Name List Only? Target Entity Disambiguation in Short Texts

Yixin Cao[1], **Juanzi Li**[1], **Xiaofei Guo**[1], **Shuanhu Bai**[2], **Heng Ji**[3], **Jie Tang**[1]
[1] Tsinghua National Laboratory for Information Science and Technology
Dept. of Computer Science and Technology, Tsinghua University, China 100084
[2] Sina Corporation, China 100084
[3] Dept. of Computer Science, Rensselaer Polytechnic Institute, USA 12180
{caoyixin2011,lijuanzi2008,sophiaguo.thu,jery.tang}@gmail.com
shuanhu@staff.sina.com.cn, jih@rpi.edu

Abstract

Target entity disambiguation (TED), the task of identifying target entities of the same domain, has been recognized as a critical step in various important applications. In this paper, we propose a graph-based model called TremenRank to collectively identify target entities in short texts given a name list only. TremenRank propagates trust within the graph, allowing for an arbitrary number of target entities and texts using inverted index technology. Furthermore, we design a multi-layer directed graph to assign different trust levels to short texts for better performance. The experimental results demonstrate that our model outperforms state-of-the-art methods with an average gain of 24.8% in accuracy and 15.2% in the F1-measure on three datasets in different domains.

1 Introduction

Currently, a growing number of people prefer to express their views and comments online. These messages, which are updated at the rate of millions per day, become a potentially rich source of information. From such a large number of texts, entity disambiguation is a critical step when extracting text information from these messages, and for various applications, such as natural language processing and knowledge acquisition (Dredze et al., 2010). Take the application of customer feedback analysis (CFA) as an example. An enterprise is typically interested in public reviews of its own products as well as those of its competitors'; the identification of these entities is thus critical for further analysis.

The product names comprise a list of entities to be identified. We refer to these entities of the same domain as **target entities**, and the identification process is called **target entity disambiguation** (Wang et al., 2012). All of target entities (e.g., *car brands*) share a common domain, which we refer to as the **target domain**. The target domain is the only constraint to target entities, which implies that the entities are in a specific domain, rather than being general things. In the case of entity recognition from short texts, the disambiguation can be performed on the document level. Given a collection of short documents, our goal is to determine which documents contain the target entities.

Challenge and Related Work

In contrast to traditional entity disambiguation tasks, TED in short texts can require as little information as a name list. There are three types of information that are utilized in Named Entity Disambiguation related tasks: knowledge resources, the context in which a target word occurs and statistical information (Navigli, 2009). However, the lack of the first two types of information makes this problem more challenging.

Knowledge Sparsity A large number of methods focus on using knowledge bases (KBs) like Wikipedia or YAGO to enrich the named entities (e.g., *in-links and out-links*) (Hoffart et al., 2011; Bunescu and Pasca, 2006; Milne and Witten, 2008; Kulkarni et al., 2009; Han et al., 2011; Mihalcea and Csomai, 2007; Shen et al., 2012). These methods compared the context of the entities and their reference pages in the KBs through a similarity measurement. However, we tested 32 different names of General Motors (GM) car brands, and only four of the brands exist in Wikipedia. This circumstance is not unusual. For another larger dataset that included 2468 stock names, we found only 340 of them had their reference pages in Wikipedia. Thus, the methods that rely heavily on KBs might not be appropriate here.

Shortness of the Texts The context in which a

Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pages 654–664,
Lisbon, Portugal, 17-21 September 2015. ©2015 Association for Computational Linguistics.

target entity occurs plays an important role in disambiguation. (Cassidy et al., 2012; Li et al., 2013) applied the underlying topical coherence information to a set of mentions for entity linking. In (Wang et al., 2012), mentionRank leverages some additional features, such as co-mention; however, people prefer to share their comments in brief or even informal words on social media platforms, such as Twitter, which becomes increasingly important as an information source. We have counted 350,498 microblogs, 37,379 tweets and 34,018 text snippets in various domains in Chinese and English from Twitter, Sina Weibo and Google. After preprocessing, their average length are 16, 5 and 11 words, respectively. To alleviate the shortness issue, additional information has been used to expand the context, such as the user's interest (Shen et al., 2013) and related documents (Guo et al., 2013). Such information is still sparse or unavailable in this case, and thus, the existing methods might not be suitable.

Large Scale Hundreds of millions of tweets are posted daily on Twitter (Liu et al., 2013). When scaling to a large document collection, the disambiguation task becomes increasingly important as the ambiguity increases (Cucerzan, 2007). MentionRank, the only state-of-the-art method for TED, is a graph-based model that focuses on a small set of entities (e.g., 50 or 100 entities) and conducts experiments on thousands of documents (Wang et al., 2012). However, the graph has a quadratic growth as the number of documents increases. In our experiments, a dataset that included 2,468 target entities and 350,498 microblogs generated a directed graph with billions of edges, which required more computer memory than was available even when using a sparse matrix. Therefore, the scalability remains a challenge.

