A Generative Entity-Mention Model for Linking Entities with Knowledge Base

Xianpei Han **Le Sun**
Institute of Software, Chinese Academy of Sciences
HaiDian District, Beijing, China.
{xianpei, sunle}@nfs.iscas.ac.cn

Abstract

Linking entities with knowledge base (entity linking) is a key issue in bridging the textual data with the structural knowledge base. Due to the name variation problem and the name ambiguity problem, the entity linking decisions are critically depending on the heterogenous knowledge of entities. In this paper, we propose a generative probabilistic model, called *entity-mention model*, which can leverage heterogenous entity knowledge (including *popularity knowledge*, *name knowledge* and *context knowledge*) for the entity linking task. In our model, each name mention to be linked is modeled as a sample generated through a three-step generative story, and the entity knowledge is encoded in the distribution of entities in document $P(e)$, the distribution of possible names of a specific entity $P(s|e)$, and the distribution of possible contexts of a specific entity $P(c|e)$. To find the referent entity of a name mention, our method combines the evidences from all the three distributions $P(e)$, $P(s|e)$ and $P(c|e)$. Experimental results show that our method can significantly outperform the traditional methods.

1 Introduction

In recent years, due to the proliferation of knowledge-sharing communities like *Wikipedia*[1] and the many research efforts for the automated knowledge base population from Web like the *Read the Web*[2] project, more and more large-scale knowledge bases are available. These knowledge bases contain rich knowledge about the world's entities, their semantic properties, and the semantic relations between each other. One of the most notorious examples is *Wikipedia*: its 2010 English version contains more than 3 million entities and 20 million semantic relations. Bridging these knowledge bases with the textual data can facilitate many different tasks such as entity search, information extraction and text classification. For example, as shown in Figure 1, knowing the word *Jordan* in the document refers to a basketball player and the word *Bulls* refers to a NBA team would be helpful in classifying this document into the *Sport/Basketball* class.

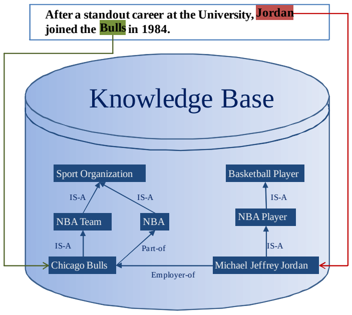

Figure 1. A Demo of Entity Linking

A key issue in bridging the knowledge base with the textual data is linking the entities in a document with their referents in a knowledge base, which is usually referred to as the *Entity Linking* task. Given a set of name mentions $M = \{m_1, m_2, \ldots, m_k\}$ contained in documents and a knowledge base *KB* containing a set of entities $E = \{e_1, e_2, \ldots, e_n\}$, an entity linking system is a function $\sigma : M \rightarrow E$ which links these name mentions to their referent entities in *KB*. For example, in Figure 1 an entity linking system should link the name mention *Jordan* to the entity *Michael Jeffrey Jordan* and the name mention *Bulls* to the entity *Chicago Bulls*.

The entity linking task, however, is not trivial due to the name variation problem and the name ambiguity problem. Name variation means that an entity can be mentioned in different ways such as *full name*, *aliases*, *acronyms* and *misspellings*. For

[1] http://www.wikipedia.org/
[2] http://rtw.ml.cmu.edu/

example, the entity *Michael Jeffrey Jordan* can be mentioned using more than 10 names, such as *Michael Jordan*, *MJ* and *Jordan*. The name ambiguity problem is related to the fact that a name may refer to different entities in different contexts. For example, the name *Bulls* can refer to more than 20 entities in Wikipedia, such as the NBA team *Chicago Bulls*, the football team *Belfast Bulls* and the cricket team *Queensland Bulls*.

Complicated by the name variation problem and the name ambiguity problem, the entity linking decisions are critically depending on the knowledge of entities (Li et al., 2004; Bunescu & Pasca, 2006; Cucerzan, 2007; Milne & Witten, 2008 and Fader et al., 2009). Based on the previous work, we found that the following three types of entity knowledge can provide critical evidence for the entity linking decisions:

- **Popularity Knowledge**. The popularity knowledge of entities tells us the likelihood of an entity appearing in a document. In entity linking, the entity popularity knowledge can provide a *priori* information to the possible referent entities of a name mention. For example, without any other information, the popularity knowledge can tell that in a Web page the name "*Michael Jordan*" will more likely refer to the notorious basketball player *Michael Jeffrey Jordan*, rather than the less popular Berkeley professor *Michael I. Jordan*.
- **Name Knowledge**. The name knowledge tells us the possible names of an entity and the likelihood of a name referring to a specific entity. For example, we would expect the name knowledge tells that both the "*MJ*" and "*Michael Jordan*" are possible names of the basketball player *Michael Jeffrey Jordan*, but the "*Michael Jordan*" has a larger likelihood. The name knowledge plays the central role in resolving the name variation problem, and is also helpful in resolving the name ambiguity problem.
- **Context Knowledge**. The context knowledge tells us the likelihood of an entity appearing in a specific context. For example, given the context "*__wins NBA MVP*", the name "*Michael Jordan*" should more likely refer to the basketball player *Michael Jeffrey Jordan* than the Berkeley professor *Michael I. Jordan*. Context knowledge is crucial in solving the name ambiguities.

