Entity Disambiguation with Freebase

Zhicheng Zheng*, Xiance Si†, Fangtao Li†, Edward Y. Chang† and Xiaoyan Zhu*
*Tsinghua University
Email: zhengzc04@gmail.com, zxy-dcs@tsinghua.edu.cn
†Google Inc.
Email: {sxc,lifangtao,edchang}@google.com

***Abstract*—Entity disambiguation with a knowledge base becomes increasingly popular in the NLP community. In this paper, we employ Freebase as the knowledge base, which contains significantly more entities than Wikipedia and others. While huge in size, Freebase lacks context for most entities, such as the descriptive text and hyperlinks in Wikipedia, which are useful for disambiguation. Instead, we leverage two features of Freebase, namely the naturally disambiguated mention phrases (aka aliases) and the rich taxonomy, to perform disambiguation in an iterative manner. Specifically, we explore both generative and discriminative models for each iteration. Experiments on $2,430,707$ English sentences and $33,743$ Freebase entities show the effectiveness of the two features, where 90% accuracy can be reached without any labeled data. We also show that discriminative models with proposed split training strategy is robust against overfitting problem, and constantly outperforms the generative ones.**

***Keywords*-Freebase, Name Entity Disambiguation, Naturally Disambiguated, Type Taxonomy**

I. Introduction

With the development of web resources, it is a common problem that one phrase may refer to different entities in different context. For example, "apple" may refer to a company in *Apple released the new iPad yesterday*, or a fruit in *Apple juice contains Vitamin C and sugar*. It is important to find the exact meaning of a phrase given the context. This not only benefits the semantic understanding of text, but also can improve the performance for related tasks, such as machine translation and information retrieval.

Therefore, entity disambiguation with a knowledge base becomes increasingly popular in the natural language processing community. Given a phrase and its context, it aims to identify the corresponding entity in the knowledge base. Currently, the most used knowledge base is Wikipedia, which has been widely studied in previous works [1], [2]. In this paper, we focus on another important knowledge base on the Web: Freebase [3].

Freebase is an entity database built manually by the community. It has two main advantages compared to Wikipedia. First, it has rich types and well defined schemas for the entities, thus is considered more as a structured database. The types of an entity determine the attribute schema of the entity, hence the users seldom assign meaningless types for an entity. Compared to the categories of Wikipedia, Freebase

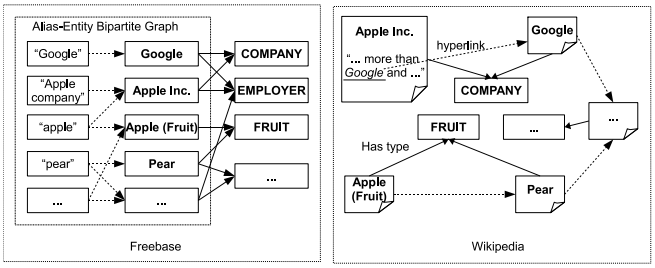


Figure 1. Example structures of Freebase and Wikipedia

enjoys a better type taxonomy and more complex schemas. The well structured database is not only convenient for a human to browse, but also very suitable for the machine to use. Second, it contains a lot more entities. According to the current statistics on each site, Freebase is about five times larger than English Wikipedia, measured by number of entities (22 million entities vs 3.9 million).

Although Freebase accumulates a large amount of entities with well defined structure, it is still challenging to conduct entity disambiguation with it. Figure 1 demonstrates the difference between Freebase and Wikipedia, where we can see that Freebase lacks the textual descriptions for its entities. For 85.1% of Freebase entities, there is no textual content at all. This is very different from the Wikipedia, which enjoys rich textual content and hyperlinks among similar entities. Since most of previous studies rely on the rich textual content and the hyperlinks to perform entity disambiguation with Wikipedia, applying them to Freebase is difficult. Therefore, we need to reconsider the problem with the restriction of limited textual context for Freebase.

Before describing our method, we want to share the following observations: First, although Freebase contains many ambiguous entities, there are still a large number of phrases which refer to only one entity. We refer to these phrases as naturally disambiguated ones. We can leverage them as labeled data, thus create a training data set with sentences containing them. Second, Freebase's rich taxonomy allows us to transfer the contextual knowledge from those naturally disambiguated phrases to others. Each Freebase entity maps to one or more types, while in real situation they often have tens of types. For example, the

entity *Google* is a *SEARCH ENGINE*, a *COMPANY*, an *EMPLOYER*, a *PATENT ASSIGNEE* and many more. In this example, although we don't need to disambiguate the phrase "Google", it can still serve as the training data for those types, then help classify the other entities with the same type, such as *Apple*. This rich taxonomy thus has great value in disambiguation.

