Combining Textual and Graph-based Features for Named Entity Disambiguation Using Undirected Probabilistic Graphical Models

Sherzod Hakimov, Hendrik ter Horst, Soufian Jebbara, Matthias Hartung, Philipp Cimiano
{shakimov, hterhors, sjebbara, mhartung, cimiano}@cit-ec.uni-bielefeld.de

Semantic Computing Group
Cognitive Interaction Technology – Center of Excellence (CITEC)
Bielefeld University
33615 Bielefeld, Germany

Abstract. Named Entity Disambiguation (NED) is the task of disambiguating named entities in a natural language text by linking them to their corresponding entities in a knowledge base such as DBpedia, which are already recognized. It is an important step in transforming unstructured text into structured knowledge. Previous work on this task has proven a strong impact of graph-based methods such as PageRank on entity disambiguation. Other approaches rely on distributional similarity between an article and the textual description of a candidate entity. However, the combined impact of these different feature groups has not been explored to a sufficient extent. In this paper, we present a novel approach that exploits an undirected probabilistic model to combine different types of features for named entity disambiguation. Capitalizing on Markov Chain Monte Carlo sampling, our model is capable of exploiting complementary strengths between both graph-based and textual features. We analyze the impact of these features and their combination on named entity disambiguation. In an evaluation on the GERBIL benchmark, our model compares favourably to the current state-of-the-art in 8 out of 14 data sets.

Keywords: entity disambiguation, collective entity disambiguation, named entity disambiguation, probabilistic graphical models, factor graphs

1 Introduction

The problem of resolving the real-world reference of entity mentions in textual data, which are already recognized, by linking them to unique identifiers in a knowledge base has received substantial attention in recent years. This *entity disambigation* task is an important first step towards capturing the semantics of textual content.

Earlier approaches to entity disambiguation resolved mentions independently of each other (e.g., DBpedia Spotlight [7], etc.). Recently, several approaches

have been presented that perform *collective entity disambiguation*, attempting to resolve several mentions at the same time within one inference step. Such joint inference approaches can capture dependencies in the choice of identifiers for different mentions.

The features used in entity disambiguation models vary widely. Many approaches rely on features that measure textual coherence. This is typically implemented by a measure of similarity between the context in which a mention appears and the context of the linking candidate. These contexts are of a textual nature and Bag-of-Words (BOW) based similarity as measured by cosine similarity, for instance, can be applied here. A prominent representative of systems using textual coherence is DBpedia Spotlight. Other approaches rely on graph connectivity features exploiting the connectedness between different disambiguation candidates in a knowledge base. Examples of these are the Babelfy [21] and AGDISTIS [26] systems. Finally, recent work has shown the power of using prior probabilities as features on the task. For instance, Tristram et al. [25] have shown that using the PageRank of linking candidates alone can yield quite high results.

Building on these previous results, in this paper we present a novel system that performs collective entity disambiguation by combining all the abovementioned types of features within one model that is trained discriminatively. In particular, we propose an undirected probabilistic graphical model based on factor graphs. Each factor in the model measures the suitability of the resolution of some mention to a given linking candidate, relying on a set of features that are linearly combined by weights. For inference during training and testing, we rely on a Markov Chain Monte Carlo (MCMC) [2] approach. For training, we rely on the SampleRank [29] algorithm.

We evaluate our approach on standard benchmarking data sets for the entity disambiguation task as available in the GERBIL framework [27]. We show the impact of the features we propose in isolation and in combination. We thus enhance our understanding of the features that work well on the task. Overall, we show that our system outperforms state-of-the-art systems on 8 out of 14 publicly available datasets.

All the data and code used to build our approach are publicly available.[1]

2 Related Work

Given the variety of previous approaches to named entity disambiguation, we structure our discussion of related work according to the features and combinations of features that have been proposed.

One of the first named entity disambiguation systems, DBpedia Spotlight [7], mainly relied on features scoring the textual coherence between the context of the mention and the context of a given linking candidate. Recent approaches include different types of similarities. One example is the approach by Liu et

[1] https://github.com/ag-sc/NED

al. [19], which considers entity-entity similarity, mention-entity similarity, and mention-mention similarity. The prior probability of a mention is shown to be a strong indicator. A related system is the one of Hoffart et al. [13], which also combines a popularity prior, mention-entity similarity as well as a score of the graph-based coherence between linking candidates. All these features are combined in a linear model. Our approach is related, but extends the feature set used by the above-mentioned approaches, studying in particular the impact of each feature in isolation and in combination.

The connectedness between different linking candidates can also be estimated by the Topic-sensitive PageRank [12] of a linking candidate given another competing candidate or via a random walk over the KB graph, as in Guo & Barbosa [10]. Other approaches relying mainly on graph connectedness include Babelfy [21], TagMe [23] as well as the approaches by Hakimov et al. [11], Alhelbawy & Gaizauskas [1], Usbeck et al. [26], and Jin et al. [15].

By combining different sources of information comprising knowledge about entities, names, context, and the Wikipedia graph in a probabilistic framework, Barrena et al. [3] observe complementary effects between these features. However, they impose strong independence assumptions (i) on the level of features, which essentially renders their model an instance of Naïve Bayes classification, and (ii) on the level of entities as well.