Unfortunately, in entity linking system, the modeling and exploitation of these types of entity knowledge is not straightforward. As shown above, these types of knowledge are heterogenous, making it difficult to be incorporated in the same model. Furthermore, in most cases the knowledge of entities is not explicitly given, making it challenging to extract the entity knowledge from data.

To resolve the above problems, this paper proposes a generative probabilistic model, called *entity-mention model*, which can leverage the heterogeneous entity knowledge (including popularity knowledge, name knowledge and context knowledge) for the entity linking task. In our model, each name mention is modeled as a sample generated through a three-step generative story, where the entity knowledge is encoded in three distributions: the entity popularity knowledge is encoded in the distribution of entities in document $P(e)$, the entity name knowledge is encoded in the distribution of possible names of a specific entity $P(s|e)$, and the entity context knowledge is encoded in the distribution of possible contexts of a specific entity $P(c|e)$. The $P(e)$, $P(s|e)$ and $P(c|e)$ are respectively called the *entity popularity model*, the *entity name model* and the *entity context model*. To find the referent entity of a name mention, our method combines the evidences from all the three distributions $P(e)$, $P(s|e)$ and $P(c|e)$. We evaluate our method on both Wikipedia articles and general newswire documents. Experimental results show that our method can significantly improve the entity linking accuracy.

Our Contributions. Specifically, the main contributions of this paper are as follows:

1) We propose a new generative model, the *entity-mention model*, which can leverage heterogenous entity knowledge (including popularity knowledge, name knowledge and context knowledge) for the entity linking task;

2) By modeling the entity knowledge as probabilistic distributions, our model has a statistical foundation, making it different from most previous *ad hoc* approaches.

This paper is organized as follows. The entity-mention model is described in Section 2. The model estimation is described in Section 3. The experimental results are presented and discussed in Section 4. The related work is reviewed in Section 5. Finally we conclude this paper in Section 6.

2 The Generative Entity-Mention Model for Entity Linking

In this section we describe the generative entity-mention model. We first describe the generative story of our model, then formulate the model and show how to apply it to the entity linking task.

2.1 The Generative Story

In the entity mention model, each name mention is modeled as a generated sample. For demonstration, Figure 2 shows two examples of name mention generation. As shown in Figure 2, the generative story of a name mention is composed of three steps, which are detailed as follows:

(i) Firstly, the model chooses the referent entity e of the name mention from the given knowledge base, according to the distribution of entities in document *P(e)*. In Figure 2, the model chooses the entity "*Michael Jeffrey Jordan*" for the first name mention, and the entity "*Michael I. Jordan*" for the second name mention;

(ii) Secondly, the model outputs the name s of the name mention according to the distribution of possible names of the referent entity *P(s|e)*. In Figure 2, the model outputs "*Jordan*" as the name of the entity "*Michael Jeffrey Jordan*", and the "*Michael Jordan*" as the name of the entity "*Michael I. Jordan*";

(iii) Finally, the model outputs the context c of the name mention according to the distribution of possible contexts of the referent entity *P(c|e)*. In Figure 2, the model outputs the context "*joins Bulls in 1984*" for the first name mention, and the context "*is a professor in UC Berkeley*" for the second name mention.

2.2 Model

Based on the above generative story, the probability of a name mention m (its context is c and its name is s) referring to a specific entity e can be expressed as the following formula (here we assume that s and c are independent given e):

$$P(m,e) = P(s,c,e) = P(e)P(s\,|\,e)P(c\,|\,e)$$

This model incorporates the three types of entity knowledge we explained earlier: *P(e)* corresponds to the popularity knowledge, *P(s|e)* corresponds to the name knowledge and *P(c|e)* corresponds to the context knowledge.

