid this problem, the authors of Uzbeck et al. [29] proposed GERBIL - General Entity Annotator Benchmark, an easy-to-use platform for the agile comparison of annotators using multiple data sets and uniform measuring approaches. Being a web-based platform it can be also used to publish the disambiguation results. The reported results of our approach and competitive systems are based on this platform and serve as comparable results for future systems.

In the following, we present nine well-known and publicly available data sets which are integrated in GERBIL and are used in our evaluation. All data sets provide different characteristics in form of surface form frequency and length of surrounding context (cf. Table 1). We evaluate our algorithm on all these data sets to demonstrate the robustness across different documents/data sets.

1. **ACE2004:** This data set from Ratinov et al. [25] is a subset of the ACE2004 coreference documents and contains 57 news articles comprising 253 surface forms.
2. **AIDA/CO-NLL-TestB:** The original AIDA data set [14] was derived from the CO-NLL 2003 shared task and contains 1.393 news articles. Cornolti et al. [6] subdivided this corpus into a training and two test corpora. The AIDA/CO-NLL-TestB data set has 231 documents with 19.40 entities on average. We note that this data set comprises several table-similar documents which mostly lack natural-language text.
3. **AQUAINT:** Compiled by Milne and Witten [22], the data set contains 50 documents and 727 surface forms from a news corpus from the Xinhua News Service, the New York Times, and the Associated Press.
4. **DBpedia Spotlight:** The DBpedia Spotlight corpus was released with the Spotlight system [19] and served as it's benchmark data set. It also contains several non-named entities (e.g. home). With 5.69 entities per document the corpus provides enough entities for collective entity disambiguation.

5. **MSNBC:** The corpus was presented by Cucerzan et al. [7] in 2007 and contains 20 news documents with 32.90 entities per document. The data set contains a wide range of entities which are linked via direct relations in DBpedia. Thus, many (simple) approaches achieve decent results.
6. **N3-Reuters128:** This corpus is based on the Reuters-21578 corpus which contains economic news articles. Roeder et al. proposed this corpus in [27] which contains 128 short documents with 4.85 entities on average.
7. **IITB:** This manually created data set by Kulkarni et al. [17] with 103 documents displays the highest entity/document-density of all data sets ($\approx$ 109 entities/document on average). Additionally, the documents are similar to Wikipedia pages and contain sufficient textual context.
8. **Microposts-2014 Test:** The tweet data set [2] was introduced for the "Making Sense of Microposts" challenge and has very few entities per document on average [29]. As usual in tweet data, the amount of textual context is very limited.
9. **N3 RSS-500:** This corpus has been published by Gerber et al. [9] and is one of the N3 data sets [27]. It consists of data collected from 1,457 RSS feeds. The list includes all major worldwide newspapers and a wide range of topics, e.g. World, U.S., Business etc. A subset of this corpus has been created by randomly selecting 1% of the contained sentences. Finally, domain experts annotated 500 sentences manually.

All ground truth annotations in these data sets either refer to entities in DBpedia or Wikipedia. Both KBs contain the same entities whose URLs can be easily converted to the other KB URL. Thus, we simply always return DBpedia URLs to GERBIL, which automatically processes the URLs according to the underlying data sets. We note that the GERBIL version that we use does not consider *NIL* annotations when computing the F1, recall and precision values. If our service returns a *NIL* annotation, GERBIL treats it like "not annotated". Thus, for our abstaining experiment we manually evaluate the accuracy on the IITB data set, which also considers *NIL* annotations.

7. EVALUATION

In the following, we compare the results of our approach with those attained by other disambiguation frameworks. To this end, we use GERBIL v1.1.4 and evaluate the approaches on the D2KB (i.e. link to a KB) task. In Section 7.1 we directly compare the approaches on the basis of its results achieved with GERBIL. In Section 7.2 we discuss our results in contrast to other works that are not publicly available. Section 7.3 provides a parameter study of the semantic embeddings and in Section 7.4 we evaluate the abstaining mechanism of our approach.