In contrast, we aim at *collective entity disambiguation* in this paper, and frame the task as an inference problem in a probabilistic graphical model to disambiguate all mentions in a text through joint prediction. Previous work on joint entity disambiguation comprises the graph-based approach by Alhelbawy and Gaizauskas [1], for instance: All candidates pertaining to the NEs in the text are represented as nodes in a so-called solution graph that serves as input to a ranking model based on PageRank. As features for the ranking, both an initial confidence (corresponding to prior popularity or mention-entity similarity, respectively) and edge weights in the graph (corresponding to entity-entity coherence) are taken into account. Houlsby and Ciaramita [14] apply a generative probabilistic model, viz. Latent Dirichlet Allocation (LDA; [4]), to the task. They construct a "knowledge base-specific" topic model where each topic corresponds to a Wikipedia article. The word-topic proportions inferred by LDA for each entity mention are directly used in order to link the mention to its most likely Wikipedia concept.

More recently, Ganea et al. [9] and Zwicklbauer et al. [30] have proposed collective entity disambiguation methods as well. Ganea et al. [9] have proposed a joint probabilistic model for collective entity disambiguation that is not trained on any particular data set, but relies on sufficient statistics over all hyperlinks in Wikipedia, considering each anchor text as a mention and the Wikipedia page it refers to as the ground truth entity label. These statistics essentially capture co-occurrence probabilities of mention-entity and entity-entity pairs. Zwicklbauer et al. [30] proposed a method using semantic embeddings of entities for entity disambiguation. They embed entities using Word2Vec [20] by constructing sequences of entities using random walks over the RDF Graph.

Undirected probabilistic graphical models have been successfully applied to a variety of related NLP tasks: Passos et al. [22] propose a method for learning neural phrase embeddings to be applied to Named Entity Recognition by leveraging factor graphs. Singh et al. [24] use factor graphs for cross-document coreference resolution. Our approach differs in that we apply factor graphs for NED while using features specific to the task. We give an overview of how we formulate the NED task with factor graphs in Section 3.2.

3 Named Entity Disambiguation with Undirected Factor Graphs

In this work, we present an approach based on imperatively defined factor graphs that addresses Named Entity Disambiguation (NED) with textual and graph-based features. By employing factor graphs, our system is able to disambiguate entity mentions in a document separately and collectively, benefitting from both paradigms. Before we give a formal description of our factor graph approach, we present our candidate retrieval component for retrieving URI candidates for a given entity mention.

3.1 Candidate Retrieval

To reduce the number of possible candidate URIs for a given mention, we implement a retrieval component based on an index that retrieves a subset of k related candidates for this mention. The retrieval component is designed as to provide a high recall, while keeping k as small as possible. Our index is constructed using two different data sources of mention-related surface forms, in particular DBpedia and Wikipedia anchors. In the following, we briefly describe both data sets as well as the generation of our index.

DBpedia data We create an index of surface forms of named entities using DBpedia data sets in their 2015-04 version.[2] We collected a set of labeleling properties from these data sets to detect surface forms. All ⟨surface form, URI⟩ pairs are extracted from these data sets while keeping track of the frequency of occurrence of each pair. In addition to label properties, we convert all redirect page URIs into surface forms and pair them with the target page URI. The data set names, label properties, surface form data and all other data sets can be found on our page.

Wikipedia anchors We extracted all links in Wikipedia pages and extracted the text mentioned in the anchor and the target link. The text of an anchor refers to the surface form, and the actual link refers to some Wikipedia page (URI). By counting the co-occurrences of ⟨surface form, URI⟩ pairs, we built another table of ⟨surface form, URI, frequency⟩ tuples.

[2] http://wiki.dbpedia.org/Downloads2015-04

Candidate Retrieval Performance In order to assess the candidate retrieval performance of our index, we compute the Recall@k, measuring in how many cases we retrieve the correct candidate among the top k results from our index. The results of Recall@k are plotted in Figure 1 for different values of k. The evaluation is based on the AIDA/CoNLL [13] and MicroPost2014 [6] training sets. We consider three settings: (i) using only the DBpedia table, (ii) using the Wikipedia anchors table, and (iii) We combine both data tables where the frequency of the same ⟨surface form, URI⟩ pairs are summed and the frequency of unique pairs are kept as they are in respective tables. Our results show that the combination approach yields the highest recall. The results also show that considering a number of $k = 10$ represents a reasonable trade-off between recall and efficiency; thus, we rely on this setting in all our experiments. The Recall@10 is 0.934 for the AIDA/CoNLL and 0.814 for the MicroPost2014 training sets, respectively. These figures represent an upper bound in terms of F-Measure for the overall task of entity disambiguation.

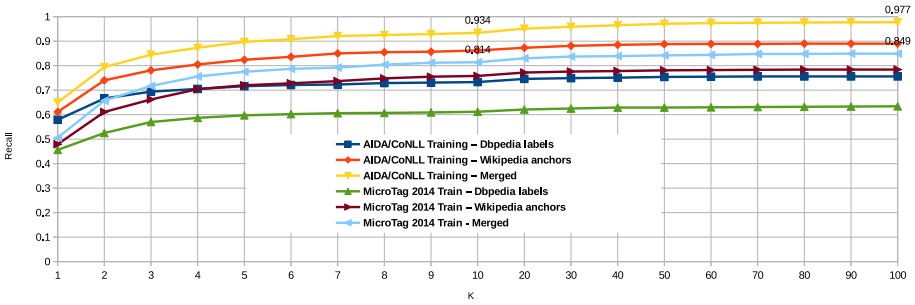

Fig. 1. Recall@k scores for candidate retrieval

3.2 Imperatively Defined Factor Graphs

In this section, we introduce the concept of factor graphs [17], following the notations in [29] and [16]. A factor graph $\mathcal{G}$ is a bipartite graph that defines a probability distribution π. The graph consists of variables V and factors Ψ. Variables can further be divided into sets of *observed* variables X and *hidden* variables Y. A factor Ψ_i connects subsets of observed variables x_i and hidden variables y_i and computes a scalar score based on the exponential of the scalar product of a feature vector $f_i(x_i, y_i)$ and a set of parameters θ_i: $\Psi_i = e^{f_i(x_i,y_i)\cdot\theta_i}$. The probability of the hidden variables given the observed variables is the product of the individual factors:

$$\pi(y|x;\theta) = \frac{1}{Z(x)} \prod_{\Psi_i \in \mathcal{G}} e^{\Psi_i} = \frac{1}{Z(x)} \prod_{\Psi_i \in \mathcal{G}} e^{f_i(x_i,y_i)\cdot\theta_i} \tag{1}$$

where $Z(x)$ is the normalization function. For a given set of observed variables, we generate a factor graph automatically making use of factor templates $\mathcal{T}$. A template $T_j \in \mathcal{T}$ defines the subsets of observed and hidden variables (x, y) with $x \in X_j$ and $y \in Y_j$ for which it can generate factors and a function $f_j(x, y)$ to generate features for these variables. Additionally, all factors generated by a given template T_j share the same parameters θ_j. With this definition, we can reformulate the conditional probability as follows:

$$\pi(y|x;\theta) = \frac{1}{Z(x)} \prod_{T_j \in \mathcal{T}} \prod_{(x,y) \in T_j} e^{f_j(x,y) \cdot \theta_j} \tag{2}$$

Thus, we define a probability distribution over possible configurations of observed and hidden variables, i.e., assigned URIs. This enables us to explore the joint space of observed and hidden variables in a probabilistic fashion.

Data Representation In the following, we show how to apply this approach of probabilistic factor graphs to the NED task. We define a document as $d = (\mathbf{w}, \mathbf{a})$ that consists of a sequence of words $\mathbf{w} = (w_1, \ldots, w_{N_w})$ and a set of annotation spans (or entity mentions) $\mathbf{a} = \{a_1, \ldots, a_{N_a}\}$. Documents, words and annotation spans constitute the observed variables X. The assigned URIs of a set of annotation spans $\mathbf{u} = \{u_1, \ldots, u_{N_a}\}$ are considered to be hidden variables Y, where u_i corresponds to annotation span a_i. A disambiguated document, i.e., the collective of words, annotation spans and assigned URIs $(\mathbf{w}, \mathbf{a}, \mathbf{u})$, is referred to as a *configuration* and in the context of sampling as a *state*. Consequently, we can apply Eq. (2) to a disambiguated document to compute its probability given the underlying factor graph. Figure 2 shows a schematic visualization of a disambiguated document along with its factor graph.

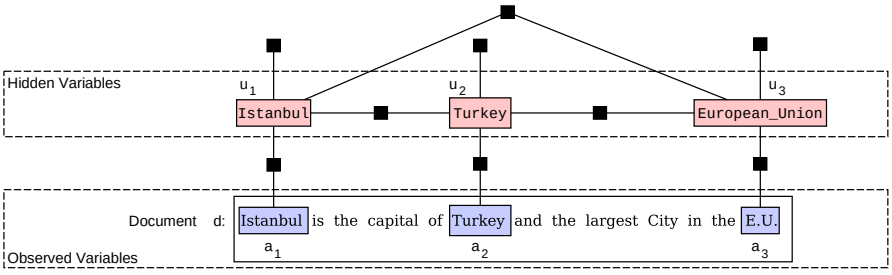

Fig. 2. An exemplary depiction of a factor graph for a disambiguated document with three NEs. The figure shows the division between observed and hidden variables as well as different factors (black boxes) between all variables.

3.3 Inference

This section shows how we infer URIs for a given document using the above formulation of factor graphs. We perform a Markov Chain Monte Carlo (MCMC) sampling procedure [2] that explores the search space of a document in an iterative fashion. The inference procedure performs a local search and can be divided into (i) generating possible successor states for a given state by applying atomic changes, and (ii) selecting a successor state from the set of generated states.

Assuming a document $d = (\mathbf{w}, \mathbf{a})$, our goal is to obtain a configuration (or state) $s^* = (\mathbf{w}, \mathbf{a}, \mathbf{u}^*)$ with the correct URI assignment $\mathbf{u}^*$ for the given annotation spans. For that, we perform an iterative sampling procedure of m steps that performs a local search at each step to find a better disambiguation for a given document.

As a first step, we create an initial state $s_0 = (\mathbf{w}, \mathbf{a}, \mathbf{u_0})$, where URIs $\mathbf{u_0}$ are randomly assigned from the top-k retrieved candidates. This state is used as the starting point of our sampling procedure.

For each annotation span a_i in the state, we retrieve a set of k candidate URIs $Cand(a_i) = \{c_{i1}, \ldots, c_{ij}, \ldots, c_{ik}\}$ from our candidate retrieval component, using the text of a_i as query. We generate $N_a \cdot k$ modified states that differ from the current state s_t in only a single atomic change. Specifically, the modified state $s'_{ij} = (\mathbf{w}, \mathbf{a}, \mathbf{u'_{ij}})$ comprises the same observed variables $\mathbf{w}$ and $\mathbf{a}$, but changes exactly one "hidden" assignment of a URI to $\mathbf{u'_{ij}} = \{u_1, \ldots, u_{i-1}, c_{ij}, u_{i+1}, \ldots, u_{N_a}\}$, while leaving all other URIs untouched. We consider this pool of generated states to be the collection of all valid states that can be reached from the current state with one atomic change.

