Entity Disambiguation based on a Probabilistic Taxonomy

Masumi Shirakawa[1]* Haixun Wang[2] Yangqiu Song[2] Zhongyuan Wang[2]
Kotaro Nakayama[3] Takahiro Hara[1]
Shojiro Nishio[1]
[1]Osaka University, Osaka, Japan
{shirakawa.masumi, hara, nishio}@ist.osaka-u.ac.jp
[2]Microsoft Research Asia, Beijing, China
{haixunw, yangs, zhowang}@microsoft.com
[3]The University of Tokyo, Tokyo, Japan
nakayama@cks.u-tokyo.ac.jp

Abstract

This paper presents a method for entity disambiguation, one of the most substantial tasks for machines to understand text in natural languages. In a natural language, terms have ambiguity, e.g. "Barcelona" usually means a *Spanish city* but it can also refer to a *professional football club*. In our work, we utilize a probabilistic taxonomy that is as rich as our mental world in terms of the concepts of worldly facts it contains. We then employ a naive Bayes probabilistic model to disambiguate a term by identifying its related terms in the same document. Specifically, our method consists of two steps: clustering related terms and conceptualizing the cluster using the probabilistic taxonomy. We cluster related terms probabilistically instead of using any threshold-based deterministic clustering approach. Our method automatically adjusts the relevance weight between two terms by taking the topic of the document into consideration. This enables us to perform clustering without using a sensitive, predefined threshold. Then, we conceptualize all possible clusters using the probabilistic taxonomy, and we aggregate the probabilities of each concept to find the most likely one. Experimental results show that our method outperforms threshold-based methods with optimally set thresholds as well as several gold standard approaches for entity disambiguation.

1 Introduction

Are computers really able to understand natural language as we human do? For many decades, researchers have tackled this, one of the most challenging but interesting issues in the area of computer science. Entity disambiguation is a fundamental task for understanding text written in natural languages. Disambiguating an entity means identifying which entity is denoted by an ambiguous term in a document. For example, given a document including "Barcelona's 2-0 victory over United in the 2009 final in Rome," how can we understand that "Barcelona" and "United" respectively mean professional football clubs "FC Barcelona" and "Manchester United F.C." while "Rome" denotes a capital city of Italy (and "the 2009 final" is the final match of UEFA Champions League in 2009)? Even some of us may fail to disambiguate them because of the lack of knowledge or misunderstanding. It is far more challenging for computers to disambiguate the terms accurately.

Concepts are substantial in entity disambiguation. Conceptualizing "Barcelona" to a *professional football club* considering its context is a disambiguation process in our mind when we see the example above. Representative work on entity disambiguation links a term to a corresponding entity in a knowledgebase without using concepts. However, toward human-like text understanding, disambiguation process should include the conceptualization of terms. Psychologist Gregory Murphy began his highly acclaimed book [20] with the statement "*Concepts are the glue that holds our mental world together.*" Still, Nature magazine book review calls it an understatement, because "*Without concepts, there would be no mental world in the first place*" [3]. Doubtless to say, the ability to conceptualize is a defining characteristic of humanity. We focus on entity disambiguation at concept-level to pass the torch to text understanding.

*The work was done at Microsoft Research Asia.

To enable machines to perform human-like conceptualization, Song et al. [24] proposed a probabilistic framework, which includes a knowledgebase and certain inferencing techniques on top of the knowledgebase. The knowledgebase, known as Probase [27], contains concepts (of worldly facts) that are as rich as those in our mental world. Probase scans billions of documents to obtain millions of concepts, and for each concept, it finds entities that make the concept concrete. Moreover, Probase scores their *isa* relationships. In Song's work, these scores are used as priors and likelihoods for various statistical inferencing over the text data. Finally, they developed a naive Bayes probabilistic model which derives the most dominant concepts from a set of terms. As an example of the conceptualization by their method, given two terms "Jaguar" and "Volvo," it derives concepts such as *company* or *car manufacturer*, but not *animal* for "Jaguar".

Song's method assumes that all the given terms belong to one concept whereas a document complexly contains various kinds of concepts in most cases. Hence it cannot be applied to entity disambiguation. Though they also proposed to use a simple threshold-based clustering of related terms, it is still not enough to obtain well-formed clusters. Moreover, detecting terms from a document is also a problem in entity disambiguation because the interpretation of terms in a document may be multiple.

In this paper, we propose a method which detects terms from a given document and disambiguates the terms by conceptualization. It first detects all terms in a document by using both linguistic and semantic information. Then, based on Song's method, it indirectly conceptualizes terms through predicting the cluster probabilistically. Specifically, the disambiguation component of the method consists of following two steps: clustering related terms in the document, and conceptualizing the cluster to obtain the dominant concepts. Here, we employ probabilistic clustering instead of threshold-based deterministic clustering for selecting related terms. This is because threshold-based clustering requires a sensitive predefined threshold that determines which terms should belong to the same cluster. For that matter, the threshold may vary with respect to each document. Meanwhile, our method automatically adjusts the relevance between the target term and the other terms depending on the topic of the document. That is, all the possible clusters are probabilistically generated and conceptualized to aggregate the probability of each concept.

The contributions of our work can be summarized as follows:

- We propose an indirect conceptualization method through probabilistic clustering to disambiguate terms in a document. We then transform a formula that models the conceptualization method in order to compute it in polynomial time. In the result, we prove that the conceptualization through probabilistic clustering equals to the conceptualization with relatedness-based smoothing. Our method is robust since it can handle a document complexly including various kinds of concepts. Moreover, it requires no parameter configuration.

- We propose a paradigm for entity disambiguation, from detecting terms to disambiguating the terms. In particular, it firstly breaks sentences from a given document and secondly detects terms using both linguistic and semantic information. Thirdly it get concepts for the terms using a probabilistic taxonomy and finally disambiguates the terms by our indirect conceptualization method. In addition, this paradigm can also handle case-insensitive documents or documents partly containing case-insensitive sentences.

- We conduct an evaluation compared to the method with deterministic clustering or state-of-the-art methods on a large dataset generated from CoNLL2003 named entity recognition shared task. From the result, our method is demonstrated to bring improvements and to be highly competitive. Especially on short texts, our method significantly outperforms comparative methods. We also evaluate our term detection method and find it effective.

The remains of the paper is organized as follows. Sec. 2 describes the related work on entity disambiguation and knowledgebases for entity disambiguation. Sec. 3 outlines our challenges for entity disambiguation. Sec. 4 details our entity disambiguation method including sentence detection, term detection and entity conceptualization. Sec. 5 shows an evaluation of our method. Finally we conclude our work in Sec. 6.

2 Related Work

2.1 Entity Disambiguation

Entity disambiguation is one of substantial tasks when we try to enable computers to understand natural texts. In many cases, entity disambiguation is performed and evaluated by linking terms in a given document to corresponding entities stored in a knowledgebase. Recently Web-scale entity disambiguation where the domain is not limited has been addressed as the technologies surrounding the Web develop in a rapid pace. Let us pick up some outstanding work.

Dill et al. [6] built a first Web-scale entity disambiguation system called SemTag, achieving good precision over the recall. Since Wikipedia appeared, it has been extensively used as a knowledge resource to disambiguate entities [5, 14, 16, 17, 19]. Cucerzan [5] introduced a vector space model to leverage the category information from Wikipedia and demonstrated the advantage of Wikipedia for entity disambiguation. Cucerzan's method considers prior and coherence: prior is the popularity of an entity and coherence is the interdependence among entities. Their method selects the most appropriate entity for each term to maximize scalar products of category vectors based on prior and coherence. note that scalar products are more efficient than cosine similarity in terms of the computational cost. The idea of prior and coherence is fundamental for entity disambiguation and most of the work uses similar metrics. Their simple vector space model has been a benchmark. Milne et al. [19] employed a machine learning approach using links within Wikipedia articles as training data. They used the commonness (prior) and relatedness to the surrounding context (coherence) as features. In addition, efficient relatedness measurement between two Wikipedia articles is introduced to cope with a large set of input terms. Kulkarni et al. [17] introduced a collective method to use pairwise coherence for the first time. They formulated the trade-off between local term-to-entity compatibility and pairwise coherence. The formula was NP-hard, hence they proposed an approximation to solve the problem. According to their experiments, it outperformed conventional methods. More recently, graph-based approaches [14, 16] achieved significant improvements in comparison to gold standard methods such as Cucerzan's method, Milne's method and Kulkarni's method. They used global-level coherence among all the possible entities for all terms in a document. It can be said that the graph-based approach is one of the most state-of-the-art methods powered by Wikipedia as of October 2011.