7.1 Results

In the following, we directly compare our approach to publicly available, state-of-the-art entity disambiguation systems, which disambiguate Wikipedia, DBpedia or YAGO entities, via GERBIL. These are the currently available versions of DBpedia Spotlight [19], AIDA [14] (new disambiguation index), Babelfy [23], WAT [24] and Wikifier [25, 5]. To the best of our knowledge, Wikifier is the current state-of-the-art entity disambiguation system which is publicly available. Wikifier and WAT use Wikipedia as underlying KB and link surface forms directly to Wikipedia pages. Babelfy also returns Wikipedia entities but uses BabelNet as KB, which was automatically created by linking Wikipedia to WordNet. In contrast, DBpedia Spotlight and AIDA rely on the RDF-KBs DBpedia and YAGO2, while additionally making use of Wikipedia knowledge.

Table 2: Micro-averaged F1, π and ρ of DoSeR on 9 data sets.

Data Set	F1	π	ρ
ACE2004	0.907	0.912	0.901
AIDA/CONLL-TestB	0.784	0.784	0.784
AQUAINT	0.842	0.85	0.838
DBpedia Spotlight	0.810	0.814	0.806
MSNBC	0.911	0.913	0.908
N3-Reuters128	0.850	0.856	0.844
IITB	0.741	0.744	0.738
Microposts-2014 Test	0.750	0.783	0.719
N3 RSS-500	0.751	0.752	0.750

Since entities within these three KBs (DBpedia, Wikipedia and YAGO2) provide sameAs relations, GERBIL maps the systems' output to the corresponding ground truth URLs. For all systems we choose the best configurations according to the authors. In addition to these frameworks, we define the strong baseline *PriorProb* which links surface forms to the entities with the highest prior probability (cf. Section 5.1). We also present the results when excluding entity-context embeddings (*DoSeR -Doc2Vec*). We investigate how well the approach performs with entity-embeddings as semantic relatedness feature only. In this case, we use the entity embeddings directly to compute the ETP (cf. Section 5.2). Table 2 lists the results on nine data sets in terms F1, precision and recall, aggregated across surface forms (micro-averaged). Table 3 shows the F1 values in comparison to the competitor systems on all data sets. The corresponding GERBIL result sheet is available on the GERBIL website[4] and can be used to make comparisons to our approach in future evaluations.

Overall, our approach attains the best averaged F1 value of all systems. Thereby, it outperforms Wikifier by 5 F1 percentage points on average. Additionally, we significantly outperform the other systems as well as the PriorProb baseline by up to 25 F1 percentage points on average. On the data sets ACE2004, MSNBC, Microposts2014-Test and N3-Reuters128 our approach performs exceptionally well (up to 12 F1 percentage points in advance). We note that our PriorProb baseline outperforms Wikifier on the Microposts-2014-Test data set because of using a newer version of DBpedia/Wikipedia. The Micropost2014-Test data set was released in 2014 and obviously queries some very new (or changed) entities. On the DBpedia Spotlight and N3 RSS-500 data sets our approach also performs best with F1 values of ≈ 0.81 (DBpedia Spotlight) and ≈ 0.75 (N3 RSS-500) respectively. Considering the AIDA/CONLL-TestB data set, our approach performs slightly better than Wikifier but performs comparatively poor with a F1 value of ≈ 0.78 compared to ≈ 0.84 by the WAT system. The reasons for this are two-fold: First, the underlying data set is still annotated with entities whose identifiers have been changed over the years with updates. Thus our service returns wrong entity URLs according to the ground truth. The same problem occurs in the AIDA system when using the newer AIDA entity index. In this case, the F1 value drops from 0.82 to 0.77. In an experiment where we disambiguate the original AIDA entities, our system achieves a F1 value of ≈ 0.84. Second, a more detailed analysis of the surface forms' textual context is necessary to perform even better. Nevertheless, our algorithm outperforms the other systems and also AIDA which was optimized on this data set. Regarding AQUAINT and IITB, Wikifier leads DoSeR by 2 percentage points F1 on both data sets.

[4] http://dx.doi.org/10.5281/zenodo.51250

Table 3: Comparing micro-averaged F1 values of DoSeR, DoSeR without Doc2Vec, the prior probability baseline as well as the publicly available entity disambiguation systems Wikifier, Spotlight, AIDA, Babelfy and WAT on nine data sets.