Contributions

To address these challenges, we propose a collective method called TremenRank to disambiguate the target entities simultaneously. The main idea is to propagate trust within a graph based on our observations that true mentions are more similar in context than false mentions in a specific domain. Specifically, inverted indexes is used to construct the graph locally that allows for an arbitrary number of target entities and documents. Furthermore, a multi-layer directed graph is designed to assign different trust levels to documents, which significantly improves the performance. The contributions of our work can be summarized as follows:

- We propose a novel graph-based method, TremenRank, to collectively identify target entities in short texts. This method constructs the graph locally to avoid storing the entire graph in memory, which provides a linear scale up with respect to the number of target entities and documents.
- We design a multi-layer directed (MLD) graph for modeling short texts that possess different trust levels during propagation, which significantly improves the performance.
- We conduct a series of experiments on three practical datasets from different domains. The experimental results demonstrate that TremenRank has a similar performance to the state-of-the-art when addressing the TED problem at a large scale, and the use of MLD graph significantly improves our method.

2 Problem Definition

Example *Suppose that GM wants to collect tweets that talk about its cars. As shown in Figure 1, we take (i) a list of car brands (target entities), and (ii) a collection of tweets (i.e., short texts) as inputs. "Car" is the target domain, and "Sonic" appears in documents d_1 and d_2, which can be characterized as two* ***mentions*** *and the latter is a* ***true mention****. The goal is to find as many true mentions as possible.*

Our method models the collection of documents as a directed graph and outputs a trust score for each document via propagation. The trust score indicates the likelihood of a document containing a true mention. For flexibility, the trust scores lie within the range of 0 to 1 and are globally comparable; thus, we can obtain the top-k documents or choose an appropriate cut-off value to balance the precision and recall for different application requirements. For example, in the application of CFA, a company expects a higher *recall* to achieve a comprehensive understanding of its product, whereas a recommendation system must provide as many *precise* microblogs as possible.

Formally, the problem of TED in short texts is defined as follows:

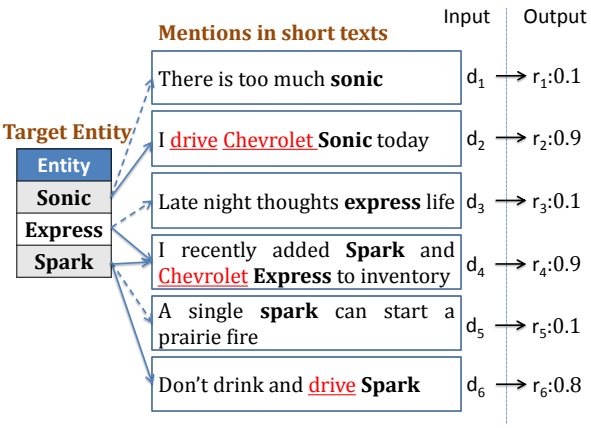

Figure 1: Illustration of TED. True mentions are shown as solid arrows, and the underlined words are their overlapping context.

Target Entity Disambiguation in Short Texts Given a list of target entities $E = \{e_i|i = 1, \ldots, m\}$, and a collection of text documents $D = \{d_j|j = 1, \ldots, n\}$, $E^{d_j} = \{e_1^j, \ldots, e_k^j\}$ is the set of target entities contained in d_j. The goal is to output the trust score $r_j \in [0, 1]$ for each document $d_j \in D$. All the target entities in E^{d_j} share the same trust score r_j.

All of the mentions in one document have the same score because they share a common context. In the graph, the context similarity between documents is computed and used as the edges normally, where the width of the context window that surround the target entity in the document is typically chosen to be 10, 20 or 50 words. However, the documents considered here are limited in length, and a user seldom changes the topic in so few words; thus, we regard the entire document as the context of its target entities. When multiple target entities occur in one document, all of them are more likely to refer to the entities in the target domain. Thus, the trust score for the document is higher, as indicated by d_4 (Figure 1).

The only constraint for the target entities is that they are in the same target domain. This constraint is reasonable in practice. The majority of applications identify a set of entities in one domain at a time. For example, a computer company focuses its attentions on the brands in the computer area, whereas an investment company is mainly interested in stocks. Additionally, even if the target domain is general and contains several small domains, an intuitive solution is to split it into several subproblems, where each subproblem focuses on the target entities in one small domain. Then, the task can be achieved by solving the subproblems individually.

3 Our Approach

TremenRank is a graph-based method that identifies target entities collectively. It propagates trust scores on the graph where each vertex denotes one document and an edge indicates the similarity between them. Considering the large scale of the problem, we obtain the neighbors of one vertex when propagating by searching two indexes instead of storing the entire graph in memory. This approach allows an arbitrary number of target entities and documents to be processed. To further improve the performance, a multi-layer directed graph is designed to treat the documents at different trust levels based on prior estimations.

3.1 Hypotheses

The documents within a domain share the characteristic of unitary similarity. This characteristic implies that all of the true mentions have a similar context due to the target domain constraint and that false mentions are distinct because their meanings belong to diversified domains. We investigated the ambiguity of 2468 stock names (target entities in experiments), and manually labeled 301 of those names. As shown in Figure 2, there are many different meanings for these names outside of the target domain, such as plant, bank, media and animal. The distribution of meanings are long-tailed; thus, we gathered a group of meanings together (the class "others").

Based on the statistical results, we can make the following hypotheses:

- The context of true mentions are similar to one another.
- The context of a false mention is different from any of the true mention.
- False mentions have distinct contexts across different entities.