With the above observations, we propose to disambiguate entity-referring phrases in natural language text with Freebase entities. Specifically, we design an iterative disambiguation framework that transfer the knowledge learned from non-ambiguous mentions to those ambiguous ones. Specifically, we experiment with two different subroutines in the framework. One is a generative model and the other one is a discriminative model. Generative models are intuitive and frequently used in relevant tasks [4]. However, picking one entity among several candidates is essentially discriminative, thus discriminative models may fit better. Within the two models, we compare the performance of using naturally disambiguated phrases and the rich taxonomy, with different percentages of labeled data (as supposed to be the overlap between Wikipedia and Freebase). Experiments with $33,743$ real world entities show that the naturally disambiguated phrases and rich taxonomy greatly boosted the accuracy of disambiguation, and a carefully designed discriminative learning process outperforms the generative one.

Our contributions are two folds: First, we tackle the problem of disambiguating entity-referring phrases with Freebase entities, which severely lack the textual context needed by traditional methods. The process could lead to automatic ways to annotate the Web with Freebase, amplifying its already great value. Second, we show that in large scale entity disambiguation, discriminative models yield better results than generative ones after solving the overfitting issue.

The rest of the paper is organized as follows. In Section 2, we present a brief survey of related works. In Section 3, we detail our approach of disambiguation. In Section 4, we carry out experiments to compare the performance of different methods. The conclusion is made in Section 5.

II. Related Work

Entity disambiguation is an important task in NLP, existing methods generally fall into three categories: unsupervised approach, supervised approach and semi-supervised approach.

The unsupervised methods are based on unlabeled corpora, and do not exploit any manually tagged corpus to provide a choice for a phrase in context [5], [6]. They mainly rely on some heuristic rules, or the pre-defined similarity metrics. These methods are very simple and easy to implement. But the overall performance can't achieve better result than supervised methods.

The supervised approaches are widely adopted by previous researchers [1], [7], [8], [9], [2], [10]. They employ the annotated data set. Generally, these methods first extract various appropriate features, such as calculating the similarity between the local context of the entity-referring phrase and the context information of candidate entities. Then they design supervised classifiers or rankers to select the best target entity for the phrase [2], [11]. Compared to unsupervised methods, the supervised methods can achieve better results. However, it is very time consuming to acquire a large number of labeled data, especially for the large scale data for the web based knowledge base, such as Wikipedia and Freebase.

To employ a small number of labeled data and a large number of unlabeled data, researchers also propose to solve the entity disambiguation problem in a semi-supervised way. Generative models, especially topic models, are powerful in describing the generation process of a corpus, hence it is widely used in those unsupervised or semi-supervised tasks [12], [13], [14], [4]. However, as we argued in Section 1, the entity disambiguation task is essentially a classification task, in which the discriminative model usually performs better than the generative model [15]. In this paper, we make a preliminary study on applying the discriminative model combined with the entity taxonomy information for the semi-supervised entity disambiguation task.

Moreover, in this paper, we focus on a new and important web knowledge base, Freebase. Currently, the most widely studied web knowledge base is Wikipedia. Freebase is very different from the Wikipedia in entity disambiguation task. In Wikipedia, the hypertext topic page for each entity are used for computing similarities [2], [11]. Gottipati et al. [16] use the words surrounding the entity-referring phrase and on the topic page as features. Han et al. [17] include the labeled entities (anchors of hyperlinks) as context too. However, in Freebase, most entities miss such kind of rich context information, making it difficult to compute context similarity by textual content. There are also some similar works on Freebase. Lee et al. [18] propose to integrate the taxonomies of Freebase and Probase [19]. However, we aim to disambiguate the entity-referring phrase in text with Freebase. As the phrase in the text does not have taxonomy information, their methods are not able to be directly applied here.

III. The Semi-supervised Framework

First, we define the notions used throughout the paper. For convenience, we call an entity-referring phrase as an *alias*, and thus a naturally disambiguated phrase as the *naturally disambiguated alias*. We denote an entity as e, a type as t and an alias as a. $T(e)$ is the set of e's types, and $E(a)$ is the set of a's corresponding entities. The whole entity database is denoted as D, containing all aliases, entities, types and the relations between them. For example, consider a small

D' containing two entities e_0 = *Apple Inc.* and e_1 = *Apple (fruit)*, two types t_0 = *COMPANY* and t_1 = *FRUIT*, and two aliases a_0 = "apple", a_1 = "apple computer". Then, we have $E(a_0) = \{e_0, e_1\}$, $E(a_1) = \{e_0\}$, $T(e_0) = \{t_0\}$ and $T(e_1) = \{t_1\}$. We use a sentence s to denote the surrounding text of one particular occurrence of an alias. In s, the words other than a are used as features of a. We denote the feature vector of s as $\vec{f^{(s)}} = \{f_i^{(s)} | 0 \leq i < F\}$, where F is the length of the feature vector, and $f_i^{(s)}$ is the count of the i-th word in s. For example, suppose that the feature vector contains 4 words $\{fruit, distribute, iPhone, juice\}$. Then the feature vector of the sentence "Apple distributes iPhone" with alias "Apple" is $\{0, 1, 1, 0\}$.