Data Set	**DoSeR**	**DoSeR (-Doc2Vec)**	**PriorProb**	**Wikifier**	**Spotlight**	**AIDA**	**Babelfy**	**WAT**
ACE2004	**0.907**	0.872	0.831	0.834	0.713	0.815	0.561	0.800
AIDA/CONLL-TestB	0.784	0.754	0.661	0.777	0.593	0.774	0.592	**0.843**
AQUAINT	0.842	0.842	0.803	**0.862**	0.713	0.532	0.652	0.768
DBpedia Spotlight	**0.810**	0.775	0.745	0.797	0.789	0.508	0.522	0.652
MSNBC	**0.911**	0.876	0.711	0.851	0.511	0.782	0.607	0.777
N3-Reuters128	**0.850**	0.810	0.700	0.703	0.577	0.596	0.534	0.644
IITB	0.741	0.738	0.711	**0.766**	0.447	0.270	0.470	0.611
Microposts-2014 Test	**0.750**	0.704	0.630	0.586	0.453	0.453	0.473	0.595
N3 RSS-500	**0.751**	0.713	0.678	0.732	0.622	0.716	0.630	0.682
Average	**0.816**	0.787	0.718	0.768	0.602	0.605	0.560	0.708

In *summary*, we state that our approach significantly outperforms other publicly available disambiguation approaches. Overall, our approach disambiguates the entities highly accurate and attains state-of-the-art or nearly state-of-the-art results on all nine data sets. Hence, our approach is very well suited for all kinds of documents available in the web (e.g. tweets, news, etc.). We emphasize that despite the huge amount of training data used to improve disambiguation robustness, our approach attains nearly identical results on all data sets with entity embeddings trained on Wikipedia only. Our results proposed in [32] show that a simple graph algorithm using entity embeddings based on Wikipedia already performs exceptionally well. In terms of performance, our approach annotates roughly as fast as the Wikifier and WAT annotation system but is slower compared to Spotlight and AIDA. The Babelfy system is the slowest and takes too much time, especially on the IITB data set. Our system has the advantage to accept multiple queries in parallel, but is not yet optimized for high-performance disambiguation.

7.2 Discussion

Comparing our results to those of other state-of-the-art approaches that are not publicly available is not an easy task. Reimplementing the respective algorithms is not an absolutely fair method to compare the approaches with our KB: Usually crucial implementation details remain unknown in the original publications, since the focus mostly lies on the algorithm instead of the implementation.

Anyhow, we use the work of Guo et al. [10] as an entry point in the following. Their approach was exclusively evaluated and optimized on the ACE2004, MSNBC and AQUAINT data sets on which the authors achieve state-of-the-art results. A direct comparison of our results and the results of [10] shows that both works perform equally well on the MSNBC data set. Furthermore, our approach performs better on the ACE2004 data set (0.906 vs. 0.877 F1) but loses on the AQUAINT data set (0.842 vs. 0.907 F1). The problem with a pure number-based comparison, however, lies in the uncertainty in the underlying KB used in the experiments. If the underlying KB has a lower number of entities, the average likelihood of a wrong disambiguation is also reduced. In order to compare our algorithm with the approach in [10], we introduce the concept of the Surface Form Ambiguity Degree (SFAD). The SFAD is based on two assumptions: First, both approaches are able to disambiguate all entities in the ground truth data set, i.e. the KB covers the entities in the data sets and contains similar entity occurrences resulting from a given corpus (important for prior computation). Second, the candidate entities retrieved from the KB contain the correct entity, i.e. the error introduced by candidate selection is zero. Under these assumptions, a varying prior probability of an entity defines the degree of ambiguity, the SFAD, for that surface form. So the SFAD describes how many entities are potentially relevant for a specific surface form. Since our approach has a (significantly) lower prior probability on these data sets, the SFAD is higher, respectively. In Table 4 we compare the differences of the best result and the result achieved with the prior alone for our approach and the Guo et al. approach. Overall, our disambiguation index contains more entities that are relevant for a surface form on average and hints that our core-algorithm (without KB and candidate selection) is more robust than the approach from Guo et al. [10]. Another evidence is that the authors reimplemented the approach used in Wikifier and achieved significantly better results on their KB as we achieved with GERBIL with Wikifier's original KB. Guo et al. also report the results of former, well-known state-of-the-art approaches (e.g. Cucerzan [7], Han et al. [12], Glow [25]), but we do not discuss the results in detail because these approaches perform worse than Wikifier and the approach of Guo et al.

