CohEEL: Coherent and Efficient Named Entity Linking through Random Walks

Toni Gruetze[a], Gjergji Kasneci[a], Zhe Zuo[a], Felix Naumann[a]

[a] *Hasso Plattner Institute, Prof.-Dr.-Helmert-Straße 2-3, 14482 Potsdam, Germany*
Email address: {firstname.lastname}@hpi.de

Abstract

In recent years, the ever-growing amount of documents on the Web as well as in digital libraries led to a considerable increase of valuable textual information about entities. Harvesting entity knowledge from these large text collections is a major challenge. It requires the linkage of textual mentions within the documents with their real-world entities. This process is called *entity linking*.
Solutions to this entity linking problem have typically aimed at balancing the rate of linking correctness (precision) and the linking coverage rate (recall). While entity links in texts could be used to improve various Information Retrieval tasks, such as text summarization, document classification, or topic-based clustering, the linking precision is the decisive factor. For example, for topic-based clustering a method that produces mostly correct links would be more desirable than a high-coverage method that leads to more but also more uncertain clusters.
We propose an efficient linking method that uses a random walk strategy to combine a precision-oriented and a recall-oriented classifier in such a way that a high precision is maintained, while recall is elevated to the maximum possible level without affecting precision. An evaluation on three datasets with distinct characteristics demonstrates that our approach outperforms seminal work in the area and shows higher precision and time performance than the most closely related state-of-the-art methods.

Keywords: Entity Linking, Named Entity Disambiguation, Random Walk, Machine Learning

1. Named Entity Linking

Semi-structured and collaboratively created Web platforms, such as Wikipedia, have motivated a wide variety of research projects aiming at knowledge harvesting. For instance, large semantic knowledge bases, such as DBpedia [1] and YAGO [2], are based on the structured assets of Wikipedia, such as infoboxes and categories. However, harvesting implicit knowledge about entities in text without consistent structure, i.e., on websites or digital libraries, is still a major challenge. To address it, reliable techniques for *Named Entity Detection* and *Disambiguation* in natural language text are needed. Especially the disambiguation process is a critical step; it involves the correct grouping of textual mentions of the same real-world entity. If these groups are linked to corresponding entities in a knowledge base, the process is referred to as *Named Entity Linking* (NEL). Quote 1

> U.S. District Court, New York: Judge Richard Berman expects to rule this week on Brady's four-game suspension appeal. Will the NFL finally prevail over Brady in the Deflategate saga?

Quote 1: Running example of a Named Entity Linking task with emphasized entity mentions.

shows an example text of a news article about the Deflategate scandal. The mentions of different entities within the text (e.g., *U.S. District Court*) are emphasized. Note that only named entities have been highlighted, where general entities, such as, concepts (e.g., judge) or temporal expressions (e.g., week) have not.

The task is to link these highlighted mentions to corresponding knowledge base entities. Dredze et al. identify three key challenges for NEL [3]:

(i) Name variations occur for various reasons, for instance to reduce the length of the actual mention, such as *"NFL"* as abbreviation for *"National Football League"*.

(ii) The absence of entities is another important challenge. For instance, the mentioned *"Deflategate"* is not covered by Wikipedia versions from 2014 or earlier. A linking algorithm has to cautiously handle such mentions and should not link them to similarly named entities, such as the data compression algorithm *DEFLATE*.

(iii) Furthermore, the entity ambiguity concerns cases where different entities are referred to by the same name, such as *"Brady"*. An algorithm with the goal to link this mention to an entity from Wikipedia has to choose between various geographical entities, such as a city in Texas and a village in Nebraska, hundreds of Persons (e.g., the famous American football quarterback and 4 time super bowl winner, the award-winning film director and producer, the American judge and Associate Justice, ...).

There are various application areas that could benefit from reliable NEL techniques. Digital libraries are usually managed by information systems that enable textual keyword search. However, state-of-the-art keyword-based search engines are not able to deal with name variations or ambiguity. Hence, previously identified entity mentions might enable users to identify documents containing information about specific entities. Furthermore, relationships between entities can be inherited from co-occurrences in documents and aggregated into large semantic relationship graphs. The identified entity mentions as well as the relationships between them might serve as a starting point for topic-based clustering and document classification to enable a categorization of the document collection.

For Web retrieval tasks, such as person or product search, reliable disambiguation tools could enable the clustering of results, so that each cluster represents only one real-world entity, allowing the user to focus on the documents of interest [4]. Encyclopedic and scholarly search as well as question answering approaches would immensely benefit from reliable NEL techniques, by using the entity links in a document corpus to identify relevant texts [5]. Several academic projects in the realm of artificial intelligence, e.g., Read the Web [6] or YAGO-NAGA [7], as well as different scientific workshops and benchmarks, such as the TAC knowledge-base-population track or the ERD challenge [8], or GERBIL benchmark repository [9], have highlighted the importance of NEL for bootstrapping relationship extraction from natural language texts.

The effectiveness of those applications highly depends on the quality of the named entity detection and disambiguation step. This quality is in turn strongly influenced by the type of collected documents. For instance, digital libraries containing scientific publications share different text characteristics than collections containing news articles, social network posts, music reviews, or even entire books of fiction. Text structure, length, and topic variability are decisive for the vocabulary and the contextual information contained in the text. In our experimental evaluation, we show that most state-of-the-art algorithms lack precision, with values between 30% and 80% depending on the text type, only expensive coherence reasoning strategies can lead to more reliable results. The approach presented in this paper focuses on the efficient retrieval of reliable, i.e., high precision, alignments of approximately 90%.

Furthermore, the efficiency of NEL algorithms is a decisive factor that is usually neglected in current state-of-the-art algorithms. For instance, the runtime of NEL algorithms is important for use cases dealing with large text corpora, such as digital libraries with millions of texts, e.g., the Internet Archive, arXiv.org, or Google Books. NEL algorithms with a runtime of

minutes per document would take two years to annotate such corpus. Another efficiency bound use-case are streaming applications. Given an infinite stream of incoming documents to be annotated, e.g., blog posts, news articles, or short messages, an NEL system has to process the texts approximately in the time window between two consecutive texts. Our contributions are the following:

1. We propose CohEEL, a supervised two-step model that enables the reliable NEL results. The model is designed to incorporate a knowledge base and *(i)* automatically adapts to different input text types, *(ii)* incorporates arbitrary mention-scoring functions, and *(iii)* considers known relations between mentioned entities within documents (i.e., coherence).
2. Based on the CohEEL model, we discuss a concrete configuration that provides reliable and coherent entity alignments to the knowledge bases YAGO and Wikipedia with a precision of approximately 90%.
3. Finally, we provide an exhaustive experimental comparison of our algorithm with state-of-the-art methods with respect to *(i)* linking quality and *(ii)* runtime.

2. Problem and Prior Work

In this section we introduce basic terms and discuss the related research for the Named Entity Linking problem. A specific discussion of the competitors used in the evaluation can be found in Section 5.

2.1. The Named Entity Linking Problem

The term named entity stands for a real-world instance and is distinct from a concept, which represents a category of named entities. Hence, NEL differs from the Wikification process, introduced in [10], which deals with the automatic identification of links to Wikipedia articles (i.e., both concepts and instances). It is also distinct from the word-sense disambiguation, which identifies the sense of a word in a sentence and is not focused on named entities [11, 12]. A prerequisite for NEL is the discovery of textual mentions that might refer to named entities.

Definition 1. *A **mention** $m = (D, p)$ is a textual reference occurring in position p within document D and referring to some named entity.*

In the following we assume that such Named Entity Recognition (NER) techniques are readily available; we employ the Stanford NER tool [13] to discover mentions. Please note, that some related work follows another definition of NEL, that includes the NER step [14].

The subsequent NEL task for a given set $M(D)$ of mentions in document D and a set $E(K)$ of named entities in knowledge base K is to find a partial mapping $f_a : M(D) \to E(K)$, such that if $m \in M(D)$ is mapped to $e \in E(K)$ then indeed m refers to the real-world entity represented by e. For instance, the mention *"Richard Berman"* shall be mapped to the Wikipedia article of Richard M. Berman. The alignment function f_a remains partial, because in some cases K might not contain the entity that is referred to by a mention. For a total function, f_a can map the mention to a designated entity *NIL* in such cases. For better readability we use E' to refer to the extended knowledge base entity set $E(K) \cup \{NIL\}$, and E to refer to $E(K)$. Furthermore, we use M to address the set of mentions of a document $M(D)$ and refer to a mapping from mention m to entity e as an *alignment*, denoted (m, e).

So far, mentions are defined only as the occurrences of named entities in a document by means of unique positions. However, each mention m within a document covers different aspects of information about the entity it refers to, most importantly the surface and the context. In related research, these aspects are commonly referred to as local ranking features, because they are independent of other entity mentions in the document [15].

The **surface** $srfc(m)$ of a mention m is the actual word or phrase used to identify the referenced real-world instance in the text. For instance, surfaces in Quote 1 are $srfc(m_1)$= *"U.S. District Court"* or $srfc(m_2)$ = *"New York"*. The **context** $ctx_n(m)$ of mention m is a multi-set of n consecutive terms surrounding the mention (including its surface). For efficiency and the fact that the semantic relatedness between a term and a mention decreases with

increasing distance, the size n of the context ctx_n is typically much smaller than the document size ($n \ll |D|$). The context is usually implemented as a sliding window over the document, the mention typically occurs in the center of the window. For instance, the context of size $n = 12$ for the mention of *"Richard Berman"* is $ctx_{12}(m_4) = \{$ *"District"*, *"Court"*, *"New"*, *"York"*, *"Judge"*, *"Richard"*, *"Berman"*, *"expects"*, *"to"*, *"rule"*, *"this"*, *"week"*$\}$.

2.2. Related Research

Traditional NEL approaches build on different combinations of surface and context feature scores to find the most appropriate entity for each mention in the knowledge base [3, 15–19]. Zuo et al. extend this method by aggregating decisions from an ensemble of different ranking functions based on sampled subset of the context as well as the surface ranks [20].

Additionally to these mention-local characteristics, many NEL are based on document-global features [15]. Because named entities within documents are usually mentioned in the context of other, often related entities, the alignments of mentions to entities in a knowledge base K should not be performed independently. Assuming that K covers salient relationships between entities, reasoning about *coherent groups* of entities may help resolve ambiguities in a joint entity linking process [21]. For instance, Quote 1 combines two topics, one surrounding the United States District Court for the Southern District of New York and Judge Richard M. Berman and another one surrounding the American football fraud scandal "Deflategate" caused by New England Patriots quarterback Tom Brady. However, finding the most coherent joint alignments of the $|M|$ mentions that co-occur in a document, can lead to an NP-hard problem [21]. Therefore, state-of-the-art approaches resort to a pair-wise coherence-based reasoning and alignment strategy. Milne and Witten identify coherent entity sets by computing the relatedness of ambiguous candidates (i.e., several entities for one mention) to other unambiguous entity candidates in the document [22]. This is done by reasoning about the relatedness of candidate entities based on Wikipedia articles that are hyperlinked to the candidates' articles. Because the relatedness score is based only on the coherence with respect to unambiguous entities, the approach does not require exhaustive reasoning over the quadratic problem space (i.e., all combinations of candidate pairs). Further related work targets to improve the performance of the relatedness score by applying Jaccard index, cosine similarity, or pointwise mutual information [15, 19]. Ratinov et al. introduce a model that is based on a linear combination of 4 local and 9 pairwise global features to rank mention candidates [15]. In a second step, the top-ranked candidate per mention is verified by using additional features, retrieved from the scored candidate list. For instance, the improvement of the ranker confidence with respect to the second-best candidate or the entropy of the surface probability score. Du et al. employ similarity measures that capture the average pair-wise proximity between candidate entities in the knowledge graph, as well as their average pair-wise conceptual similarity by means of the lowest-common-ancestor classes [23]. Kulkarni et al. attempt an efficient solution for finding collective annotations of entities in documents by transforming the problem to a local hill climbing algorithm [24]. Hoffart et al. exploit the key-phrase- and hypernymy-based relatedness between the candidate entities in the knowledge base to jointly link mentions occurring in the same document to the candidate entities [21, 25].