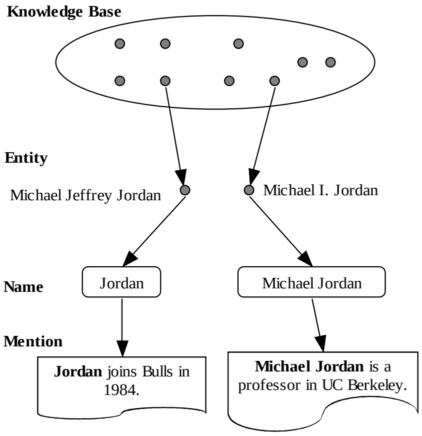

Figure 2. Two examples of name mention generation

Given a name mention m, to perform entity linking, we need to find the entity e which maximizes the probability *P(e|m)*. Then we can resolve the entity linking task as follows:

$$\mathrm{e} = \arg\max_{e} \frac{P(m,e)}{P(m)} = \arg\max_{e} \; P(e)P(s\,|\,e)P(c\,|\,e)$$

Therefore, the main problem of entity linking is to estimate the three distributions *P(e)*, *P(s|e)* and *P(c|e)*, i.e., to extract the entity knowledge from data. In Section 3, we will show how to estimate these three distributions.

Candidate Selection. Because a knowledge base usually contains millions of entities, it is time-consuming to compute all *P(m,e)* scores between a name mention and all the entities contained in a knowledge base. To reduce the time required, the entity linking system employs a candidate selection process to filter out the impossible referent candidates of a name mention. In this paper, we adopt the candidate selection method of *NLPR_KBP* system (Han and Zhao, 2009), the main idea of which is first building a name-to-entity dictionary using the *redirect links*, *disambiguation pages*, *anchor texts* of Wikipedia, then the candidate entities of a name mention are selected by finding its name's corresponding entry in the dictionary.

3 Model Estimation

Section 2 shows that the entity mention model can decompose the entity linking task into the estimation of three distributions $P(e)$, $P(s|e)$ and $P(c|e)$. In this section, we describe the details of the estimation of these three distributions. We first introduce the training data, then describe the estimation methods.

3.1 Training Data

In this paper, the training data of our model is a set of annotated name mentions $M = \{m_1, m_2, \dots, m_n\}$. Each annotated name mention is a triple $m=\{s, e, c\}$, where s is the name, e is the referent entity and c is the context. For example, two annotated name mentions are as follows:

- *Jordan* | **Michael Jeffrey Jordan** | *... wins his first NBA MVP in 1991.*
- *NBA* | **National Basketball Association** | *... is the pre-eminent men's professional basketball league.*

In this paper, we focus on the task of linking entities with Wikipedia, even though the proposed method can be applied to other resources. We will only show how to get the training data from Wikipedia. In Wikipedia, a hyperlink between two articles is an annotated name mention (Milne & Witten, 2008): its anchor text is the name and its target article is the referent entity. For example, in following hyperlink (in Wiki syntax), the *NBA* is the name and the *National Basketball Association* is the referent entity.

> "*He won his first* **[[National Basketball Association | NBA]]** *championship with the Bulls*"

Therefore, we can get the training data by collecting all annotated name mentions from the hyperlink data of Wikipedia. In total, we collected more than 23,000,000 annotated name mentions.

3.2 Entity Popularity Model

The distribution *P(e)* encodes the popularity knowledge as a distribution of entities, i.e., the $P(e_1)$ should be larger than $P(e_2)$ if e_1 is more popular than e_2. For example, on the Web the *P(Michael Jeffrey Jordan)* should be higher than the *P(Michael I. Jordan)*. In this section, we estimate the distribution *P(e)* using a model called *entity popularity model*.

Given a knowledge base *KB* which contains *N* entities, in its simplest form, we can assume that all entities have equal popularity, and the distribution *P(e)* can be estimated as:

$$P(e) = 1/N$$

However, this does not reflect well the real situation because some entities are obviously more popular than others. To get a more precise estimation, we observed that a more popular entity usually appears more times than a less popular entity in a large text corpus, i.e., more name mentions refer to this entity. For example, in Wikipedia the NBA player *Michael Jeffrey Jordan* appears more than 10 times than the Berkeley professor *Michael I. Jordan*. Based on the above observation, our entity popularity model uses the entity frequencies in the name mention data set *M* to estimate the distribution *P(e)* as follows:

$$P(e) = \frac{Count(e) + 1}{|M| + N}$$

where *Count(e)* is the count of the name mentions whose referent entity is *e*, and the $|M|$ is the total name mention size. The estimation is further smoothed using the simple *add-one smoothing* method for the zero probability problem. For illustration, Table 1 shows three selected entities' popularity.

Entity	Popularity
National Basketball Association	$1.73*10^{-5}$
Michael Jeffrey Jordan(NBA player)	$8.21*10^{-6}$
Michael I. Jordan(Berkeley Professor)	$7.50*10^{-8}$

Table 1. Three examples of entity popularity

3.3 Entity Name Model

The distribution *P(s|e)* encodes the name knowledge of entities, i.e., for a specific entity *e*, its more frequently used name should be assigned a higher *P(s|e)* value than the less frequently used name, and a zero *P(s|e)* value should be assigned to those never used names. For instance, we would expect the *P*(*Michael Jordan*|*Michael Jeffrey Jordan*) to be high, *P*(*MJ*|*Michael Jeffrey Jordan*) to be relative high and *P*(*Michael I. Jordan*|*Michael Jeffrey Jordan*) to be zero.