Named entity recognition (NER) includes a similar task to named entity disambiguation while it generates only one of a few classes (basically *person*, *organization*, *location* and *miscellaneous*) for a detected term. A term with such a class cannot be regarded as a disambiguated entity because it may be still ambiguous (e.g. a *person* "Michael" is highly ambiguous). Fine-grained named entity recognition addresses the disambiguation problem of NER by providing more specific classes such as *politician*, *writer* or *singer*. Fleischman et al. [10] tackled fine-grained classification of named entities by a supervised learning with both local and global information. Giuliano et al. [13] proposed a semi-supervised algorithm using snippets from the Web and a knowledgebase and Giuliano [12] also extended it by introducing a latent semantic kernel. Our method also belongs to fine-grained named entity recognition rather than entity linking.

2.2 Knowledgebase

Disambiguation process in human mind is to conceptualize a term considering its context in a document. To enable machines to understand human concepts, we need a knowledgebase. WordNet [8], Wikipedia[1], Cyc [18] and Freebase [4] are created by human experts or community efforts. Recently, much work has been devoted to building knowledgebases automatically. Representative systems include KnowItAll [7], TextRunner [2], WikiTaxonomy [22], DBPedia [1] and YAGO [25].

Existing knowledgebases fall short of supporting machines to perform human-like conceptualization. There are two major obstacles. First, the scale and scope of the knowledgebases is not big enough. For example, Freebase has about 2,000 categories, and Cyc, after 25 years of efforts, has 120,000 categories. In other words, they are limited in coverage and granularity in representing concepts in our mental world. Second, most of these knowledgebases are deterministic instead of probabilistic. This means, for example, although we can find the concepts that a term may belong to, it is not possible to find which concept is the most typical concept for that term.

Probase [27] is a taxonomy that contains millions of concepts learned iteratively from 1.68 billion web pages. The core taxonomy consists of the *isa* relationships extracted by using syntactic patterns including the Hearst patterns [15]. For example, we consider "... artists such as Pablo Picasso ..." as a piece of evidence for the claim that "Pablo Picasso" is an entity of the concept *artist*. In addition, these

[1] http://www.wikipedia.org/

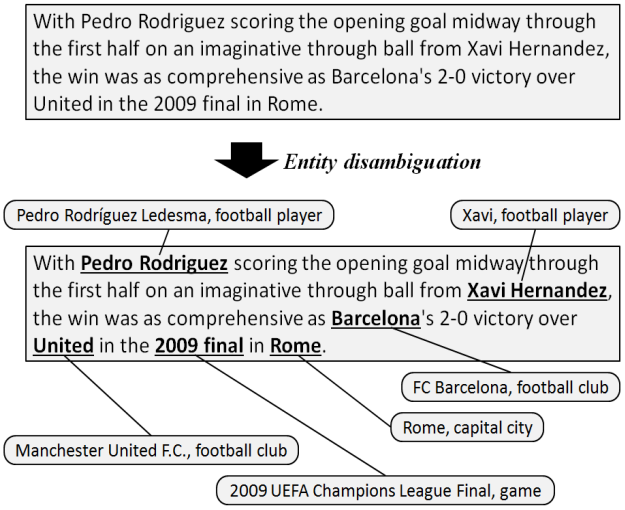

Figure 1: Example of entity disambiguation

relationships are associated with a probability that models the consensus, typicality and ambiguity. The current version of the taxonomy contains about 8 million concepts and 17 million entities. More detailed description of how the taxonomy is constructed can be found in [27]. Based on Probase, human-like conceptualization can be performed [24] (see Sec. 3).

3 Entity Disambiguation using a Probabilistic Taxonomy

In this work, we address the problem of entity disambiguation. Disambiguating terms occurring in a document is a fundamental task for computers to understand text. Disambiguation process in human mind is to conceptualize a term considering its context in a document. For example, given a sentence "Apple has shoved Microsoft aside as the company," we conceptualize "Apple" to a *company* considering that "Microsoft" is also a *company*, or the sentence contains a term "company." Entity disambiguation task also includes term detection from a target document. Figure 1 shows another example of entity disambiguation that we aim to perform, where all terms that can be entities are detected and conceptualized. The challenges in entity disambiguation task are summarized as follows:

- How do we disambiguate detected terms?
- How do we detect all the terms from a given document?

We use the conceptualization for disambiguating detected terms. This is because the conceptualization is a more direct way for disambiguating terms just as we human do. In addition, since we aim to enable machines to understand natural-language text, disambiguation process as a part of text understanding should be accomplished at a concept-level. Most work on entity disambiguation focuses on entity linking in which a term is linked to an entity in a knowledgebase. We also employed entity linking for the evaluation in our work, though, we believe that obtaining dominant concepts for a term is more valuable than identifying the entity id stored in a knowledgebase toward text understanding.

The other reason that we choose the conceptualization approach is to introduce concrete relationships to cope with short texts. Representative work is based on relatedness (similarity) between entities derived from Wikipedia. It selects entities that are related each other in terms of any relationships. This approach is reasonable and works well if it can use enough contexts. In other words, though, this approach may perform worse on a short text that provides few contexts. In an extreme case, if we observe only two terms "Chicago" and "Detroit" in a document, it is quite hard to determine what they are (they can be *cities* or *ice hockey teams* etc.) from their relatedness. To make the most of few contexts, concrete relationships (such as *isa*, *attributeof*) should be introduced because they are more confident evidence than relatedness, which is represented by a value.

Based on a probabilistic taxonomy, known as Probase (see Sec. 2.2), Song et al. [24] proposed a conceptualization method. A naive Bayes probabilistic model is employed to derive the most dominant concepts from a set of terms. E.g. given two terms "Apple" and "Microsoft," it derives concepts such

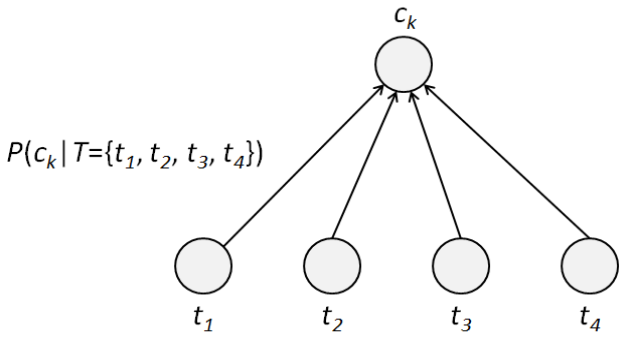

Figure 2: Conceptualization from a set of terms

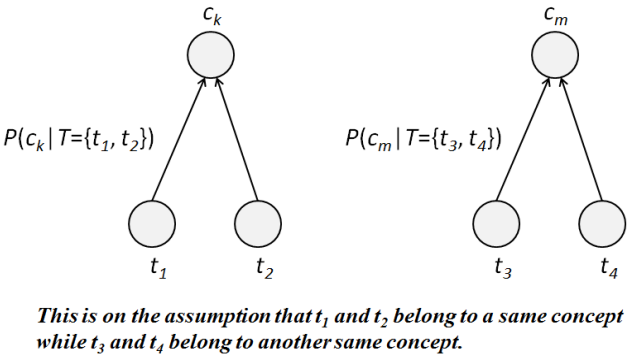

Figure 3: Conceptualization for each cluster

as *company* or *brand* according to a naive Bayes model. Figure 2 represents a conceptualization model proposed by Song et al. As the figure shows, the conceptualization method is on the assumption that all the terms belong to a same concept. Otherwise it derives much general concepts such as *name* or *thing*, which do not make sense for disambiguation. This is because all of the terms can belong to such general concepts. To enable it to work for terms that belong to multiple kinds of concepts, related terms should be clustered and conceptualized separately like Figure 3. For example, if a document consists of four terms "Apple," "Microsoft," "China" and "United States," two clusters should be created, i.e. "Apple" and "Microsoft" into a cluster of *companies*, and "China" and "United States" to a cluster of *countries*. Actually, Song et al. also proposed to use a simple threshold-based clustering of related terms based on the number of common concepts. However, since a document usually contains various kinds of concepts complexly, it is hard to obtain well-formed clusters. We give up the idea of solving the issue to cluster related terms clearly. Instead, we choose an alternative approach which does not need to determine the clusters.