In contrast to the pairwise coherence reasoning, Han et al. approach the problem by means of random walks on the "referent graph", where each entity is assumed to share portions of its alignment probability with its knowledge base neighbors [26]. Thus, the algorithm estimates the coherence of the entire solution at once and exploits the complete graph structure, instead of aggregating the values over different candidate pairs. However, due to exhaustive candidate selection in the algorithm, the holistic graph reasoning has to be performed over a large set of candidate entities, i.e., yielding millions of candidates for mid-sized documents and high runtimes. Agirre et al. tackle this issue by applying a more conservative candidate selection strategy (using a surface dictionary) and a sparser knowledge base graph [27]. By selecting only "reciprocal links" from Wikipedia, the number of edges can be reduced by an order of

magnitude, while improving the NEL accuracy significantly. Moro et al. introduce a greedy algorithm that is based on the knowledge base BabelNet and simultaneously disambiguates named entities as well as word senses [28]. The knowledge base is curated using a random walk approach that removes seldom hit neighbors (i.e., under 100 times). A document is initially represented as a graph containing all possible interpretations of entity mentions as well as word references and the direct relationships among them. To disambiguate text fragments mentioning entities and word senses, the algorithm iteratively removes the weakliest connected interpretation of the most ambiguous the mention to build a dense subgraph representation of the document. Following this iterative process, the densest subgraph is defined as the graph with highest average degree. The remaining candidates of the densest subgraph are scored based on their degree in the dense graph (w.r.t. fragments as well as other candidates). The disambiguation result is retrieved by selecting the highest scored meaning for each fragment. The authors furthermore suggest the application of a threshold to remove uncertain meanings. In contrast to this proceeding, Guo and Barbosa introduce a greedy approach that is iteratively increasing the solution set of the disambiguation task [29]. The algorithm represents entities and documents as "semantic signatures". A signature is defined as the stationary probability distribution of a random walk. For entities, the random walks are performed on the first- and second-degree neighbors, whereas the neighborhood size is narrowed by removing neighbors with degree of under 200. The document neighborhood is built according to the already disambiguated entities as well as candidate entities in the document. To improve the runtime of the algorithm, the candidate set per mention is limited to 20 entities that are selected based on a surface and a context compatibility score. Mention candidates are disambiguated iteratively: starting with the mention with the smallest candidate set, the algorithm selects an entity based on a similarity measure that is based on the KL-Divergence between document candidate. For the next iteration, the selected candidate is added to the set of disambiguated entities.

The named entity linking model introduced in this work is based on a two-phase strategy: a classification and a random-walk-based reasoning phase. Due to a supervised learning approach, our model is able to adequately prune the candidate space and to adapt to different types of texts to provide steadily high precision results. The approach does not rely on specific attributes of any knowledge base and is neither based on handcrafted thresholds, nor manually optimized combinations of different scores (e.g., neighborhood relevance degree or alignment uncertainty thresholds). This initial classification enables efficient coherence reasoning over a strongly reduced entity set in the iterative exploitation phase using random walks.

3. Identifying Reliable and Coherent Entity Alignments

CohEEL (Coherent and Efficient Named Entity Linking) is an abstract model for reliable and coherent entity linking. It makes use of a knowledge base that provides the set of entities to be discovered in arbitrary documents. We therefore assume that the knowledge base covers relevant inter-entity relations that allow reasoning about entity interdependencies (such as links in Wikipedia, co-author relationships or citations in bibliographic knowledge bases).

Figure 1 gives an overview of our approach. The first step, *Candidate Classification*, is based on different supervised learning methods and is responsible for an efficient mention-wise proposal of alignments by:

- proposing high-precision *seed alignments* S, i.e., the corresponding mentions are linked to these entities, and by
- proposing high-coverage *candidate alignments* C, i.e., more reasoning is needed about the linking of the mentions with respect to these entities.

The candidate classification is essential to prune the candidate space and perform a coherence reasoning efficiently.

The second step, *Judicious Neighborhood Exploration* (JNE), cautiously extends the seed set with high-coverage candidate entities (from the first step)

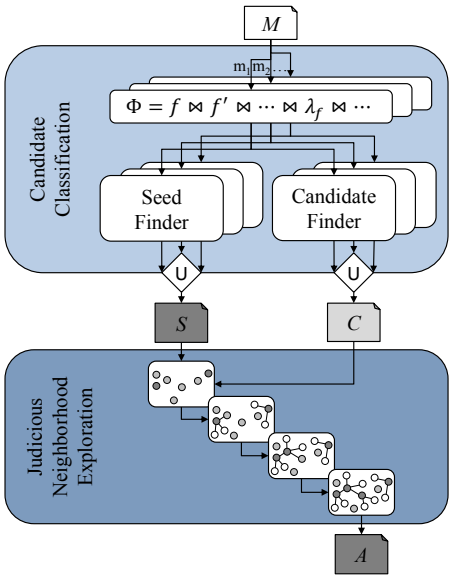


Figure 1: Two-phase model of CohEEL creating a set of alignments $\boldsymbol{A}$ for a given set of mentions $\boldsymbol{M}$ from one document.

that are related to the seed entities, i.e., candidates contained in the seed communities within the knowledge base. Hence, this step improves the recall significantly while providing a comparable precision of the found alignments. Both steps are explained in detail in the following sections.

3.1. Candidate Classification

The candidate classification step is an important step towards an efficient coherence reasoning. In theory, a mention can be linked to each entity in the knowledge base. However, only a small amount of entities cover features making them promising candidates. A candidate classifier is meant to prune the candidate space to a viable size. For the mention-wise proposal of entities, the rating for the match of a mention and a knowledge base entity is computed by means of various scoring functions.

3.1.1. Scoring Functions

Definition 2. *A **scoring function** $f : M \times E' \to \mathbb{R}$ quantifies the affinity between a mention $m \in M$ to a knowledge base entity $e \in E'$.*

Intuitively, the outcome of all scoring functions $f(m, e)$ should be correlated to the actual probability that m represents the real world entity e in the given text. First-order scores are derived from various features of mentions and entities, e.g., the compatibility of the mention surface and the entity name, or the similarity between mention context and the entity information (see discussion of mention-local and document-global characteristics in Section 2). The set of actually used scoring functions mainly depends on the information provided by the knowledge base. For instance, contextual similarity functions can be applied only for knowledge bases containing textual entity features. For instance, the context $ctx_{12}(m_4)$ of mention *"Richard Berman"* might yield important indications that the referred entity is judge Richard M. Berman and not the more prominent lobbyist Richard B. Berman. Many scoring functions have been discussed in previous work. Ratinov at al. introduce thirteen different scoring functions based on mention-local as well as document global features [15]. Piccinno and Ferragina provide more than 70 features in their WAT framework [19]. Due to performance orientation of the candidate classification step scores have to be retrieved efficiently (i.e., without complex iterative coherence reasoning phases). For instance, in our experiments, we limit the basic scoring functions to a surface and a context compatibility score, as well as an efficient relatedness measure (see Section 4.2).

Additionally, CohEEL is able to incorporate *higher-order scoring functions* that quantify the affinity between mention $m \in M$ and knowledge base entity $e \in E'$ based on a scoring function f, i.e., $\lambda : ((m, e), f) \mapsto y \in \mathbb{R}$, for short: $\lambda_f(m, e)$. An example of a higher-order score might be given by a function λ_f^{rank}. It is defined as the rank of entity e in the candidate list for m that is ordered by the scores of a first-order function f. In this case, the rank of a candidate entity is a feature with additional implicit mention-local information, i.e., how many other entities have higher f scores, which can be useful for the linking decision. Ratinov et al. apply two higher-order scores to remove uncertain alignments from the solution set [15]: the entropy of prior probability (surface compatibility) and a confidence gain of the best

candidate compared to the second match.

To not restrict the model to a specific knowledge base, we have to make sure that CohEEL relies neither on specific attributes of the knowledge base, nor on specific scoring functions. Instead, it learns to incorporate a combination of arbitrary functions (such as the ones above) and automatically derives a good feature combination based on a representative training sample to judge further alignments. Hence, in contrast to other approaches, CohEEL is not based on handcrafted rules or thresholds that combine a specific set of scoring functions. This is important, because a one-size-fits-all solution is unrealistic for the broad variety of available document types. The diversity of the employed scoring functions allows CohEEL to judge different relatedness aspects between mentions and entities.

As stated earlier, CohEEL applies candidate selection to prune the alignment candidate space for the JNE phase. For this, we cast the selection into a binary classification problem. Based on a training set, two selection models are learned, a cautious one, which is a finds for high-precision *seed alignments*, and a brave one, which produces high-coverage lists of *candidate alignments*. Note, the evidences gained by various scoring functions enable the training of classifiers for well-founded decisions that are superior to thresholds or rules based on single features.

3.1.2. Seed Finder

With the help of different first $\{f, f', \ldots\}$ and higher order scoring functions $\{\lambda, \lambda', \ldots\}$, CohEEL is able to generate a set of labeled feature vectors $\vec{\phi}$ per alignment:

$$\vec{\phi}_{(m,e)} \mapsto \langle\, f(m,e)\ , \lambda_f(m,e)\ , \lambda'_f(m,e)\ , \ldots, \\ f'(m,e)\ , \lambda_{f'}(m,e), \lambda'_{f'}(m,e), \ldots, \\ f''(m,e), \ldots\ \rangle$$

Based on a set of training documents (including correct alignments), a binary classifier is trained that assigns class-labels to an arbitrary alignment (m, e), namely the label *correct* (positive) or *incorrect* (negative). To influence how cautious or brave the classifier assigns class labels to alignments, the concept of loss matrices is applied. During training, the classification model is adjusted to minimize the expected loss defined as the sum of costs produced over all four classes of the confusion matrix [30]. In CohEEL, proper classifications are not punished and the loss matrix ($\mathbf{L}$) is used to weight the different types of misclassifications (type I/II errors). The costs for *true positives* ($\mathbf{L}_{tp}$) and *true negatives* ($\mathbf{L}_{tn}$) are set to zero.

The first classifier in CohEEL is the Seed Finder. It is configured to identify at most one entity per mention with high precision.

Definition 3. ***Seed alignments*** *are reliable, non-contradicting alignments, i.e., with a high probability of being correct.*

The entities of seed alignments (seeds for short) are subsequently used to deduce further correct alignments. For instance, the mention *"Deflategate"* in Quote 1 might be easily linked with the eponymous Wikipedia article, because no other articles are related to this name. Furthermore, the Seed Finder might be able to detect that *"Richard Berman"* refers to the same entity as the article of U.S. District Court judge Richard M. Berman but not the disputed lobbyist Richard B. Berman. This can be achieved by considering surface and specifically context compatibility scores (e.g., the context $ctx_{12}(m_4)$ mentioned in Section 2.1 is very similar to the judge's Wikipedia article). Note that this definition of seeds strongly differs from previous work where unambiguous entities were used instead (e.g., for the relatedness definition of Milne and Witten [22], see Section 4.2). For instance, there is only one matching Wikipedia article available for the mention *"Deflategate"*. Following our definition, an ambiguous entity might be chosen as a seed if the scores indicate one entity being a good choice, e.g., for the mention of *"Richard Berman"*. Furthermore, unambiguous entities might be discarded as seed if the contextual score is very low.