Considering the IITB data set, the system by Han et al. [11] performs best with a micro F1 value of 0.80. The authors did not evaluate their system on other data sets. However, their topic model approach is fully-trained on Wikipedia and takes all words into account. Since, the IITB data set consists of long documents very similar to those in Wikipedia, the system performs best on it.

In 2014, the Micropost2014-Test data set was created in the context of the workshop challenge *Making Sense of Microposts* [2]. The best system in the workshop was proposed by Microsoft which attains a micro F1 value of 0.70. To the best of our knowledge, this has been the best disambiguation approach on this data set so far, but is outperformed by our approach by ≈ 5 percentage points.

Considering the CONLL-TestB data set, the current state-of-the-art approach has been presented by Huang et al. [15] and attains a micro F1 value of 0.866. Similar to our approach, the authors learn semantic embeddings with a deep neuronal network approach from DBpedia and Wikipedia (but not with Word2Vec and Doc2Vec). Again, the approach was only evaluated on the CONLL-TestB data set as well as on a tweet data set. Experiments show that we can also further improve our results on this data set to a micro F1 value of 0.850 by (i) reducing entity candidates (lower SFAD), (ii) training the semantic embeddings on DBpedia instead of Wikipedia and (iii) using an older entity index. However, since our main goal was to create a **robust** disambiguation approach which performs well on several data sets with varying underlying document properties, we do not optimize the DoSeR algorithm on a single data set.

Table 4: Differences of the best result and the prior DoSeR and Guo et al. [10] on 3 data sets.

Approach	ACE2004	MSNBC	AQUAINT
DoSeR F1(best) - F1(prior)	0.076	0.200	0.039
Guo et al. F1(best) - F1(prior)	0.022	0.049	0.035
Δ in F1 percentage points	**5.4**	**15.1**	**0.4**

In summary, we state that a pure number-based result comparison is not always easy since multiple factors play an important role (e.g. disambiguation index, data set queries). However, the results give the hint that our approach achieves comparable accuracy, while being robust on other data sets at the same time.

7.3 Semantic Embeddings Parameter Study

The accuracy of our approach depends on a number of parameters, foremost the parameters of the semantic embeddings. In order to analyze this sensitivity, we conducted experiments in which we vary the dimension number of our semantic embeddings and report the results for both Word2Vec and Doc2Vec architectures (CBOW vs. Skip-gram and PV-DM vs. PV-DBOW).

Figure 2 (top) depicts the average micro F1 value (across all data sets) of our approach when using either the CBOW or skip-gram Word2-Vec architecture and a specific amount of dimensions. During this experiment the corresponding Doc2Vec architecture is set to PV-DM since it is better suited as we will see in the following. Basically, in our experiments, the skip-gram architecture consistently creates better entity embeddings then CBOW. This might be due to skip-gram performs better with infrequent words (entities) in the training corpus [20]. However, the difference between both architectures is $\approx 1-2$ percentage points F1. On $d = 400$ the result margin between both architectures is maximized and the average F1-value reaches its peak. It is interesting to see that even more dimensions slightly decrease the result values. We assume that this leads to some kind of overfitting and, thus, the optimal number of dimensions for entity embeddings probably depends on the number of entities and amount of training data.

We conduct the same experiment for our entity-document embeddings (Doc2Vec). In this particular case, we use the skip-gram architecture as baseline training algorithm for the entity embeddings (Word2Vec). Figure 2 (bottom) depicts the corresponding average micro F1 values for various dimensions and both Doc2Vec architectures. The PV-DM architecture for Doc2Vec performs better if the number of dimensions is higher than $d = 400$. We assume that the context consideration in the PV-DM architecture leads to an advance. However, the best average F1 value is achieved with $d = 1000$, whereby the difference between PV-DBOW and PV-DM is maximal ≈ 2 percentage points F1.