To cope with a document that contains various kinds of concepts, we propose an indirect conceptualization method using a probabilistic clustering (Sec. 4.5). The core of our method is probabilistic clustering instead of deterministic clustering for applying a naive Bayes model. It computes the probability for each state of the cluster and conceptualizes all the clusters to aggregate the probabilities of concepts. To compute it in polynomial time, we transform a formula that models the conceptualization through probabilistic clustering. As a result, we prove that the conceptualization through probabilistic clustering is equivalent to the conceptualization using relatedness-based smoothing.

Term detection is also a challenging problem in a paradigm of entity disambiguation. This is due to both the incompleteness of a knowledgebase and the multiple interpretation of terms. For example, the organization "Children's Care Foundation" can be also detected as "Children," "Care" and "Foundation." Generally the longest term "Children's Care Foundation" is appropriate as the interpretation among all the possible terms. This identification process is based on how we identify an entity on text. That is, if we know the entity, we can easily detect it. However, if a knowledgebase does not contain "Children's Care Foundation," can we prevent it from detecting the incorrect short terms "Children," "Care" and "Foundation" separately? In practice, a human may be able to accurately detect the term "Children's Care Foundation" even if the person does not know it at all. This is based on how we detect a term when we do not have knowledge about it. In this case, we usually rely on its surrounding words to predict the correct segmentation of the term. Since we have empirically learned that successive capitalized

words are much likely to be one entity, we can identify "Children's Care Foundation" instead of take it apart. To enable machines to perform this human-like term detection, we leverage both linguistic and semantic information. In particular, our method uses entries stored in a knowledgebase to catch terms in a document. Besides, it utilizes POS tagging to predict which phrase is an entity. In our method, we combine both approaches to detect terms as a person does.

Our two challenges on entity disambiguation are supported by some preprocessing and postprocessing such as sentence breaking, concept acquisition and truecasing. We give a full detail of the methodology including term detection from next section.

4 Methodology

In this section, we describe a method for entity disambiguation based on Probase, a probabilistic taxonomy. We assume a document as the input, and the goal is to obtain dominant concepts for each entity found in the document. As the main component is entity disambiguation (Sec. 4.5), our method is divided into several components. First of all, we explain the overview of the method.

4.1 Overview

Figure 4 is an overview of our method. Broadly speaking, our method consists of:

- Sentence breaking
- Term detection
- Entity conceptualization
- Entity disambiguation

Given a document, our method first splits the document into sentences. Here each sentence is checked whether it should be processed as case-insensitive. Secondly, all possible terms (which can be entities or concepts) are detected from the sentences by using both semantic and linguistic information. In particular, candidates of the entity are determined according to a knowledgebase entries and proper noun phrase chunking, while all the concepts are simply detected by a knowledgebase using trie index. Thirdly, it obtains concepts for all the candidates of the entity with their probability. Finally, all the candidates are disambiguated using other terms found in the document. In particular, related terms are clustered and then conceptualized using a naive Bayes model. Here we employ probabilistic clustering instead of deterministic clustering. That is, it computes the probability for each state of the cluster based on the relatedness. Then it conceptualizes all the clusters to aggregate the probabilities of concepts. To reduce the calculation time to be polynomial, we formulate it using indirect inference. In the result, dominant concepts are obtained for each candidate of the entity. According to the dominant concepts, truecases are obtained for case-insensitive candidates. Some of the candidates may be discarded in the last.

4.2 Sentence Breaking

In our method, the input is assumed as a document. First, instead of detecting terms directly from a document, the document is split into sentences. Although this process can be skipped, the advantages of sentence breaking are not trivial; the potential of a POS tagger which is required in the next process (Sec. 4.3) can be brought out, and positions where case information may be missing can be identified.

Sentence breaking in English is basically done by using punctuations (i.e. period ".", exclamation mark "!", question mark "?" and linefeed). However, there are not a few cases when a punctuation does not mean the end of a sentence. Especially, with regard to period, the problem becomes challenging compared to the others. Basically a period may be used in: the end of a sentence (e.g. "Tom handed his book to Mary."), abbreviation (e.g. "George W. Bush"), acronym (e.g. "U.S."), ellipsis (e.g. "We have nothing to do..."), and number (e.g. "30.8 thousand dollars"). The problem arises because an abbreviation or an acronym may occur in the end of a sentence. We then employ simple rules to detect the end of sentences: 1) if it is period, then it is the end of a sentence, 2) if the preceding token is included in one of the terms defined in a knowledgebase, then it is not the end of a sentence, 3) if the next token is capitalized, then it is the end of a sentence. According to an article[2], it achieves about 95% of the accuracy. This simple rule-base approach may be enough to our requirement because our goal is not to

[2]http://www.attivio.com/blog/57-unified-information-access/263-doing-things-with-words-part-two-sentence-boundary-detection.html

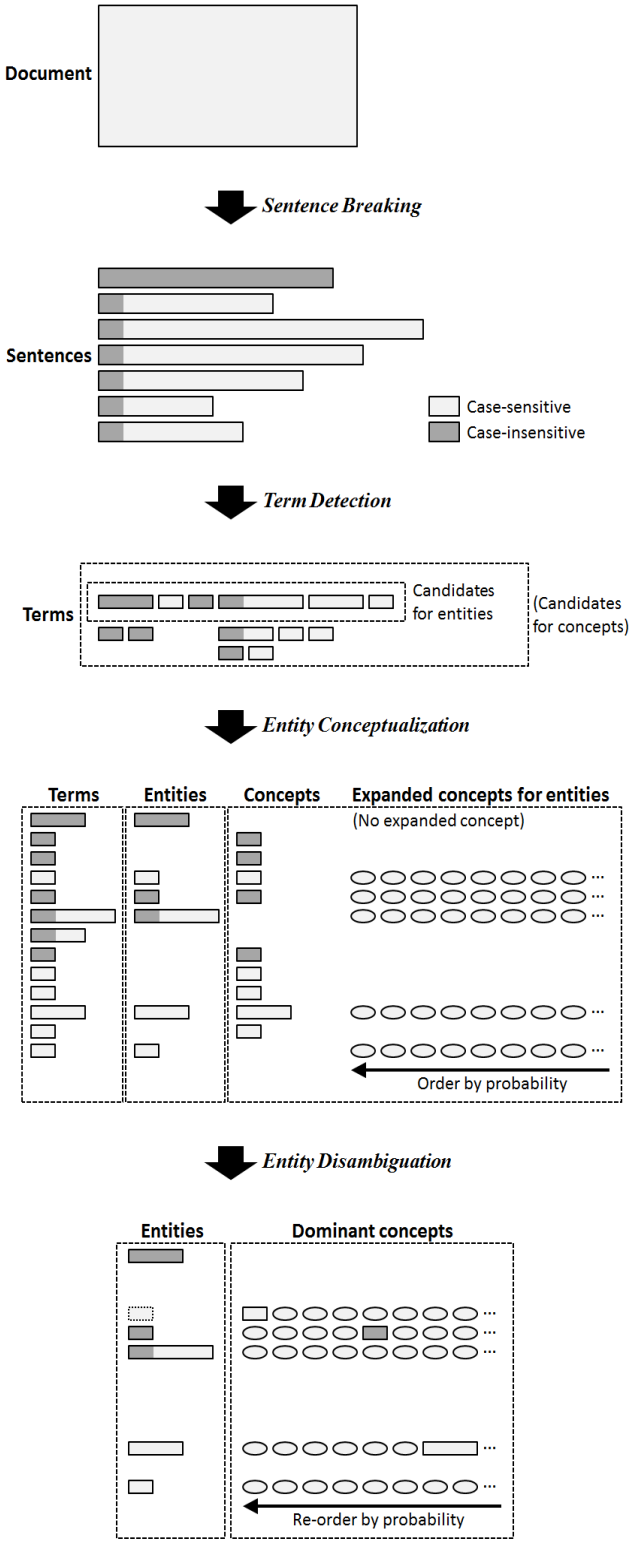

Figure 4: An overview of the method

completely detect sentence boundaries, but to obtain a complete term list from a document. In the same way, our method also removes redundant symbols using a knowledgebase.

After breaking sentences, our method identifies positions where case information may be missing. Generally the beginning of each sentence is capitalized because of orthographic conventions. Hence the first word of each sentence is dealt with as a case-insensitive word. In addition, if most of the words in a sentence are capitalized, it is likely that these words are capitalized on purpose (e.g. a title of an article). Then case information in the sentence may be missing.