For example, the true mentions in d_2, d_4 and d_6 (Figure 1) describe car brands of GM and share common pieces of text: "drive" or "Chevrolet". However, "sonic", "express" and "spark" in d_1, d_3 and d_5 are all false mentions; in their contexts, they refer to sound, giving opinions, and a small amount of fire, respectively. These false mentions are different in context from one another and from any true mention.

The assumptions resemble the main insight of MentionRank except the co-mention (multiple entities occur in one document). Co-mention seldom happens in short texts, and can be treated as the same because they share a common context. In conclusion, the assumptions suggest that a collective method could perform better than a method that disambiguates entities separately, because more comprehensive information on the entities from multiple "collaborators" (i.e., the mentions have similar contexts) has been used (Chen and Ji, 2011; Kulkarni et al., 2009; Pennacchiotti and Pantel, 2009; Will et al., 2010; Fernandez et al., 2010; Cucerzan, 2011; Guo et al., 2011; Han and Sun, 2011; Ratinov et al., 2011; Kozareva et al., 2011; Dalton and Dietz, 2013).

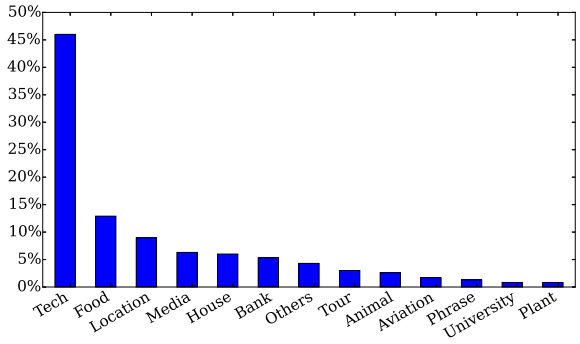

Figure 2: Statistics on the ambiguity in stocks.

3.2 Graph-based Method

Based on these assumptions, we build a graph to represent the documents and their relations, and we perform a TrustRank-like algorithm on the graph. We are given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ that consists of a set $\mathcal{V}$ of N documents (vertices) and a set $\mathcal{E}$ of directed edges, where each edge $(d_i, d_j) \in \mathcal{E}$ denotes that d_i points to d_j, and $o(d_i)$ is the out-neighbors of d_i.

3.2.1 Similarity Measurement

We constructed the edges of the graph according to the similarity relations between documents. Most similarity measurements (Artiles et al., 2010; Miller, 1995) could be used in the proposed method. After some exploration, we found that Jaccard similarity performs better. It can be efficiently calculated through simple set operations:

$$\omega_{ij}^J = J(d_i, d_j) = \frac{\left|\mathcal{W}^{d_i} \bigcap \mathcal{W}^{d_j}\right|}{\left|\mathcal{W}^{d_i} \bigcup \mathcal{W}^{d_j}\right|} \quad (1)$$

where ω_{ij}^J denotes the weight of edge (d_i, d_j) using Jaccard similarity, and $\mathcal{W}^{d_i}$ is the set of words contained in d_i. ω_{ij} varies from 0 to 1, where a value closer to 1 indicates that its two nodes are more similar. We link only similar nodes by choosing an appropriate threshold η. In other words, we have $(d_i, d_j) \in \mathcal{E}$, only if $\omega_{ij}^J > \eta$. This is the foundation to construct the graph locally using inverted indexes.

3.2.2 Inverted Index

When the scale becomes excessively large, such as the 350,000 pieces in our dataset, the number of edges will increase into the billions, producing computational and storage-based difficulties. Considering that the propagation begins with document d_i and that we are required to find all of its out-neighbors $o(d_i)$ through a traversal of the entire dataset, then the complexity is $O(n^2)$. Alternatively, we can represent the entire graph with a matrix; however, its billions of elements would be difficult to store and calculate.

To address the large scale problem, we construct the graph locally via inverted index technology. During propagation, the neighbors of the documents are obtained by searching the indexes in real time. Two types of indexes are used: the document-to-word index and the word-to-document index. The former index records $\mathcal{W}^{d_i}$ of each document $d_i \in D$; the latter index records the occurrence of each word $\mathcal{D}^{w_k} = \{d_j | w_k \in W\}$, where $W = \{w_1, \ldots, w_N\}$ is the word dictionary. Combining these two indexes, the total out-neighbors of a document can be obtained in constant time as follows:

1. Obtain all of the words $\mathcal{W}^{d_i}$ of d_i via searching the document-to-word index.
2. Find the occurrences of each word $w_k \in \mathcal{W}^{d_i}$ in the word-to-document index: $D^{d_i} = \{d_j | \cup_{w_t \in \mathcal{W}^{d_i}} \mathcal{D}^{w_t}\}$. Each document $d_j \in D^{d_i}$ shares at least one common word with d_i.
3. Count the frequency of d_j, which indicates the number of overlapping words between d_i and d_j. Then, the frequency $f_{ij} = |\mathcal{W}^{d_i} \cap \mathcal{W}^{d_j}|, i \neq j$ and Equation 1 becomes:

$$\omega_{ij}^{J} = \frac{f_{ij}}{|\mathcal{W}^{d_i}| + |\mathcal{W}^{d_j}| - f_{ij}} \quad (2)$$

4. Calculate the similarity weight. We obtain the out-neighbors $o(d_i) = \{d_j | \omega_{ij}^{J} \geq \eta\}$.

3.2.3 Trust Propagation

Similar to TrustRank, an individual document propagates the trust score to its neighbors and those who have more neighbors will receive more trust. Intuitively, trust attenuates along the edge. There are several ways for attenuation to take place, such as trust dampening, trust splitting, and a combination of them (Gyöngyi et al., 2004). For example, each document $d_i \in D$ has a trust score r_i, and its out-neighbors obtain $\alpha \cdot r_i$ (or $\frac{r_i}{|o(d_i)|}$) through trust dampening (or splitting). We adopt the third method, in which a document dampens its split trust with the attenuation coefficient α.

