Scalable graph-based method for individual named entity identification

Sammy Khalife **Michalis Vazirgiannis**
LIX, CNRS, Ecole Polytechnique
Institut Polytechnique de Paris, 91128 Palaiseau, France
khalife@lix.polytechnique.fr
mvazirg@lix.polytechnique.fr

Abstract

In this paper, we consider the named entity linking (NEL) problem. We assume a set of queries, named entities, that have to be identified within a knowledge base. This knowledge base is represented by a text database paired with a semantic graph, endowed with a classification of entities (ontology). We present state-of-the-art methods in NEL, and propose a new method for individual identification requiring few annotated data samples. We demonstrate its scalability and performance over standard datasets, for several ontology configurations. Our approach is well-motivated for integration in real systems. Indeed, recent deep learning methods, despite their capacity to improve experimental precision, require lots of parameter tuning along with large volume of annotated data.

1 Introduction

1.1 Basic concepts and definitions

The purpose of *Named entity discovery* (NED) in information retrieval is two-fold. First, it aims at extracting pre-defined sets of words from text documents: this corresponds to Named entity recognition (NER). These words are representations of *named entities* (such as names, places, locations, ...). Then, these *entity mentions* paired with their context are seen as *queries* to be identified within database: this corresponds to named entity linking (NEL). NEL is also refered as named entity disambiguation. The interest in NEL has grown recently in several fields: in bioinformatics, to obtain locations of viral sequences from databases (Weissenbacher et al., 2015), or to process biomedical litterature (Zheng et al., 2015). It also revealed to be useful in recruitment in order to identify employer names in a database (Liu et al., 2018). Firstly, it is important to stress that the subtask of NED, Named entity recognition (NER), is not trivial since we do not have an exhaustive list of the possible spelling of named entities. Moreover their text representation can change (for example, “J. Kennedy" vs. “John Kennedy"). In this paper we focus on the second task, *Named entity linking (NEL)*.

Named entity (and Mention/Query): An entity is a real-world object and usually has a physical existence. It is denoted with a proper name. In the expression “Named Entity", the word “Named" aims to restrict the possible set of entities to only those for which one or many rigid designators stands for the referent (Nadeau and Sekine, 2007). When a named entity appears in a document, its surface form can also be refered as a *mention*. Finally, a *query* refers to the mention, the context where it appears, and a type of entity considered.

Ontology: In this paper, our definition of an ontology is represented as a tree of entity types. In the following, the variable $\boldsymbol{T}$ represents the total number of nodes of this tree minus one (we don't count the root node since it is uninformative). Originally, entities had a very limited number of types (Nadeau and Sekine, 2007), such as person (*PER*), organization (ORG), and localization (*GPE*) (i.e $\boldsymbol{T} = 3$). These types play a central role for named entity recognition and identification. An example of ontology is in Fig. 1. More recently, due to the increase in the volume of the web semantics data, fine-grained classifications are available, with hundreds of entity types, similarly to DBPedia ontology[1] (Lehmann et al., 2015).

Knowledge base/graph: A Knowledge base is a database providing supplementary descriptive and semantic information about entities. The semantic information is contained in a knowledge graph, where a node represents an entity, and an edge represents a semantic relation. The knowledge graph

[1] http://wiki.dbpedia.org/services-resources/ontology

Proceedings of the Thirteenth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-13), pages 17–25
Hong Kong, November 4, 2019.

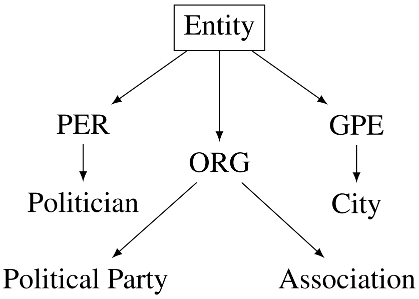

Figure 1: Example of ontology, $\boldsymbol{T} = 7$

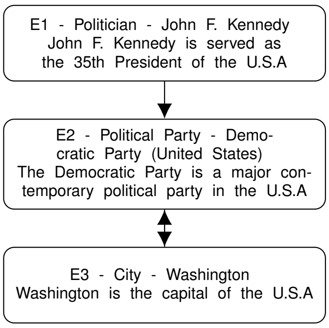
E1 - Politician - John F. Kennedy
John F. Kennedy is served as the 35th President of the U.S.A
E2 - Political Party - Democratic Party (United States)
The Democratic Party is a major contemporary political party in the U.S.A
E3 - City - Washington
Washington is the capital of the U.S.A

Figure 2: Representation of a *unweighted directed semantic graph* (Wikipedia/NIST TAC-KBP Challenge 2010). An edge between two entities E_1 and E_2 represents a url link from E_1 web page to E_2 web page.

can be of any kind (directed, weighted, ...). See Fig. 2 for an example.