Intuitively, the name model can be estimated by first collecting all (entity, name) pairs from the name mention data set, then using the maximum likelihood estimation:

$$P(s \mid e) = \frac{Count(e,s)}{\sum_s Count(e,s)}$$

where the *Count(e,s)* is the count of the name mentions whose referent entity is *e* and name is *s*. However, this method does not work well because it cannot correctly deal with an unseen entity or an unseen name. For example, because the name "*MJ*" doesn't refer to the *Michael Jeffrey Jordan* in Wikipedia, the name model will not be able to identify "*MJ*" as a name of him, even "*MJ*" is a popular name of *Michael Jeffrey Jordan* on Web.

To better estimate the distribution *P(s|e)*, this paper proposes a much more generic model, called *entity name model*, which can capture the variations (including *full name*, *aliases*, *acronyms* and *misspellings*) of an entity's name using a statistical translation model. Given an entity's name *s*, our model assumes that it is a translation of this entity's full name *f* using the IBM model 1 (Brown, et al., 1993). Let Σ be the vocabulary containing all words may be used in the name of entities, the entity name model assumes that a word in Σ can be translated through the following four ways:

1) It is retained (translated into itself);
2) It is translated into its acronym;
3) It is omitted(translated into the word *NULL*);
4) It is translated into another word (misspelling or alias).

In this way, all name variations of an entity are captured as the possible translations of its full name. To illustrate, Figure 3 shows how the full name "*Michael Jeffrey Jordan*" can be transalted into its misspelling name "*Micheal Jordan*".

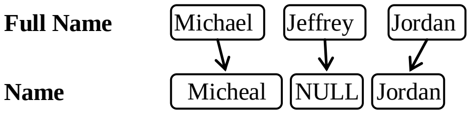

Figure 3. The translation from *Michael Jefferey Jordan* to *Micheal Jordan*

Based on the translation model, *P(s|e)* can be written as:

$$P(s\,|\,e) = \frac{\varepsilon}{(l_f + 1)^{l_s}} \prod_{j=1}^{l_s} \sum_{i=0}^{l_f} t(s_i \mid f_j)$$

where ε is a normalization factor, *f* is the full name of entity *e*, l_f is the length of *f*, l_s is the length of the name *s*, s_i the i^{th} word of *s*, f_j is the j^{th} word of *f* and $t(s_i|f_j)$ is the lexical translation probability which indicates the probability of a word f_j in the full name will be written as s_i in the output name.

Now the main problem is to estimate the lexical translation probability $t(s_i|f_j)$. In this paper, we first collect the (*name*, *entity full name*) pairs from all annotated name mentions, then get the lexical translation probability by feeding this data set into an IBM model 1 training system (we use the GIZA++ Toolkit[3]).

Table 2 shows several resulting lexical translation probabilities through the above process. We can see that the entity name model can capture the different name variations, such as the acronym (*Michael*→*M*), the misspelling (*Michael*→*Micheal*) and the omission (*St.* → *NULL*).

Full name word	Name word	Probability
Michael	*Michael*	*0.77*
Michael	*M*	*0.008*
Michael	*Micheal*	$2.64{*}10^{-4}$
Jordan	*Jordan*	*0.96*
Jordan	*J*	$6.13{*}10^{-4}$
St.	*NULL*	*0.14*
Sir	*NULL*	*0.02*

Table 2. Several lexical translation probabilities

3.4 Entity Context Model

The distribution *P(c|e)* encodes the context knowledge of entities, i.e., it will assign a high *P(c|e)* value if the entity *e* frequently appears in the context *c*, and will assign a low *P(c|e)* value if the entity *e* rarely appears in the context *c*. For example, given the following two contexts:

C1: __*wins NBA MVP.*

C2: __*is a researcher in machine learning.*

Then *P(C1|Michael Jeffrey Jordan)* should be high because the NBA player *Michael Jeffrey Jordan* often appears in C1 and the *P(C2|Michael Jeffrey Jordan)* should be extremely low because he rarely appears in C2.