The transformation of the original problem into a classification problem, allows us to configure CohEEL such that it yields high precision. This is achieved by applying loss matrix $\mathbf{L}^s$, which defines higher costs

for *false positive* alignments ($\mathbf{L}^s_{fp}$) than for *false negatives* ($\mathbf{L}^s_{fn}$). In this way we achieve a bias of the classifier decisions towards higher precision. For instance, the cost model with a ratio of $\mathbf{L}^s_{fp}/\mathbf{L}^s_{fn} = 9$ punishes *false positives* nine times stronger than *false negatives* and guide the learner to create more reliable classifiers with an expected precision $\mathbb{E}(|P_{min}|)$ of around 90% or more. Hence, we can define the cost model as follows:

$$\mathbf{L}^s_{fp} = \frac{\mathbb{E}(P_{min})}{1 - \mathbb{E}(P_{min})}, and\ \mathbf{L}^s_{fn} = 1$$

However, the classification results (i.e., in terms of confusion matrix) are not directly transferable to the Seed Finder performance, because the classifier might identify several candidates per mention as *correct*. These contradicting alignments are undue, hence, the Seed Finder is required to select at most one entity per mention. This problem becomes more severe in case of lower costs ratios $\mathbf{L}^s_{fp}/\mathbf{L}^s_{fn}$, that let the classifier make braver decisions. To resolve these contradicting cases, we decided to reject all of these entities. There are various other strategies (e.g., randomly pick one entity, select the entity with highest scores, ...), however, our implementation prevents a decrease of precision of the seed alignments, due to the addition uncertain alignments.

3.1.3. Candidate Finder

In contrast to the Seed Finder, the goal of the Candidate Finder is to identify a set of entities C_m per mention m.

Definition 4. *A **candidate alignment** set C_m for a mention m contains with high probability the alignment with the actual entity e referred to by m.*

The Candidate Finder is configured to create a candidate list with a high coverage rate (recall). A straightforward implementation would be an "always *correct*" classifier that declares all entities from the knowledge base as candidates ($C_m = E$). However, for practical reasons such a classifier would be useless, because the following JNE phase would be costly and inaccurate. Hence, another important goal is to prune the candidate lists per mention without losing correct entities. Given a specific maximal expected candidate set cardinality $\mathbb{E}(|C_m|)$ for the Candidate Finder, the appropriate loss matrix $\mathbf{L}^c$ is given by

$$\mathbf{L}^c_{fp} = 1, and\ \mathbf{L}^c_{fn} = \mathbb{E}(|C_m|) - 1$$

and maximizes the number of *true positives* while limiting the average candidate set size to $\mathbb{E}(|C_m|)$. The intuition behind this cost model is simple: to minimize the risk of missing a correct candidate (*fn*), the classifier accepts n-times more incorrect candidates (*fp*), where $n = \mathbb{E}(|C_m|) - 1$. Note that the maximum expected candidate set cardinality $\mathbb{E}(|C_m|)$ has also a large influence on the runtime behavior of the second phase of the algorithm (i.e., neighborhood exploration). The influence of the $\mathbb{E}(|C_m|)$ on the runtime of CohEEL is further discussed in the next section.

For instance, given the *"U.S. District Court"* mention surface in Quote 1, the number of compatible Wikipedia articles exceeds 90. Considering the mention context, containing the terms "New" and "York", the Candidate Finder is might be empowered to prune the candidate set to the four courts of New York, namely the Court for the Northern, Eastern, Southern, and Western District of New York.

In the next step, CohEEL leverages the seeds to establish new alignments to candidate entities such that the coherence (in terms of semantic relations) between all the entities in the alignments is maximized.

3.2. Judicious Neighborhood Exploration

Typically, documents contain only few distinct groups of topically related entities. The goal of neighborhood exploration is to identify candidates that are semantically related to the identified seeds for a given document. The prerequisite for this step is a knowledge base that covers relevant inter-entity relations. NEL algorithms are able to deduce further alignments from a given set of entity assignments by exploiting such knowledge. However, there is a trade-off between quality and completeness: the more aggressive the deduction process is, the more additional alignments can be found but the higher the risk of finding incorrect alignments. Furthermore, reasoning

over a set of entity candidates for a large document might be computationally expensive.

To efficiently find coherent candidate entities, we apply an iterative exploration model that in each iteration uses the current set of seeds to efficiently discover the most *coherent* candidate, and adds it to the seed-set for the next iteration. To measure the coherence between seeds and candidates, we apply a Random Walk with Restarts algorithm (RWR) [31]. Random walks on graphs estimate the probability $p_s^{(t)}(n)$ of finding a random walker at vertex n at time t after starting from vertex s. The steady state probability $p_s(n)$ can be viewed as a measure of affinity between s and n. For us, $p_s(n)$ represents the probability of finishing at entity n after following knowledge base links starting from entity s and thus represents the proximity between s and n. In other words, we define coherence as probability that a random walker, that starts at a seed entity (e.g., Wikipedia article) and follows knowledge base links (hyperlinks to other articles), finishes at a candidate. The application of random walks has the following benefits: (i) it captures the structure of the complete neighborhood graph, and (ii) in comparison to traditional graph distances, it captures all facets of relationships between two nodes in the graph. The steady state probabilities of a random walk corresponds to the eigenvector $\vec{\mathbf{p}}$ of the neighborhood transition matrix and can be efficiently approximated using the Power method [32].

Algorithm 1 outlines the details of our approach. Given a set S of seed alignments and a set C of candidate alignments, the approach tries to identify a set of coherent alignments A. The functions $M(A)$ and $E(A)$ retrieve the union of all mentions/entities within one set of alignments, and $N_G(A)$ retrieves the transition matrix of the neighborhood of already disambiguated entities in the knowledge base. The neighborhood graph is a subgraph of the knowledge base, containing all first and second order neighbors of disambiguated entities. More details about the construction are given in the next subsection.

Between Lines 4 and 22 the iterative exploration is executed by extending the A with the alignments of the most promising candidate entity e_i. It finishes if no additional alignment could be added. In

Algorithm 1: Judicious Neighborhood Exploration

Data: seed alignments $S = \{s_1, s_2, ...\}$,
candidate alignments $C = \{c_1, c_2, ...\}$,
where $S, C \subset \mathcal{P}(M \times E)$
Result: alignments $A = \{a_1, a_2, ...\}$, where $A \subseteq S \cup C$

1 $A := S$;
2 $A' := \emptyset$;
3 /* running iterative exploration */
4 **while** $A' \neq A$ **do**
5 /* remove candidates w/ covered mentions */
6 $C := C \setminus (M(A) \times E)$;
7 /* transition matrix of the knowledge base neighborhood of $E(A)$ */
8 $\widetilde{\mathbf{N}} := N_G(A)$;
9 /* get starting vector w/ same start probability for each entity in A */
10 $\vec{\mathbf{s}} := \begin{cases} \mathbf{s}_i = |E(A)|^{-1} & \text{if } e_i \in E(A) \\ \mathbf{s}_i = 0 & \text{otherwise} \end{cases}$
11 $\vec{\mathbf{p}} := \vec{\mathbf{s}}$;
12 $it := 0$;
13 /* power iterations until convergence */
14 **repeat**
15 $\vec{\mathbf{p}}' := \vec{\mathbf{p}}$;
16 $\vec{\mathbf{p}} := (1-\alpha)\widetilde{\mathbf{N}}\vec{\mathbf{p}} + \alpha\vec{\mathbf{s}}$;
17 $it := it + 1$;
18 **until** $f_\theta(\|\vec{\mathbf{p}} - \vec{\mathbf{p}}'\|, it)$;
19 $A' := A$;
20 /* add all alignments of candidate entity e with highest affinity ($\max \mathbf{p}_e, \mathbf{p}_e > 0$) */
21 $A := A \cup (C \cap (M \times \arg\max_{e \in E(C)} \mathbf{p}_e))$;
22 **end**

Lines 6-12 all necessary variables for the RWR are initialized, and in Lines 11 to 18 the power iterations are executed to identify the eigenvector $\vec{\mathbf{p}}$ of the neighborhood graph until the convergence criterion is fulfilled. As parameters for the random walk, we suggest to use the established values of $\alpha = 0.15$ and $\theta_c = 10^{-8}$. Where the former parameter influences the skew of steady state probabilities towards the seed entities, the latter one mainly influences the

runtime of the random walk. We discuss the convergence separately.

The eigenvector of the transition matrix represents the steady state probabilities of all candidates in the neighborhood after starting a RWR from A. Consequently, Line 21 extends the set of seeds for the next RWR by the most probable candidate alignments, i.e., those with highest affinity from the prior alignments. This step can be understood as a maximum a posteriori estimation of the candidates entities. Note, we do not use any confidence threshold to narrow the alignments, but add each valid candidate having a non-zero steady state probability ($\mathbf{p}_e > 0$). A probability of zero means that the entity cannot be reached during a random walk and is thus unrelated.

To finally create a complete result set for a document, the alignments A are extended by additional links to *NIL* alignments, if *(i)* neither the Seed nor Candidate Finder is able to identify a promising entity for mention m or *(ii)* the JNE phase did not identify any relevant connection from any alignment entity $E(A)$ to an entity in the candidate list C_m for mention m. Specifically *(ii)* is rather conservative, but ensures a high linking quality (we do not link candidates without strong indication) and underlines the focus on high precision results. A braver version could choose the candidate based on candidate classifier scores to increase recall while adding potential false positive alignments In the following, *NIL* alignments are treated as alignments to entities with no representation in the knowledge base.

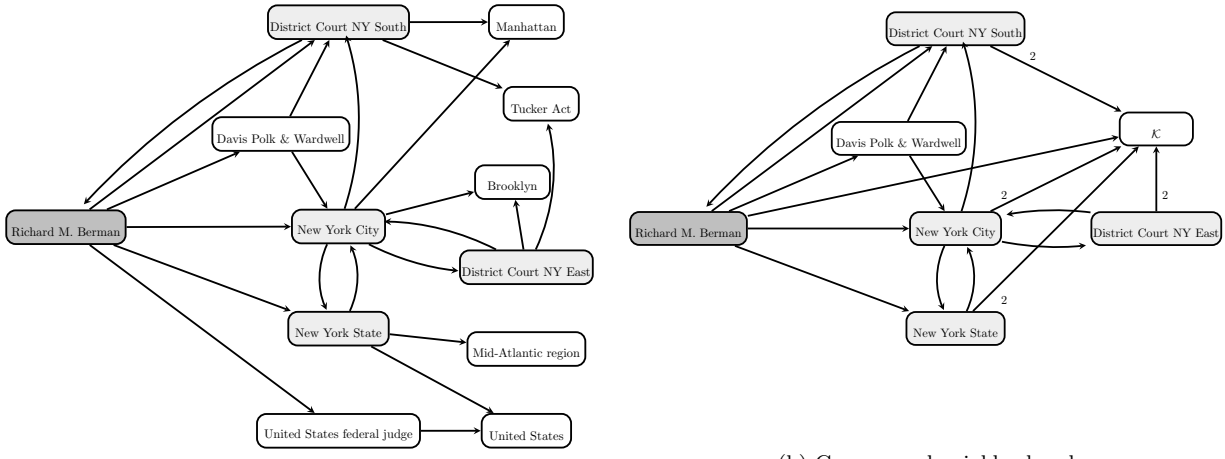
3.2.1. Neighborhood construction

The neighborhood graph is defined as the subgraph of the knowledge base, containing all direct neighbors of all relevant entities (i.e., $E(A \cup C)$) as well as the neighbors of second order (neighbors of neighbor) of disambiguated entities in the document. Figure 2a depicts the reduced neighborhood of the legal topic discussed in Quote 1. The mention *"Richard Berman"* is represented by the dark gray entity Richard M. Berman aligned by the Seed Finder (see discussion in Section 3.1.2). The mentions *"U.S. District Court"* and *"New York"* are not disambiguated yet. However, as discussed in Section 3.1.3 the Candidate Finder identified four potential alignments for the U.S. Court (e.g., out of 94 Courts listed in Wikipedia): U.S. District Courts for the Northern, Eastern, Southern, and Western District of New York, as well as two candidates for mention New York: the U.S. State and the populous City. Because two of the four U.S. District Court entities (Northern and Western District) are not part of the second order neighborhood of Richard M. Berman, we do not show them in the graph. These two candidates would end up with a steady state probability of zero. Furthermore, we added only a subset (at most 2) of neighborhood entities (white nodes) for seeds or candidates for brevity and clarity of the figure.