Named entity linking (NEL): Given a named entity query, the purpose of named entity linking is to identify the corresponding ground truth entity (*gold entity*) in a database (*knowledge base*). For a detailed description of a concrete competition in entity linking, we refer to (Ji et al., 2014).

Individual & collective linking: Linking can be done *individually* or *collectively*. In the first case, queries are independent. In the collective framework, we consider a set of queries that usually originates from the same document, and for which gold entities (i.e *ground truth* entities) should have some proximity, or coherence. In this work, we propose individual linking approach.

1.2 Contributions

In this work, we provide a brief survey of existing methods for named entity linking. Then, we investigate a method for individual named entity linking. The first step of this method, refered as *entity filtering*, reduces entity candidates to top $\boldsymbol{K}$ entities for one *query*. The second step, refered as *entity identification*, aims at identifying the true entity among the remaining $\boldsymbol{K}$ candidates, based on a new graph-based algorithm. We include an experimental evaluation of our method with several datasets, with an analysis of the impact of parameter $\boldsymbol{K}$, the ontology parameter $\boldsymbol{T}$, and a detailed comparison with existing approaches. The implementation used for experiments is available at our repository[2]. We do not include in this paper work on *Fine-grained named entity recognition* (Ling and Weld, 2012). Moreover, we do not include NIL-detection problem (detect if a query is referring to an entity that is not in the knowledge base, for instance (Ji et al., 2014)).

2 Related work

In the following subsections, we present three families of algorithms for named entity linking. ***Notations:*** $\boldsymbol{E} = \{1, ..., E\} \subset \mathbb{N}$: indexes of entities and $\boldsymbol{Q} = \{1, ..., Q\} \subset \mathbb{N}$: indexes of queries, $\hat{e}_i$: system's output entity index for query index q_i.

2.1 Graphs for NED

Formulation: Given a scoring function defined between queries and entities, let $W_{i,j}$ the corresponding score between the query i and the entity j. For individual disambiguation, one wants to perform independent query-entity attribution. A straightforward formulation is:

$$\hat{e}_i = \arg\max_{j \in E} W_{i,j} \tag{1}$$

In this case, the total cost is separable in the variable i, but the score $W_{i,j}$ can use the knowledge graph structure: this is the case in our approach. For the sake of completeness, we give a description of the collective linking formulation. In this framework the optimization formulation is different: the underlying *gold entities* should respect some arbitrary semantic coherence. The coherence information is represented within a *coherence function* $\psi : \boldsymbol{E}^Q \rightarrow \mathbb{R}$ between the entity candidates. Usually ψ is defined from the knowledge graph structure. For example ψ can be defined using the opposite sign of the shortest-path function on the *knowledge graph*. With these notations, the set of selected entities are formally defined as:

$$\hat{e_1}, ..., \hat{e_Q} = \arg\max_{j_1,..,j_q \in E^Q} [(\sum_{l=1}^{Q} W_{l,j_l}) + \psi(j_1, ..., j_Q)] \tag{2}$$

[2] If needed, please send your request at khalife@lix.polytechnique.fr.

Eq. (2) can be formulated as a boolean integer program. Its *NP-hardness* (Cucerzan, 2007) does not allow to solve the general case for an important number of queries. (Ratinov et al., 2011) evaluated local and global approaches to find approximate solutions of an approximation of Eq. (2) with Eq. (3), given a new coherence function $\tilde{\psi}$, and for each query q_l a disambiguation context of entities C_l:

$$\hat{e_1}, ..., \hat{e_Q} = \underset{j_1,..,j_q \in E^Q}{\arg\max} [(\sum_{l=1}^{Q} W_{l,j_l} + \sum_{k \in C_l} \tilde{\psi}(j_l, j_k))] \tag{3}$$

The formulation with Eq. (3) is halfway between individual and collective linking: it suggests to select a convenient set of disambiguation contexts, and then solving locally for each query. In the same time, it still enforces some coherence among the predited entities. Collective linking also has other formulations: (Han et al., 2011) proposed a collective formulation for entity linking decisions, in which evidence can be reinforced into high-probability decisions.

For individual graph based linking, a rule based on the importance of the entity node in the knowledge graph rule has been studied experimentally (Guo et al., 2011).