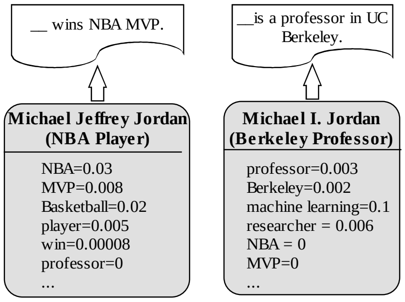

Figure 4. Two entity context models

To estimate the distribution *P(c|e)*, we propose a method based on language modeling, called *entity context model*. In our model, the context of each name mention *m* is the word window surrounding *m*, and the window size is set to 50 according to the experiments in (Pedersen et al., 2005). Specifically, the context knowledge of an entity *e* is encoded in an unigram language model:

$$M_e = \{P_e(t)\}$$

where $P_e(t)$ is the probability of the term *t* appearing in the context of *e*. In our model, the term may indicate a word, a named entity (extracted using the *Stanford Named Entity*

[3] http://fjoch.com/GIZA++.html

Recognizer[4]) or a Wikipedia concept (extracted using the method described in (Han and Zhao, 2010)). Figure 4 shows two entity context models and the contexts generated using them.

Now, given a context *c* containing n terms $t_1t_2...t_n$, the entity context model estimates the probability *P(c|e)* as:

$$P(c \mid e) = P(t_1 t_2 ... t_n \mid M_e) = P_e(t_1) P_e(t_2) .... P_e(t_n)$$

So the main problem is to estimate $P_e(t)$, the probability of a term *t* appearing in the context of the entity *e*.

Using the annotated name mention data set *M*, we can get the maximum likelihood estimation of $P_e(t)$ as follows:

$$P_{e\_ML}(t) = \frac{Count_e(t)}{\sum_t Count_e(t)}$$

where $Count_e(t)$ is the frequency of occurrences of a term *t* in the contexts of the name mentions whose referent entity is *e*.

Because an entity *e*'s name mentions are usually not enough to support a robust estimation of $P_e(t)$ due to the sparse data problem (Chen and Goodman, 1999), we further smooth $P_e(t)$ using the Jelinek-Mercer smoothing method (Jelinek and Mercer, 1980):

$$P_e(t) = \lambda P_{e\_ML}(t) + (1-\lambda) P_g(t)$$

where $P_g(t)$ is a general language model which is estimated using the whole Wikipedia data, and the optimal value of λ is set to 0.2 through a learning process shown in Section 4.

3.5 The NIL Entity Problem

By estimating *P(e)*, *P(s|e)* and *P(c|e)*, our method can effectively link a name mention to its referent entity contained in a knowledge base. Unfortunately, there is still the *NIL entity problem* (McNamee and Dang, 2009), i.e., the referent entity may not be contained in the given knowledge base. In this situation, the name mention should be linked to the NIL entity. Traditional methods usually resolve this problem with an additional classification step (Zheng et al. 2010): a classifier is trained to identify whether a name mention should be linked to the NIL entity.

Rather than employing an additional step, our entity mention model seamlessly takes into account the NIL entity problem. The start assumption of our solution is that "If a name mention refers to a specific entity, then the probability of this name mention is generated by the specific entity's model should be significantly higher than the probability it is generated by a general language model". Based on the above assumption, we first add a pseudo entity, the NIL entity, into the knowledge base and assume that the NIL entity generates a name mention according to the general language model P_g, without using any entity knowledge; then we treat the NIL entity in the same way as other entities: if the probability of a name mention is generated by the NIL entity is higher than all other entities in Knowledge base, we link the name mention to the NIL entity. Based on the above discussion, we compute the three probabilities of the NIL entity: *P(e)*, *P(s|e)* and *P(c|e)* as follows:

$$P(NIL) = \frac{1}{|M| + N}$$

$$P(s \mid NIL) = \prod_{t \in s} P_g(t)$$

$$P(c \mid NIL) = \prod_{t \in c} P_g(t)$$

4 Experiments

In this section, we assess the performance of our method and compare it with the traditional methods. In following, we first explain the experimental settings in Section 4.1, 4.2 and 4.3, then evaluate and discuss the results in Section 4.4.

4.1 Knowledge Base

In our experiments, we use the Jan. 30, 2010 English version of Wikipedia as the knowledge base, which contains over 3 million distinct entities.

4.2 Data Sets

To evaluate the entity linking performance, we adopted two data sets: the first is *WikiAmbi*, which is used to evaluate the performance on Wikipedia articles; the second is *TAC_KBP*, which is used to evaluate the performance on general newswire documents. In following, we describe these two data sets in detail.

WikiAmbi: The *WikiAmbi* data set contains 1000 annotated name mentions which are randomly selected from Wikipedia hyperlinks data set (as shown in Section 3.1, the hyperlinks between Wikipedia articles are manually annotated name mentions). In *WikiAmbi*, there were 207 distinct

[4] http://nlp.stanford.edu/software/CRF-NER.shtml

names and each name contains at least two possible referent entities (on average 6.7 candidate referent entities for each name) [5]. In our experiments, the name mentions contained in the *WikiAmbi* are removed from the training data.