The final trust score is determined by two parts: the trust from a document's neighbors and its prior estimation. Then, TremenRank can iteratively propagate via the following function:

$$r_i = \alpha \cdot \sum_{o(d_i)} \frac{r_j}{|o(d_j)|} + (1 - \alpha) \cdot \mathcal{T}(d_i) \quad (3)$$

where $\mathcal{T}(d_i)$ is the prior estimation of the document d_i and is a constant during iterative propagation. Here we simply assign a uniform distribution to all of the documents $\mathcal{T}(d_i) = \frac{1}{|D|}$. A more precise estimation will be discussed later.

At the beginning of the propagation, we initialize the trust scores of the documents with the prior estimation. Then, the trust scores of the documents are updated iteratively until convergence[1]. True mentions receive high scores because they are likely to connect with more trustworthy documents and thus receive more trust through the first term in Equation 3. In contrast, false mentions are dissimilar to most other documents; thus, their scores gradually attenuate (the second term in Equation 3). These scores are globally comparable. One can normalize the final scores by dividing them by the largest score, but the relative ordering of the documents will not change. Thus, one document that is more likely to contain any true mention will receive a higher trust score and be ranked higher.

3.3 Multi-layer Directed Graph

During propagation, all of the documents are initialized with the same trust score and attenuate their trust at the same level. However, different documents should be treated differently. For example, the tweet "*I drive a big car* ***suburban***" is more trustworthy than "*doctors use* ***GMC*** *report system to process harmful patients*", because the former tweet contains the credible context feature "car", which is the name of the target domain. The latter clearly irrelevant text should not be trusted or even be regarded as noise. In this subsection, we first discuss how to make more precise prior estimation of documents based on some of the characteristics of the target domain. Additionally, based on the prior estimation, an MLD graph is built to assign different trust levels to the documents.

3.3.1 Prior Estimation

Ideally, a true mention is similar to true mentions only. To take advantage of the approximate isolation of true mentions, we first extract a set of true mentions to start the propagation. The documents that contain these true mentions are called seeds.

Formally, the entire set of documents D is divided into two groups: (i) the seed set $D^s = \{d_1, \ldots, d_s\}$, and (ii) a subset $D^* = \{d_{s+1}, \ldots, d_n\}$. We estimate a prior trust score for each document via the function $\mathcal{T}$:

$$\mathcal{T}(d_i) = \begin{cases} \frac{1-\epsilon}{|D^*|} & d_i \in D^*, \\ \frac{\epsilon}{|D^s|} & d_i \in D^s. \end{cases} \quad (4)$$

where $|D^s|$ and $|D^*|$ represent the size of the two sets for normalization. $\epsilon \in (0, 1]$ is used for smoothing and indicates the likelihood that we can trust the seeds that actually contain true mentions. In the experiments, we set $\epsilon = 0.9$ based on the accuracy of the seeds extraction method.

There are several methods for extracting seeds, such as manual annotation and pattern-based method. Patterns could be easily derived from the characteristics of the target domain, such as the domain name, product type or unique ID[2]. These methods can typically identify entities with a high accuracy, but their low recall limits the portion of

[1] We select the attenuation coefficient $\alpha = 0.85$ and the number of iterations to be 20, which have been regarded as the standards in the PageRank literature(Page et al., 1999; Krishnan and Raj, 2006). Our experiments also show that 20 iterations are sufficient to achieve convergence.

[2] The exact patterns used in our datasets are detailed in Section 4

true mentions that can be found. We use the results of the pattern-based method as our seeds, which have an accuracy higher than 90%. We do not present the experimental results here due to space limitations.

3.3.2 Graph Construction

The similarity measurement does not consider directions, and thus, the documents are mutually connected. In this section, we construct a layered structure in the graph, where each layer denotes a document trust level that contains any true mention. Thus, we define that the propagation direction from the high trust level to the low trust level.

Figure 3 shows an example of an MLD graph. The blue nodes of the seeds in the top layer are the most trustworthy, and the other white nodes in the higher layer are less similar to the seeds which implies that they are at lower trust levels. Thus, as we move farther away from the seeds, trust attenuates at a constant speed α along with the layers. The nodes in the same layer are also connected.