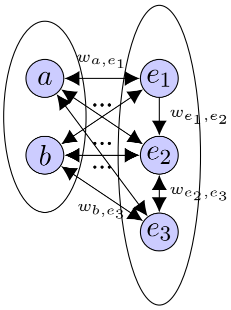

Figure 3: Directed query/entity bipartite weighted graph. Nodes e_1, e_2, e_3 are entities in the knowledge base (same as Fig. 2). Nodes a and b are entity queries extracted from text documents.

Dense subgraphs, PageRank: Other graph-based approaches have been developed. (Hoffart et al., 2011), and (Alhelbawy and Gaizauskas, 2014) proposed to link efficiently a query to its corresponding entity using the weighted undirected bipartite graph (Fig. 3). The idea is to extract a dense subgraph in which every query node is connected to exactly one entity, yielding the most likely disambiguation. In general, this combinatorial optimization problem is *NP-hard* with respect to the number of nodes, since they generalize Steiner-tree problem (Hoffart et al., 2011). However heuristics to solve this problem have been experimented: (Hoffart et al., 2011) and (Alhelbawy and Gaizauskas, 2014) proposed a discarding algorithm using taboo search and local similarities with polynomial complexity. Adaptations of PageRank algorithm were carried out to provide each entity a popularity score: (Usbeck et al., 2014) built a weighted graph of all queries and entities based on local and global similarities, and capitalize on the Hyperlink-Induced Topic Search (HITS) algorithm to produce node authority scores. Then, within similar entities to queries, only entities with high authority will be retained.

2.2 Probabilistic graphical models

Another interesting idea is to consider named entity queries as random variables and their golden/true entities as hidden states. Unlike character recognition where $|E| = |E_i| = 26$ for latin alphabet, the number of possible states S per entity is large (usually $S \geq 10^6$). Since Viterbi algorithm has a $\mathcal{O}(N\,|S|^2)$ complexity, where N is the number of observations, inference is inefficient. To overcome this issue, (Alhelbawy and Gaizauskas, 2013) considers a reduced set of candidates per query: $e_i \in E_i$ using query text information. Using annotation, an Hidden Markov Model (HMM) is trained on the reduced set of candidates. Inference is made using message passing (Viterbi algorithm) to find the most probable named entity sequence. Another approach using probabilistic graphical model has been provided by (Ganea et al., 2016), with a factor graph that uses popularity-based prior.

2.3 Embeddings and deep architectures

Recent advances in neural networks conception suggested to use word embeddings and convolutional neural networks to solve the named entity linking problem. (Sun et al., 2015) proposed to maximize a corrupted cosine similarity between a query, its annotated gold entity and a false entity. An example of learning representations for entities using a neural architecture is achieved in (Yamada et al., 2017), a linking system based on the similarity of average of pre-trained entity embeddings has been proposed (Yamada et al., 2016), with a $O(QE^2)$ complexity. Finally other architectures have been proposed (Sil et al., 2018; Raiman and

Raiman, 2018), the latter using a fine-grained ontology type system and reaching promising results on several datasets.

3 Methodology

In this section we present a novel graph-based method for NEL. As a preprocessing step, we propose a new but simple entity filtering method using information retrieval techniques to obtain a limited number of entity candidates. The novelty of our method lies in the second subsection where we present a new graph-based method for final entity identification. Source code is available at our repository[2].

3.1 Entity filtering

To discard wrong entity candidates, we use the three sources of information in the query $\boldsymbol{q} = (\boldsymbol{m}, \boldsymbol{c}, \hat{\boldsymbol{t}})$: the mention name, the information contained in the rest of document, and the entity type. Obviously, the richer is the ontology (i.e the larger is $\boldsymbol{T}$), the easier the NEL problem (but harder is the NER problem). In order to improve existing entity filtering algorithms, we propose a routine based on three main components below. The algorithm is summarized in algorithm 1).

a* - preProcess*: For trivial queries having a mention name equal to an existing entity name and type, we implemented a naive match preprocessing. If a mention has the same name and the same type, its gold entity is labelled as the corresponding entity.

b* - acronymDetection *&* acronymScore*: Acronym detection and expansion is a common topic in bioinformatics. We refer to (Ehrmann et al., 2013) as a survey of acronym detection methods. We implemented a simple rule-based decision for acronym detection, following (Gusfield, 1997): a string is tagged as an acronym if there are two or more capital letters, and that consecutive distance between two capital letters is always one. The similarity score for acronym extension is chosen as the length of longest common substring (Apostolico and Guerra, 1987) between the acronym and capital letters of the target.

c* - JN *&* contextScore*: When the named entity mention is not tagged as an acronym, comparison with entity titles is performed by computing N-grams for $N \in \{2, 3, 4\}$, and use Jaccard Index of mention name and entity title. It is refered as JN in algorithm 1. We also mesure similarity between the context of the query and the text description of an entity in the knowledge base. We experimented several techniques: TF-IDF, BM25 , BM25+ based on the probabilistic retrieval framework developed in the 1970s and 1980s (we refer to (Robertson et al., 2009) for a recent description). The experimental results were very similar in term of recall. We present results obtained with TFIDF (cf. Sec. 4).