4.3 Term Detection

Given a set of sentences obtained in Sec. 4.2, our method detects all possible terms (i.e. entities and concepts). To be precise, the problem of term detection is to obtain a set of terms $T = \{t_l, l \in 1, ..., L\}$ and to determine whether each of the terms can be an entity $t_l \in E$ or a concept $t_l \in C$ (though all terms can be concepts i.e. $C = T$ in our method because they are far less harmful in our disambiguation process even if they are incorrect). Note that t_l inherits case-insensitive positions from Sec. 4.2, and T does not allow duplication of terms.

Generally, detecting terms defined in a knowledgebase is done in a theoretically optimal way such as trie index [11]. That is, all terms which has a corresponding entry in a knowledgebase should be detected. However, in practice, it is not optimal in two reasons. One comes from the incompleteness of a knowledgebase. Although there are a number of knowledgebases as described in Sec. 3, there is no knowledgebase covering all entities and concepts in our mental world. It is obvious that terms cannot be detected if they are unknown to a knowledgebase. The other reason is that even if all detected terms are literally written in a document, they may be neither entities nor concepts. For example, when "New York Times" is found in a document, it may be detected as "New York Times," "New York," "New," "York," and "Times" for candidates of the entity. In such case, a simple and reliable way to detect the correct entity is to adopt the longest term among them. However, given that a knowledgebase is incomplete, the longest term may be absent from the knowledgebase. Then the shorter terms are regarded as entities, which are actually incorrect. E.g. "Children's Care Foundation" is an entity but not in Probase, then it is detected as "Children," "Care" and "Foundation" separately.

After all, if all terms are detected completely, the simple and reliable approach which adopts the longest term as an entity should work well. In other words, what is required here is to detect unknown terms to a knowledgebase. To solve this problem, we rely on the fact that unknown terms to a knowledgebase are basically proper nouns. Therefore we adopt a POS tagging technique for chunking proper noun phrases which should be candidates for the entity. Our method obtains a set of terms T derived from all the possible terms defined in Probase as well as all the proper noun phrases detected by a POS tagger. Then we can use the simple and reliable approach. That is, among terms conflicting at a same position, only the longest term t_l is regarded as an entity $t_l \in E$. While rarely, two terms are intersectional. In such case, a term with higher observed frequency is selected as an entity.

This approach should work well if the knowledgebase is clean i.e. does not contain noisy data. However, because Probase contains noisy data, it is possible that the longest term is still incorrect. Noisy terms are obtained because Probase is constructed automatically using syntactic patterns including the Hearst patterns as described in Sec 3. For example, from a sentence "... vigorous and reliable varieties such as 'Navelina' anywhere in the world, but China and Chile" which is found in the Web document, the Hearst pattern *such as* extracts "'Navelina' anywhere in the world," "but China" and "Chile" as entities of *vigorous and reliable varieties*. Thus, Probase unfortunately obtain incorrect entities "'Navelina' anywhere in the world" and "but China" and we have to remove these noisy terms from Probase. Here, we employ two indicators of the entity for detecting noisy terms according to Parameswaran's work [21]. The indicators are: 1) the observed frequency of the term is not low, and 2) the term is *better* than its sub/super-terms. In Parameswaran's work, given the term is k-gram, the term is checked whether *better* or not than both the prefix and the postfix $(k-1)$-grams (sub-terms) based on the ratio of their frequencies against that of their sub-terms. Instead of the ratio of frequencies, our method utilizes noun phrase chunking to determine whether the term is *better* than the sub-terms. This approach is much more intuitive because most of the entities are generally written as a noun phrase, and whether a term is a noun phrase or not is strongly related to whether the term is an entity or not. If either of the $(k-1)$-grams is a noun phrase, then the $(k-1)$-grams is *better* than the term of k-grams. Otherwise the k-grams is *better* than its sub-terms. Consequently, 1) if the observed frequency of the k-grams is lower than a predefined threshold θ (we heuristically determined $\theta = 3$ in our evaluation, Sec. 5), and 2) if either of the $(k-1)$-grams is a noun phrase, then the term of k-grams is removed as a noise.

Figure 5 is an example of term detection. In all the cases except fifth one, the longest term is regarded as an entity and the others can be only concepts. As for the fifth case, the longest term "but China"

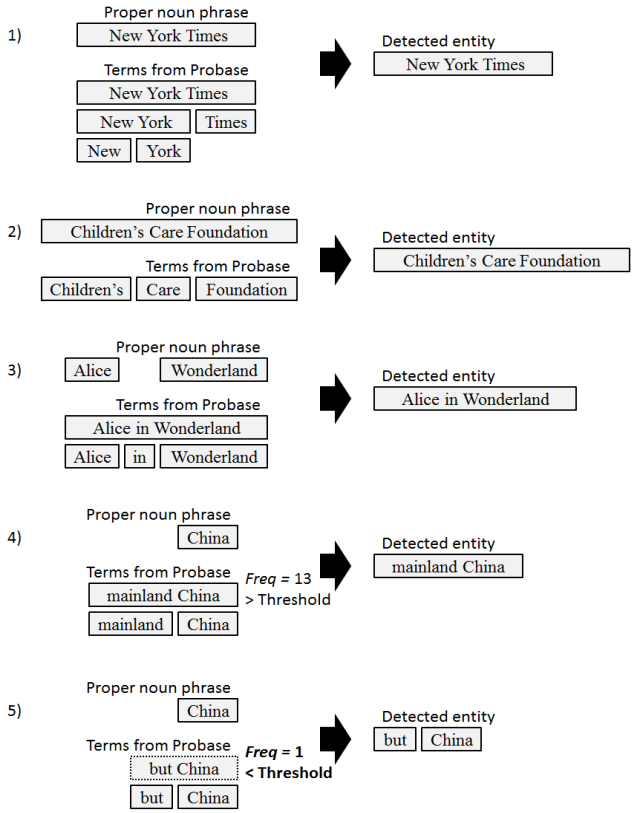

Figure 5: An example of term detection

satisfies the condition of noisy entity and is removed.

Our method also resolves in-document coreferences for entities. We employ a simple coreference technique used in Cucerzan's work [5]. In his work, short capitalized entities are mapped to longer entities (e.g. "Bush" can be mappled to "George W. Bush"). Our method follows his approach. Mapped short entities are then removed from the list of entities E.

4.4 Entity Conceptualization

For each term t_l in T, the probability of a concept given a term, i.e. $P(c_k|t_l)$ can be obtained using Probase. These expanded concepts are used in the main component of our method, entity disambiguation (Sec. 4.5).

In Probase, a term can be both an entity and a concept. If a term t is an entity $t^{(e)}$, the probability $P(c_k|t^{(e)})$ can be simply derived from Probase. If t is a concept $t^{(c)}$, the probability $P(c_k|t^{(c)})$ equals to 1 only if $c_k = t^{(c)}$. The problem is that how to combine the two probabilities when we observe a term which can be both an entity and a concept. Intuitively, the ratio of the observed frequency of an entity $t^{(e)}$ and a concept $t^{(c)}$ indicates how probable the term becomes an entity or a concept. Table 1 shows the observed frequencies of several terms. According to the frequencies, while "company," "fruits" and "animal" are likely to be concepts, "Apple," a capitalized term, is never a concept. "apple," "jaguar" and "cat" tend to be entities, but they may be also concepts with a little probabilities, which means it depends on the context of a document in which they occur. These figures fit in our intuition.