The data set used for training is a set of natual language sentences. Each sentence s consists of an alias $a^{(s)}$ and its feature vector $\vec{f^{(s)}}$. For each sentence s in the labeled data set (denoted as S_L), the entity $e^{(s)}$ refered by its alias $a^{(s)}$ is explicitly given. Our goal is to generate correct assignments for the sentences in the unlabeled data set (denoted as S_U). An assignment means assign an entity $e^{(s)}$ to $a^{(s)}$ in a sentence s from $E(a^{(s)})$. A correct assignment means the assigned entity is just the one referred by the alias in the sentence.

With those notions, Figure 2 shows our semi-supervised entity disambiguation framework. The motivation is to spread the correct assignments from labeled data and naturally disambiguated aliases to the others. At the beginning, we initialize the assignments for all aliases. Specifically, for sentences in S_L, we use the correct assignment; for sentences in S_U, we randomly assign a corresponding entity(line 1 in Figure 2). Note that for sentences containing naturally disambiguated aliases, even in S_U, the correct assignment is the only choice. In each iteration, we train the model with assignments from the previous iteration (line 3 in Figure 2), and use the new model to re-generate assignments only for the unlabeled sentences (line 4 in Figure 2). The assignments for the labeled sentences keep unchanged throughout iterations. The algorithm stops after a desired number (K) of iterations (line 6 in Figure 2).

There are two ways to spread the knowledge from correct assignments to ambiguous ones. The first way is from the correctly assigned aliases to the other aliases of the same entity (the naturally disambiguated aliases). The second way is from the correctly assigned entities to the other entities of the same type (the rich taxonomy). Hence, when implementing the subroutines $Train$ and $Assign$, both ways should be considered. In the rest of this section, we will present the design of both generative and discriminative subroutines.

A. Generative Model

We introduce the baseline generative model first, then describe how naturally disambiguated aliases and rich taxonomy are used. The baseline generative approach models

Input: S_L, S_U, D, K;
Output: S_U;
1: Initialize assignments in S_U;
2: **for** $(k = 0; k < K; k++)$ **do**
3: Model $m = Train(S_L \cup S_U, D)$;
4: $Assign(S_U, D, m)$;
5: **end for**
6: **return** S_U;

Figure 2. The Disambiguation Framework

the entity as generated from the alias, and its surrounding features as generated from both the entity and alias. The joint distribution of e, a and $\vec{f}$ in each sentence is: $p(e, a, \vec{f}) = p(a)p(e|a)p(\vec{f}|e, a)$. Hence the conditional probability of e in the sentence is:

$$p(e|a, \vec{f}) = \frac{p(e, a, \vec{f})}{p(a, \vec{f})} \propto p(e|a) \cdot p(\vec{f}|e, a) \tag{1}$$

Note that $p(a)$ is the same for all candidate entities, so we don't need to calculate $p(a)$.

We assume $p(e|a)$ follows the Dirichlet distribution $Dir(\vec{1})$, and $p(\vec{f}|e, a)$ follows the Multinomial distribution $\vec{p}(\cdot|e, a) = \{p(i|e, a) | 0 \leq i < F\}$. Hence, $p(e|a, \vec{f})$ is calculated as:

$$p(e|a, \vec{f}) \propto p(e|a) \prod_{f_i \in \vec{f}} p(i|e, a)^{f_i} \tag{2}$$

In the $Train$ subroutine, we update the parameters with the following formulas:

$$p(e|a) = \frac{n(e, a)}{\sum_{e'} n(e', a)} \tag{3}$$

$$p(i|e, a) = \frac{n(i, e, a)}{\sum_j n(j, e, a)} \tag{4}$$

where $n(\cdot)$ is a counting function.

In the $Assign$ subroutine, we choose e with maximum $p(e|a, \vec{f})$ as the assignment.

To incorporate the naturally disambiguated aliases, we assume that all features of an given entity are independent with the aliases, i.e. $p(i|e, a) = p(i|e)$. Accordingly, we update the Eq. 4 as

$$p(i|e) = \frac{n(i, e)}{\sum_j n(j, e)} \tag{5}$$

To incorporate the rich taxonomy, we assume that the feature distribution of all entities sharing the same type are similar, and they follow the Dirichlet distribution with the parameter $\vec{1} + \lambda \cdot \sum_{t \in T(e)} \vec{p}(\cdot|t)$, where λ balances the influence of the types, and $\vec{p}(\cdot|t)$ is the feature distribution for type t, which is approximately calculated as:

$$
\begin{aligned}
p(i|t) &= \sum_{\forall e, t \in T(e)} p(i, e|t) \\
&\approx \sum_{\forall e, t \in T(e)} p(i|e)p(e|t) \qquad (6) \\
&\propto \sum_{\forall e, t \in T(e)} p(i|e)p(t|e) \sum_{\forall a, e \in E(a)} p(e|a)
\end{aligned}
$$

where we approximate $p(t|e)$ as $\frac{1}{|T(e)|}$. We update $\vec{p}(\cdot|t)$ in each iteration, and then update $\vec{p}(\cdot|e)$ with the following equation:

$$
p(i|e) = \frac{n(i, e) + \lambda \cdot \sum_{t \in T(e)} p(i|t)}{\sum_j n(j, e) + \lambda * |T(e)|} \qquad (7)
$$