TAC_KBP: The *TAC_KBP* is the standard data set used in the Entity Linking task of the TAC 2009 (McNamee and Dang, 2009). The *TAC_KBP* contains 3904 name mentions which are selected from English newswire articles. For each name mention, its referent entity in Wikipedia is manually annotated. Overall, 57% (2229 of 3904) name mentions's referent entities are missing in Wikipedia, so *TAC_KBP* is also suitable to evaluate the NIL entity detection performance.

The above two data sets can provide a standard testbed for the entity linking task. However, there were still some limitations of these data sets: First, these data sets only annotate the salient name mentions in a document, meanwhile many NLP applications need all name mentions are linked. Second, these data sets only contain well-formed documents, but in many real-world applications the entity linking often be applied to noisy documents such as product reviews and microblog messages. In future, we want to develop a data set which can reflect these real-world settings.

4.3 Evaluation Criteria

We adopted the standard performance metrics used in the Entity Linking task of the TAC 2009 (McNamee and Dang, 2009). These metrics are:

- Micro-Averaged Accuracy (**Micro-Accuracy**): measures entity linking accuracy averaged over all the name mentions;
- Macro-Averaged Accuracy (**Macro-Accuracy)**: measures entity linking accuracy averaged over all the target entities.

As in TAC 2009, we used **Micro-Accuracy** as the primary performance metric.

4.4 Experimental Results

We compared our method with three baselines: (1) The first is the traditional *Bag of Words* based method (Cucerzan, 2007): a name mention's referent entity is the entity which has the highest cosine similarity with its context – we denoted it as *BoW*; (2) The second is the method described in (Medelyan et al., 2008), where a name mention's referent entity is the entity which has the largest average semantic relatedness with the name mention's unambiguous context entities – we denoted it as *TopicIndex*. (3) The third one is the same as the method described in (Milne & Witten, 2008), which uses learning techniques to balance the semantic relatedness, commoness and context quality – we denoted it as *Learning2Link.*

4.4.1 Overall Performance

We conduct experiments on both ***WikiAmbi*** and ***TAC_KBP*** datasets with several methods: the baseline *BoW*; the baseline *TopicIndex;* the baseline *Learning2Link*; the proposed method using only popularity knowledge (***Popu***), i.e., the *P(m,e)=P(e)*; the proposed method with one component of the model is ablated(this is used to evaluate the independent contributions of the three components), correspondingly ***Popu+Name***(i.e., the *P(m,e)=P(e)P(s|e)*), ***Name+Context***(i.e., the *P(m,e)=P(c|e)P(s|e)*) and ***Popu+Context*** (i.e., the *P(m,e)=P(e)P(c|e)*); and the full entity mention model (***Full Model***). For all methods, the parameters were configured through 10-fold cross validation. The overall performance results are shown in Table 3 and 4.

	Micro-Accuracy	**Macro-Accuracy**
BoW	*0.60*	*0.61*
TopicIndex	*0.66*	*0.49*
Learning2Link	*0.70*	*0.54*
Popu	*0.39*	*0.24*
Popu + Name	*0.50*	*0.31*
Name+Context	*0.70*	*0.68*
Popu+Context	*0.72*	*0.73*
Full Model	***0.80***	***0.77***

Table 3. The overall results on *WikiAmbi* dataset

	Micro-Accuracy	**Macro-Accuracy**
BoW	*0.72*	*0.75*
TopicIndex	*0.80*	*0.76*
Learning2Link	*0.83*	*0.79*
Popu	*0.60*	*0.53*
Popu + Name	*0.63*	*0.59*
Name+Context	*0.81*	*0.78*
Popu+Context	*0.84*	*0.83*
Full Model	***0.86***	***0.88***

Table 4. The overall results on *TAC-KBP* dataset

From the results in Table 3 and 4, we can make the following observations:

1) Compared with the traditional methods, our entity mention model can achieve a significant

[5] This is because we want to create a highly ambiguous test data set

performance improvement: In *WikiAmbi* and *TAC_KBP* datasets, compared with the *BoW* baseline, our method respectively gets 20% and 14% micro-accuracy improvement; compared with the *TopicIndex* baseline, our method respectively gets 14% and 6% micro-accuracy improvement; compared with the *Learning2Link* baseline, our method respectively gets 10% and 3% micro-accuracy improvement.

2) By incorporating more entity knowledge, our method can significantly improve the entity linking performance: When only using the popularity knowledge, our method can only achieve 49.5% micro-accuracy. By adding the name knowledge, our method can achieve 56.5% micro-accuracy, a 7% improvement over the *Popu*. By further adding the context knowledge, our method can achieve 83% micro-accuracy, a 33.5% improvement over *Popu* and a 26.5% improvement over *Popu+Name*.

3) All three types of entity knowledge contribute to the final performance improvement, and the context knowledge contributes the most: By respectively ablating the popularity knowledge, the name knowledge and the context knowledge, the performance of our model correspondingly reduces 7.5%, 5% and 26.5%.