To be able to perform the random walk more efficiently, we compress the neighborhood graph without influencing the results of the RWR ranking (see Figure 2b). Because the ranking is only relevant for candidate entities, we can merge all "dead ends" in the graph to a new default node $\mathcal{K}$. Dead ends are vertices that do not refer (directly or indirectly) to any relevant entity. They are either sinks or exclusively refer to other dead ends.

To retain the degree of the remaining edges, we represent the neighborhood as a multigraph where the number of edges to the dead end node corresponds to the amount of merged neighbors. This is important to maintain the structural properties of the graph for the random walk. The $\mathcal{K}$-node can be interpreted as a state of a surfer who has left the direct neighborhood graph and is browsing the rest of the knowledge base. We are aware of the fact, that the probability of a random walker returning from $\mathcal{K}$ to the graph is not zero, but would be correlated to the (global) in-degree of the neighborhood entities. However, because we do not want to favor hub entities in the neighborhood (i.e., New York has a much higher in-degree than U.S. Courts), we decided to ignore this fact. The return of a surfer to the neighborhood is modeled as a restart of the surfer from one of the already aligned entities $E(A)$.

In a final step, we represent the neighborhood graph as stochastic transition matrix $\widetilde{\mathbf{N}}$. It is built based on the matrix representation of the graph. The matrix contains integer elements that represent the number of edges between two nodes. To guarantee convergence of a random walk, we have to prevent the



(a) Knowledge base neighborhood

(b) Compressed neighborhood

Figure 2: Two graph representations during neighborhood graph construction of the JNE for entities forming the legal topic subgraph of Quote 1.

graph from containing dangling nodes or being periodic. For now, at least $\mathcal{K}$ is a dangling node. The easiest way to ensure both properties is by adding "stalling edges". Broadly speaking, a stalling edge models a pausing of the surfer on the same article. In a final step, the matrix is normalized such that the entries contain actual transition probabilities. An executed random walk on $\widetilde{\mathbf{N}}$ would yield a new alignment from *"U.S. District Court"* to the entity of the Court in the Southern New York District ($\mathbf{p}_e = 5.8\%$) that is ranked above the Eastern District Court ($\mathbf{p}_e = 1.0\%$), as well as the City of New York ($\mathbf{p}_e = 5.7\%$) and New York state ($\mathbf{p}_e = 4.4\%$).

3.2.2. Convergence

The convergence of a RWR is usually modeled by observing the changes of the probability vector in the last power iteration $\Delta\vec{\mathbf{p}} = \|\vec{\mathbf{p}} - \vec{\mathbf{p}}'\|$. If the changes iterations fall below a tolerance level ($\Delta\vec{\mathbf{p}} \leq \theta_c$), the RWR is defined to be converged.

Lemma (Rate of Convergence [32])**.** *Given a non-bipartite graph, a Random Walk with Restart – with a restart probability α and a convergence tolerance level θ_c – converges approximately after $\log_{1-\alpha}(\theta_c)$ power iterations.*

For the common values $\alpha = 0.15$ and $\theta_c = 10^{-8}$, a RWR converges on average after roughly 113 iterations. To further reduce the number of matrix operations, we use the following trick: Because we employ the steady state probabilities of the random walk to rank candidate entities, the random walk could already been terminated after the candidate ranking is steady. However, for different document types, the number of iterations leading to a stable selection of the most probable candidate entity might differ (i.e., due to varying amount of ambiguous mentions and candidates per mention). Therefore, we use characteristics exploited from data used to train Seed and Candidate Finder (see Section 4.4). We deduce the number of iterations θ_{it}, after which the top-ranked candidate entity is assumed to be stable. To learn the threshold, we propose the following heuristic: The JNE phase is executed for all documents of the training set until convergence. For each document, we derive the iteration it_s at which the top-ranked entity is stable. The value of θ_{it} is then set to the 99^{th}-percentile over all it_s of the training set documents. The 99^{th}-percentile is used to find a reliable threshold that does not depend on outliers. We are aware of the fact that it_s might decrease the linking quality of our approach, but we clearly favor the runtime

benefits. We define the convergence criteria as:

$$f_\theta(\Delta\vec{\mathbf{p}}, it) := \begin{cases} \texttt{true}, & \Delta\vec{\mathbf{p}} \leq \theta_c \ \textbf{\textit{or}}\ it > \theta_{it} \\ \texttt{false}, \text{ otherwise} \end{cases}$$

Given a reasonably low threshold θ_{it} (see also Section 4.4), the runtime of each RWR is driven by the dimensionality of the compressed transition matrix $\widetilde{\mathbf{N}}$. The number of seed entities is limited by the number of mentions $|M|$. Assuming a fixed entity candidate set size $\mathbb{E}(|C_m|)$ provided by the Candidate Finder, the entity count is bound by the mention count, instead of the knowledge base size. Hence, the dimensionality of $\widetilde{\mathbf{N}}$ is quadratically limited by the number of mentions. Furthermore, the number of steps for the iterative exploration is also constrained by the number of mentions, because each iteration adds at least one alignment.

3.2.3. Discussion

In this section we showed how to identify the most promising candidate entities by applying an iterative exploration model. The model efficiently discovers the most coherent candidate with respect to a set of seeds in each iteration. This is a rather conservative proceeding in comparison to exploring the complete knowledge base graph for any connections between candidates as proposed by Han et al. and Agirre et al. [26, 27]. CohEEL surveys only the first and second order seed communities to find matching candidates.

Guo and Barbosa follow a similar incremental strategy, by selecting one entity at a time and rerun the random walk over the new seed set. Their selection strategy differs in two aspects [29]: first, their algorithm selects the most appropriate candidate from the least ambiguous mention per iteration, instead of the most coherent candidate in the document. Second, instead of picking the entity with the highest random walker visiting probability, the approach measures the compatibility of document and candidates based on semantic signatures. A signature is the steady state probability vector of a random walk starting form the candidate entity SS_e or from the disambiguated entities of the document SS_d (similar to our seeds). Hence, these two distributions represent the probability that a random walker ends at a specific entity in the neighborhood of e or d. The signatures are then compared using the Zero-KL Divergence, a non-symmetric measure to estimate the information loss when SS_d is used to approximate SS_e. Thus, the approach selects the candidate, whose neighborhood distribution is best represented by the neighborhood of the already disambiguated entities. Practically speaking, Guo and Barbosa rank candidates based on the likeliness that a random walker starting from seeds ends with the same probability at an entity as a walker starting from the candidate, whereas we rank based on the likeliness of hitting candidate during a random walk starting from the seeds. Note, we do not claim that our interpretation of coherence yields better results, however, it is more conservative and specifically considers directed knowledge base relationships. This interpretation furthermore yields an advantage in efficiency. By applying our neighborhood compression without affecting the steady state probabilities of any candidate, we are able to shrink the random walk graphs. Because CohEEL executes at most $|M|$ random walks per document, each linking one mention per iteration, the runtime depends only on the compressed neighborhood of the already found alignments. By applying θ_{it}, CohEEL is able to minimize the runtime of each individual random walk.

4. Applying CohEEL

We demonstrate the value of CohEEL by exemplarily applying it to a combination of the open knowledge bases YAGO and Wikipedia. Due to the fact that YAGO is based on Wikipedia and contains links to the Wikipedia entities, an integration of both knowledge bases is straightforward. Note, because we solve the NEL problem, ontological entities, such as `<yago:person>`, and facts and relationships, such as `<rdf:type>`, `<rdfs:subClassOf>`, are ignored. Note, previous work discusses the application of ontological constraints to as presented by Dalvi et al. [33]. Most of the discussed parameters are generic and could be applied to other knowledge bases. However, some parameters also depend on specific knowledge base features (i.e., scoring functions). Furthermore, we show that the CohEEL model is able

to autonomously adapt to different text types, providing a steady linking quality.

We now provide a discussion of all parameters for CohEEL, namely Scoring Functions (Section 4.2), Candidate Classification models (Section 4.3), and Neighborhood Exploration strategies (Section 4.4). Within each section, we provide experiments that compare the influence of the individual parameters and provide a configuration used for the subsequent experiments. A detailed comparison of the discussed configuration with state-of-the-art algorithms is presented in Section 5.

4.1. Experimental Setup

The experiments in the following sections are based on ground truth alignments of text mentions to the knowledge base YAGO [2], which were also used in the AIDA project [21, 25]. The aligned entities comprise all instances from YAGO, but do not include the concepts of the knowledge base.

4.1.1. Datasets

As discussed earlier, different types of document collections share different text characteristics. Table 1 shows the properties of the three datasets used to evaluate CohEELs adaptability:

The **news** article dataset contains 100 randomly picked Reuters articles from the CoNLL-YAGO dataset [20]. The articles were manually aligned with entities from YAGO.

The **encyclopedic** text corpus consists of 335 Wikipedia articles selected in 2006 [17]. The annotated named entities in this dataset are retrieved by translating the linked Wikipedia pages within each article to the corresponding YAGO entities.

The synthetic **micro** corpus consists of 50 short text snippets and was introduced in the AIDA project [25]. Every text snippet consists of few (usually one) hand-crafted sentences about different ambiguous mentions of named entities and has similar properties as content of microblogging platforms, such as Twitter.

Table 1: Dataset overview.

	texts	**mentions per text**	**words per text**
news	100	24.4	171.4
encyclopedic	335	15.1	365.5
micro	50	3	12.6

4.1.2. Quality Measures

We measure precision P, recall R, specificity S, and F_β-measure to investigate the varying linking quality of the tested approaches. Given the gold standard alignments G for a dataset, we first distinguish between entity (G_e) and *NIL* alignments (G_{NIL}). Entity alignments link to an actual entity $e \in E$ in the knowledge base, whereas mentions linking to *NIL* indicate that the actual entity meaning is not present in the knowledge base. Besides the gold standard, each algorithm performs the NEL job and generates two according sets (A_e and A_{NIL}). The quality measures can be calculated as follows:

$$P = \frac{|A_e \cap G_e|}{|A_e|} \quad R = \frac{|A_e \cap G_e|}{|G_e|}$$

$$S = \frac{|A_{NIL} \cap G_{NIL}|}{|G_{NIL}|} \quad F_\beta = \frac{(1+\beta^2) \cdot P \cdot R}{(\beta^2 \cdot P) + R}$$

These definitions are basically the micro-averaged versions over all corpus documents. Where precision measures the fraction of correctly linked entities given the set of found alignments (A_e), the recall (also known as sensitivity) measures the proportion of correctly linked entities (G_e). A high sensitivity shows a low number of type II errors. Furthermore, specificity measures the proportion of correctly identified *NIL* alignments. Low specificity reveals a high number of type I errors. Thus, it is easy to increase the performance with respect to one measure by lowering another one. For instance, by replacing all A_{NIL} by linking to a prominent entity might yield in an increase of sensitivity (recall) but a decrease of specificity. Furthermore, this strategy might decrease the precision of the result. Additionally to the three measures, we show F-measure figures. F_1 is defined as the harmonic mean of precision and recall, where $F_{.25}$ favors precision over recall. To compare the actual linking quality and eliminate the influence of the NER quality, all results are based on an identical set of mentions over the tested documents.