In *summary*, we state that the skip-gram architecture for Word2-Vec and the PV-DM architecture for Doc2Vec perform best. It is interesting to see that the number of optimal dimensions for entity embeddings must be geared to the underlying corpora.

7.4 Abstaining

Abstaining is an important task in disambiguation algorithms when it comes to disambiguating surface forms whose referent entity is not in the entity set Ω. It is also used if there is uncertainty about the correct entity due to insufficient context information.

In this experiment, our algorithm returns the pseudo-entity *NIL* in the following situations: (i) if no entity candidates can be found during the candidate generation step (cf. Section 5.1), and (ii) if the algorithm is uncertain about the correct entity after the last PR iteration (cf. Algorithm 1). For experimental purpose, we downloaded the original IITB data set (information on our GitHub page), which additionally contains 7652 *NIL* annotations in addition to the default annotations (18897 annotations overall), and report the disambiguation *accuracy*. We also rerun the GERBIL experiments with abstaining to investigate to what extent the results decrease on data sets which do not provide *NIL* ground truth annotations.

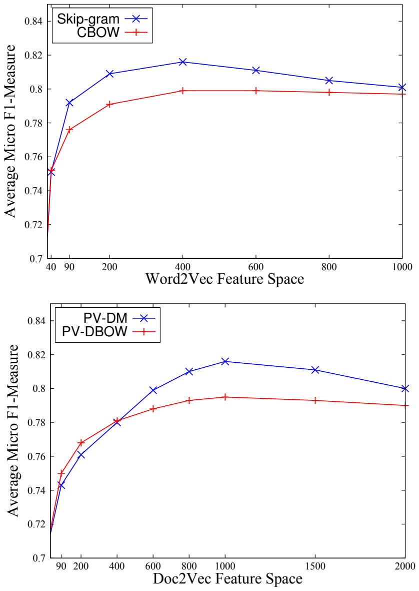

Figure 2: Comparison of Word2Vec and Doc2vec architectures with various feature-space dimensions.

Conducting the experiment on the manually downloaded IITB data set results in a disambiguation accuracy of 0.757 (micro-averaged). With returning 6120 *NIL* annotations overall, our algorithm does not find candidates for surface forms in 3823 cases ($\approx 62.5\%$) and abstains 2297 surface forms ($\approx 37.5\%$). When we tune our abstaining parameter to abstain more aggressive, our overall accuracy slightly decreases. Unfortunately, the authors of the topic-model, state-of-the-art approach [11] on this data set do not provide abstaining results for comparison in their work.

However, Table 5 reports the micro F1 values of our algorithm with abstaining on all data sets in the GERBIL evaluation. As a result of GERBIL not querying surface forms with *NIL* annotations in the ground truth, our results (slightly) decrease. Nevertheless, the amount of abstained surface forms is very limited and, thus, our approach still outperforms Wikifier on 6 out of 9 data sets. On the Microposts2014-Test data set the F1 decrease is the highest with 7 percentage points F1. Obviously, our algorithm is sometimes uncertain about the correct entity and abstains, which is due a small amount of surface forms per document. In other words, our algorithm lacks sufficient evidence about the correct entity and, hence, abstains due to exceeding the abstaining threshold. In *summary*, we state that our algorithm is able to successfully abstain entity annotations if evidence of the correct entity is missing. Our abstaining mechanism performs well even if data sets do not provide *NIL* annotations (as simulated by GEBRIL).

8. CONCLUSION AND FUTURE WORK

In this work, we present a new collective, graph-based entity disambiguation approach that utilizes semantic entity and entity-context embeddings for robust entity disambiguation. Robust there-

Table 5: Micro F1 values of our approach with abstaining on data sets without *NIL* annotations.