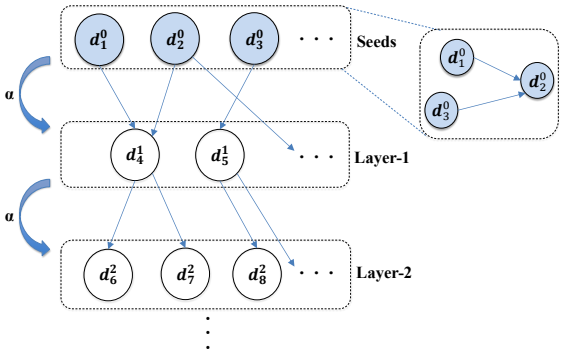

Figure 3: Example of an MLD graph.

The construction algorithm[3] is presented in Algorithm 1, where $D^l = \{d_i^l | l \geq 0\}$ is the set of nodes in layer l, and the nodes in $\bar{D} = \{\bar{d}_j | \bar{d}_j \notin D^l, l \geq 0\}$ are not connected. To simplify our notation, the seeds are set to be in layer 0. Note that $(\cap_{l \geq 0} D^l) \cap \bar{D} = D$.

TremenRank is different from the standard TrustRank and MentionRank in several respects. First, TremenRank is designed to process short texts at a large scale. Second, through a well-designed MLD graph, we consider documents to consist of different trust levels rather than be represented by a unified distribution. Third, TrustRank randomly samples a set of seeds and checks them manually, which limits the number of seeds available; however, the proposed method extracts the seeds automatically and uses large amounts of seeds to produce better prior estimations. Finally, disambiguation occurs at the document level in the proposed method; we assign the same scores to the entities that occur in one document, because they typically share common context features in short texts.

Algorithm 1: Construction of an MLD graph.

Input: D^s, D^*, η, indexes $\mathcal{W}^D, \mathcal{D}^W$
Output: $\mathcal{G}$
Initialize seeds in layer $l = 0$, $\bar{D} = D^*$;
foreach *layer* $l \geq 0$ **do**
  Find the out-neighbors of D^l from $\bar{D}$;
  foreach *document* $d_i^l \in D^l$ **do**
    put $o(d_i^l)$ in the next layer D^{l+1};
    update $\bar{D} = \bar{D} - D^{l+1}$;
  end
  if $|\bar{D}| = 0$ *or* $|D^{l+1}| = 0$ **then**
    break;
  else
    $l = l + 1$;
  end
end

4 Empirical Evaluation

4.1 Data Preparation

Because there is no publicly available benchmark dataset for TED, we constructed three datasets of different domains: Stock, Car and Herb[4]. All of these datasets came from the needs of real-life projects.

1. **Stock** - We collected 2,468 stock names from a stock exchange website, and identified their candidate mentions by string matching from Sina Weibo (Counterpart of Twitter in China). Each stock has a unique ID, which has little ambiguity. Thus, we used this regular expression pattern to extract seeds.

2. **Car** - We collected 32 car brands of General Motors and a group of tweets that contain at least one mention of these brands via the Twitter API. In the seeds extraction step, we use the domain name "car" as the patterns.

[3] A key parameter for the structure of MLD Graph is η, which will be discussed in Section 4.2.

[4] All of the dataset related sources used in this study are listed at http://newsminer.org/TEDs.

3. **Herb** - We randomly selected 1,119 Chinese herb names and collected 40 pieces of text descriptions in the search results per entity from Google. To extract the seeds, we simply added the domain name to the key words (e.g., *"天虫(Worms)* 中草药*(Chinese Medical Herb)"*).

Table 1: Statistics on the datasets.

	E	D	#edges	Positive(%)
Stock	2,468	350,498	2.049B	43
Car	32	37,379	4.91M	7
Herb	1,119	34,018	9.66M	68

Table 1 shows some of the statistics of the datasets created. We chose these datasets because (i) the identification of entities in these domains meets the practical requirements of many applications, (ii) the target entities are ambiguous and (iii) there is little information in the existing KBs. Before applying TremenRank, we preprocessed the plain texts through word segmentation, low frequency words filtering and stop words filtering.

4.2 Experiment

Baseline Methods TED in short texts is a relatively new problem, and there are few specific methods for solving it. To validate the performance of TremenRank and the improvement produced by the MLD graph, we selected the baseline from three different perspectives: (i) a context-based method that identifies target entities separately; (ii) a classic supervised method SVM to classify a document by whether it contains any true mentions; and (iii) the only state-of-the-art MentionRank for TED, which is a collective ranking method.

- **The Context-based method** mines a frequent item set of true mentions and identifies the documents that contain more frequent items.

- **SVM** classifies the documents into two classes: documents that are in the target domain and those that are not. Using context words as features, we train and test SVM on the labeled set with a 10-fold cross validation.

- **MentionRank** is a graph-based method that disambiguates at the entity level. Difficult to apply to the entire large datasets directly, it has been applied to the labeled set.

Evaluation Metrics The performance of the disambiguation task is typically evaluated by *accuracy*, but in TEDs we are also interested in *precision*, *recall* and the *F1-measure* because different applications focus on different aspects. For example, in the application of CFA, a company expects a higher *recall* to collect as many reviews as possible, while in financial news recommendations, users prefer to read more *accurate* microblogs. Because the entire dataset is too large to evaluate directly, we randomly sampled 800 mentions for each dataset and labelled them manually to calculate the performance metrics[5].