NIL Entity Detection Performance. To compare the performances of resolving the NIL entity problem, Table 5 shows the micro-accuracies of different systems on the *TAC_KBP* data set (where **All** is the whole data set, **NIL** only contains the name mentions whose referent entity is NIL, **InKB** only contains the name mentions whose referent entity is contained in the knowledge base). From Table 5 we can see that our method can effectively detect the NIL entity meanwhile retaining the high InKB accuracy.

	All	**NIL**	**InKB**
BoW	*0.72*	*0.77*	*0.65*
TopicIndex	*0.80*	***0.91***	*0.65*
Learning2Link	*0.83*	***0.90***	*0.73*
Full Model	***0.86***	***0.90***	***0.79***

Table 5. The NIL entity detection performance on the *TAC_KBP* data set

4.4.2 Optimizing Parameters

Our model needs to tune one parameter: the Jelinek-Mercer smoothing parameter λ used in the entity context model. Intuitively, a smaller λ means that the general language model plays a more important role. Figure 5 plots the tradeoff. In both *WikiAmbi* and *TAC_KBP* data sets, Figure 5 shows that a λ value 0.2 will result in the best performance.

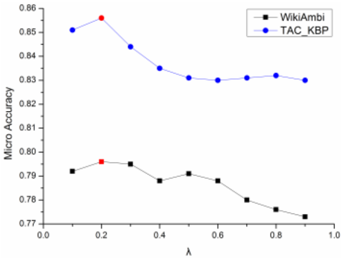

Figure 5. The micro-accuracy vs. λ

4.4.3 Detailed Analysis

To better understand the reasons why and how the proposed method works well, in this Section we analyze our method in detail.

The Effect of Incorporating Heterogenous Entity Knowledge. The first advantage of our method is the entity mention model can incorporate heterogeneous entity knowledge. The Table 3 and 4 have shown that, by incorporating heterogenous entity knowledge (including the name knowledge, the popularity knowledge and the context knowledge), the entity linking performance can obtain a significant improvement.

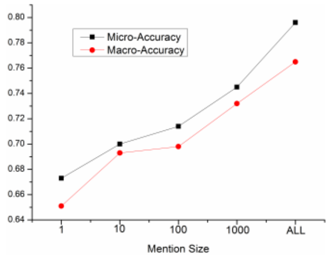

Figure 6. The performance vs. training mention size on *WikiAmbi* data set

The Effect of Better Entity Knowledge Extraction. The second advantage of our method is that, by representing the entity knowledge as probabilistic distributions, our model has a statistical foundation and can better extract the entity knowledge using more training data through the *entity popularity model*, the *entity name model* and the *entity context model*. For instance, we can train a better entity context model *P(c|e)* using more name mentions. To find whether a better entity knowledge extraction will result in a better performance, Figure 6 plots the micro-accuray along with the size of the training data on name mentions for *P(c|e)* of each entity *e*. From Figure 6, we can see that when more training data is used, the performance increases.

4.4.4 Comparision with State-of-the-Art Performance

We also compared our method with the state-of-the-art entity linking systems in the TAC 2009 KBP track (McNamee and Dang, 2009). Figure 7 plots the comparison with the top five performances in TAC 2009 KBP track. From Figure 7, we can see that our method can outperform the state-of-the-art approaches: compared with the best ranking system, our method can achieve a 4% performance improvement.

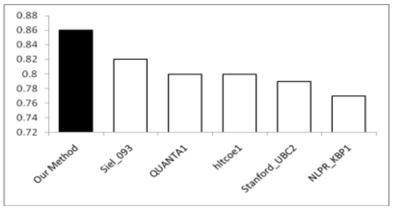

Figure 7. A comparison with top 5 TAC 2009 KBP systems

5 Related Work

In this section, we briefly review the related work. To the date, most entity linking systems employed the context similarity based methods. The essential idea was to extract the discriminative features of an entity from its description, then link a name mention to the entity which has the largest context similarity with it. Cucerzan (2007) proposed a *Bag of Words* based method, which represents each target entity as a vector of terms, then the similarity between a name mention and an entity was computed using the cosine similarity measure. Mihalcea & Csomai (2007), Bunescu & Pasca (2006), Fader et al. (2009) extended the *BoW* model by incorporating more entity knowledge such as popularity knowledge, entity category knowledge, etc. Zheng et al. (2010), Dredze et al. (2010), Zhang et al. (2010) and Zhou et al. (2010) employed the learning to rank techniques which can further take the relations between candidate entities into account. Because the context similarity based methods can only represent the entity knowledge as features, the main drawback of it was the difficulty to incorporate heterogenous entity knowledge.