4.2. Scoring Functions

As discussed in Section 3.1.1, CohEEL applies a set of scoring functions to qualify the compatibility between mentions and entities.

4.2.1. First-Order Scores

Based on the knowledge base features, we apply three well-known scoring functions that are based on different entity features and proved successful in various related work: *(i)* the surface prominence function f_{prom} relies on the entity name variations, *(ii)* the context scoring function f_{ctx} is based on the textual knowledge about the entities, and *(iii)* the relatedness scoring function f_{rel} qualifies the compatibility to other aligned entities in the document.

The **surface prominence** score $f_{prom}(m, e)$ is based on the compatibility of mention surface $srfc(m)$ and entity e. In related work it is commonly referred to as commonness, mention-entity-prior, surface form, or anchor-title compatibility [15, 17, 18, 20, 22, 34]. It computes a score based on the mention surface $srfc(m)$ and entity e by leveraging the information contained in the Wikipedia link labels. A link label is a textual mention that is hyperlinked to a Wikipedia article that describes a named entity. To derive candidate entities for a given surface, we compute the relative frequency by which a Wikipedia article is hyperlinked from any occurrence of this label. Hence, the surface prominence score can be seen as an estimate of the conditional log-probability for entity e given the surface of mention m: $f_{prom}(m, e) = \log P(e|srfc(m))$. Note that entities never referred to by m would end up with a log-probability of $-\infty$. However, these entities are not considered as candidates and ignored by default.

The above scoring function considers only the "prominence" of entities with respect to the mention surfaces. For the running example of mention "Brady" in Quote 1, the prominence score yields a relatively low value for the New England Patriots quarterback. The quarterback is linked only in 4% of all "Brady" mentions, while the Texan city is referred to in 20% of the cases. By additionally exploiting contextual information, CohEEL is able to gain evidence that the mention should be aligned to the football player nevertheless.

Second, to efficiently exploit the **context** of mentions, this configuration builds on statistical language models. These have been widely used in information retrieval for ranking result documents to keyword queries [35]. For all knowledge base entities, a language model is built based on the relative frequency of terms that occur in their respective Wikipedia articles. The statistical language model $L(\boldsymbol{\theta}_e)$ of an entity e represents a probability distribution over terms (word unigrams) occurring in the context of the entity. Given the textual context $c = ctx_n(m)$ of a mention, for all instances e that have $s = srfc(m)$ as a label, CohEEL estimates the probability that c is generated by $L(\boldsymbol{\theta}_e)$, i.e., $f_{ctx}(m, e) = \log P(c|L(\boldsymbol{\theta}_e))$ and scores alignments based on these log-probabilities. The context of a mention is represented by terms that surround it in the document. As proposed by Pedersen et al. in [16], we set the range of terms surrounding the mention to $n = 50$. In previous research similar approaches were used to measure the context compatibility in terms of cosine similarity between mention context and entity texts [15, 29] or probability score detonating that the context is generated by a candidate article [18–20, 26].

Third, we exploit the context of the mention by considering the relations between entity candidates within a document. This is done based on the **relatedness** measure introduced by Milne and Witten[22]. The score calculates the pairwise relatedness between two entities based on the set of Wikipedia articles referring to the two entities. To avoid a cyclic definition of the relatedness measure, Milne and Witten propose to calculate f_{rel} as the weighted average of the relatedness between a candidate and all unambiguous entity candidates in the document for all other mentions (E_D^u). Due to this simplified definition of seed entities, the score can be calculated efficiently without an extensive coherence reasoning phase. Similar relatedness measures are shown to outperform the measure of Milne and Witten [15, 19]. However, for this work we decided to limit to this proven score.

Figure 3 compares the alignment performance, to depict the differences of the scoring functions with respect to different text types. An alignment is derived by selecting the entity e_a for mention m that maximizes the value of the scoring function f (i.e.,

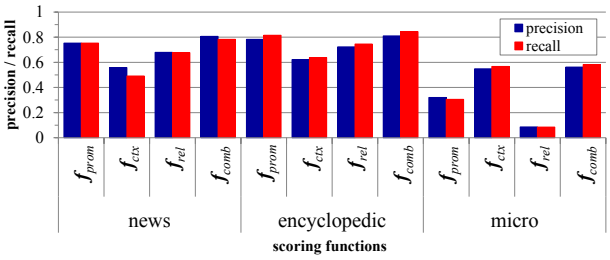


Figure 3: Performance of different first-order functions and a linear combination (f_{comb}) of them.

$e_a = \arg\max_{e \in K} f(m, e)$). In addition to the introduced first-order scoring functions, we further show performance of a fourth score (f_{comb}). The score is a linear combination of the other scores. The weights for the individual scores are equal for all datasets and are derived without any holdout strategies. This is similar to the strategy of manually finding a good combination of scores for these three kinds of texts to improve the individual score-based algorithms with the aforementioned features.

The performance ratio between the four scoring functions are similar for the *news* and the *encyclopedic* dataset. The superior first-order score for these two datasets is f_{prom}. This is not surprising, given that for many cases the selection of prominent entities is appropriate for news or encyclopedic texts. For instance, given the mention "Michael Jordan", the entity of the famous NBA player is usually a good alignment choice and only about 25% of the mentions in these datasets should be linked differently. Ignoring the prominence of the candidate, the other two first-order scores cannot compete, especially the context score shows a low performance. However, the performance of f_{comb} shows, that a combination of all three scores yield an increase in alignment quality. The linear combination improves the best first-order score (f_{prom}) by 4-7% in precision and recall (3-5 percentage points). The improvement of the linear combination in comparison to f_{rel} and f_{ctx} is 12-18% and 30-60%, respectively.

On the *micro* dataset however, divergent results can be identified. Here, the context score outperforms the other first-order scores. This is due to the synthetic nature of the dataset, containing documents with mainly ambiguous mentions, meaning that there are only seldom obvious entity alignments per document ($E^u_D = \emptyset$). For instance, the dataset contains a document: "David and Victoria named their children Brooklyn, Romeo, Cruz, and Harper Seven." It is obvious, that the members of the Beckham family cannot be easily aligned because the first name mentions are very ambiguous. Many similarly named, prominent entities can be found, e.g., David — the second king of Israel, Victoria — the state in south-east of Australia, Brooklyn — the borough of New York City, ... Focusing on the contextual information, the candidate ranking yield better results. For instance, the Wikipedia article of David Beckham contains textual information about the other family members. As for the other two datasets, the linear combination improves all three first-order scores. Where the increase in comparison to f_{ctx} is 3%, f_{prom} is outperformed by 75-90% and f_{rel} is beaten by more than 500%. This shows that a combination of different scores can lead to an improvement of linking results and underlines the independent operating principles of the scores.

4.2.2. Higher-Order Scores

In addition to first-order scoring functions, CohEEL enables the application of higher-order scoring functions. Higher-order scores cover proportions between first-order scores of the candidates for one mention. For the remainder of this work, we introduce three higher-order scores that cover different aspects of the candidate list of a mention:

- $\lambda^{rank}_{\mathbf{f}}$ is the rank of e in a list ordered by the first-order score f (see also Section 3.1). It covers the global position of e among all candidates for mention m and thus correlates with cardinality of the set of entities providing higher f-scores with respect to mention m.

- $\lambda^{\Delta top}_{\mathbf{f}}$ is the decrease in base score f in comparison to the highest scored entity e' for the same mention (m). It indicates the loss of f-score that is encountered by aligning e with m in contrast to e'.

$\lambda_{\mathbf{f}}^{\mathbf{\Delta}succ}$ is the f-score increase in comparison to the next lower scored entity e' for mention m. This score provides evidence by how much e outperforms each lower-scored entity at the minimum.

In an evaluation of the alignment performance, the former two higher-order scores would perform as well as the applied first-order function, because they retain the order of the candidates. The latter one does not provide clear candidate rankings per mention and would thus fail in this evaluation.

4.3. Candidate Classification

We now discuss a configuration for the candidate classification step of CohEEL. We compare the influence of different classification models for both modules, the Seed Finder and the Candidate Finder. Furthermore, we discuss the influence of different higher-order score combinations.

4.3.1. Seed Finder

The Seed Finder has to be configured to achieve reliable alignments, i.e., with a high precision. Out of various different existing classification models, we decided to apply the entropy-based decision tree learner C4.5 [36]. Decision trees already proved in similar research fields [37, 38]. In comparison to the other models, decision trees can explicitly model the dependencies between attributes without any assumptions, i.e., linearity in the data. Because the C4.5 learner applies a reduced error pruning strategy, it is less sensitive to outliers and reduces the danger of overfitting, given a representative training set. Furthermore, the learner implicitly perform a feature selection, i.e., only features with significant information gain are considered.

We now show the influence of the higher-order scores on the seed selector. Our classifier implementations are based on the machine learning algorithms provided by the Weka framework [39]. To this end, we use a configurations that apply all three previously defined first-order functions. Furthermore, the influence of different higher-order score combinations is compared:

$_{no}\lambda$ a configuration without any higher-order functions: $\vec{\phi}_{_{no}\lambda}=\langle f_{prom}, f_{ctx}, f_{rel}\rangle$

$\mathbf{\Delta}_{top}$ a configuration with only one higher-order score for each first-order score:
$\vec{\phi}_{\Delta top}=\left\langle f_{prom}, \lambda_{f_{prom}}^{\Delta top}, f_{ctx}, \lambda_{f_{ctx}}^{\Delta top}, f_{rel}, \lambda_{f_{rel}}^{\Delta top}\right\rangle$

λ^* a configuration with all three higher-order functions resulting in all twelve scores:
$\vec{\phi}_{\lambda^*}=\left\langle f_{prom}, \lambda_{f_{prom}}^{rank}, \lambda_{f_{prom}}^{\Delta top}, \lambda_{f_{prom}}^{\Delta succ}, f_{ctx}, \ldots, f_{rel}, \ldots, \lambda_{f_{rel}}^{\Delta succ}\right\rangle$

Figure 4 shows the performance for different feature vector configurations $\vec{\phi}$ based on C4.5. To measure how accurately the models perform on unknown data, the depicted precision and recall values were measured using 10-fold cross-validation. The applied loss matrix $\mathbf{L}_{fp}^{s}/\mathbf{L}_{fn}^{s}=9$ punishes *false positives* nine times stronger than *false negatives* to guide the learner to create more reliable classifiers with precision of around 90%. The three main observations are: *(i)* the application of additional higher-order scores allow C4.5 to construct more intricate decision rules. This influence might vary for other classification models (not shown in the figure). For instance, a Naïve Bayes classifier follows a feature independence assumption. By introducing more strongly correlated features, the model is skewed, leading to lower precision. *(ii)* The classification quality of the model on the *micro* dataset is lower than on the other datasets. This is expected, because the first-order scores showed decreased performance as well. Furthermore, the models for $_{no}\lambda$ and Δtop are too cautious and do not identify any entity alignment as correct (i.e., seed) leading to a theoretic precision of 1 and a recall of 0. Only a combination of higher-order scores enables the model to generate rules for positive classifications. Especially $\lambda_{f}^{\Delta succ}$ seems to help the classifier, while solely providing only insufficient indications for certain decisions. *(iii)* The C4.5

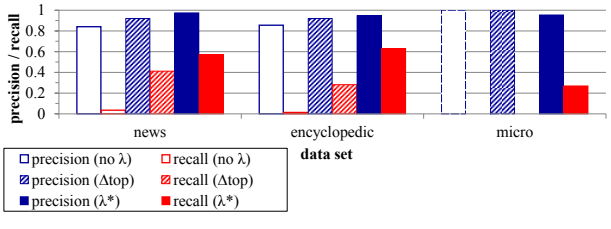


Figure 4: Performance of different scoring functions and the C4.5 classification model for the Seed Finder.