We analyze the complexity of the the generative model that uses naturally disambiguated aliases and the rich taxonomy as follows. In the $Train$ subroutine, we have to count the number for all alias-entity pairs and word-entity pairs. The count functions only apply to all the sentences in both S_U and S_L once. Hence, its time complexity is $O((|S_U|+|S_L|)\cdot l)$, where l is the average length of sentences in the two data sets. In the $Assign$ subroutine, given a sentence with alias a in S_U, we calculate the conditional probability of $p(e|a, \vec{f})$ for each entity $e \in E(a)$. Hence, its time complexity is $O(|S_U| \cdot l \cdot Q_{S_U})$, where Q_{S_U} is the average size of candidate entities for the sentences in S_U. During the iterative process, we need to store the numbers for all alias-entity pairs and word-entity pairs. In the worst case, we need to store all the word-entity pairs. Hence, the space complexity is $O(|A| \cdot Q_{E(a)} + |E| \cdot F)$, where $Q_{E(a)}$ is the average number of candidate entity of all aliases. In practice, we filter the long tail part of the word-entity pairs in order to minimize the space requirement.

B. Discriminative Model

We also apply a discriminative model in the $Train$ and $Annotation$ subroutines. Specifically, we choose the Maximum Entropy (ME) model as the concrete implementation, due to its validated performance.

The baseline method for applying the ME model is: train a muliti-class ME classifier for each alias a with the given data set (in the $Train$ subroutine), each class is an entity in $E(a)$. Then, we use the classifier to generate assignments for the sentences with the corresponding alias in S_U (in the $Assign$ subroutine). The features used for the classifier is the context word feature vector $\vec{f^{(s)}}$. The classifier produces the probability $p(e|a, \vec{f^{(s)}})$.

Although quite intuitive, the above training procedure could suffer from overfitting. As you can see, the training data of iteration m is the output of the classifier in iteration $m-1$. Overlapping of training data and output result, even between consecutive iterations, could introduce overfitting. For example, given the sentence *Apple opened a new retail shop in Bangalore*, where the alias "Apple" is what we want to disambiguate, and *Bangalore* is only used in this sentence among the entire data set. Then, the classifier would learn that *Bangalore* is a strong signal indicating "Apple" is a company. This is clearly overfitting.

To avoid overfitting, we introduce a *split training* strategy to tackle the problem: In each iteration, we randomly split the whole data set into two disjoint subsets S_A and S_B. We train the model on S_A, then use the model to generate assignments for S_B. So the model never got a chance to be trained by it's own output. We will show the improvements brought by split training in Section IV.

The baseline method uses alias-related parameters to do classification. To incorporate the naturally disambiguated aliases, we need to collect the information around an entity, so to use naturally disambiguated aliases as training data for other ambiguous aliases. Hence, instead of training a multi-class classifier for each alias, we train a binary classifier for recognizing each entity. In each iteration, we construct a training set for each entity pair. The positive instances in the training set are the sentences whose assignments are current entity. The negative instances are the rest sentences (one vs. the rest). Obviously, there are too many negative instances, hence we randomly sample a fixed portion of the negative instances into the training set to obtain balanced training data. The classifier output the probability of current sentence containing the given entity, denoted as $p(e|\vec{f^{(s)}})$. Then, in the $Annotate$ subroutine, we choose the e that maximize $p(e|\vec{f^{(s)}})$ as the assignment.

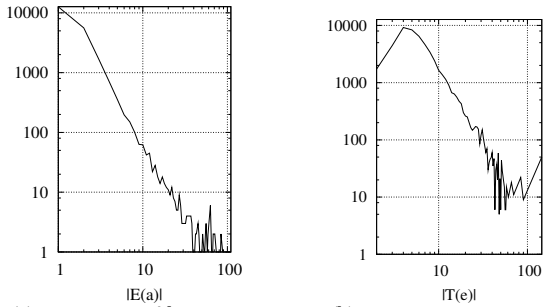
We follow the same method to incorporate the rich taxonomy of Freebase. Besides the per-entity classifiers, we additionally train a binary classifier for recognizing each type. The positive training sample for a type t are the sentences whose assigned entities have type t. The negative instances are the rest sentences. The selection of negative samples follows the same way as incorporating naturally disambiguated alias. For a given sentence with alias a and features $\vec{f}$, the classifier will give the probability of the mentioning is of type t, as $p(t|f^{(s)})$.

We combine the probability from per-entity and per-type classifiers in a linear mixture way. The final weight for choosing the assiged entity e is computed as:

$$
\omega(e) = p(e|\vec{f}) + \lambda' \cdot \sum_{t \in T(e)} p(t|\vec{f}) \qquad (8)
$$

where, λ' is the parameter balancing between per-entity and per-type classifiers. The assignment for the sentence is the candidate entity with the highest score.