Next, we compute the probability of each generated state s'_{ij} using Eq. (2) and obtain a probability distribution over all generated states.[3] We select a single candidate state s'_t by sampling from the distribution of generated states[4] to obtain a potential successor state. We accept the sampled successor state s'_t as our next state s_{t+1} if it has a higher probability than the previous state s_t:

$$s_{t+1} = \begin{cases} s'_t, & \text{if } \pi(s'_t) > \pi(s_t) \\ s_t, & \text{otherwise} \end{cases} \tag{3}$$

Following this procedure for m iterations yields a sequence of states $(s_0, \ldots, s_m)$ that are sampled from the distribution defined by the underlying factor graphs. The final state s_m in this sequence represents the predicted configuration s^*. With a reasonable choice of model parameters θ (see Section 3.4 below), it is expected that the URI assignments $\mathbf{u}^*$ in s^* constitute a good disambiguation of the entity mentions in the document. A more pseudo-algorithmic description of the inference procedure is given in Algorithm 1 and a schematic visualization of the generation of neighboring states is shown in Figure 3.

[3] After re-normalizing the probabilities such that $\sum_{s'_{ij}} \pi(s'_{ij}) = 1$.

[4] Our experiments show that a greedy approach that always prefers the state with the highest probability works best.

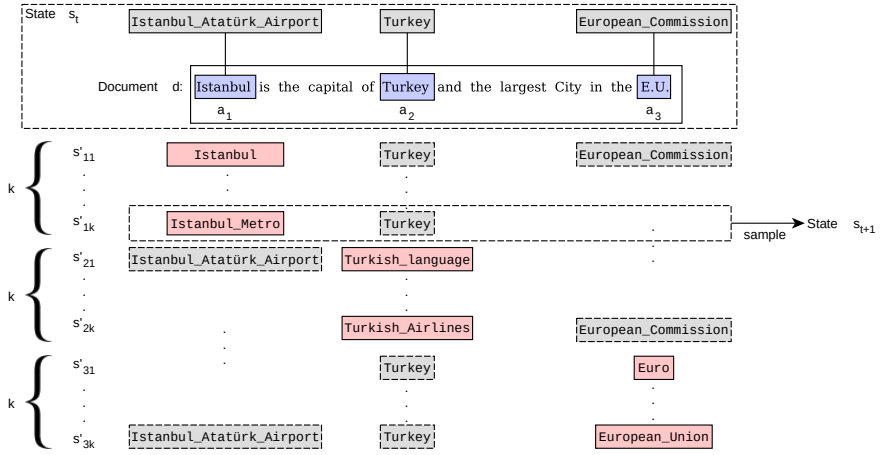

Fig. 3. An exemplary depiction of the sampling procedure. Starting from state s_t we generate states $\{s_{ij}\}$ in its local neighborhood performing only atomic modifications. Specifically, we generate a state for each annotation span and each retrieved candidate URI. Each state is scored according to the current model and the successor state s_{t+1} is selected from these generated states.

3.4 Learning Model Parameters

In order to optimize parameters θ, we use an implementation of the SampleRank [29] algorithm. The SampleRank algorithm obtains gradients for these parameters from pairs of states (e.g. s_t and s'_t) by observing the individual steps in the inference routine. For that, the algorithm requires a preference function $\mathbb{P}(s', s)$ that indicates which of two states is "objectively" preferred. We implement $\mathbb{P}$ based on an objective function $\mathbb{O}(s)$ that computes a score for a state compared to the ground truth assignments for the respective training document in terms of the ratio of correctly linked entity mentions $N_{a,correct}$ of a state s and the total number of entity mentions N_a in s, i.e., $\mathbb{O}(s) = N_{a,correct}/N_a$. The preference function is thus:

$$\mathbb{P}(s', s) = \begin{cases} 1, & \text{if } \mathbb{O}(s') > \mathbb{O}(s) \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

We modify the original SampleRank algorithm such that we select the best scoring successor state based on the objective function score $\mathbb{O}(.)$ rather than the probability given by the model in Eq. (2). This small modification ensures that the training procedure is guided towards a good solution when updating the model parameters.

The final training algorithm is similar to the inference procedure in Algorithm 1, with two changes, however: Line 5 of the inference algorithm is changed to $s' \leftarrow \arg\max_{s'' \in \{s'_{ij}\}}(\mathbb{O}(s''))$ and an additional call is inserted between line

Algorithm 1 Inference procedure

```
1: function INFERENCE(w, a, u_0)
2:     s ← (w, a, u_0)
3:     for t=1,2 ... m do
4:         {s'_ij} ← NEIGHBORS(s)
5:         s' ← arg max_{s''∈{s'_ij}}(π(s''))
6:         if π(s') > π(s) then
7:             s ← s'
8:         else
9:             break
10:        end if
11:    end for
12:    return s
13: end function
```

```
1: function NEIGHBORS(s)
2:     for i=1,2 ... N_a do
3:         {c_j} ← CANDIDATES(s.a_i)
4:         for j=1,2 ... k do
5:             s'_ij ← s
6:             s'_ij.u_j ← c_j
7:         end for
8:     end for
9:     return {s'_ij}
10: end function
```

6 and 7: $\theta \leftarrow \textsc{Update}(s', s, \theta)$, in order to update the model parameter at each step according to the regular SampleRank algorithm [29].

3.5 Templates

In the following, we describe the templates that are used to automatically instantiate the factors between variables in a configuration and, thus, for the extraction of the features that determine the probability of a configuration (see Eq. (2)). Throughout this discussion, we use a_i to denote the ith annotation span, and $\mathit{Cand}(a_i) = \{c_1, \ldots, c_k\}$ to denote the set of entity candidates for a_i. Further, we denote the actually assigned URI for a_i as $u_i \in \mathit{Cand}(a_i)$.