Results and Analysis

The overall performances of TremenRank and those of the baseline methods on all of the datasets are shown in Table 2. The following is indicated in the results:

- TremenRank+MLD outperforms all of the baselines with all of the datasets, because it collectively identifies target entities and treats the documents differently based on a precise prior estimation.

- The collective methods tend to perform better. Although, on the Car dataset, SVM achieves the second best performance, the use of MLD graph in TremenRank outperforms all of the methods tested. This is because false mentions that occupy a large proportion of the dataset produce too much noise, whose negative impacts can be reduced through the train set in the supervised method or a precise prior estimations.

- Compared with MentionRank, TremenRank processes a larger number of entities and documents while achieving a similar performance. Combined with a MLD graph, TremenRank shows significant improvements including an average gain of 24.8% in accuracy and 15.2% in the F1-measure on the three datasets.

We also investigate the influence of the main elements in TremenRank below.

Similarity Threshold Different similarity thresholds result in various structures of the MLD graph, and have a great impact on the performance of our

[5] A detailed explanation about these metrics can be found in http://en.wikipedia.org/wiki/Precision_and_Recall.

Table 2: Overall results on the three datasets

Method	Stock				Car				Herb			
	Accu	Prec	Recall	F-score	Accu	Prec	Recall	F-score	Accu	Prec	Recall	F-score
Context	0.704	0.332	0.545	0.410	0.733	0.333	0.256	0.476	0.670	0.200	0.074	0.108
SVM	0.746	0.377	0.998	0.547	0.839	0.371	0.923	0.529	0.567	0.988	0.564	0.718
MentionRank	0.458	0.424	0.828	0.561	0.464	0.301	0.715	0.423	0.644	0.691	0.824	0.752
TremenRank	0.477	0.424	0.803	0.555	0.542	0.310	0.720	0.433	0.737	0.736	0.827	0.779
TremenRank+MLD	0.683	0.575	0.844	**0.684**	0.827	0.624	0.731	**0.673**	0.800	0.774	0.908	**0.836**

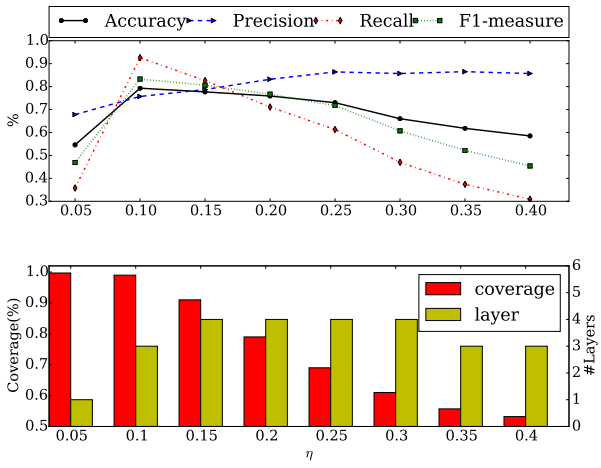

Figure 4: Selection of the Similarity Threshold

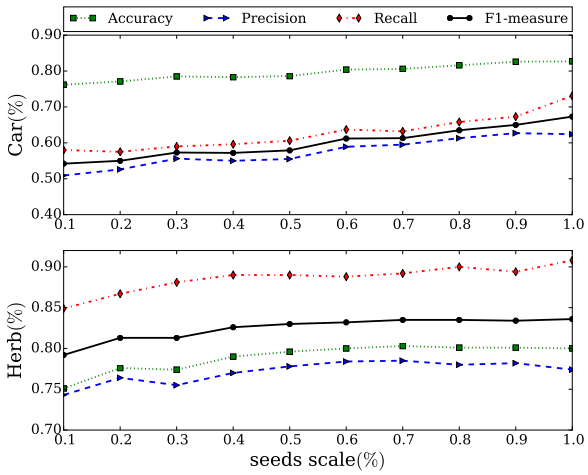

Figure 5: Influence of Seed Scale

method. In this subsection, we study a heuristical method that help choose the best eta according two factors (Both of them can be obtained before propagation). Figure 4 presents the experimental results for different values of η on the dataset of Herb. The performance of TremenRank is showed on the top, and the bottom is the corresponding graph structure represented by two factors: the graph coverage and the number of layers. We can see that precision increases until it becomes steady with the growth of η, and other measurements reach their peaks when $\eta = 0.1$.

This is in accordance with the change of the graph structure. On one hand, the number of layers l declines sharply when η is less than 0.1, this indicates little difference in the trust level of documents in the propagation. On the other hand, our method has no effect on the vertices outside of the graph, so the performance is directly proportional to the graph coverage rate. Therefore, a proper η should ensure a high coverage as well as adequate layers. In experiments, we choose η as 0.1, 0.15, 0.1 for the datasets of Stock, Car and Herb respectively.

Influence of the Seed Scale As the basis of the prior estimation, the seeds have significant influence on the performance of the proposed method via the MLD graph. Intuitively, a larger set of seeds should lead to a more precise estimation and thus better performance. In the experiments on the Car and Herb datasets, we split their seed set into ten parts, and add one part each time. As Figure 5 shows, when increasing the percentage of the seeds gradually, the performance has an overall upward trend (e.g., a 14.6% and 4.4% gain in the F1-measure for the Car and Herb datasets, respectively). This trend occurs because the potential context features of the seeds utilized for propagation increase as the absolute number of seed documents rises. For example, the tweet *"Cadillac Uber airport is classy"* obtains a low score of 0.013 with 50% of the seeds, whereas it is identified successfully with the score of 0.588 when all of the seeds are used for propagation because more context features are introduced, such as *"airport" and "Uber"*. Thus, the proposed method achieves better performance as the seed set grows. This arrangement is helpful for a company that seldom changes its business area and then accumulates seeds continuously to improve performance.