In practice, we use our in-house implementation of ME model which uses a squared regularization term. Since analyze the time complexity of a ME model is well studied and beyond the scope of this paper, we only analyze the



(a) Entities Per Alias (b) Types Per Entity

Figure 3. Entity and type distribution in the data set.

space complexity for our discriminative model. For each ME model, we need to store $O(F)$ parameters. Hence, the total space complexity of the discriminative model is $O(F \cdot (|E| + |T|))$.

IV. Experiments

The experiments are designed to answer the following questions:

1) Could the naturally disambiguated aliases and the rich taxonomy help improve the performance of entity disambiguation?
2) Which approach performs better, the generative one or the discriminative one?

In addition, we also want to look into the following more specific questions: (1) Does the proposed split training strategy improve the performance of the discriminative model? (2) Is labeled data still important while we have naturally disambiguated aliases? (3) How do the parameters affect each model? (4) What is the time and space cost of our methods, as it is supposed to work on a large-scale data set?

A. Data Sets and Metrics

We perform all experiments on a sampled subset of Freebase. Specifically, we sample the entities of the following types: *professional athletes*, *academics*, *actors*, *films*, *books*, *hotels*, and *tourist attractions*. The rationale behind is that we want to include both traditional Named Entity Recognition types (person, location, organization) as well as more diverse types that are common in Freebase, such as books and movies. The sampled subset contains $33,743$ entities that have $21,931$ distinct aliases and belong to $2,056$ types. Among all distinct aliases, $12,611$ are naturally disambiguated. Fig. 3 show the distribution of #entities per alias and #types per entity.

We use sentences from Wikipedia articles as the sentence corpus. Specifically, for each alias in our sampled subset, we find the sentences in Wikipedia that use the alias as anchor text to another Wikipedia page. Since there are explicit links between Freebase and Wikipedia (some Freebase entities have a *Wikipedia url* property), we are able to automatically map the hyperlinks to Freebase entities. We sample at most $3,000$ sentences for each alias, and ensure that all $21,931$ aliases has at less 10 sentences. In the end, we get $2,430,707$ sentences.

In real world data, not all the Freebase entities enjoy a collection of labeled data. To synthesize the scenario, we divide the sampled entities into a labeled entity set and a test entity set. The test entity set serves as the entities lack of labeled data, hence the true assignment involving any entity in the test entity set are hidden to our models; they only serve as evaluation data. We randomly select $9,960$ entities (30% of the whole entity set) as the labeled entity set, and treat the rest as the test entity set. Similarly, we devide the sentence set into a labeled sentence set and a test sentence set. There are $1,719,172$ sentences containing aliases related to at least one test entity set, they are used as the test sentence set.

For convenience, we name the methods without using of any Freebase specific features as the *Baseline* methods, the methods incorporating the naturally disambiguated aliases as the *NDA* methods, and the methods incorporating rich taxonomy information as the *RT* methods.

We measure the performances with the assignment accuracy, or accuracy in short. The accuracy is computed as the number of correct assignments divided by the total number of assignments.

B. Evaluation of the Baseline Method

Firstly, we evaluate the accuracy of baseline methods. We compare the following three methods: (1) generative model; (2) discriminative model without split training; (3) discriminative model with split training.

We randomly select 0%, 20%, 40%, and 80% sentences from the labeled sentence set as labeled data. We show both accuracy on the rest of labeled sentence set and on the test sentence set.

Fig. 4 shows the accuracy of different baseline methods on the rest of labeled sentence set, which corresponds to the performance on the entities with sufficient labeled data. Our observations are as follows: (1) Without split training, the ME model overfits to the training samples, thus the accuracy doesn't change between iterations. The split training prevents overfitting and improves the accuracy of the discriminative model over 18.2%, 18.6%, 19.3% with 20%, 40%, 80% labeled data respectively. (2) As the proportion of labeled data increases, so does the accuracy. When there is sufficient labeled data for the entities, the baseline methods work pretty well (over 95%). (3) Except for the case with no labeled data, the discriminative model combined with split training achieves the best performance. With 20%, 40% and 80% labeled data, the discriminative model with split training outperforms the generative model by 2.6%, 2.2% and 1.8% respectively. The results confirm that the