Based on the intuition, we define the probability as below:

$$P(t = t^{(e)}) = \frac{Freq(t^{(e)})}{Freq(t^{(e)}) + Freq(t^{(c)})} \tag{1}$$

$$P(t = t^{(c)}) = \frac{Freq(t^{(c)})}{Freq(t^{(e)}) + Freq(t^{(c)})} \tag{2}$$

where $Freq(t_l^{(e)})$ and $Freq(t_l^{(c)})$ are the observed frequency of an entity $t^{(e)}$ and a concept $t^{(c)}$ respectively. Therefore the probability of a concept c_k given a term t is computed in a generative mixture model, written

Table 1: The observed frequency

Term	Freq of entity	Freq of concept
apple	2390	231
Apple	9300	0
apple (case-insensitive)	11760	231
company	550	164718
fruits	2384	17757
jaguar	109	3
cat	1712	448
animal	1021	37818

by:

$$P(c_k|t) = P(c_k|t^{(e)})P(t = t^{(e)}) + P(c_k|t^{(c)})P(t = t^{(c)}) \tag{3}$$

4.5 Entity Disambiguation

Given a set of terms $T = \{t_l, l \in 1, ..., L\}$, our method disambiguates each of the term by means of the conceptualization, i.e. it predicts the dominant concepts (e.g. *company*, *brand*, etc.) of a term (e.g. "Apple"). In practice, only the terms which can be an entity $t_l \in E$ are disambiguated in this process. The conceptualization of a term is based on the method proposed by Song et al. [24], in which the whole document is conceptualized using a naive Bayes probabilistic model as below:

$$P(c_k|T) \propto P(c_k) \prod_{l=1}^{L} P(t_l|c_k) \propto \frac{\prod_l^L P(c_k|t_l)}{P(c_k)^{L-1}} \tag{4}$$

where $P(c_k|t_l)$ is the probability of a concept c_k given a term t_l, and $P(c_k)$ is approximately proportional to the observed frequency of c_k. $P(c_k|t_l)$ can be derived from Probase as shown in Sec. 4.4. We apply Eq. (4) for conceptualizing each term in the document, i.e. conceptualizing each term t by using the other terms in the same document to organize a subset of terms T_t from T and compute $P(c_k|T_t)$.

Actually Eq. (4) may not work for conceptualizing the whole document. This is because a document generally contains many kinds of concepts and Eq. (4) is on the assumption that all terms in T belong to a same concept. If we try to conceptualize terms which belong to several kinds of concepts by Eq. (4), the result, i.e. the dominant concepts for these terms may be quite vague such as *name* and *thing*. This is because such concepts are the most probable common concepts of the input terms. To apply it for conceptualizing each term t in the document, it needs to select related terms to organize T_t. For example, we consider a document consisting of entities "Apple," "Microsoft," "U.S.," and "China." The disambiguation of "Apple" here is supposed to be *company*, *brand* and so on, and not *fruits*. In this case, we have to select only "Microsoft" for disambiguating "Apple" because only "Microsoft" belongs to the same concept of "Apple." Actually "U.S." and "China" also belong to the same concept of "Apple", but this concept should be semantically much more vague than that of "Microsoft" and "Apple." To be concrete, both of the probabilities of the common concept c_1 of "U.S." (or "China") and "Apple" $P(c_1|$"U.S."$)$, $P(c_1|$"Apple"$)$ is far less than the probability of the common concept c_2 of "Microsoft" and "Apple" $P(c_2|$"Microsoft"$)$, $P(c_2|$"Apple"$)$.

Consequently, for each term t, the disambiguation is done by the following two steps:

1. Organize a subset of terms T_t from T. T_t consists of terms related to t.
2. Compute $P(c_k|T_t)$ to conceptualize T_t.

In the first step, related terms $t_l \in T_t$ are selected by how much their concepts are related to that of the term t. Namely, a term t_l is related to t if both concept vectors $\{P(c_k|t_l), k \in 1, ..., K\}$ and $\{P(c_k|t), k \in 1, ..., K\}$ are related. If the relatedness exceeds a predefined threshold, we regard the term is related to the target term. Practically the relatedness is computed by using well-known metrics such as Cosine similarity, Jaccard index or Dice coefficient. Figure 6 denotes the direct disambiguation method based on Song's conceptualization mechanism. Given a full set of terms in the document $T = \{t_1, t_2, t_3, t_4\}$ and a predefined threshold θ, it first computes the relatedness between each term and t. In the case of Figure 6, only t_1 and t_2 satisfy θ, then the subset of terms T_t is determined as $T_t = \{t_1, t_2\}$. After that, Song's conceptualization method is applied to T_t to obtain the dominant concepts of t at the document.

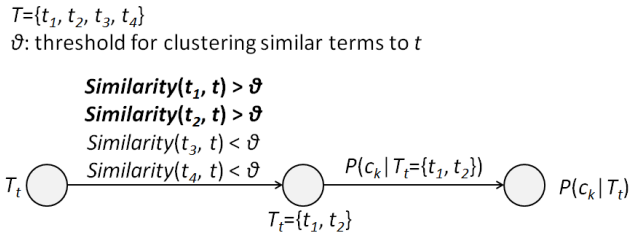

Figure 6: Entity disambiguation by related terms

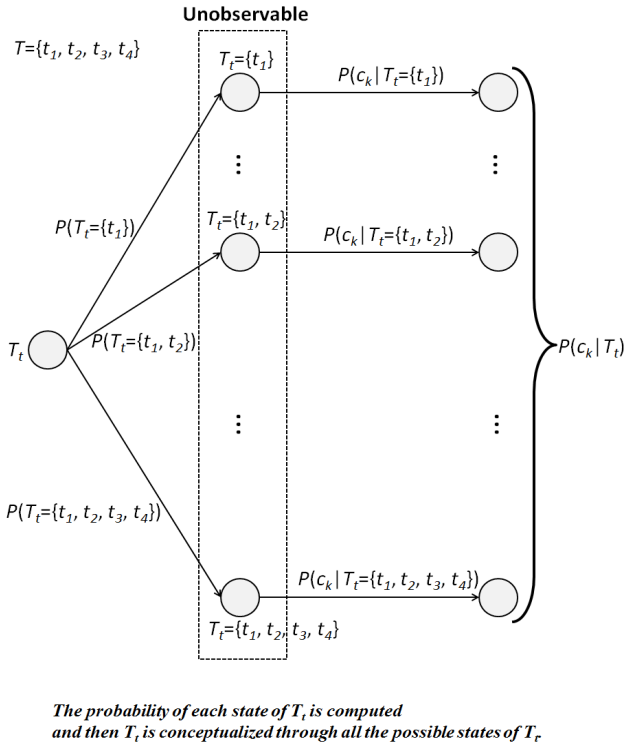

Figure 7: Indirect entity disambiguation through probabilistic clustering

According to the evaluation (Sec. 5), this two-step method successfully disambiguate entities which occur in a news article accomplishing a very competitive accuracy only if the threshold for clustering related terms is defined properly. However, it is difficult to define the appropriate threshold on ahead. On our hypothesis, the appropriate threshold is different depending on the target document. In fact, there were several cases that the method with improper threshold could disambiguate while the method with the most appropriate threshold could not. In order to cope with this problem, the threshold needs to be defined automatically according to the target document.

Actually, this problem cannot be solved squarely. The challenging problem lies in that there is no way to know the true subset of terms T_t which really affects t. Therefore we propose a method to probabilistically predict T_t and indirectly compute $P(c_k|T_t)$. Figure 7 denotes the disambiguation method that we propose. In contrast to Figure 6, the method does not determine the state of T_t. Instead, it first predict the probabilities for all the possible states of T_t since T_t is unobservable. E.g. if the full set of terms in the document is $T = \{t_1, t_2, t_3, t_4\}$, our method computes $P(c_k|T_t)$ for all the states (totally 15 states) of T_t.

To formulate our method, we introduce an auxiliary variables r_l to indicate the state whether t_l affects the term t. Particularly $r_l = 1$ if t_l affects t, and $r_l = 0$ if t_l does not affect t. Given the independence assumption for terms, the probability that T_t consists of t_l with its variable $r_l = 1$ is given by:

$$P(T_t = \{t_l, r_l = 1\}) = \prod_{r_l=1} P(t_l \in T_t) \prod_{r_l=0} P(t_l \notin T_t)$$

$$= \prod_{r_l=1} P(t_l \in T_t) \prod_{r_l=0} (1 - P(t_l \in T_t)) \tag{5}$$

where $P(t_l \in T_t)$ denotes the probability that t_l belongs to T_t (or t_l affects t). We define $P(t_l \in T_t)$ by computing the relatedness between both concept vectors $\{P(c_k|t_l), k \in 1, ..., K\}$ and $\{P(c_k|t), k \in 1, ..., K\}$. In our evaluation, cosine metric was employed to compute the relatedness. This was heuristically determined according to the stability of the result in our preliminary experiment. From Eq. (4) and (5), our disambiguation method is formulated as below:

$$P(c_k|T_t) \propto \sum_{(r_l)\in\{0,1\}^L} \left(P(T_t = \{t_l, r_l = 1\}) \frac{\prod_{r_l=1} P(c_k|t_l)}{P(c_k)^{Count(r_l=1)-1}} \right) \tag{6}$$

where $Count(r_l = 1)$ is the number of r_l which equals to 1, in turn the number of terms which affects t. Eq. (6) performs the conceptualization for each state of T_t and sums up them with multiplied by the probability that T_t becomes the state. In fact, this equation needs exponential time and cannot be applied. Hence we have to reduce it to polynomial time. The Eq. (6) is transformed into:

$$\sum_{(r_l)\in\{0,1\}^L} \left(\prod_{r_l=1} P(t_l \in T_t) \prod_{r_l=0} \big(1 - P(t_l \in T_t)\big) \frac{\prod_l^L P(c_k|t_l)^{r_l} P(c_k)^{1-r_l}}{P(c_k)^{L-1}} \right) \tag{7}$$

and then:

$$\frac{\sum_{(r_l)\in\{0,1\}^L} \left(\prod_{r_l=1} P(t_l \in T_t) P(c_k|t_l) \prod_{r_l=0} \big(1 - P(t_l \in T_t)\big) P(c_k) \right)}{P(c_k)^{L-1}} \tag{8}$$