Data Set	F1	Change in F1 percentage points
ACE2004	0.892	-1.65
AIDA/CONLL-TestB	0.782	-0.26
AQUAINT	0.820	-2.61
DBpedia Spotlight	0.773	-4.57
MSNBC	0.906	-0.55
N3-Reuters128	0.809	-4.82
IITB	0.722	-2.56
Microposts-2014 Test	0.607	-7.07
N3 RSS-500	0.738	-1.73

by refers to the property of achieving (better than) state-of-the-art results over a wide range of very different data sets. Our approach is also able to abstain if no appropriate entity can be found for a specific surface form. We evaluate our approach against 5 strong, publicly available entity disambiguation systems on 9 data sets and show that our approach outperforms all other system by a significant margin on nearly all data sets. We also discuss the influence of the quality of the underlying knowledge base on the disambiguation accuracy and compare our results to those of other non-publicly available state-of-the-art algorithms. We also provide our approach as well as the underlying knowledge base as open source solution.

Future work includes the evaluation of non-collective disambiguation. This is important if queries provide only a single surface form to disambiguate. We also want to optimize the disambiguation performance and provide in-depth performance tests.

9. ACKNOWLEDGMENTS

The presented work was developed within the EEXCESS project funded by the European Union Seventh Framework Programme FP7/2007-2013 under grant agreement number 600601. Additionally, it was partially funded by the IRYXYS Research Center.

10. REFERENCES

[1] A. Alhelbawy and R. J. Gaizauskas. Collective named entity disambiguation using graph ranking and clique partitioning approaches. In *Proc. of the 25th Int. Conf. on Computational Linguistics, Technical Papers*, pages 1544–1555, 2014.

[2] A. E. C. Basave, G. Rizzo, A. Varga, M. Rowe, M. Stankovic, and A.-S. Dadzie. Making sense of microposts named entity linking challenge. volume 1141 of *CEUR*, pages 54–60, 2014.

[3] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. In *7th WWW*, pages 107–117, Amsterdam, The Netherlands, 1998. Elsevier Science Publishers B. V.

[4] R. Bunescu and M. Pasca. Using encyclopedic knowledge for named entity disambiguation. In *Proc. of the 11th Conf. of the European Association for Computational Linguistics (EACL-06)*, pages 9–16, Trento, Italy, 2006.

[5] X. Cheng and D. Roth. Relational inference for wikification. In *Proc. of the Conf. on Empirical Methods in NLP*, EMNLP '13, 2013.

[6] M. Cornolti, P. Ferragina, and M. Ciaramita. A framework for benchmarking entity-annotation systems. In *Proc. of the 22nd Int. Conf. on World Wide Web*, WWW '13, pages 249–260, Republic and Canton of Geneva, Switzerland, 2013. International World Wide Web Conferences Steering Committee.

[7] S. Cucerzan. Large-scale named entity disambiguation based on wikipedia data. In *EMNLP-CoNLL*, volume 7, pages 708–716, 2007.

[8] P. Ferragina and U. Scaiella. Fast and accurate annotation of short texts with wikipedia pages. *IEEE Softw.*, 29(1):70–75, Jan. 2012.

[9] D. Gerber, A.-C. Ngonga Ngomo, S. Hellmann, T. Soru, L. Bühmann, and R. Usbeck. Real-time RDF extraction from unstructured data streams. In *Proceedings of ISWC*, 2013.

[10] Z. Guo and D. Barbosa. Robust entity linking via random walks. In *Proc. of the 23rd ACM Int. Conf. on Information and Knowledge Management*, CIKM '14, pages 499–508, New York, NY, USA, 2014. ACM.

[11] X. Han and L. Sun. An entity-topic model for entity linking. In *Proc. of the 2012 Joint Conf. on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pages 105–115, Stroudsburg, PA, USA, 2012. ACL.

[12] X. Han, L. Sun, and J. Zhao. Collective entity linking in web text: A graph-based method. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '11, pages 765–774, New York, NY, USA, 2011. ACM.

[13] Z. He, S. Liu, M. Li, M. Zhou, L. Zhang, and H. Wang. Learning entity representation for entity disambiguation. In *ACL*, 2013.

[14] J. Hoffart, M. A. Yosef, I. Bordino, H. Fürstenau, M. Pinkal, M. Spaniol, B. Taneva, S. Thater, and G. Weikum. Robust disambiguation of named entities in text. In *Proc. of the Conf. on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 782–792, Stroudsburg, PA, USA, 2011. ACL.