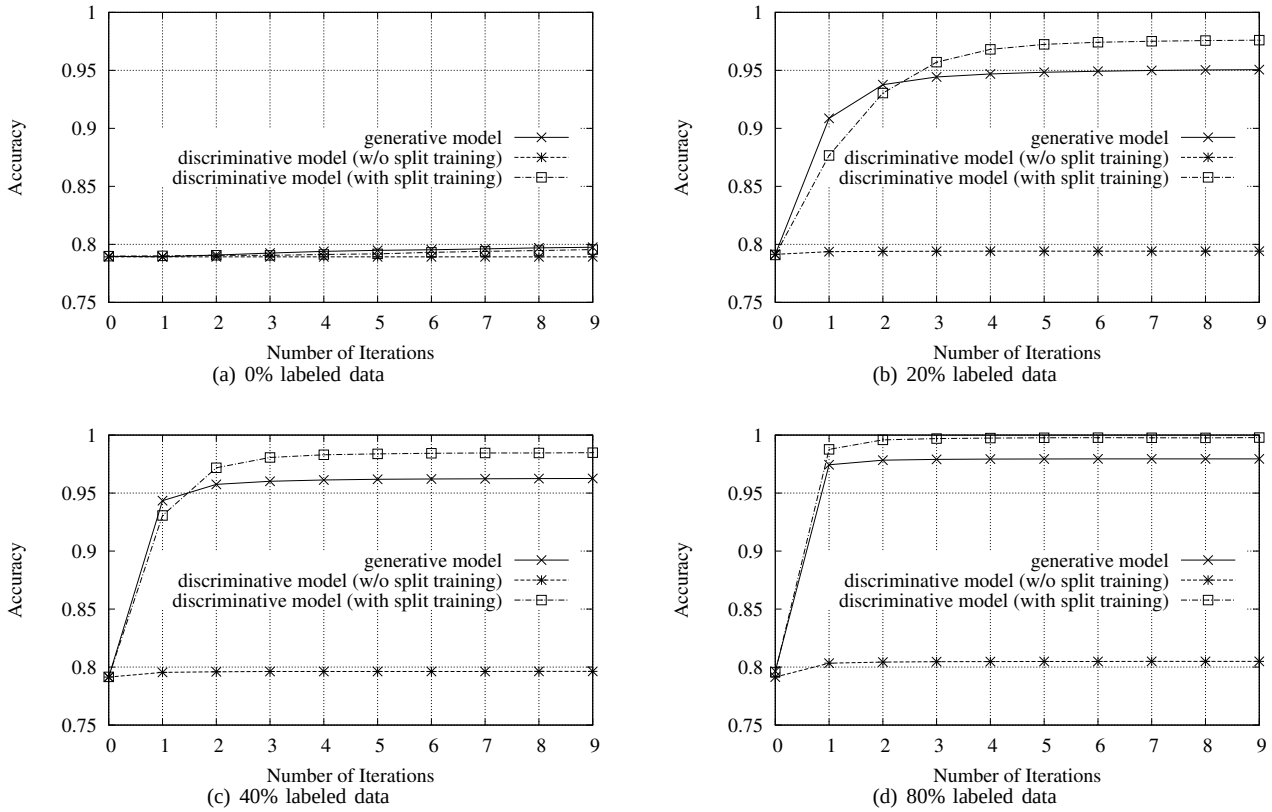


Figure 4. The baseline method with different portions of labeled sentence set, evaluated on the labeled sentence set.

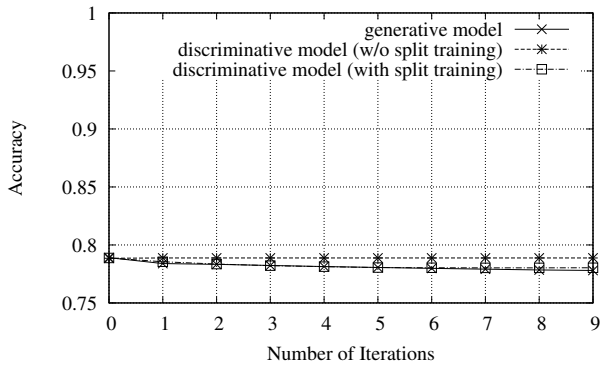


Figure 5. Baseline methods with 80% labeled data (on the test sentence set)

discriminative model with split training is very effective in disambiguating aliases with sufficient labeled data for each candidate entity.

Fig. 5 shows the accuracy of the baseline method on the test sentence set, which represents the performance on those entities without labeled data. Although we have conducted the experiments with different proportions of labeled data, the accuracy is not sensitive to that. Thus we only show the results with 80% labeled data. The poor performance is as expected as the baseline method is trained on a per-alias basis, it cannot scale to unknown aliases.

From the results of the baseline method, we can see that 1) the split training could help overcome the overfitting problem of the discriminative model; 2) if the aliases refer to the Freebase entities without labeled data, the baseline method are not able to disambiguate them.

C. Evaluation of the NDA Method

Secondly, we measure the accuracy on the test sentence set for different NDA methods. As the proportion of labeled data increases, the accuracy on the test sentence set does not change too much. The reason is that there are many naturally disambiguated aliases in the test set (about 65% of the sentences contain one). Those naturally disambiguated aliases serve as training data that renders real labeled data of no additional use. The results are shown in Fig. 6.