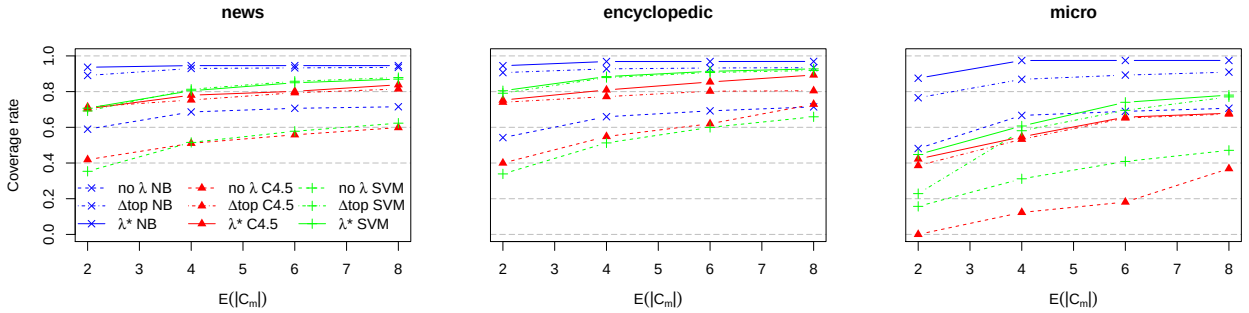


Figure 5: Performance of different scoring function, expected candidate set cardinality ($\mathbb{E}(|\mathbf{C_m}|)$), and classification mode combinations for the Candidate Finder.

decision tree with λ^* outperforms the other configurations with a precision always around 95% while still supplying competitive recall values. For the following experiments we use the C4.5 decision tree algorithm with λ^* to model the Seed Finder.

4.3.2. Candidate Finder

As discussed in Section 3.1.3, the Candidate Finder is meant to provide a list of candidates per mention. Such list should contain the correct entity with a high probability. It is important, that the correct candidate is contained in the list. Incorrect entities in the list are removed in the JNE phase. For instance, the classifier should add all entities that have a high value for one score, even if other candidates might have better overall scores. Therefore, we propose to use Naïve Bayes, a probabilistic classification model. Due to the independence assumption, the model labels an alignment as *correct* as soon as there are sufficient indications, and it does not revise its decision based on the other scores. For instance, given an alignment with a first-order score (f) that is low in comparison to the *correct* alignments in the training set and a high ranking (e.g., $\lambda_f^{rank} = 1$) of the same alignment, Naïve Bayes would still use the latter feature as an indicator of the alignment being *correct*. Other models, such as C4.5, instead would reason over these two features holistically and probably label such an instance as *incorrect*.

Next, we compare the performance of Naïve Bayes (NB) with two established classification models, the entropy-based decision tree learner C4.5, and Support vector machines (SVM), a linear classification model. Besides the classification model, the selection of an appropriate expected candidate set cardinality $\mathbb{E}(|C_m|)$ is important to enable an efficient JNE phase. For the knowledge bases Wikipedia and YAGO, the average number of entities per ambiguous name is of 4.57. Hence, we argue that a value of $\mathbb{E}(|C_m|) \approx 4.57$ is sufficient for the most linking tasks of various with datasets. Figure 5 compares the performance of different classification and cost model combinations on the introduced datasets. To measure the performance of the configurations, we employ the coverage measure. It corresponds to the recall of an optimal selector that always picks the correct entity ($e \in C_m \cap G_e$ or *NIL* if $C_m \cap G_e = \emptyset$) from the candidate set. Hence, the coverage is the maximum possible recall value CohEEL can achieve in future steps (i.e., during JNE). Additionally to the classification and cost model, we evaluate the influence of the different $\vec{\phi}$ configurations, introduced in the previous section. As expected, the overall trend is that the Naïve Bayes classifier outperforms the other two models in terms of coverage. The performance of the algorithms for the *news* and *encyclopedic* datasets is comparable and the performance of the different strategies varies more on the *micro* dataset. The expected candidate set cardinality $\mathbb{E}(|C_m|)$ clearly influences the candidate coverage. An increase in $\mathbb{E}(|C_m|)$ increases the coverage. Furthermore, the addition of higher-order scores, enables a large increase of cov-

erage for all classification models. The difference between Δtop and λ^* configurations is not definite, but in general, the λ^* feature vector performs better.

Hence, we argue that the Naïve Bayes is a good choice for the Candidate Finder and is applied to all introduced features (λ^*) in the following experiments. Furthermore, we apply the rather conservative loss matrix derived from $\mathbb{E}(|C_m|) = 6$ to support datasets with more ambiguous mentions without subverting the JNE performance.

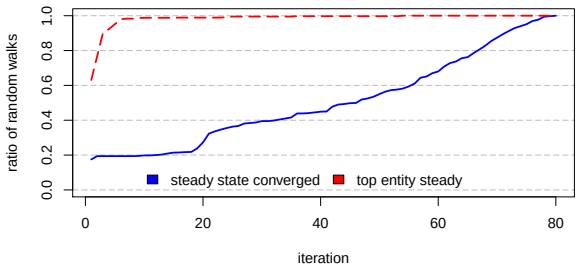


Figure 6: Convergence behavior of steady state probabilities and top scored candidate entities based on the *encyclopedic* dataset.

4.4. Judicious Neighborhood Exploration

In the JNE phase, CohEEL combines the alignments identified by Seed and Candidate Finder. The relationship graph from the knowledge base is used to identify coherent sets of entities within a document. Due to the directed nature of the inter-instance relations in YAGO, we construct directed neighborhood graphs. However, other knowledge bases might contain undirected relations, for instance a bibliographic knowledge base with co-author relationships.

As stated earlier, the RWR is known to converge depending on restart probability α and convergence tolerance level θ_c. We apply the common values of $\alpha = 0.15$ and $\theta_c = 10^{-8}$, implying that the power method for the RWR converges after roughly 113 iterations. To further reduce the number of iterations and thus the runtime of the JNE phase, we introduced θ_{it}. This parameter depends on the dataset, i.e., number of ambiguous mentions and candidates per mention. As stated earlier, the value of θ_{it} can be learned from the training set that was used to train Seed and Candidate Finder by observing the iteration it_s on which top-ranked entity is stable for all documents in the training set.

Figure 6 compares the convergence of the steady state probability vector ($\vec{\mathbf{p}}$) and the iteration it_s after which the most promising candidate was steady (finally selected) during the power iterations. The statistics are exemplary collected over all RWRs for the *encyclopedic* dataset. What surprises first is that the number of necessary power iterations is approximately 80 (instead of 113 as expected). The theoretic convergence bounds are defined for any graph size. However, in our setup the document size (hundreds of mentions) limits the graph size leading to thousands of entities and a faster convergence in comparison to the theoretic bounds, which is observed for graphs that are several orders of magnitudes higher (e.g., PageRank with billions of nodes).

In 62% of the RWR executions the most promising candidate is already steady after one power iteration and changes only in 3% of the cases after the fifth iteration. For the *encyclopedic* dataset, the value of $\theta_{it} = 9$ was derived. This reduces the number of necessary iterations nearly by an order of magnitude. On the *news* dataset, the iteration bound was set to $\theta_{it} = 16$, whereas on the *micro* dataset the value was $\theta_{it} = 5$. This emphasizes the variance of the parameters on different datasets and shows the runtime efficiency increase of the RWR due to the convergence criteria determination.

Next, the linking quality of the previously discussed configurations for Seed and Candidate Finder are compared with the performance of the combined CohEEL model after applying the Judicious Neighborhood Exploration (JNE). Table 2 shows precision and recall results for the CohEEL configuration after *(i)* applying the Judicious Neighborhood Exploration (JNE), based on the alignments of *(ii)* Seed Finder, and *(iii)* Candidate Finder for the three different datasets. The candidate coverage is the maximum possible recall value the model can achieve (Section 4.3.2). The 95% confidence intervals were determined by executing all experiments using a 10-fold cross validation and incorporating the variance between the different folds.

As can be seen, the precision of the Seed Finder

Table 2: Performance of the CohEEL modules and their 95% confidence intervals over a 10-fold cross validation.

		news	ency.	micro
JNE	Prec.	$91.09\%^{+.07}_{-.14}$	$89.19\%^{+.06}_{-.09}$	$90.48\%^{+.10}_{-.32}$
	Recall	$73.99\%^{+.10}_{-.16}$	$72.59\%^{+.03}_{-.10}$	$40.43\%^{+.14}_{-.38}$
Seeds	Prec.	$97.36\%^{+.03}_{-.19}$	$94.66\%^{+.04}_{-.04}$	$95.35\%^{+.05}_{-.27}$
	Recall	$57.05\%^{+.17}_{-.16}$	$63.00\%^{+.09}_{-.08}$	$26.95\%^{+.11}_{-.16}$
Candidate Coverage		$94.57\%^{+.04}_{-.03}$	$96.95\%^{+.02}_{-.02}$	$96.63\%^{+.03}_{-.07}$

reaches around 95% for all three datasets. This emphasizes the effectiveness of this classifier. Furthermore, the Candidate Finder supports a high coverage of correct entities of around 95% for all three datasets. This shows that the Seed and Candidate Finders do not corrupt the reasoning process in the Judicious Neighborhood Exploration. Due to the limited size of the *micro* dataset, the confidence interval is larger in comparison to the other datasets.

It is apparent from the results that the neighborhood exploration phase is able to significantly increase the number of alignments while decreasing the precision provided by the Seed Finder only slightly. While the decrease in precision is only around 5-6%, the JNE is able to increase the recall by 15-50% for all three datasets. For instance, the candidate classifier provided 10,215 alignments for the *news* dataset. These comprise 666 of 773 correct alignments missing in the seed set. During the JNE phase, CohEEL additionally selects 280 correct and 116 incorrect candidates. Recalling that these are the difficult mentions, in which all of the tested classification models failed to provide reliable alignments, this shows the quality of the Judicious Neighborhood Exploration. It remains to say that the recall is nowhere near the theoretical possible value given by the candidate set coverage. This is due to the cautious nature of the neighborhood exploration phase that uses only two-step neighborhood expansion. Following this restriction, the knowledge base does not indicate a relevant relationship. For instance, the algorithm is not intended to identify relationships between two distinct topics like the legal and the American football topic presented in Quote 1.

5. Comparative Evaluation

Next, we provide an evaluation of the discussed CohEEL configuration based on the introduced datasets (Section 4.1.1) and compare it with state-of-the-art approaches.