Relative Term Frequency This template instantiates a factor between each assigned URI u_i and its corresponding annotation span a_i in order to reflect the co-occurrence of u_i and a_i in our index (see Section 3.1). The feature value $\mathit{RTF}(a_i, u_i)$ for such a factor is defined by the term-candidate frequency normalized across all candidate URIs that are retrieved for a_i:

$$\mathit{RTF}(a, u) = \frac{\mathit{freq}(a, u)}{\sum_{c \in \mathit{Cand}(a)} \mathit{freq}(a, c)} \tag{5}$$

Edit Similarity In this template, we add a factor between each annotation span a_i and its assigned URI u_i that reflects the string similarity $l(a_i, u_i)$ between those two based on the Levenshtein distance [18]. We use the maximum length to normalize the string similarity $l(a_i, u_i)$ and it is calculated as follows $v(a_i, u_i) = 1 - \frac{l(a_i, u_i)}{\max(len(a_i), len(u_i))}$ which is added as a feature to the factor. Further, we create n equally distributed thresholds $t_j \in (0, 1]$. For each $t_j \leq v(a_i, u_i)$, an additional boolean feature is added.

Document Similarity Following [5], we hypothesize a positive impact of the textual context on named entity disambiguation, in particular for NEs that share the same surface form (e.g., *apple*). We represent the content of a document as a bag-of-words vector that is constructed from all of its tokens. Each document is preprocessed by applying tokenization, case normalization, stemming and stopwords removal. Vector components are weighted by their term frequency *tf* and their inverse document frequency *idf*. The latter is computed on the Wikipedia abstract corpus. Given a document d, we denote its document vector as $\boldsymbol{v_d}$. The document vector of an assigned URI u_i is denoted as $\boldsymbol{v_{u_i}}$ which is computed analogously from its corresponding Wikipedia abstract. For each u_i in d, we add a factor to the factor graph whose feature is defined by the cosine similarity of $\boldsymbol{v_d}$ and $\boldsymbol{v_{u_i}}$:

$$cos(\boldsymbol{v}, \boldsymbol{w}) = \frac{\sum_{i=1}^{n} \boldsymbol{v}_i \boldsymbol{w}_i}{\sqrt{\sum_{i=1}^{n} \boldsymbol{v}_i^2} \sqrt{\sum_{i=1}^{n} \boldsymbol{w}_i^2}} \tag{6}$$

Relative Page Rank The Relative Page Rank template instantiates a factor for each assigned URI u_i to measure its a-priori popularity in Wikipedia. The PageRank scores $PR(u_i)$ are computed on a subgraph of the Wikipedia PageLinks data set[5] excluding all category, disambiguation and file pages. We calculate the PageRank scores based on the approach explained in [21] that uses the random walk algorithm by Das Sarma et al.[8]. We normalize the raw PageRank scores over all $c \in Cand(a_i)$ as described in Eq. (7) and add the relative score $RPR(a_i, u_i)$ as a feature to the factor.

$$RPR(a, u) = \frac{PR(u)}{\sum_{c \in Cand(a)} PR(c)} \tag{7}$$

Topic-specific PageRank To measure the degree of coherence between all assigned URIs $(u_1, \ldots, u_{N_a})$ in a state, we introduce the Topic-specific PageRank template. Topic-specific PageRank [12] is computed on the Wikipedia graph using the random walk with restart (RWR) algorithm as described by Moro et al. [21]. Following the notation by Moro et al. [21], we set the RWR parameters as follows: the minimum hit threshold $\eta = 100$, the restart probability $alpha = 0.85$, the number of iterations $n = 1.000.000$ and the transition probability P as uniformly distributed for all neighbor nodes.

For each pair of URIs $p_{ij} = (u_i, u_j)$ where $i \neq j$, we add a new factor to the factor graph connecting u_i and u_j. The feature value for p_{ij} is determined by the sum of the Topic-Specific PageRank values of $TSPR(u_i, u_j)$ and $TSPR(u_j, u_i)$. For pairs where $u_i = u_j$ the feature value is set to 1 in order to encourage repetitions of the same URI.

[5] http://wiki.dbpedia.org/Downloads2015-04

Table 1. Micro F_1 scores of models trained on combinations of features RPR (Relative PageRank), RTF (Relative Term Frequency), ES (Edit Similarity), DS (Document Similarity), TSPR (Topic Specific PageRank) as defined in Section 3.5

Feature Combinations	AIDA/CoNLL Test-A	MicroPost2014 Test
RPR	0.720	0.66
RTF	0.619	0.60
ES	0.500	0.49
DS	0.230	0.29
TSPR	0.725	0.29
RPR + RTF	0.723	0.68
RPR + ES	0.724	0.67
RPR + TSPR	0.747	0.65
RPR + DS	0.720	0.66
RPR + RTF + ES	0.718	**0.70**
RPR + RTF + TSPR	0.747	0.67
RPR + RTF + DS	0.721	0.68
RPR + RTF + ES + DS	0.737	0.69
RPR + RTF + ES + TSPR	**0.781**	0.64
RPR + RTF + ES + TSPR + DS	0.775	0.65

4 Experiments

In this section, we present our experimental results on different data sets. First, we evaluate the performance of different subsets of features on development data from the AIDA/CoNLL and Micropost2014 data sets in Section 4.1. In Section 4.2, we compare the performance of our model using the best feature configuration on the GERBIL benchmark [27] (version 1.2.2), addressing the D2KB task in which the named entities are pre-annotated so that only the actual disambiguation, but not the recognition, is evaluated.