All the NDA methods perform better than the baseline method, we can see that (1) using naturally disambiguated entities improves the generative model by 3.4%, the discriminative model (without split training) by 4.5% and the discriminative model (split training) by 5.8%. With the relations between the aliases and entities, NDA methods spread the correct assignments from the naturally disambiguated

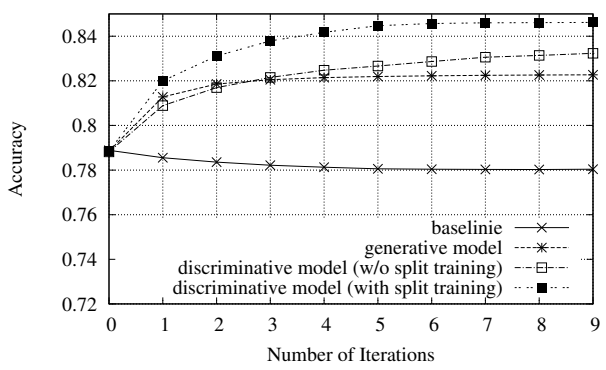


Figure 6. NDA methods with no labeled data, i.e., all labeled sentences are treated as unlabeled. The results change little when labeled sentences are used.

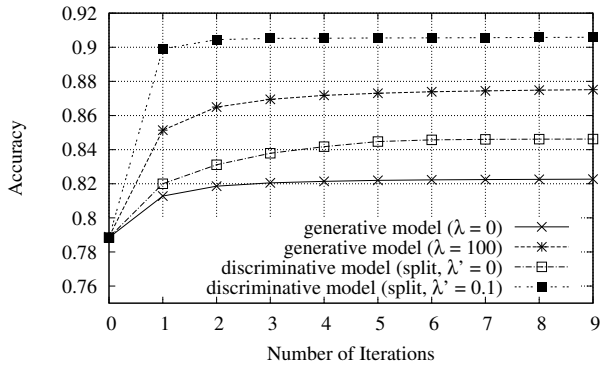


Figure 7. RT methods compared with NDA methods

aliases to the others. (2) The accuracy of the discriminative model (without split training) increases between iterations, suffering less overfitting. The reason is, for the NDA method, the assignments in each iteration are voted by several per-entity classifiers. Thus a overfitted wrong assignment from one classifier could be corrected by the other classifiers. However, the split training still improves the discriminative model without it by 1.3%. (3) The discriminative model with split training outperforms the generative model over 2.4%. Incorporating naturally disambiguated aliases results in significant improvements on the aliases that need disambiguation.

D. Evaluation of the RT Method

Finally, we examine the quality of disambiguation with the rich taxonomy (RT methods). First, we compare NDA methods and RT methods in Fig. 7 Note that for RT methods, the result is not sensitive to the proportion of labeled data either, so we just give the results without labeled data. In detail, we compare the following 4 configurations: (1) generative model ($\lambda = 0$); (2) generative model ($\lambda = 100$); (3) discriminative model with split training ($\lambda' = 0$); (4) discriminative model with split training, $\lambda' = 0.1$). Here $\lambda = 0$ and $\lambda' = 0$ means the methods are just NDA methods. Since we have confirmed the effectiveness of split training, we omit the results without it.

We have two observations. (1) Incorporating the rich taxonomy information improves the performance of both generative model and discriminative model by 5.3% and 5.9%, to 87.5% and 90.5% respectively. The rich taxonomy information allows the methods to spread the correct assignments to the other entities of the same type. (2) The discriminative model constantly outperforms the generative model, by 2.7% with the RT method.

To show the detailed gain of employing rich taxonomy, we evaluate the performance of both models with different weight λ: (1) For the generative model, we use $\lambda = 0$, 1, 100, 1000; (2) For the discriminative model, we use $\lambda = 0$, 0.01, 0.1, 1. Fig. 8 shows the results. We can see that generative model is sensitive to the choice of λ, while discriminative model is not that sensitive to it once its larger than 0. The difference of sensitivity corresponds to the objective each model is optimized to. The generative model is optimized towards higher likelihood, while the discriminative model only focus on the accuracy of classification results.