Besides CohEEL we evaluate a strong scoring function baseline as well as nine state-of-the-art approaches. The baseline $\boldsymbol{f_{comb}}$ derives alignments by selecting the entity e_a for mention m that maximizes the value of the scoring function f_{comb} (i.e., $e_a = \arg\max_{e \in K} f_{comb}(m, e)$). The scoring function is based on a linear combination of surface, context and relatedness scoring functions (for details of the weights see Section 4.2) and outperforms the individual scores in our tests.

The nine state-of-the-art approaches that indeed achieve a high quality in the disambiguation and linking task are:

First, the approach of Cucerzan (referred to as **LED**) extends the term-based feature vectors of Wikipedia entities by information, such as key phrases and categories, from other articles that link to it [17]. The project was conducted at Microsoft and the code is proprietary. Hence, we reimplemented the algorithm according to the descriptions in the paper.

Second, we compare with an approach that is based on the algorithm of Milne and Witten (referred to as **L2LW**) [22]. We implemented the algorithm according to the descriptions in the paper based on the Weka toolkit. In contrast to the evaluation of the prior work that was based on a knowledge base consisting of only 700 Wikipedia articles, we applied the algorithm to the complete knowledge bases YAGO and Wikipedia. For commonness and relatedness measures, we used the scores f_{prom} and f_{rel} introduced in Section 4.2. Subsequently, we trained the algorithm based on all three datasets.

Third, we evaluate **BEL**, an approach that builds on a majority-voting over multiple ranking classifiers [20]. BEL is based on the idea of bagging the contextual information of each mention. We used the original implementation, which operates on the knowledge base of YAGO.

UKB, the algorithm of Agirre et al., runs a personalized PageRank approach over the Wikipedia graph built over reciprocal article links [27]. We used the implementation as well as the knowledge base graph provided on the project website.

Fifth, **AIDA** (also known as GRAPH-KORE) provides an algorithm that uses graph-based connectivity between candidate entities of multiple mentions (e.g., derived from the type, subclassOf edges of the knowledge graph or from the incoming links in Wikipedia articles) to determine the most promising linking of the mentions [25]. For the experiments, we used the implementation provided on the project website.

Sixth, we provide the results of the approach of Guo and Barbosa (**REL-RW**), which applies an iterative algorithm based on semantic entity signatures derived from stationary random walk distribution over the knowledge base neighborhood [29]. Unfortunately, the source code was not available as open source at the time of the evaluation, however, the authors generously executed the algorithm on their hardware to produce the alignments for the discussed datasets.

Finally, we used the General Entity Annotation Benchmark Framework **GERBIL** [9] to retrieve the results for the three algorithms based on their external Web services: DBpedia **Spotlight** disambiguates entities using a generative probabilistic model that based on surface and context features [18]. **WAT** applies different mention-local features to build an entity graph for each document and use link-based ranking algorithms on the entity graph containing edges weighted by relatedness between the entities [19]. **Babelfy** identifies a densest subgraph by iteratively removing the weakliest connected entity from a semantic document representation, which is derived from the multilingual knowledge base BabelNet [28].

To enable a fair comparison of the linking quality of approaches not based on the same knowledge base version (i.e., LED, Spotlight, UKB, WAT, Babelfy, and REL-RW), we replaced the alignments of entities without representation in our knowledge base (e.g., concepts like summer) by *NIL* links.

Table 3 provides a performance comparison over all eight algorithms. The six rightmost algorithms are the approaches that perform a coherence reasoning to improve the linking quality. It is apparent that these approaches produce the best results for all of the applied quality measures. Furthermore, it is visible that the performance of all competitors is lower for the synthetic micro dataset than for the two long text datasets.

All competitors of CohEEL provide balanced precision-recall values per dataset leading to large numbers of incorrect entity alignments of up to 71%. In contrast, CohEEL consistently outperforms all the other approaches in terms of precision and is able to reduce the false positive rate to around 10%. Only one competitor is able to achieve comparable precision values on the two long text corpora (news and encyclopedic): REL-RW. Whereas it performs well for long text, it suffers from low precision of only 60% on the micro dataset. For this dataset, all competitors suffer from low precision. For the best competitor Babelfy, one in four entity alignments are incorrect, whereas the other competitors even provide a third or more incorrect entities. Only CohEEL is able to produce a higher precision. This emphasizes the effectiveness of the model with a focus on producing reliable alignments. In terms of recall, CohEEL is mostly on par with the non-coherence-based approaches (five leftmost algorithms) but is outperformed by the coherence-based approaches. This leads to an upper midfield position of our approach when ranked by the harmonic mean between precision and recall (F_1 measure), favoring balanced precision-recall performance. However, in comparison to the strongest competitor REL-RW, CohEEL loses only 2 to 8 percent points in F_1 in all datasets.

Recalling that the focus of our system lies on producing reliable alignments with high precision, a comparison based on the $F_{.25}$-measure that favors precision to recall is more reasonable. The performance of REL-RW is impressive on the long text corpora

Table 3: Performance comparison of state-of-the-art algorithms and CohEEL. Right-sided: approaches that apply a graph-based coherence reasoning strategy

						news					
	f_{comb}	LED	L2LW	Spotlight	BEL	UKB	WAT	AIDA	Babelfy	REL-RW	CohEEL
P	80.6%	63.4%	72.1%	79.3%	80.1%	83.3%	84.8%	78.7%	76.6%	88.1%	**91.1%**
R	78.2%	64.9%	69.7%	73.8%	80.1%	82.8%	83.1%	80.4%	76.5%	**85.4%**	74.0%
S	78.4%	61.2%	71.2%	81.0%	73.1%	75.2%	74.9%	66.0%	70.2%	**96.6%**	88.7%
F_1	79.4%	64.1%	70.9%	76.5%	80.1%	83.0%	84.0%	79.5%	76.5%	**86.8%**	81.7%
$F_{.25}$	80.1%	63.7%	71.6%	78.1%	80.1%	83.2%	84.5%	79.0%	76.6%	**87.6%**	87.1%
						encyclopedic					
	f_{comb}	LED	L2LW	Spotlight	BEL	UKB	WAT	AIDA	Babelfy	REL-RW	CohEEL
P	82.0%	63.5%	72.3%	84.2%	82.2%	87.0%	81.6%	79.5%	77.3%	88.0%	**89.2%**
R	82.2%	68.8%	70.4%	70.9%	82.8%	58.6%	73.7%	85.6%	77.0%	**89.0%**	72.6%
S	65.3%	50.1%	67.6%	76.5%	68.6%	79.0%	72.1%	52.2%	65.5%	80.8%	**83.3%**
F_1	82.1%	66.0%	71.4%	77.0%	82.5%	70.0%	77.4%	82.4%	77.1%	**88.5%**	80.0%
$F_{.25}$	82.0%	64.5%	71.9%	81.2%	82.3%	79.3%	79.9%	80.6%	77.2%	**88.2%**	85.3%
						micro					
	f_{comb}	LED	L2LW	Spotlight	BEL	UKB	WAT	AIDA	Babelfy	REL-RW	CohEEL
P	56.2%	40.1%	29.4%	48.8%	54.8%	63.7%	59.6%	64.4%	72.7%	60.0%	**90.5%**
R	58.2%	41.8%	24.8%	44.7%	48.2%	61.0%	59.6%	66.7%	**73.8%**	55.3%	40.4%
S	28.6%	14.3%	57.1%	42.9%	42.9%	42.9%	42.9%	28.6%	28.6%	57.1%	**85.7%**
F_1	57.1%	41.0%	26.9%	46.7%	51.3%	62.3%	59.6%	65.5%	**73.2%**	57.6%	55.9%
$F_{.25}$	56.6%	40.5%	28.4%	47.9%	53.4%	63.1%	59.6%	64.8%	**72.9%**	59.0%	72.5%

and only CohEEL is able to produce comparable results. However, for the micro dataset the CohEEL improves the $F_{.25}$ of REL-RW by 14 percent points and is on par with the best competitor Babelfy. All other algorithms are outperformed by 8–44 percent points.

The previous discussion covered only the detection of knowledge base entities. Specifically, most competitors show better results in terms of sensitivity (recall). However, in terms of specificity, i.e., the proportion of *NIL* alignments that are correctly identified, the performance of the algorithms shows a different behavior. CohEEL favors cautiousness for linking knowledge base entities and thus achieves high specificity values and is the only approach that detects around 85% of the out-of-knowledge base entities for all three datasets.

Overall, the three iterative graph-based coherence reasoning approaches Babelfy, REL-RW, and CohEEL outperform the other systems. Depending on the dataset, either Babelfy or REL-RW provides the best F_1 results, because of the stronger precision-recall-balance in comparison to CohEEL. REL-RW in particular on the news and encyclopedic dataset. This is probably due to the focus of the configuration towards longer texts. The evaluation in the original publication is performed on news corpora, namely MSNBC, AQUAINT, and ACE2004 [29]. The strong performance drop of REL-RW on the micro dataset, which shows other text characteristics, is drastic. On this dataset Babelfy outperforms all other competitors by nearly ten percent points in F_1. In contrast, CohEEL is able to consistently provide high precision and specificity values. This underscores the advantage of our supervised approach that is able to automatically adopt to different input text types and assure high specificity and precision values.

Figure 7 depicts that, executed on Intel Xeon

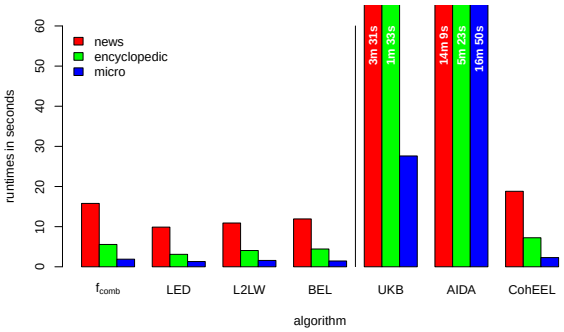


Figure 7: Runtime comparison of state-of-the-art algorithms and CohEEL.

2.67GHz machine with 32 CPU-cores and 256GB RAM, the runtime of most of the tested algorithms per document varies in the seconds range (between 1 and 15s). This is due to the fact, that f_{comb}, LED, and BEL are not based on an exhaustive coherence reasoning. Furthermore, the relatedness score of L2LW is based on a greedy approach that calculates coherence based on unambiguous entities and does not consider different combinations of ambiguous entities. Hence, the runtime of all four approaches is mainly determined by the calculation of the scoring functions.

AIDA applies an expensive reasoning over all candidate combinations, leading to a document processing time more than one order of magnitude higher than the other competitors. The measured runtimes of AIDA for the *micro* dataset are surprisingly high ($16min\,50s$), since each document contains only three mentions on average. However, these mentions are highly ambiguous, yielding a large number of potential entities, e.g., all persons named "David" in combination to all persons named "Victoria". UKB shows slightly better performance than AIDA. As discussed in Section 2.2, this can be explained by an efficient RWR implementation that is based on a sparser knowledge base graph containing only relevant (reciprocal) Wikipedia links. However, depending on the dataset, UKB analyzes each document within 0.5 to 3.5min on average. This shows that the coherence reasoning phase is a costly operation, even on thoroughly curated knowledge base graphs.

In contrast to AIDA and UKB, the coherence reasoning of CohEEL, i.e., the JNE phase, is only a small factor and occupies between 5% and 20% of the runtime. Furthermore, the candidate classification phase has negligibly low runtimes of only 1-5ms per mention. Thus, the overall runtime of CohEEL lies between 2 and 18s per document depending on the dataset and is therefore comparable with the approaches without coherence reasoning. This is important for different scenarios: if large text collections (millions of documents) have to be processed, texts have to be analyzed in seconds to annotate the whole corpus in acceptable time (days). Furthermore, in online scenarios, such as an entity linking web service, where links have to be retrieved per document immediately, runtimes larger than several seconds are not acceptable.