Algorithm 1 Entity filtering (generation of entity candidates)

```
Require: Parameter $\boldsymbol{K}$, Query ($\boldsymbol{q} = (\boldsymbol{m}, \boldsymbol{c}, \hat{\boldsymbol{t}})$), Entities $(\boldsymbol{e}_j, \boldsymbol{t}_j)_{1 \le j \le \boldsymbol{E}}$
preProcess($\boldsymbol{q}$, $(\boldsymbol{e}_j, \boldsymbol{t}_j)_{1 \le j \le \boldsymbol{E}}$)
$\boldsymbol{ds} = []$
$\boldsymbol{y}_{acr} \leftarrow$ acronymDetection($\boldsymbol{m}$)
for $j = 1 \rightarrow E$ do
    if $t_j == \hat{t}$ then
        if $\boldsymbol{y}_{acr} == 1$ then
            $s_n$ = acronymScore($\boldsymbol{m}, \boldsymbol{e}_j$)
        else
            $s_n$ = JN($\boldsymbol{m}, \boldsymbol{e}_j$)
        end if
        $s_t = \frac{1}{2}(s_n$ + contextScore($\boldsymbol{c}, \boldsymbol{e}_j$))
        Sorted insertion by value of $\{j : s_t\}$ in $\boldsymbol{ds}$
    end if
end for
return $\boldsymbol{ds}[: \boldsymbol{K}]$ ($\boldsymbol{K}$ top entities )
```

3.2 Graph-based identification

In this section, we present our graph-based method for *named entity identification*. This graph-based method uses enriched features extraction from the knowledge graph, in order to re-rank top entity candidates.

Feature extraction: Let $\boldsymbol{q}$ and $\boldsymbol{e}$ respectively a query and an entity. $\boldsymbol{T}$ still represents the number of distinct entity types in the ontology. Let s be a scoring function between a query and an entity. Let $\mathcal{N}_t(\boldsymbol{e})$ the set of entity neighbors of type t (cf. Fig 4 for an example). By convention if $\mathcal{N}_t(\boldsymbol{e}) = \emptyset$, then $s(\boldsymbol{q}, \mathcal{N}_t(\boldsymbol{e})) \triangleq 0$. $f(\boldsymbol{q}, \boldsymbol{e})$ is the filtering score obtained with algorithm 1. We define the features vector associated with the couple $(\boldsymbol{q}, \boldsymbol{e})$, $\boldsymbol{X}^{\boldsymbol{q},\boldsymbol{e}}$ as the scores concatenation:

$$\begin{aligned} (\boldsymbol{X}^{\boldsymbol{q},\boldsymbol{e}})_0 &= f(\boldsymbol{q}, \boldsymbol{e}) \\ \forall t \in \{1, ..., \boldsymbol{T}\},\ (\boldsymbol{X}^{\boldsymbol{q},\boldsymbol{e}})_t &= s(\boldsymbol{q}, \mathcal{N}_t(\boldsymbol{e})) \end{aligned} \tag{4}$$

The label of a couple $(\boldsymbol{q}, \boldsymbol{e})$ is defined as:

$$Y^{\boldsymbol{q},\boldsymbol{e}} = \begin{cases} 1 & \text{if } \boldsymbol{e} \text{ is the gold entity of } \boldsymbol{q} \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

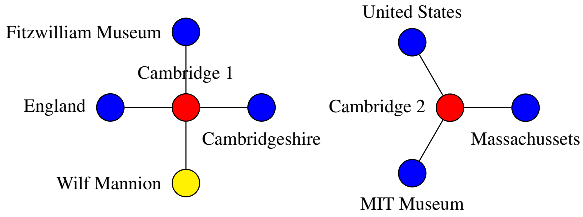

Figure 4: Two homonyms: Cambridge cities. Each color is assigned to a node in the ontology. If t_1 is associated to the entity type *Country*, and t_2 to *Football player*, then
$\mathcal{N}_{t_1}(\text{Cambridge 1}) = \{\text{England}\}$
$\mathcal{N}_{t_2}(\text{Cambridge 1}) = \{\text{Wilf Mannion}\}$
$\mathcal{N}_{t_1}(\text{Cambridge 2}) = \{\text{United States}\}$
$\mathcal{N}_{t_2}(\text{Cambridge 2}) = \emptyset$

Supervised NEL: With this formulation, we can train *NEL* standard regressors or classifiers in a supervised learning framework. At inference, the couple $(q, \hat{e})$ maximizing the prediction score yields predicted entity $\hat{e}$. If same scores are returned for different couples, we return the first candidate. (This situation didn't occur in practice). The feature extraction procedure and inference are summed up in algorithm 2 and algorithm 3 respectively.