Robustness to Noise Because seeds play an important role in the MLD graph, we further test the

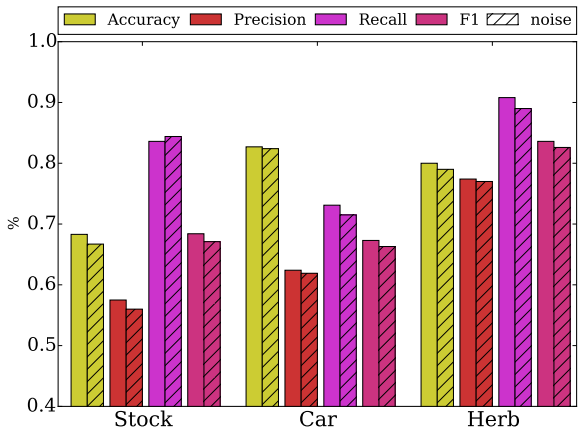

Figure 6: Robustness to Noise

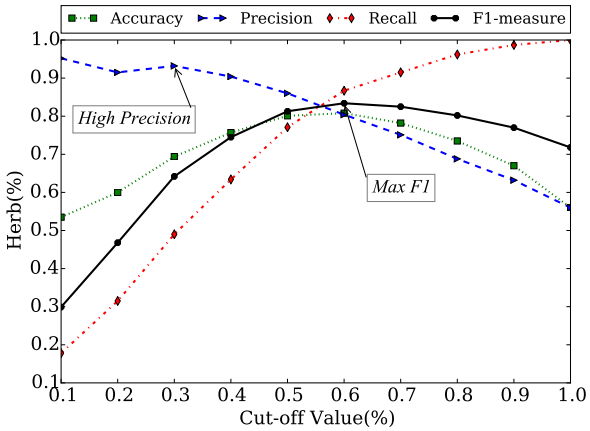

Figure 7: Cut-off Value

robustness of our method with respect to the quality of the seeds. We randomly sample documents outside of the seed set (considered as noise) to replace 20% of the seeds on all the datasets; these results are shown in Figure 6. The introduced noise only leads to a limited decrease in performance (average 1.3%, 1% and 1% in the F1), which is much better than when only using TremenRank (Table 2). More specifically, the experiments that use artificial noise occasionally achieve a higher recall (e.g., on the Stock dataset); this result could occur because the unknown true mentions add some useful edges to the graph, which is helpful when finding more true mentions of the target entities.

Cut-off Value According to the globally comparable trust scores, we can (i) rank all of the documents, and trade off the performance metrics by choosing an appropriate cut-off value γ for various applications, or (ii) rank the documents of an individual entity separately and obtain its top-k mentions.

In the experiments, we set γ as different percentage of the ranked documents. As Figure 7 shows, the recommendation system could use $\gamma = 30\%$ to achieve a 93.2% precision and a 64.2% recall, or a company could use $\gamma = 60\%$ for more reviews that have a relatively high precision.

Efficiency We implemented TremenRank in Java and ran it on a single PC with the Windows 8.1 64-bit operation system. With an Intel(R) Core(TM) i3-3240 (3.40GHz) CPU and 4GB memory, our program converges within 5 iterations, and only consumes approximately 700MB of memory when running steadily. The overall identification times of the Stock, Car and Herb datasets are 12h, 5min and 18min, respectively. The computation time increases exponentially with the increase of the amount of data due to the excessive computations required to search the indexes, which can be optimized in future work.

5 Conclusions and Future Work

In this paper, we addressed a new and increasingly important problem in social content analysis in a challenging scenario: disambiguation of a list of homogenous entities in short texts using names only. We proposed a graph-based method called TremenRank to identify target entities collectively; this method can also hold an arbitrary number of target entities and documents. The performance of this method can be further improved via a well-designed MLD graph. The experimental results show that the proposed method has a significant improvement compared to other approaches.

In the future, we are interested in refining the prior estimation by using the ontology and extending this work to detect the target entities that are not in a list while performing the disambiguation task.

Acknowledgments We'd like to acknowledge Lei Hou, Chi Wang and Hongzhao Huang for their impressive discussions on this paper. The work is supported by 973 Program (No. 2014CB340504), NSFC-ANR (No. 61261130588), Tsinghua University Initiative Scientific Research Program (No. 20131089256), Science and Technology Support Program (No. 2014BAK04B00), and THU-NUS NExT Co-Lab.

References

Javier Artiles, Andrew Borthwick, Julio Gonzalo, Satoshi Sekine, and Enrique Amigó. 2010. Weps-3 evaluation campaign: Overview of the web people search clustering and attribute extraction tasks. In *CLEF (Notebook Papers/LABs/Workshops)*.