Please note that we cannot supply runtime measurements of DBpedia Spotlight, WAT, Babelfy, and REL-RW, because we did not execute the programs on our hardware. However, as discussed in Section 3.2.3, we argue that CohEEL is able to outperform REL-RW: Assuming a similar document neighborhood graph sizes as UKB – both algorithms are based on the Wikipedia link graph – we expect a comparable runtime of REL-RW, meaning in the order of minutes per document. This assumption is due to the complexity of the RWR over a large (uncompressed) neighborhood graph.

6. Conclusion

We introduced CohEEL, a novel and efficient named entity linking model that uses a random walk strategy to combine the results of a precision-oriented and a recall-oriented classifier while maintaining a high precision and elevating the recall to practically viable levels. We introduced a configuration of this model based on YAGO and Wikipedia and compared it with state-of-the-art NEL approaches. While showing superior behavior in terms of precision, CohEEL also provides competitive F-measure values. Furthermore, we showed that CohEEL's efficiency is practical and thus enables online scenarios, where entity links have to be retrieved in near-real-time.

We are working on a distributed version of CohEEL based on Apache Flink to help to process and anno-

tate huge text collections, i.e., dozens of millions of documents[1]. We plan to extend CohEEL with the ability to automatically detect possible mention surfaces in texts, and thus reduce its dependency on a named entity recognizer. Furthermore, we plan to test the influence of the entity alignments found by CohEEL on different information retrieval tasks, such as document clustering or question answering. Another goal is to investigate CohEEL's performance on other types of texts, such as scientific texts (e.g., life sciences, social sciences, etc.) or user generated content (e.g., blog articles, product reviews, etc.) based on appropriate knowledge bases. Future research not covered by this work targets at improving knowledge base graphs for the coherence reasoning to solve the NEL task [27–29, 33]. Specifically, the quality as well as efficiency of CohEEL might be improved by gathering dense subgraph structure information, such as k-trusses [40]. CohEEL provides a strong basis for further investigations in this important research area.

Acknowledgments

This research was funded by the German Research Society, DFG grant no. FOR 1306.

7. References

[1] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, Z. G. Ives, DBpedia: A nucleus for a web of open data, in: The Semantic Web, Vol. 4825 of Lecture Notes in Computer Science, Springer, 2007, pp. 722–735. doi:10.1007/978-3-540-76298-0_52.

[2] J. Hoffart, F. M. Suchanek, K. Berberich, E. Lewis-Kelham, G. de Melo, G. Weikum, YAGO2: Exploring and querying world knowledge in time, space, context, and many languages, in: Proceedings of the International Conference on World Wide Web (WWW), 2011, pp. 229–232. doi:10.1145/1963192.1963296.

[3] M. Dredze, P. McNamee, D. Rao, A. Gerber, T. Finin, Entity disambiguation for knowledge base population, in: Proceedings of the International Conference on Computational Linguistics (COLING), 2010, pp. 277–285.

[4] T. Gruetze, G. Kasneci, Z. Zuo, F. Naumann, Bootstrapping Wikipedia to answer ambiguous person name queries, in: International Workshop on Information Integration on the Web (IIWeb), 2014, pp. 56–61. doi:10.1109/ICDEW.2014.6818303.

[5] M. Khalid, V. Jijkoun, M. de Rijke, The impact of named entity normalization on information retrieval for question answering, in: Advances in Information Retrieval, Vol. 4956 of Lecture Notes in Computer Science, Springer, 2008, pp. 705–710. doi:10.1007/978-3-540-78646-7_83.

[6] A. Carlson, J. Betteridge, B. Kisiel, B. Settles, E. R. Hruschka Jr., T. M. Mitchell, Toward an architecture for never-ending language learning, in: Proceedings of the National Conference on Artificial Intelligence (AAAI), 2010, pp. 1306–1313.

[7] G. Kasneci, F. M. Suchanek, G. Ifrim, M. Ramanath, G. Weikum, NAGA: Searching and ranking knowledge, in: Proceedings of the IEEE International Conference on Data Engineering (ICDE), 2008, pp. 953–962. doi:10.1109/ICDE.2008.4497504.

[8] D. Carmel, M.-W. Chang, E. Gabrilovich, B.-J. P. Hsu, K. Wang, ERD 2014: Entity recognition and disambiguation challenge, SIGIR Forum 48 (2) (2014) 63–77. doi:10.1145/2701583.2701591.

[9] 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, L. Wesemann, GERBIL: General

[1] https://hpi.de/naumann/projects/coheel

entity annotator benchmarking framework, in: Proceedings of the International Conference on World Wide Web (WWW), 2015, pp. 1133–1143.

[10] R. Mihalcea, A. Csomai, Wikify!: Linking documents to encyclopedic knowledge, in: Proceedings of the International Conference on Information and Knowledge Management (CIKM), 2007, pp. 233–242. doi:10.1145/1321440.1321475.

[11] R. Sinha, R. Mihalcea, Unsupervised graph-based word sense disambiguation using measures of word semantic similarity, in: Proceedings of the International Conference on Semantic Computing (ICSC), 2007, pp. 363–369. doi:10.1109/ICSC.2007.107.

[12] E. Agirre, A. Soroa, Personalizing PageRank for word sense disambiguation, in: Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics (EACL), 2009, pp. 33–41.

[13] M.-C. d. Marneffe, B. MacCartney, C. D. Manning, Generating typed dependency parses from phrase structure parses, in: Proceedings of the International Conference on Language Resources and Evaluation (LREC), 2006, pp. 449–454.

[14] B. Hachey, W. Radford, J. Nothman, M. Honnibal, J. R. Curran, Evaluating entity linking with Wikipedia, Artificial Intelligence 194 (2013) 130–150. doi:10.1016/j.artint.2012.04.005.

[15] L. Ratinov, D. Roth, D. Downey, M. Anderson, Local and global algorithms for disambiguation to Wikipedia, in: Proceedings of the Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (HLT), 2011, pp. 1375–1384.

[16] T. Pedersen, A. Purandare, A. Kulkarni, Name discrimination by clustering similar contexts, in: Proceedings of the International Conference on Intelligent Text Processing and Computational Linguistics (CICLing), 2005, pp. 226–237. doi:10.1007/978-3-540-30586-6_24.

[17] S. Cucerzan, Large-scale named entity disambiguation based on Wikipedia data, in: Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL), 2007, pp. 708–716.

[18] J. Daiber, M. Jakob, C. Hokamp, P. N. Mendes, Improving efficiency and accuracy in multilingual entity extraction, in: Proceedings of the International Conference on Semantic Systems (I-SEMANTICS), 2013, pp. 121–124. doi:10.1145/2506182.2506198.

[19] F. Piccinno, P. Ferragina, From TagME to WAT: A new entity annotator, in: Proceedings of the International Workshop on Entity Recognition & Disambiguation (ERD), 2014, pp. 55–62. doi:10.1145/2633211.2634350.

[20] Z. Zuo, G. Kasneci, T. Gruetze, F. Naumann, BEL: Bagging for entity linking, in: Proceedings of the International Conference on Computational Linguistics (COLING), 2014, pp. 2075–2086.

[21] J. Hoffart, M. A. Yosef, I. Bordino, H. Fürstenau, M. Pinkal, M. Spaniol, B. Taneva, S. Thater, G. Weikum, Robust disambiguation of named entities in text, in: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), 2011, pp. 782–792.

[22] D. Milne, I. H. Witten, Learning to link with Wikipedia, in: Proceedings of the International Conference on Information and Knowledge Management (CIKM), 2008, pp. 509–518. doi:10.1145/1458082.1458150.

[23] F. Du, Y. Chen, X. Du, Linking entities in unstructured texts with RDF knowledge bases, in: Proceedings of Asia-Pacific Web Conference (APWeb), Vol. 7808 of Lecture Notes in Computer Science, Springer, 2013, pp. 240–251. doi:10.1007/978-3-642-37401-2_25.

[24] S. Kulkarni, A. Singh, G. Ramakrishnan, S. Chakrabarti, Collective annotation of Wikipedia entities in web text, in: Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2009, pp. 457–466. doi:10.1145/1557019.1557073.

[25] J. Hoffart, S. Seufert, D. B. Nguyen, M. Theobald, G. Weikum, KORE: Keyphrase overlap relatedness for entity disambiguation, in: Proceedings of the International Conference on Information and Knowledge Management (CIKM), 2012, pp. 545–554. doi:10.1145/2396761.2396832.

[26] X. Han, L. Sun, J. Zhao, Collective entity linking in web text: A graph-based method, in: Proceedings of the International ACM SIGIR Conference on Research and development in Information Retrieval, 2011, pp. 765–774. doi:10.1145/2009916.2010019.

[27] E. Agirre, A. Barrena, A. Soroa, Studying the Wikipedia hyperlink graph for relatedness and disambiguation, CoRR abs/1503.01655.
URL http://arxiv.org/abs/1503.01655

[28] A. Moro, A. Raganato, R. Navigli, Entity linking meets word sense disambiguation: a unified approach, Transactions of the Association for Computational Linguistics 2 (2014) 231–244.

[29] Z. Guo, D. Barbosa, Robust entity linking via random walks, in: Proceedings of the International Conference on Information and Knowledge Management (CIKM), 2014, pp. 499–508. doi:10.1145/2661829.2661887.

[30] C. M. Bishop, Pattern Recognition and Machine Learning, Springer, 2006.

[31] H. Tong, C. Faloutsos, J.-Y. Pan, Random walk with restart: fast solutions and applications, Knowledge and Information Systems 14 (3) (2008) 327–346. doi:10.1007/s10115-007-0094-2.

[32] A. N. Langville, C. D. Meyer, Deeper inside PageRank, Internet Mathematics 1 (3) (2004) 335–380. doi:10.1080/15427951.2004.10129091.

[33] B. Dalvi, E. Minkov, P. P. Talukdar, W. W. Cohen, Automatic gloss finding for a knowledge base using ontological constraints, in: Proceedings of the International Conference on Web Search and Data Mining (WSDM), 2015, pp. 277–285.

[34] P. N. Mendes, M. Jakob, A. García-Silva, C. Bizer, DBpedia Spotlight: Shedding light on the web of documents, in: Proceedings of the International Conference on Semantic Systems (I-SEMANTICS), 2011, pp. 1–8. doi:10.1145/2063518.2063519.

[35] C. Zhai, J. Lafferty, A study of smoothing methods for language models applied to information retrieval, ACM Transactions on Information Systems 22 (2). doi:10.1145/984321.984322.

[36] J. R. Quinlan, C4.5: Programs for machine learning, Vol. 1, Morgan Kaufmann, 1993.

[37] J. F. McCarthy, W. G. Lehnert, Using decision trees for coreference resolution, in: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), 1995, pp. 1050–1055.

[38] V. Ng, C. Cardie, Improving machine learning approaches to coreference resolution, in: Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL), Association for Computational Linguistics, 2002, pp. 104–111. doi:10.3115/1073083.1073102.

[39] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, I. H. Witten, The WEKA data mining software: An update, SIGKDD Explorations 11 (1) (2009) 10–18.

[40] J. Cohen, Graph twiddling in a MapReduce world, Computing in Science and Engineering 11 (4) (2009) 29–41. doi:10.1109/MCSE.2009.120.