Algorithm 2 Feature extraction using knowledge graph and ontology

```
Require: Knowledge Graph $\boldsymbol{G}$, Query $\boldsymbol{q}$, Entity candidate $\boldsymbol{e}$
    with initial filtering score $s^0$, Types $(\boldsymbol{t}_j)_{1\leq j\leq \boldsymbol{T}}$
1: $\boldsymbol{X}^{q,e} = [s^0]$
2: Get neighbor nodes of $\boldsymbol{e}$
3: for $j = 1$ to $\boldsymbol{T}$ do
4:     Aggregate text description of neighbors of type $t_j$
5:     Compute score $s_{t_j}$ between $\mathcal{N}_{t_j}(\boldsymbol{e})$ and the query $\boldsymbol{q}$
6:     Append $s_{t_j}$ to $X^{q,e}$
7: end for
8: return Score vectors $(\boldsymbol{X}^{q,e})_{1\leq j\leq \boldsymbol{T}+1}$
```

Algorithm 3 Named entity identification (Inference)

```
Require: Knowledge base $\boldsymbol{B}$ and its graph $\boldsymbol{G_B}$, queries
    $(\boldsymbol{q}_i)_{1\leq i\leq M}$, scoring threshold $\boldsymbol{K}$, trained predictor $\hat{\boldsymbol{F}}$
1: for $i = 1$ to $M$ do
2:     Use filtering on query $\boldsymbol{q}_i$ and $\boldsymbol{B}$, return a list of $\boldsymbol{K}$
       top ranked entities $(\boldsymbol{e}_h^1)_{1\leq h\leq K}$
3:     Use algorithm 2 using $\boldsymbol{G_B}$, on $\boldsymbol{K}$ entity candidates,
       return new score vectors
4:     Evaluate $\hat{\boldsymbol{F}}$ on each vector score and use maximum a
       posteriori to infer estimated gold entity $\hat{\boldsymbol{g}}_i$
5: end for
6: return $(\hat{\boldsymbol{g}}_i)_{1\leq i\leq M}$ (list of estimated gold entities)
```

Graph-based scoring functions: In the identification step, features defined from Eq. (4) require the choice of a scoring function. First of all, several representations for $\boldsymbol{q}$ and $\boldsymbol{e}$ are possible. In our first experiment, we used the standard TFIDF representation for the supervised learning procedure described previously, and the corresponding scoring function with cosine similarity. This allowed to increase slightly empirical accuracy over entity filtering.

In order to explore a broader class of scoring functions, let us introduce graph of words (GoW) representations. GoW is a representation built over a sequence of objects in order to capture sequential relationships. Given a window size, nodes are added to the graph by their string representation, and edges are added between nodes in the same sliding window. This representation has proven its efficiency for several information retrieval problems (Rousseau and Vazirgiannis, 2013).

Indeed, bag of words representations can be considered as a special case of graph of words representations, for which edge deleting operations have been applied. Here, we consider that the query context and the entity description are both composed of at least 10 words for GoW to be meaningful. The final step to define a scoring function as in Eq. (4), is to compare the two graph structures (one from the query context and the other from the entity description).

Given two graphs G and H, determining if G is isomorphic H allows to measure graph similarities (Cordella et al., 2004). However, for several applications, including the topic of this paper, isomorphic conditions are too rigid since two documents can be similar without isomorphic GoWs. Also, we are interested in graph similarity measures taking in account structure (word relations) and node attributes (words). For this reason, graph kernels have been popularized as a powerful tool to measure graph similarity in a continuous fashion.

Following the notations of Sec. 3.2, and k a graph kernel, we considered the family of scoring functions: $(\boldsymbol{q}, \boldsymbol{e}) \mapsto k(GoW_{\boldsymbol{q}}, GoW_{\mathcal{N}_t(\boldsymbol{e})})$ in our experiments. If $\mathcal{N}_t(\boldsymbol{e})$ contains more than one node, we concatenate their text content and compute a GoW. As mentioned previously, this family of functions countains some of the bag-of-words scoring functions, such as TFIDF. We obtained better empirical results using standard graph kernels (cf. next paragraph for examples). It should be noted that we could not use graph kernels for the first step (entity filtering), since the computation time would be too long. On the contrary, the identification step takes as input a limited amount of entity candidates, which makes the computation time reasonable.