4.1 Model Training and Feature Selection

We trained several models with various combinations of features as defined in Section 3.5, using training data from the AIDA/CoNLL [13] and Micropost2014 [6] data sets. We trained and tested models on documents where each annotation has a valid link in a knowledge base, e.g. DBpedia. Each model was trained by iterating 5 times over the training data using the training split of both data sets. Micro F_1 scores for each model are shown in Table 1. Note that the main focus of these experiments is to determine the optimal feature combination. Due to differences in the underlying data splits and evaluation, the results reported in Table 1 are not comparable to official GERBIL results.

The results show that the single best-performing features are the PageRank (RPR) and the Topic-specific PageRank (TSPR) features, which yield an F_1 score of around 0.72. PageRank acts as a strong prior, while TSPR models the connectedness between different linking candidates in a pairwise fashion. Both features are mildly complementary, which can be seen from the fact that they yield the best combination of two features on AIDA/CoNLL, obtaining an F_1 score of 0.747. No combination of three features improves upon this result. The overall best model capitalizes on four features: PageRank, TSPR, relative term

frequency and edit similarity. This combination yields an overall performance of F_1=0.78.

On the MicroPost2014 data set that contains a significantly smaller amount of annotations compared to AIDA/CoNLL (2.1 vs. 20 annotations per document on average), the best model combines PageRank, relative term frequency and edit similarity, yielding an F_1 score of 0.70. Obviosuly, the Topic-specific PageRank is less effective in this text genre, due to its considerably lower degree of connectedness and the lower number of links to explore. Being the only feature that incorporates connectedness of all candidates, the strong individual performance of TSPR on AIDA/CoNLL is indicative of the advantages of a collective disambiguation strategy over an approach that resolves entities independently of one another.

4.2 Comparative Evaluation

In this experiment, we use the best-performing model configurations as determined by feature selection (see Table 1). Our system uses the model with RPR + RTF + ES + TSPR features when the number of annotations in a given document is higher than 3. When the number of annotations is equal or lower than 3, the model with RPR + RTF + ES features is used. We use the top 10 candidates from the Candidate Retrieval component of the system as explained in Section 3.1. Returning 100 or more candidates increases the runtime while having no significant improvement on performance.

We compare our system to other state-of-the-art systems on 14 publicly available data sets via GERBIL version 1.2.2 [27]. We implemented a web service called NERFGUN (**N**amed **E**ntity disambiguation by **R**anking with **F**actor **G**raphs over **Un**directed edges), with the two best models after feature selection and submitted to GERBIL. The results of our system[6] and state-of-the-art annotation systems[7] that are integrated into GERBIL are presented in Table 2. Since Ganea et al. [9] and Zwicklbauer et al. [30] evaluated their systems with respect to older versions of GERBIL (version 1.1.4) and these systems do not have submitted a publicly available API to the framework, we cannot fairly compare to them. Thus, we ommited these systems from comparison. This is in particular the case because the evaluation metrics have changed in recent versions of the GERBIL framework. Note that all results presented in Table 2 are based on GERBIL version 1.2.2.

In Table 2 we report Micro F_1 and Macro F_1 measures of compared systems for 14 data sets. Based on Micro F_1 and Macro F_1 measures, NERFGUN obtains the best result on 8 out of 14 data sets. Kea [28] outperforms all systems on the AQUAINT and the DBpedia Spotlight data. On the AQUAINT, the results of our system are comparable to the best annotation system (Micro F_1 0.73 compared to 0.77, Macro F_1 0.72 compared to 0.76, respectively). Babelfy achieves

[6] Our results, GERBIL v1.2.2:
http://gerbil.aksw.org/gerbil/experiment?id=201604290045

[7] State-of-the-art annotation systems' results, GERBIL v1.2.2:
http://gerbil.aksw.org/gerbil/experiment?id=201604270003

the highest score for the KORE50 data set with 0.74 and 0.71 while NERFGUN obtains 0.44 and 0.40 for Micro F_1 and Macro F_1 measures respectively.

Table 2. Macro F_1 and Micro F_1 measures for the D2KB task (named entity disambiguation) based on GERBIL v1.2.2; N/A : Not Available. The best scoring system for each data set is highlighted (using **boldface** for the best Micro F_1 result and *italics* for Macro F_1, respectively).