[15] H. Huang, L. Heck, and H. Ji. Leveraging deep neural networks and knowledge graphs for entity disambiguation. *CoRR*, abs/1504.07678, 2015.

[16] S. S. Kataria, K. S. Kumar, R. R. Rastogi, P. Sen, and S. H. Sengamedu. Entity disambiguation with hierarchical topic models. In *Proc. of the 17th ACM SIGKDD international Conf. on Knowledge Discovery and Data Mining*, KDD '11, pages 1037–1045, New York, NY, USA, 2011. ACM.

[17] S. Kulkarni, A. Singh, G. Ramakrishnan, and S. Chakrabarti. Collective annotation of wikipedia entities in web text. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '09, pages 457–466, New York, NY, USA, 2009. ACM.

[18] Q. V. Le and T. Mikolov. Distributed representations of sentences and documents. In *Proc. of the 31th Int. Conf. on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, pages 1188–1196, 2014.

[19] P. N. Mendes, M. Jakob, A. García-Silva, and C. Bizer. Dbpedia spotlight: Shedding light on the web of documents. In *Proceedings of the 7th International Conference on Semantic Systems*, I-Semantics '11, pages 1–8, New York, NY, USA, 2011. ACM.

[20] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013.

[21] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.*, pages 3111–3119, 2013.

[22] D. Milne and I. H. Witten. Learning to link with wikipedia. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, CIKM '08, pages 509–518, New York, NY, USA, 2008. ACM.

[23] A. Moro, A. Raganato, and R. Navigli. Entity linking meets word sense disambiguation: a unified approach. *TACL*, 2:231–244, 2014.

[24] F. Piccinno and P. Ferragina. From tagme to wat: A new entity annotator. In *First Int. Workshop on Entity Recognition/Disambiguation*, ERD '14, pages 55–62, NY, USA, 2014. ACM.

[25] L. Ratinov, D. Roth, D. Downey, and M. Anderson. 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*, HLT '11, pages 1375–1384, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics.

[26] R. Řehůřek and P. Sojka. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, May 2010. ELRA.

[27] M. Röder, R. Usbeck, S. Hellmann, D. Gerber, and A. Both. N3 - a collection of datasets for named entity recognition and disambiguation in the nlp interchange format. In *The 9th edition of the Language Resources and Evaluation Conference, 26-31 May, Reykjavik, Iceland*, 2014.

[28] R. Usbeck, A.-C. Ngonga Ngomo, M. Röder, D. Gerber, S. A. Coelho, S. Auer, and A. Both. Agdistis - graph-based disambiguation of named entities using linked data. In *14th ISWC*, pages 457–471, New York, NY, USA, 2014. Springer-Verlag New York, Inc.

[29] R. Usbeck, M. Röder, A.-C. Ngonga Ngomo, C. Baron, A. Both, M. Brümmer, D. Ceccarelli, M. Cornolti, D. Cherix, B. Eickmann, P. Ferragina, C. Lemke, A. Moro, R. Navigli, F. Piccinno, G. Rizzo, H. Sack, R. Speck, R. Troncy, J. Waitelonis, and L. Wesemann. Gerbil: General entity annotator benchmarking framework. In *Proc. of the 24th Int. Conf. on World Wide Web*, WWW '15, pages 1133–1143, Republic and Canton of Geneva, Switzerland, 2015. International World Wide Web Conferences Steering Committee.

[30] S. White and P. Smyth. Algorithms for estimating relative importance in networks. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '03, pages 266–275, New York, NY, USA, 2003. ACM.

[31] W. Xie, D. Bindel, A. Demers, and J. Gehrke. Edge-weighted personalized pagerank: Breaking a decade-old performance barrier. In *21th SIGKDD*, pages 1325–1334, NY, USA, 2015. ACM.

[32] S. Zwicklbauer, C. Seifert, and M. Granitzer. Doser - a knowledge-base agnostic framework for entity disambiguation using semantic embeddings. In *The Semantic Web. Latest Advances and New Domains*, ESWC '16. Springer, 2016.