Graph-of-words window, graph kernels & regressors: We selected as a graph-of-word window $w = 4$ (same results were obtained for $w \in \{3, 4, 5, 6\}$), with different graph kernels, including Shortest-path kernel, Weisfeiler-Lehman Kernel. The accuracy results for each graph kernel were very close, but higher than with TFIDF scoring (1% to 2% better). In Sec. 4, we report results for the pyramid match graph kernel, for its low complexity among standard kernels (Nikolentzos et al., 2017). Finally, we used several standard classifiers: regression trees, support vector machines, and logistic regression. We obtained better results with logistic regression (reported in Table 1).

Computational complexity: The total complexity (filtering and identification) is: $\mathcal{O}(M(E + \boldsymbol{KT}G))$. We report this in Table 2, along with some experimental computing times.

4 Experimental setup and evaluation

The source code of our experiments along with documentation, and datasets samples are available at our repository[2].

4.1 Datasets, entity types, and ontology:

We used CONLL and NIST TAC-KBP 2009-2010 as datasets. Each query contains its gold entity id and type. TAC-KBP: the corresponding knowledge base is composed of 818741 entities. TAC09 contains 1675 test queries, and TAC10 1074 for train and 1020 for test. CONLL/AIDA is composed of 22516 queries for training and 4379 queries for test.

The other methods, mainly deep learning (DL) in Table 1 use millions of training examples from Wikipedia's anchor links and corresponding entities. In our method, we did not use this additional training data, but only those provided by the original challenges.

Also, we considered a more recent Knowledge base (Wikipedia 2016 dump with 2880838 entities) since the original Wikipedia 2010 dump is not available anymore. The ontology we considered is available on DBPedia[1]. We provide the script that builds the complete knowledge base and ontology in our repository. To generate fine-grained ontology knowledge bases, we describe the procedure (along with the code) in our repository. We must remind that our method does not include fined-grained entity recognition from the queries: we suppose this given as input in the data. For the implementation of graph kernels, we used the GraKeL software library (Siglidis et al., 2018).

4.2 Results:

We compare our methods with most performing baselines. Table 1 sums up our experimental results (averaged $\boldsymbol{P}$@1 is also referred as accuracy (Sun et al., 2015)). We included standard deviation of the accuracy, but could not include p-significance of our method, due to the difficulty to reproduce other baselines experiments (no source code is publicly available, or filtering method is not detailed). Our method yields remarkable accuracy on TAC09 dataset, CONLL/AIDA and TAC10 datasets. It performs better than any existing graph-based methods, outperforms all existing methods on two NIST TAC09 and TAC10, and is competitive with state-of-the arts methods on CONLL/AIDA. We also report impact of parameter K on average precision $\boldsymbol{P}$@1 (accuracy). Results are in Fig. 5. Low values of $\boldsymbol{K}$, corresponding to limited exploration, leading to decreasing accuracy. High values of K yield too many entity candidates and an imbalanced learning problem, resulting in a decrease of accuracy. Results are similar for $5 \leq \boldsymbol{K} \leq 10$, and allow a 2% to 3% improvement over filtering. As expected, the precision is strictly increasing with respect to $\boldsymbol{T}$ but the variation is bounded by 5% for $\boldsymbol{T} \in [3, 249]$.

Table 2: Computing times rounded to the minute. $Q = 1000$, $E = 2.8 \times 10^6$, $G \leq 200$, $\boldsymbol{K} = 7$, $\boldsymbol{T} = 249$. Setup 1: Single CPU with 32Gb Ram, 4-cores 2.40GHz. Setup 2: Distributed cluster with variety of 20 CPU processors equivalent to setup 1 (Spark/Hadoop technology)

Component	Complexity	Time (mn.)	
		Setup 1	**Setup 2**
Filtering	$\mathcal{O}(ME)$	196	15
Identification	$\mathcal{O}(Q\boldsymbol{KT}G)$	153	10

5 Conclusion

In this paper, we proposed a new methodology concerning the problem of named entity linking.

Table 1: Comparison with state-of-the art methods for $\boldsymbol{K} = 7$ and $\boldsymbol{T} = 249$. PGMs stands for probabilistic graphical model.