Razvan C Bunescu and Marius Pasca. 2006. Using encyclopedic knowledge for named entity disambiguation. In *EACL*, volume 6, pages 9–16.

Taylor Cassidy, Heng Ji, Lev-Arie Ratinov, Arkaitz Zubiaga, and Hongzhao Huang. 2012. Analysis and enhancement of wikification for microblogs with context expansion. In *COLING*, volume 12, pages 441–456. Citeseer.

Zheng Chen and Heng Ji. 2011. Collaborative ranking: A case study on entity linking. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 771–781. Association for Computational Linguistics.

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

Silviu Cucerzan. 2011. Tac entity linking by performing full-document entity extraction and disambiguation. In *Proceedings of the Text Analysis Conference*, volume 2011.

Jeffrey Dalton and Laura Dietz. 2013. A neighborhood relevance model for entity linking. In *Proceedings of the 10th Conference on Open Research Areas in Information Retrieval*, pages 149–156. LE CENTRE DE HAUTES ETUDES INTERNATIONALES D'INFORMATIQUE DOCUMENTAIRE.

Mark Dredze, Paul McNamee, Delip Rao, Adam Gerber, and Tim Finin. 2010. Entity disambiguation for knowledge base population. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 277–285. Association for Computational Linguistics.

Norberto Fernandez, Jesus A Fisteus, Luis Sanchez, and Eduardo Martin. 2010. Webtlab: A cooccurencebased approach to kbp 2010 entity-linking task. In *Proc. TAC 2010 Workshop*.

Yuhang Guo, Wanxiang Che, Ting Liu, and Sheng Li. 2011. A graph-based method for entity linking. In *IJCNLP*, pages 1010–1018. Citeseer.

Weiwei Guo, Hao Li, Heng Ji, and Mona T Diab. 2013. Linking tweets to news: A framework to enrich short text data in social media. In *ACL (1)*, pages 239–249. Citeseer.

Zoltán Gyöngyi, Hector Garcia-Molina, and Jan Pedersen. 2004. Combating web spam with trustrank. In *Proceedings of the Thirtieth international conference on Very large data bases-Volume 30*, pages 576–587.

Xianpei Han and Le Sun. 2011. A generative entity-mention model for linking entities with knowledge base. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 945–954. Association for Computational Linguistics.

Xianpei Han, Le Sun, and Jun Zhao. 2011. Collective entity linking in web text: a graph-based method. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 765–774. ACM.

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

Zornitsa Kozareva, Konstantin Voevodski, and Shang-Hua Teng. 2011. Class label enhancement via related instances. In *Proceedings of the conference on empirical methods in natural language processing*, pages 118–128. Association for Computational Linguistics.

Vijay Krishnan and Rashmi Raj. 2006. Web spam detection with anti-trust rank. In *AIRWeb*, volume 6, pages 37–40.

Sayali Kulkarni, Amit Singh, Ganesh Ramakrishnan, and Soumen Chakrabarti. 2009. Collective annotation of wikipedia entities in web text. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 457–466. ACM.

Yang Li, Chi Wang, Fangqiu Han, Jiawei Han, Dan Roth, and Xifeng Yan. 2013. Mining evidences for named entity disambiguation. In *KDD'13*, pages 1070–1078.

Xiaohua Liu, Yitong Li, Haocheng Wu, Ming Zhou, Furu Wei, and Yi Lu. 2013. Entity linking for tweets. In *ACL (1)*, pages 1304–1311.

Rada Mihalcea and Andras Csomai. 2007. Wikify!: linking documents to encyclopedic knowledge. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 233–242. ACM.

George A. Miller. 1995. Wordnet: A lexical database for english. *Commun. ACM*, 38(11):39–41.

David Milne and Ian H Witten. 2008. Learning to link with wikipedia. In *Proceedings of the 17th ACM conference on Information and knowledge management*, pages 509–518. ACM.

Roberto Navigli. 2009. Word sense disambiguation: A survey. *ACM Computing Surveys (CSUR)*, 41(2):10.

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. The pagerank citation ranking: Bringing order to the web.

Marco Pennacchiotti and Patrick Pantel. 2009. Entity extraction via ensemble semantics. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*, pages 238–247. Association for Computational Linguistics.

Lev Ratinov, Dan Roth, Doug Downey, and Mike Anderson. 2011. Local and global algorithms for disambiguation to wikipedia. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 1375–1384. Association for Computational Linguistics.

Wei Shen, Jianyong Wang, Ping Luo, and Min Wang. 2012. Linden: linking named entities with knowledge base via semantic knowledge. In *Proceedings of the 21st international conference on World Wide Web*, pages 449–458. ACM.

Wei Shen, Jianyong Wang, Ping Luo, and Min Wang. 2013. Linking named entities in tweets with knowledge base via user interest modeling. In *KDD*, pages 68–76.

Chi Wang, Kaushik Chakrabarti, Tao Cheng, and Surajit Chaudhuri. 2012. Targeted disambiguation of adhoc, homogeneous sets of named entities. In *WWW*, pages 719–728.

Radford Will, Hachey Ben, Nothman Joel, Honnibal Matthew, and R.Curran James. 2010. Cmcrc at tac10: Document-level entity linking with graph-based re-ranking. In *In Proc. Text Analysis Conference (TAC 2010)*.