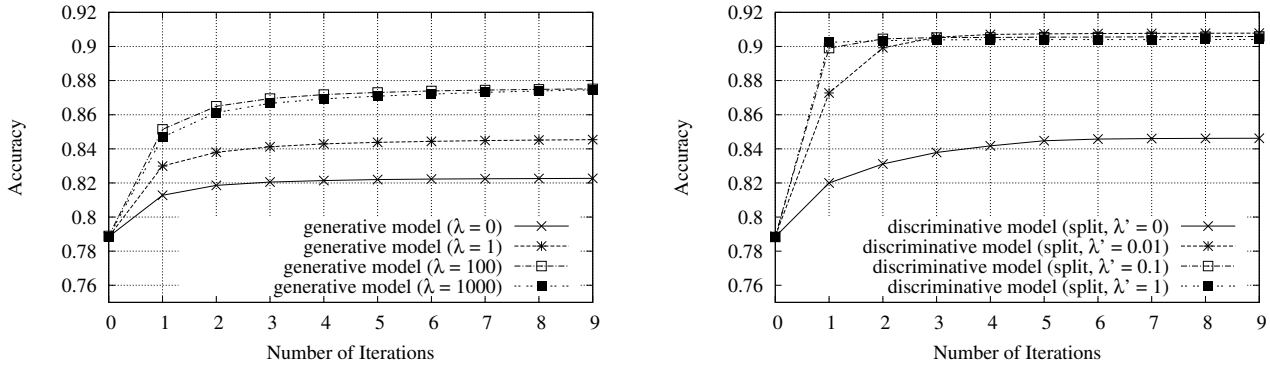
V. Conclusions

In this paper, we use an iterative semi-supervised framework to perform entity disambiguation with Freebase. Two features in Freebase are employed to complement insufficient textual context: naturally disambiguated mention phrases and the rich taxonomy. We conduct experiments on a large scale data set consisting $2,430,707$ English sentences and $33,743$ Freebase entities. The results show that leveraging naturally disambiguated mention phrases and the rich taxonomy helps improve the accuracy significantly. Specifically, even on a high baseline with accuracy over 80%, incorporating the naturally disambiguated aliases improves the accuracy by $3.5\% \sim 6\%$, and incorporating the rich taxonomy improves the accuracy over 5%. Moreover, we use both generative and discriminative models in our framwork, and find that the discriminative model outperforms the generative one constantly.

We have two directions for the future works. First, we plan to design more principled discriminative model instead of using stock classification algorithms, as the effectiveness of discriminative models is clear. Second, we plan to add mention phrase identification into our framework, as not all matching phrase are true mentions to entities, e.g. "jobs" could refer to *Steve Jobs* as well as job positions.

References

[1] R. Bunescu and M. Pasca, "Using encyclopedic knowledge for named entity disambiguation," in *Proceedings of EACL*, vol. 6, 2006, pp. 9–16.



(a) Generative Model (b) Discriminative Model

Figure 8. Accuracy of RT models with different weights for per-type classifier.

[2] M. Dredze, P. McNamee, D. Rao, A. Gerber, and T. Finin, "Entity disambiguation for knowledge base population," in *Proceedings of the 23rd International Conference on Computational Linguistics*. Association for Computational Linguistics, 2010, pp. 277–285.

[3] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor, "Freebase: a collaboratively created graph database for structuring human knowledge," in *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*. ACM, 2008, pp. 1247–1250.

[4] S. Kataria, K. Kumar, R. Rastogi, P. Sen, and S. Sengamedu, "Entity disambiguation with hierarchical topic models," in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2011, pp. 1037–1045.

[5] G. Mann and D. Yarowsky, "Unsupervised personal name disambiguation," in *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*. Association for Computational Linguistics, 2003, pp. 33–40.

[6] C. Wang, K. Chakrabarti, T. Cheng, and S. Chaudhuri, "Targeted disambiguation of ad-hoc, homogeneous sets of named entities," in *Proceedings of the 21st international conference on World Wide Web*. ACM, 2012, pp. 719–728.

[7] X. Han and J. Zhao, "Named entity disambiguation by leveraging wikipedia semantic knowledge," in *Proceeding of the 18th ACM conference on Information and knowledge management*. ACM, 2009, pp. 215–224.

[8] D. Milne and I. Witten, "Learning to link with wikipedia," in *Proceeding of the 17th ACM conference on Information and knowledge management*. ACM, 2008, pp. 509–518.

[9] R. Mihalcea and A. Csomai, "Wikify!: linking documents to encyclopedic knowledge," in *CIKM*, vol. 7, 2007, pp. 233–242.

[10] S. Kulkarni, A. Singh, G. Ramakrishnan, and S. Chakrabarti, "Collective annotation of wikipedia entities in web text," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2009, pp. 457–466.

[11] Z. Zheng, F. Li, M. Huang, and X. Zhu, "Learning to link entities with knowledge base," in *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, 2010, pp. 483–491.

[12] I. Bhattacharya and L. Getoor, "A latent dirichlet model for unsupervised entity resolution," in *SDM*, 2006.

[13] J. Boyd-Graber, D. Blei, and X. Zhu, "A topic model for word sense disambiguation," in *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, 2007, pp. 1024–1033.

[14] L. Shu, B. Long, and W. Meng, "A latent topic model for complete entity resolution," in *Data Engineering, 2009. ICDE'09. IEEE 25th International Conference on*. Ieee, 2009, pp. 880–891.

[15] A. Jordan, "On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes," *Advances in neural information processing systems*, vol. 14, p. 841, 2002.

[16] S. Gottipati and J. Jiang, "Linking entities to a knowledge base with query expansion," in *Empirical Methods in Natural Language Processing (EMNLP)*, 2011.

[17] X. Han and J. Zhao, "Structural semantic relatedness: a knowledge-based method to named entity disambiguation," in *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2010, pp. 50–59.

[18] T. Lee, Z. Wang, H. Wang, and S. Hwang, "Web scale taxonomy cleansing," *Proceedings of the VLDB Endowment*, vol. 4, no. 12, 2011.

[19] W. Wu, H. Li, H. Wang, and K. Zhu, "Towards a probabilistic taxonomy of many concepts," Technical Report MSR-TR-2011-25, Microsoft Research, Tech. Rep., 2011.