Systems	Measures	ACE2004	AIDA/CoNLL-Complete	AIDA/CoNLL-Test A	AIDA/CoNLL-Test B	AIDA/CoNLL-Training	AQUAINT	DBpediaSpotlight	IITB	KORE50	Microposts2014-Test	Microposts2014-Train	MSNBC	N3-Reuters-128	N3-RSS-500
AGDISTIS	Micro F1	0.63	0.54	0.54	0.52	0.55	0.52	0.27	0.47	0.32	0.67	0.33	0.43	**0.61**	**0.65**
	Macro F1	0.77	0.52	0.49	0.53	0.53	0.51	0.28	0.49	0.3	0.64	0.6	0.61	***0.61***	***0.71***
AIDA	Micro F1	0.14	0.54	0.54	0.52	0.55	0.16	0.19	0.18	0.65	0.3	0.36	0.44	0.44	0.39
	Macro F1	0.44	0.5	0.47	0.5	0.5	0.16	0.17	0.19	0.59	0.28	0.57	0.58	0.38	0.32
Babelfy	Micro F1	0.52	0.66	0.65	0.68	0.66	0.68	0.53	N/A	**0.74**	0.64	0.48	0.51	0.45	N/A
	Macro F1	0.69	0.6	0.59	0.62	0.61	0.68	0.52	N/A	***0.71***	0.59	0.63	0.61	0.39	N/A
DBpedia Spotlight	Micro F1	0.47	0.5	0.48	0.52	0.5	0.53	0.71	0.3	0.45	0.39	0.5	0.49	0.2	0.34
	Macro F1	0.67	0.49	0.47	0.5	0.5	0.52	0.69	0.29	0.41	0.39	0.66	0.61	0.17	0.27
Dexter	Micro F1	0.52	0.51	0.49	0.5	0.52	0.52	0.29	0.21	0.2	0.38	0.41	0.43	0.37	0.36
	Macro F1	0.68	0.48	0.45	0.47	0.48	0.51	0.26	0.21	0.14	0.4	0.59	0.56	0.3	0.31
Entityclassifier.eu NER	Micro F1	0.49	0.5	0.47	0.47	0.51	0.41	0.25	0.14	0.29	0.45	0.41	0.48	0.34	0.37
	Macro F1	0.66	0.48	0.47	0.46	0.48	0.38	0.2	0.16	0.26	0.44	0.6	0.6	0.32	0.34
FOX	Micro F1	0	0.51	0.49	0.47	0.51	0	0.15	0.02	0.29	0.02	0.23	0.32	0.57	0.55
	Macro F1	0.37	0.48	0.44	0.47	0.49	0	0.12	0.02	0.25	0.02	0.5	0.49	0.55	0.59
FREME NER	Micro F1	0.69	0.6	0.59	0.57	0.61	**0.78**	**0.82**	0.43	0.32	0.53	**0.65**	**0.65**	0.42	0.51
	Macro F1	0.81	0.6	0.57	0.59	0.6	***0.78***	***0.83***	0.42	0.3	0.56	***0.78***	***0.76***	0.38	0.48
Kea	Micro F1	0.65	0.62	0.61	0.6	0.63	0.77	0.74	0.48	0.59	**0.7**	0.64	**0.65**	0.44	0.51
	Macro F1	0.76	0.59	0.56	0.59	0.6	0.76	0.73	0.47	0.53	0.67	0.77	0.74	0.39	0.46
NERD-ML	Micro F1	0.56	0.2	0	0.01	0.28	0.59	0.55	0.43	0.32	0.54	0.5	0.51	0.38	0.41
	Macro F1	0.72	0.12	0.01	0.01	0.17	0.57	0.53	0.42	0.26	0.54	0.65	0.62	0.31	0.35
WAT	Micro F1	0.65	0.71	0.7	**0.71**	0.71	0.72	0.66	0.41	0.61	0.65	0.6	0.63	0.44	0.51
	Macro F1	0.77	0.68	0.66	0.68	0.68	0.72	0.68	0.4	0.51	0.62	0.74	0.73	0.37	0.43
xLisa	Micro F1	0.47	0.15	0.41	0.4	0.4	0.42	0.22	**0.57**	0.24	0.45	0.24	0.51	0.43	0.32
	Macro F1	0.63	0.45	0.37	0.33	0.37	0.37	0.22	***0.58***	0.25	0.38	0.24	0.63	0.37	0.29
NERFGUN	Micro F1	**0.73**	**0.72**	**0.71**	**0.71**	**0.72**	0.70	0.49	N/A	0.40	0.65	**0.65**	N/A	0.57	**0.65**
	Macro F1	***0.85***	***0.72***	***0.68***	***0.71***	***0.72***	0.69	0.51	N/A	0.37	***0.79***	0.76	N/A	0.58	0.65

5 Conclusion and Future Work

We have proposed a new approach to collective entity disambiguation that frames the task as a joint inference problem. Our approach relies on an undirected probabilistic graphical model to model dependencies between different factors that score the suitability of assignments of named entities to identifiers in a KB. The model is defined through imperatively defined factor graphs and in particular by templates that 'roll out' the factor graph structure for a given input text by generating corresponding factors. Our model allows to combine and investigate different features in terms of their impact on the task. In particular, our

model considers three text-based features, namely i) term frequency scores, ii) a similarity measure based on the Levenshtein distance and iii) the document similarity. Further, our model includes features measuring the degree of connectedness between pairs of linking candidates via the Topic Specific PageRank Score and the PageRank of each linking candidate as a prior. We have shown that a combination of all features with exception of the document similarity feature performs best on the AIDA/CoNLL data sets. Based on Micro F_1 and Macro F_1 measures we outperform well-known annotation systems such as DBpedia Spotlight, Babelfy, WAT and AGDISTIS in 8 out of 14 datasets. In future work, we will extend the approach to also solve the problem of recognition of entities, thus performing named entity recognition and linking within one model in which statistical dependencies between both tasks can be modeled.

Acknowledgements

This work was supported by the Cluster of Excellence Cognitive Interaction Technology 'CITEC' (EXC 277) at Bielefeld University, which is funded by the German Research Foundation (DFG).