The numerator of Eq. (8) can be calculated separately by each t_l. It is derived by recurrence formula, starting from L by:

$$\begin{aligned}
&\Big(P(t_L \in T_t)P(c_k|t_L) + \big(1 - P(t_L \in T_t)\big)P(c_k)\Big) \\
&\sum_{(r_l)\in\{0,1\}^{L-1}} \left(\prod_{r_l=1} P(t_l \in T_t) P(c_k|t_l) \prod_{r_l=0} \big(1 - P(t_l \in T_t)\big) P(c_k) \right) \\
&= \Big(P(t_{L-1} \in T_t)P(c_k|t_{L-1}) + \big(1 - P(t_{L-1} \in T_t)\big)P(c_k)\Big) \\
&\quad \Big(P(t_L \in T_t)P(c_k|t_L) + \big(1 - P(t_L \in T_t)\big)P(c_k)\Big) \\
&\quad \sum_{(r_l)\in\{0,1\}^{L-2}} \left(\prod_{r_l=1} P(t_l \in T_t) P(c_k|t_l) \prod_{r_l=0} \big(1 - P(t_l \in T_t)\big) P(c_k) \right) \\
&= \cdots \\
&= \Big(P(t_1 \in T_t)P(c_k|t_1) + \big(1 - P(t_1 \in T_t)\big)P(c_k)\Big) \\
&\quad \vdots \\
&\quad \Big(P(t_L \in T_t)P(c_k|t_L) + \big(1 - P(t_L \in T_t)\big)P(c_k)\Big)
\end{aligned} \tag{9}$$

After all, the conceptualization of t in a document is computed as:

$$P(c_k|T_t) \propto \frac{\prod_l^L \Big(P(t_l \in T_t)P(c_k|t_l) + \big(1 - P(t_l \in T_t)\big)P(c_k)\Big)}{P(c_k)^{L-1}} \tag{10}$$

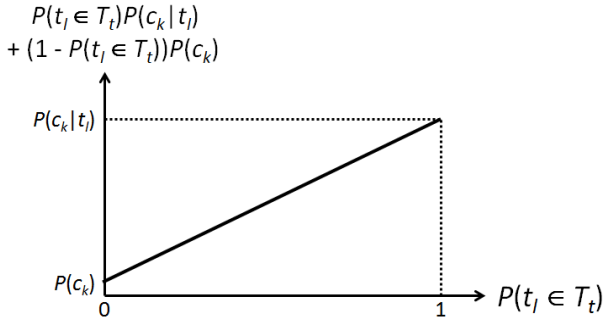

Figure 8: Similarity-based smoothing

where $P(c_k)$ is approximately computed by the observed frequency of a concept c_k divided by the total frequency of all concepts.

Eq. (10), eventually, becomes a quite similar form to the original conceptualization method shown as Eq. (4). In particular, $P(c_k|t_l)$ in Eq. (4) is replaced by a linear combination $P(t_l \in T_t)P(c_k|t_l) + \big(1 - P(t_l \in T_t)\big)P(c_k)$. This formula properly acts as a relatedness-based smoothing. Figure 8 describes the mechanism of the smoothing. Along with the probability $P(t_l \in T_t)$, the value moves on the line from $P(c_k|t_l)$ to $P(c_k)$. The effect of the smoothing becomes larger as $P(t_l \in T_t)$ decreases. If $P(t_l \in T_t) = 0$, the value becomes $P(c_k)$ and is reversed with $P(c_k)$ in the denominator in Eq. (10), consequently affecting nothing to the output.

According to the dominant concepts, truecases are obtained for case-insensitive terms. Specifically, a term that belongs to the most dominant concept with the highest probability is selected as the representative of the terms. For example, when we obtain the most dominant concept "company" for a case-insensitive term "apple," we then select a capitalized term "Apple" for representing the case-insensitive term because "Apple" is a *company* with the highest probability among all the possible terms "apple," "Apple" and "APPLE."

5 Evaluation

5.1 Setup

Our method disambiguates entities by predicting their dominant concepts. However, to the best of our knowledge, there is no gold standard metric for evaluating fine-grained named entity recognition methods. Therefore we evaluate our method in terms of entity linking. In particular, we compare our method with Hoffart's method [16]. This is for the following reasons. First, it is a state-of-the-art method for entity disambiguation as of October 2011. Second, it is compared with several benchmarks including Kulkarni's method [17] and Cucerzan's method [5], and indirectly with other simpler methods such as Milne's method [19]. Third, it is evaluated by using a dataset which is available online. The dataset can be generated based on CoNLL2003 named entity recognition shared task[3] [23]. All the proper nouns were hand-annotated with corresponding entities both in Wikipedia and Freebase. Each term was disambiguated by two annotators (students), and in case of conflict, they were resolved by Hoffart et al. This answer data is available at their Web page[4]. According to Hoffart's work, the statistics of the CoNLL2003 dataset is as shown in Table 2. The setup in the evaluation basically follows the experiment conducted in Hoffart's work because we want to compare our method in the same condition for the sake of comparability.

The evaluation metrics are macro-average precision and micro-average precision following Hoffart's work.

- Macro average precision: precision of entity linking first averaged by all terms in each document and then averaged by all documents. This reflects
- Micro average precision: precision of entity linking averaged by detected terms across all documents.

Especially we emphasize macro-average precision for evaluating disambiguation methods. This is because the length of each document in CoNLL2003 is quite different and short documents far less affect micro-average precision. Moreover, NIL entities (entities which are not defined in the target knowledgebase)

[3] http://www.cnts.ua.ac.be/conll2003/ner/
[4] http://www.mpi-inf.mpg.de/yago-naga/aida/

Table 2: CoNLL2003 dataset statistics

Statistics	Value
Articles	1,393
Terms (total)	34,956
Terms with no entity	7,136
Words per article (avg.)	216
Terms per article (avg.)	25
Distinct Terms per article (avg.)	17
Terms with candidate in KB (avg.)	21
Entities per term (avg.)	73
Initial annotator disagreement (%)	21.1

Table 3: Results of entity disambiguation on CoNLL2003 dataset (values in %)

Metric	Our methods		AIDA [16]	Kulkarni et al. [17]	Cucerzan [5]
	Similar-or-not (optimal)	Similarity-based			
Macro precision	83.14	**83.57**	82.02	76.74	43.74
Micro precision	80.26	81.40	82.29	72.87	51.03

were ignored in accordance with Hoffart's manner. That is, we considered only terms which should be linked to a known entity in the knowledgebase. After all, we ignored approximately 20% of the terms in the dataset.

We compared our method with AIDA (Hoffert et al.) [16], Kulkarni's method [17] and Cucerzan's method [5] on 231 of CoNLL2003 test documents. All methods except our method were trained by a SVM classifier on 946 of CoNLL2003 training documents to determine weights for combining components. In addition, the parameters of AIDA were tuned by using 216 of CoNLL2003 development documents. Our method does not need any parameter tuning. To demonstrate the effectiveness of the relatedness-based method (probabilistic clustering, Eq. (10)), we also evaluate the related-or-not method (deterministic clustering, Eq. (4)). As for the threshold for clustering the related terms in the related-or-not method, we checked the precision by several values on 216 of CoNLL development documents and employed the best one.