	Method	Nil detection	Train. size	$\boldsymbol{P}$@1 **(Accuracy)** ± std %		
				TAC09	TAC10	AIDA
(Ganea et al., 2016)	PGM	No	$\sim 10^6$	/	/	87.39
(Ganea and Hofmann, 2017)	PGM/DL	No	$\sim 10^6$	/	/	92.22
(Sun et al., 2015)	DL	No	$\sim 10^6$	82.26	83.92	/
(Yamada et al., 2016)	DL	No	$\sim 10^6$	/	85.2	93.1
(Yamada et al., 2017)	DL	No	$\sim 10^6$	/	87.7	94.3
(Globerson et al., 2016)	DL	Yes	$\sim 10^6$	/	87.2	92.7
(Sil et al., 2018)	DL	Not detailed	$\sim 10^6$	/	87.4	93.0
(Raiman and Raiman, 2018)	DL	Not detailed	$\sim 10^6$	/	90.85	**94.87**
(Guo et al., 2011)	Graphs	Yes	$\sim 10^4$	84.89	82.40	/
(Hoffart et al., 2011)	Graphs	No	$\sim 10^4$	/	/	81.91
Our method	Graphs	No	$\sim 10^3, 10^4$	**93.67**± 0.06	**94.70**± 0.05	93.56± 0.06

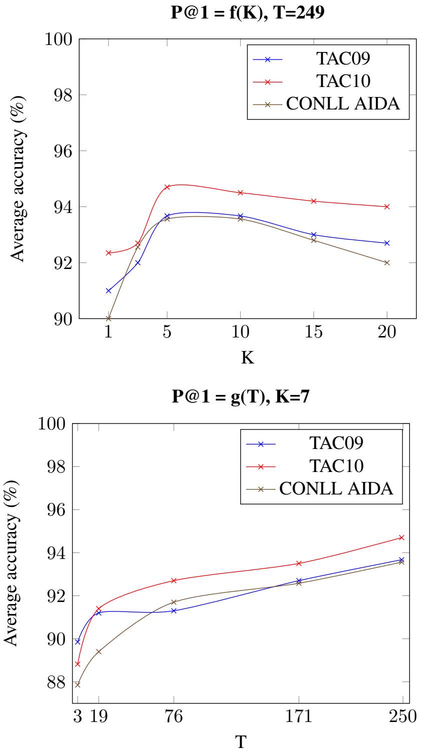

Figure 5: Impact of $\boldsymbol{K}$ and $\boldsymbol{T}$ on average $\boldsymbol{P}$@1.

First, we presented an entity filtering algorithm to return entity candidates that improves over trivial association rules. Then, each entity candidate is matched with a new representation built on a subgraph centered on their node. These representations use information contained in the ontology of the knowledge base. Finally, we used standard supervised learning to identify entities in the top candidates from filtering. We showed experimentally with standard datasets that named entity linking systematically improves over filtering using graph-based identification (for $2 \leq \boldsymbol{K} \leq 10$), up to 3%. Our experiments show that our method is competitive with state-of-the-art, and is stable with respect to $\boldsymbol{K}$ and $\boldsymbol{T}$, has a linear complexity and reasonable experimental computing time. Our linking system is relatively easy to implement, with few hyper-parameters. Last but not least, it does not require lots of data compared with deep learning to reach good experimental performance: only a few thousands of training samples were used to reach these results.

References

Ayman Alhelbawy and Robert Gaizauskas. 2013. Named entity disambiguation using hmms. In *2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)*, volume 3, pages 159–162. IEEE.

Ayman Alhelbawy and Robert Gaizauskas. 2014. Graph ranking for collective named entity disambiguation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 75–80.

A. Apostolico and C. Guerra. 1987. The longest common subsequence problem revisited. *Algorithmica*, 2(1):315–336.

Luigi P Cordella, Pasquale Foggia, Carlo Sansone, and Mario Vento. 2004. A (sub) graph isomorphism algorithm for matching large graphs. *IEEE transactions on pattern analysis and machine intelligence*, 26(10):1367–1372.

Silviu Cucerzan. 2007. Large-scale named entity disambiguation based on wikipedia data. In *Proceedings of the 2007 EMNLP-CoNLL*, pages 708–716.

Maud Ehrmann, Leonida Della Rocca, Ralf Steinberger, and Hristo Tanev. 2013. Acronym recognition and processing in 22 languages. *arXiv preprint arXiv:1309.6185*.

Octavian-Eugen Ganea, Marina Ganea, Aurelien Lucchi, Carsten Eickhoff, and Thomas Hofmann. 2016. Probabilistic bag-of-hyperlinks model for entity linking. In *Proceedings of the 25th International Conference on World Wide Web*, pages 927–938. International World Wide Web Conferences Steering Committee.