References

1. Alhelbawy, A., Gaizauskas, R.J.: Graph ranking for collective named entity disambiguation. In: Proc. of ACL (Short Papers). pp. 75–80. Baltimore, MD (2014)
2. Andrieu, C., de Freitas, N., Doucet, A., Jordan, M.I.: An Introduction to MCMC for Machine Learning. Machine Learning 50, 5–43 (2003)
3. Barrena, A., Soroa, A., Agirre, E.: Combining mention context and hyperlinks from wikipedia for named entity disambiguation. In: Proceedings of ⋆SEM. pp. 101–105. Denver, CO (2015)
4. Blei, D.M., Ng, A., Jordan, M.: Latent Dirichlet Allocation. Journal of Machine Learning Research 3, 993–1022 (2003)
5. Bunescu, R.C., Pasca, M.: Using encyclopedic knowledge for named entity disambiguation. In: Proceedings of EACL. pp. 9–16 (2006)
6. Cano, A.E., Rizzo, G., Varga, A., Rowe, M., Stankovic, M., Dadzie, A.S.: Making sense of microposts: (# microposts2014) named entity extraction & linking challenge. In: CEUR Workshop Proceedings. vol. 1141, pp. 54–60 (2014)
7. Daiber, J., Jakob, M., Hokamp, C., Mendes, P.N.: Improving efficiency and accuracy in multilingual entity extraction. In: Proceedings of SEMANTICS (2013)
8. Das Sarma, A., Molla, A.R., Pandurangan, G., Upfal, E.: Fast distributed pagerank computation. Theoretical Computer Science 561, Part B, 113–121 (2015), Special Issue on Distributed Computing and Networking
9. Ganea, O.E., Horlescu, M., Lucchi, A., Eickhoff, C., Hofmann, T.: Probabilistic bag-of-hyperlinks model for entity linking. In: Proceedings of WWW (2016)
10. Guo, Z., Barbosa, D.: Robust entity linking via random walks. In: Proceedings of CIKM. pp. 499–508. Shanghai, China (2014)
11. Hakimov, S., Oto, S.A., Dogdu, E.: Named entity recognition and disambiguation using linked data and graph-based centrality scoring. Proceedings of the Workshop on Semantic Web Information Management (SWIM) pp. 1–7 (2012)

12. Haveliwala, T.H.: Topic-sensitive PageRank. In: Proceedings of WWW. pp. 517–526 (2002)
13. Hoffart, J., Yosef, M.A., Bordino, I., Fürstenau, H., Pinkal, M., Spaniol, M., Taneva, B., Thater, S., Weikum, G.: Robust disambiguation of named entities in text. In: Proceedings of EMNLP. pp. 782–792. Edinburgh, Scotland, UK (2011)
14. Houlsby, N., Ciaramita, M.: A Scalable Gibbs Sampler for Probabilistic Entity Linking. In: Proceedings of ECIR. pp. 335–346 (2014)
15. Jin, Y., Kcman, E., Wang, K., Loynd, R.: Entity linking at the tail: Sparse signals, unknown entities and phrase models. In: Proceedings of WSDM (2014)
16. Klinger, R., Cimiano, P.: Joint and pipeline probabilistic models for fine-grained sentiment analysis: Extracting aspects, subjective phrases and their relations. Proceedings of ICDMW pp. 937–944 (2013)
17. Kschischang, F.R., Frey, B.J., Loeliger, H.A.: Factor Graphs and Sum Product Algorithm. IEEE Transactions on Information Theory 47(2), 498–519 (2001)
18. Levenshtein, V.I.: Binary codes capable of correcting deletions, insertions, and reversals. Soviet Physics Doklady 163(4), 707–710 (1966)
19. Liu, X., Li, Y., Wu, H., Zhou, M., Wei, F., Lu, Y.: Entity linking for tweets. In: Proceedings of ACL. pp. 1304–1311. Sofia, Bulgaria (2013)
20. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)
21. Moro, A., Raganato, A., Navigli, R.: Entity linking meets word sense disambiguation: a unified approach. Transactions of the Association for Computational Linguistics 2, 231–244 (2014)
22. Passos, A., Kumar, V., McCallum, A.: Lexicon infused phrase embeddings for named entity resolution. arXiv preprint arXiv:1404.5367 (2014)
23. Piccinno, F., Ferragina, P.: From TagME to WAT. A New Entity Annotator. In: Proceedings of ACM Workshop on Entity Recognition & Disambiguation. pp. 55–62 (2014)
24. Singh, S., Subramanya, A., Pereira, F., McCallum, A.: Large-scale cross-document coreference using distributed inference and hierarchical models. In: Proceedings of ACL, Vol. 1. pp. 793–803 (2011)
25. Tristram, F., Walter, S., Cimiano, P., Unger, C.: Weasel. A Machine Learning based Approach to Entity Linking Combining Different Features. In: Proceedings of ISWC Workshop on NLP and DBpedia (2015)
26. Usbeck, R., Ngomo, A.C.N., Röder, M., Gerber, D., Coelho, S.A., Auer, S., Both, A.: AGDISTIS. Graph-based Disambiguation of Named Entities Using Linked Data. The Semantic Web–ISWC 2014 pp. 457–471 (2014)
27. Usbeck, R., Röder, M., Ngonga Ngomo, A.C., Baron, C., Both, A., Brümmer, M., Ceccarelli, D., Cornolti, M., Cherix, D., Eickmann, B., et al.: GERBIL. General Entity Annotator Benchmarking Framework. In: Proceedings of WWW. pp. 1133–1143 (2015)
28. Waitelonis, J., Sack, H.: Named entity linking in #tweets with kea. In: Proceedings of 6th workshop on Making Sense of Microposts – Named Entity Recognition and Linking (NEEL) Challenge, at WWW2016 (2016)
29. Wick, M., Rohanimanesh, K., Culotta, A., McCallum, A.: SampleRank. Learning preferences from atomic gradients. NIPS Workshop on Advances in Ranking pp. 1–5 (2009)
30. Zwicklbauer, S., Seifert, C., Granitzer, M.: Doser-a knowledge-base-agnostic framework for entity disambiguation using semantic embeddings. In: European Semantic Web Conference. pp. 182–198. Springer (2016)