Since our method outputs dominant concepts ordered by the probability for each detected term, we have to assign one of the entities for each term based on the dominant concepts. To link terms to corresponding entities, we utilized Freebase. The entity linking was done by following procedure: search Freebase entities by the term using the Web interface and obtain the entities which the dominant concept appears in the Freebase types or the first statement of. If the candidates are more than one, we obey the rank by Freebase Web interface, i.e. we choose an entity with the highest rank. In case that a term which should be linked to an entity in Freebase is actually not detected by our method, we immediately regarded it as incorrect.

Furthermore, we also evaluated our term detection method on the same test set. We compared our method to the Stanford NER Tagger [9] because AIDA uses it for identifying terms in documents. In this experiment, we used the Stanford NER Tagger trained on CoNLL2003 training documents as a comparative method. Since our method utilizes proper noun chunking for detecting terms, we need a POS tagger. In our evaluation, we employed the Stanford POS Tagger [26] to find proper noun phrases from documents. The Stanford POS Tagger is trained using Penn Treebank tagset (Wall Street Journal), which ensuring the applicability of our method. As metrics for evaluating term detection methods, we employed precision, recall and F_1-measure across all documents. We do not ignore NIL entities in the evaluation for term detection. This is because identifying such entities is crucial for some applications including knowledgebase population and query suggestion.

5.2 Results and Discussion

Table 3 and Table 4 respectively shows the results of entity disambiguation and term detection on 231 of CoNLL2003 test documents.[5] First of all, for both macro-average and micro-average precision,

[5]The values of Kulkarni's method and Cucerzan's method were calculated for 229 documents because 2 of the 231 test documents could not be processed by Kulkarni's method. We reused the experimental results by Hoffart et al.

Table 4: Results of term detection on CoNLL2003 dataset (values in %)

Metric	Our method	Stanford NER [9]
Precision	92.84	94.38
Recall	**95.30**	92.22
F_1-measure	**94.06**	93.29

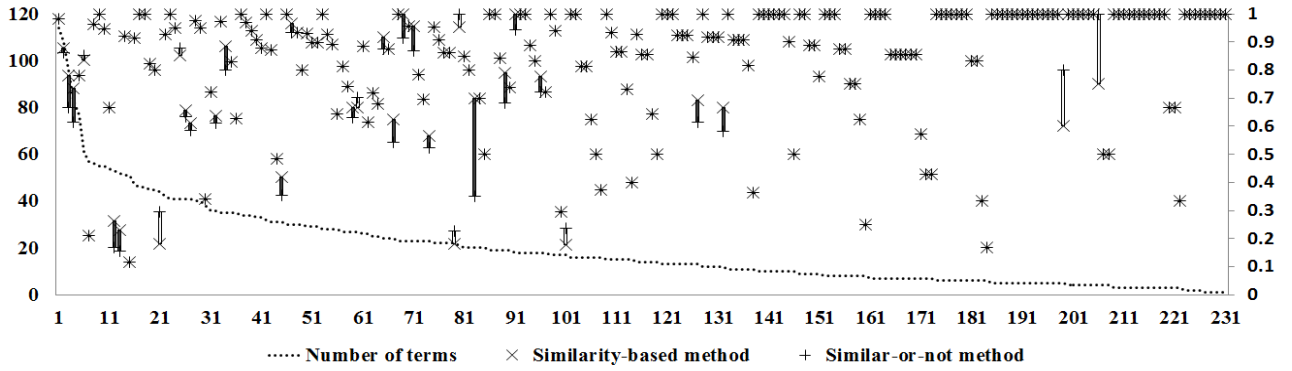

Figure 9: Precision of our methods on each document ordered by the number of terms

our relatedness-based method outperforms the related-or-not method that is almost optimally tuned using the development documents. This demonstrates the effectiveness of our relatedness-based method in terms of not only the reduction of parameter tuning but also the precision. We can say that our method automatically adjusts the effect of other related terms depending on the topic in the document. For example, if a document contains terms "Liverpool," "Barcelona" and "London," it may talk about *Europian football*, and "Liverpool" and "Barcelona" are entities of *football clubs* while "London" is an entity of *city*. Actually "Liverpool" and "Barcelona" can be also *cities* if "London" is used to disambiguate them, but our relatedness-based method can differentiate "Liverpool" and "Barcelona" from "London" according to their relatedness. In fact, the relatedness of "Liverpool" and "Barcelona" is relatively higher than that of "Liverpool" and "London" ("Barcelona" and "London"), which enables our method to differentiate them. Similar-or-not method can perform this only if the threshold is configured to separate them.

Our method also outperforms all the other methods in macro-average precision though micro-average precision is a little lower than that of AIDA. The improvement of macro-average precision indicates our method is robust across all the test documents and effective on short texts. Figure 9 represents the precision of our methods on each of 231 test documents ordered by the number of terms that should be resolved. The number of terms in a document approximately corresponds to the length of the document. To emphasize the difference between relatedness-based method and related-or-not method, single lines and double lines are drawn between them (single lines mean improvements by relatedness-based method). On short documents (top right of Figure 9), our methods seem to perform overwhelmingly good. In order to testify this, we calculated precisions on divided 4 subsets of test documents according to the length of a document. 4 subsets consist of approximately the same number of terms. Table 5 shows the macro-average and micro-average precision on the subsets. Only on subset 4, which consists of 135 out of 231 test documents, we recalculated precisions for AIDA with the same parameters as the original work. The Web interface of AIDA[6] had a limitation of text-length of an input and only the documents in subset 4 were fully processed. From the results on subset 4, both of our methods achieve significant improvements in comparison with AIDA. This is because they use extra information of concepts derived from Probase, as also mentioned in the work on short text conceptualization [24]. In short text, enhancing information using a knowledgebase is crucial to disambiguate terms. In addition, it is more favorable when enhanced information is more informative. For example, from a very short-text document containing two terms "INDIANAPOLIS" and "Philadelphia" as *American football teams*, our method correctly predicted both of them by leveraging their concepts (*isa* relationships). AIDA also used extra information of similarity based on Wikipedia, but only identified "INDIANAPOLIS" correctly. This represents that concepts are more informative than similarity. In fact, it is intuitive that *isa* relationships are informative compared to similarity or relatedness, which only denotes a value.

[6]https://d5gate.ag5.mpi-sb.mpg.de/webaida/

Table 5: Results of entity disambiguation on divided 4 subsets of CoNLL2003 dataset according to the length of documents (values in %)

Subset	Metric	Similar-or-not (optimal)	Similarity-based	AIDA [16]
Subset 1 (long texts)	Macro precision	69.54	71.94	-
	Micro precision	72.85	75.79	-
Subset 2	Macro precision	79.89	80.24	-
	Micro precision	79.73	80.00	-
Subset 3	Macro precision	80.18	81.85	-
	Micro precision	82.45	84.01	-
Subset 4 (short texts)	Macro precision	**86.53**	**86.30**	78.83
	Micro precision	**83.96**	**83.92**	82.93

Not only for short-text documents, our methods achieves high precision across all the documents. However, in some of the long-text documents (bottom left of Figure 9), our method works badly, which may resulting in a little worse performance in micro-average precision than AIDA (Table 3). In practice, it also causes the deterioration of precisions in subset 1 (Tabletb:4results). We delved into the cause of the low precision on the long-text documents and found that our method worked quite bad when almost all of the terms in a long-text document had same two concepts. For example, if a document contains "Toronto," "Buffalo," "Anaheim" and other dozens of related terms which can be both *cities* and *ice hockey teams*, our method tends to output the wrong dominant concept *city* for all of the terms based on the naive Bayes approach. This results in very poor precision in the document and affects the micro-average precision. To grasp the topic correctly, we need more information like context terms other than *isa* concepts. In fact, this document also contained a context term "National Hockey League" to catch the correct concept. Our method only uses *isa* relationships to disambiguate entities for the sake of the versatility of our probabilistic clustering approach. We can extend our method to be more specialized for entity disambiguation by using other kinds of relationships or context information.

It is noteworthy that our method does not need any parameter tuning while AIDA requires several parameters. Although Hoffart et al. mentions that one of the parameters of AIDA affects the deterioration of the precision within 1%, AIDA should work worse on the condition that there is no available training/development dataset. In addition to this, our method originally outputs the dominant concepts, does not link terms to entities. We calculated the accuracy of the dominant class and found that our method achieved **85.77%** of the macro-average precision and 83.75% of the micro-average precision.