Octavian-Eugen Ganea and Thomas Hofmann. 2017. Deep joint entity disambiguation with local neural attention. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2619–2629.

Amir Globerson, Nevena Lazic, Soumen Chakrabarti, Amarnag Subramanya, Michael Ringaard, and Fernando Pereira. 2016. Collective entity resolution with multi-focal attention. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 621–631.

Yuhang Guo, Wanxiang Che, Ting Liu, and Sheng Li. 2011. A graph-based method for entity linking. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 1010–1018.

Dan Gusfield. 1997. *Algorithms on strings, trees and sequences: computer science and computational biology*. Cambridge university press.

Xianpei Han, Le Sun, and Jun Zhao. 2011. 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*, pages 765–774. ACM.

Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard Weikum. 2011. Robust disambiguation of named entities in text. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 782–792. Association for Computational Linguistics.

Heng Ji, Joel Nothman, Ben Hachey, et al. 2014. Overview of tac-kbp2014 entity discovery and linking tasks. In *Proc. Text Analysis Conference (TAC2014)*, pages 1333–1339.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. 2015. Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*, 6(2):167–195.

Xiao Ling and Daniel S Weld. 2012. Fine-grained entity recognition. In *Twenty-Sixth AAAI Conference on Artificial Intelligence*.

Qiaoling Liu, Josh Chao, Thomas Mahoney, Alan Chern, Chris Min, Faizan Javed, and Valentin Jijkoun. 2018. Lessons learned from developing and deploying a large-scale employer name normalization system for online recruitment. In *Proceedings of the 24th ACM SIGKDD*, pages 556–565. ACM.

David Nadeau and Satoshi Sekine. 2007. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26.

Giannis Nikolentzos, Polykarpos Meladianos, and Michalis Vazirgiannis. 2017. Matching Node Embeddings for Graph Similarity. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, pages 2429–2435.

Jonathan Raphael Raiman and Olivier Michel Raiman. 2018. Deeptype: multilingual entity linking by neural type system evolution. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Lev Ratinov, Dan Roth, Doug Downey, and Mike Anderson. 2011. 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*, pages 1375–1384. Association for Computational Linguistics.

Stephen Robertson, Hugo Zaragoza, et al. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389.

François Rousseau and Michalis Vazirgiannis. 2013. Graph-of-word and tw-idf: New approach to ad hoc ir. In *Proceedings of the 22Nd ACM International Conference on Information & Knowledge Management*, CIKM '13, pages 59–68, New York, NY, USA. ACM.

Giannis Siglidis, Giannis Nikolentzos, Stratis Limnios, Christos Giatsidis, Konstantinos Skianis, and Michalis Vazirgiannis. 2018. Grakel: A graph kernel library in python. *arXiv preprint arXiv:1806.02193*.

Avirup Sil, Gourab Kundu, Radu Florian, and Wael Hamza. 2018. Neural cross-lingual entity linking. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Yaming Sun, Lei Lin, Duyu Tang, Nan Yang, Zhenzhou Ji, and Xiaolong Wang. 2015. Modeling mention, context and entity with neural networks for entity disambiguation. In *IJCAI*, pages 1333–1339.

Ricardo Usbeck, Axel-Cyrille Ngonga Ngomo, Michael Röder, Daniel Gerber, Sandro Athaide Coelho, Sören Auer, and Andreas Both. 2014. Agdistis-graph-based disambiguation of named entities using linked data. In *International semantic web conference*, pages 457–471. Springer.

Davy Weissenbacher, Tasnia Tahsin, Rachel Beard, Mari Figaro, Robert Rivera, Matthew Scotch, and Graciela Gonzalez. 2015. Knowledge-driven geospatial location resolution for phylogeographic models of virus migration. *Bioinformatics*, 31(12):i348–i356.

Ikuya Yamada, Hiroyuki Shindo, Hideaki Takeda, and Yoshiyasu Takefuji. 2016. Joint learning of the embedding of words and entities for named entity disambiguation. *CoNLL 2016*, page 250.

Ikuya Yamada, Hiroyuki Shindo, Hideaki Takeda, and Yoshiyasu Takefuji. 2017. Learning distributed representations of texts and entities from knowledge base. *Transactions of the Association for Computational Linguistics*, 5:397–411.

Jin G Zheng, Daniel Howsmon, Boliang Zhang, Juergen Hahn, Deborah McGuinness, James Hendler, and Heng Ji. 2015. Entity linking for biomedical literature. *BMC medical informatics and decision making*, 15(1):S4.