As for term detection, our method also achieves improvements on recall (Table 4) compared to the Stanford NER Tagger which is used in AIDA system. Specifically, our method accurately detected 2.46% out of 7.16% of all terms, i.e. over one-third of terms that the Stanford NER Tagger could not detect. Although precision is sacrificed slightly, F_1-measure still exceeds that of the Stanford NER Tagger. It is effected by the combination of linguistic and semantic information. Our method basically leverages entries stored in a knowledgebase to catch terms in a document. This corresponds to that we can identify an entity on text if we know it. Besides, it utilizes POS tagging to predict which phrase is an entity. This corresponds to that we predict an unknown entity by depending on its surrounding words. For the Stanford NER Tagger, it mostly performs using the second approach. Since our method combines both the first and second approaches, it can detect terms that the Stanford NER Tagger cannot. For example, from a portion of a document "... make his Sheffield Wednesday comeback against ..." the Stanford NER Tagger could not identify "Sheffield Wednesday" (a *professional football club*) as a term. This is because it is difficult to predict it as an entity without the knowledge. On the other hand, since we have an entry "Sheffield Wednesday" in the knowledgebase, it is easily detected. In another example, from "... (a Newmont-Santa Fe deal) ..." and "... estimated by Bre-X to contain ...," the Stanford NER Tagger detected "Newmont-Santa Fe" and "Bre-X" as terms because both contain a hyphen bridging capitalized words. In fact, "Bre-X" is correct but "Newmont-Santa Fe" should be detected "Newmont" and "Santa Fe" separately. In our method, since the knowledgebase knows both Newmont and Santa Fe can be terms while "Newmont-Santa Fe" cannot, "Newmont-Santa Fe" is divided into two terms. In the case of "Bre-X," it can be a term according to the knowledgebase and thus it is detected without separation.

6 Conclusion

In this paper, we propose a method for disambiguating entities which occur in a document. Focusing on a term in a given document, the method first predicts a cluster consisting of related terms found in the same document, and then conceptualize the cluster using Bayesian inference. Instead of deterministically clustering related terms, we employ a probabilistic clustering technique, because the former requires to define a sensitive threshold for determining which term should be in the same cluster of the target term. All the possible clusters are conceptualized to aggregate the probability of each concept. Compared to the deterministic clustering approach or some competitive entity disambiguation methods, our method achieves improvements, especially on short texts.

In our future work, we focus on much deeper semantic tasks toward text understanding. For example, detecting relationships among entities in a document may be quite useful for computers to understand text. Text summarization also requires deeper semantic analysis of text including keyword extraction, entity disambiguation, conceptualization and so on. Still, entity disambiguation has been one of tasks since our method works not enough for real-world documents. Our method in this work only provides a formal model based on a naive Bayes probabilistic model whereas real-world documents contain a lot of noises which cannot be formalized. It is thought that some heuristic rules or particular methods are needed to cope with the noise.

References

[1] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. G. Ives. DBpedia: A Nucleus for a Web of Open Data. In *Proceedings of International Semantic Web Conference, Asian Semantic Web Conference (ISWC/ASWC)*, pages 722–735, Nov. 2007.

[2] M. Banko, M. J. Cafarella, S. Soderland, M. Broadhead, and O. Etzioni. Open Information Extraction from the Web. In *Proceedings of International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2670–2676, Jan. 2007.

[3] P. Bloom. Glue for the mental world. *Nature*, 421(6920):212–213, Jan. 2003.

[4] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor. Freebase: A Collaboratively Created Graph Database For Structuring Human Knowledge. In *Proceedings of ACM SIGMOD International Conference on Management of Data (SIGMOD)*, pages 1247–1249, June 2008.

[5] S. Cucerzan. Large-Scale Named Entity Disambiguation Based on Wikipedia Data. In *Proceedings of Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 708–716, June 2007.

[6] S. Dill, N. Eiron, D. Gibson, D. Gruhl, R. Guha, A. Jhingran, T. Kanungo, S. Rajagopalan, A. Tomkins, J. A. Tomlin, and J. Y. Zien. SemTag and Seeker: Bootstrapping the Semantic Web via Automated Semantic Annotation. In *Proceedings of International World Wide Web Conference (WWW)*, pages 178–186, May 2003.

[7] O. Etzioni, M. Cafarella, D. Downey, S. Kok, A.-M. Popescu, T. Shaked, S. Soderland, D. S. Weld, and A. Yates. Web-Scale Information Extraction in KnowItAll (Preliminary Results). In *Proceedings of International World Wide Web Conference (WWW)*, pages 100–110, May 2004.

[8] C. Fellbaum. *WordNet: An Electronic Lexical Database*. The MIT Press, May 1998.

[9] J. R. Finkel, T. Grenager, and C. Manning. Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling. In *Proceedings of Meeting of the Association for Computational Linguistics (ACL)*, pages 363–370, June 2005.

[10] M. Fleischman and E. Hovy. Fine Grained Classification of Named Entities. In *Proceedings of International Conference on Computational Linguistics (COLING)*, Aug. 2002.

[11] E. Fredkin. Trie Memory. *Communications of the ACM (CACM)*, 3(9):490–499, Sept. 1960.

[12] C. Giuliano. Fine-Grained Classification of Named Entities Exploiting Latent Semantic Kernels. In *Proceedings of Conference on Computational Natural Language Learning (CoNLL)*, pages 201–209, June 2009.

[13] C. Giuliano and A. Gliozzo. Instance-Based Ontology Population Exploiting Named-Entity Substitution. In *Proceedings of International Conference on Computational Linguistics (COLING)*, pages 265–272, Aug. 2008.

[14] X. Han, L. Sun, and J. Zhao. Collective Entity Linking in Web Text: A Graph-Based Method. In *Proceedings of International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 765–774, July 2011.

[15] M. A. Hearst. Automatic Acquisition of Hyponyms from Large Text Corpora. In *Proceedings of International Conference on Computational Linguistics (COLING)*, pages 539–545, Aug. 1992.

[16] J. Hoffart, M. A. Yosef, I. Bordino, H. Furstenau, M. Pinkal, M. Spaniol, B. Taneva, S. Thater, and G. Weikum. Robust Disambiguation of Named Entities in Text. In *Proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 782–792, July 2011.

[17] S. Kulkarni, A. Singh, G. Ramakrishnan, and S. Chakrabarti. Collective Annotation of Wikipedia Entities in Web Text. In *Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 457–465, June/July 2009.

[18] D. B. Lenat and R. V. Guha. *Building Large Knowledge-Based Systems: Representation and Inference in the Cyc Project*. Addison-Wesley, Jan. 1990.

[19] D. Milne and I. H. Witten. Learning to Link with Wikipedia. In *Proceedings of ACM Conference on Information and Knowledge Management (CIKM)*, pages 509–518, Oct. 2008.

[20] G. L. Murphy. *The Big Book of Concepts*. The MIT Press, Aug. 2002.

[21] A. Parameswaran, H. Garcia-Molina, and A. Rajaraman. Towards The Web Of Concepts: Extracting Concepts from Large Datasets. In *Proceedings of International Conference on Very Large Data Bases (VLDB)*, pages 566–577, Sept. 2010.

[22] S. P. Ponzetto and M. Strube. Deriving a Large Scale Taxonomy from Wikipedia. In *Proceedings of National Conference on Artificial Intelligence (AAAI)*, pages 1440–1445, July 2007.

[23] E. F. T. K. Sang and F. D. Meulder. Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. In *Proceedings of Conference on Computational Natural Language Learning (CoNLL)*, May/June 2003.

[24] Y. Song, H. Wang, Z. Wang, H. Li, and W. Chen. Short Text Conceptualization Using a Probabilistic Knowledgebase. In *Proceedings of International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2330–2336, July 2011.

[25] F. M. Suchanek, G. Kasneci, and G. Weikum. YAGO: A Core of Semantic Knowledge. In *Proceedings of International World Wide Web Conference (WWW)*, pages 697–706, May 2007.

[26] K. Toutanova, D. Klein, C. Manning, and Y. Singer. Feature-Rich Part-of-Speech Tagging with a Cyclic Dependency Network. In *Proceedings of Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL)*, pages 252–259, May/June 2003.

[27] W. Wu, H. Li, H. Wang, and K. Q. Zhu. Towards a Probabilistic Taxonomy of Many Concepts. Technical Report MSR-TR-2011-25, Microsoft Research, Mar. 2011.