Recently there were also some entity linking methods based on inter-dependency. These methods assumed that the entities in the same document are related to each other, thus the referent entity of a name mention is the entity which is most related to its contextual entities. Medelyan et al. (2008) found the referent entity of a name mention by computing the weighted average of semantic relatedness between the candidate entity and its unambiguous contextual entities. Milne and Witten (2008) extended Medelyan et al. (2008) by adopting learning-based techniques to balance the semantic relatedness, commoness and context quality. Kulkarni et al. (2009) proposed a method which collectively resolves the entity linking tasks in a document as an optimization problem. The drawback of the inter-dependency based methods is that they are usually specially designed to the leverage of semantic relations, doesn't take the other types of entity knowledge into consideration.

6 Conclusions and Future Work

This paper proposes a generative probabilistic model, the *entity-mention model*, for the entity linking task. The main advantage of our model is it can incorporate multiple types of heterogenous entity knowledge. Furthermore, our model has a statistical foundation, making the entity knowledge extraction approach different from most previous ad hoc approaches. Experimental results show that our method can achieve competitive performance.

In our method, we did not take into account the dependence between entities in the same document. This aspect could be complementary to those we considered in this paper. For our future work, we can integrate such dependencies in our model.

Acknowledgments

The work is supported by the National Natural Science Foundation of China under Grants no. 60773027, 60736044, 90920010, 61070106 and 61003117, and the National High Technology Development 863 Program of China under Grants no. 2008AA01Z145. Moreover, we sincerely thank the reviewers for their valuable comments.

References

Adafre, S. F. & de Rijke, M. 2005. *Discovering missing links in Wikipedia*. In: Proceedings of the 3rd international workshop on Link discovery.

Bunescu, R. & Pasca, M. 2006. *Using encyclopedic knowledge for named entity disambiguation*. In: Proceedings of EACL, vol. 6.

Brown, P., Pietra, S. D., Pietra, V. D., and Mercer, R. 1993. *The mathematics of statistical machine translation: parameter estimation*. Computational Linguistics, 19(2), 263-31.

Chen, S. F. & Goodman, J. 1999. *An empirical study of smoothing techniques for language modeling*. In Computer Speech and Language, London; Orlando: Academic Press, c1986-, pp. 359-394.

Cucerzan, S. 2007. *Large-scale named entity disambiguation based on Wikipedia data*. In: Proceedings of EMNLP-CoNLL, pp. 708-716.

Dredze, M., McNamee, P., Rao, D., Gerber, A. & Finin, T. 2010. *Entity Disambiguation for Knowledge Base Population*. In: Proceedings of the 23rd International Conference on Computational Linguistics.

Fader, A., Soderland, S., Etzioni, O. & Center, T. 2009. *Scaling Wikipedia-based named entity disambiguation to arbitrary web text*. In: Proceedings of Wiki-AI Workshop at IJCAI, vol. 9.

Han, X. & Zhao, J. 2009. *NLPR_KBP in TAC 2009 KBP Track: A Two-Stage Method to Entity Linking*. In: Proceeding of Text Analysis Conference.

Han, X. & Zhao, J. 2010. *Structural semantic relatedness: a knowledge-based method to named entity disambiguation*. In: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics.

Jelinek, Frederick and Robert L. Mercer. 1980. *Interpolated estimation of Markov source parameters from sparse data*. In: Proceedings of the Workshop on Pattern Recognition in Practice.

Kulkarni, S., Singh, A., Ramakrishnan, G. & Chakrabarti, S. 2009. *Collective annotation of Wikipedia entities in web text*. In: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 457-466.

Li, X., Morie, P. & Roth, D. 2004. *Identification and tracing of ambiguous names: Discriminative and generative approaches*. In: *Proceedings of the National Conference on Artificial Intelligence*, pp. 419-424.

McNamee, P. & Dang, H. T. 2009. *Overview of the TAC 2009 Knowledge Base Population Track*. In: Proceeding of Text Analysis Conference.

Milne, D. & Witten, I. H. 2008. *Learning to link with Wikipedia*. In: Proceedings of the 17th ACM conference on Conference on information and knowledge management.

Milne, D., et al. 2006. *Mining Domain-Specific Thesauri from Wikipedia: A case study*. In Proc. of IEEE/WIC/ACM WI.

Medelyan, O., Witten, I. H. & Milne, D. 2008. *Topic indexing with Wikipedia*. In: Proceedings of the AAAI WikiAI workshop.

Mihalcea, R. & Csomai, A. 2007. *Wikify!: linking documents to encyclopedic knowledge*. In: Proceedings of the sixteenth ACM conference on Conference on information and knowledge management, pp. 233-242.

Pedersen, T., Purandare, A. & Kulkarni, A. 2005. *Name discrimination by clustering similar contexts*. Computational Linguistics and Intelligent Text Processing, pp. 226-237.

Zhang, W., Su, J., Tan, Chew Lim & Wang, W. T. 2010. *Entity Linking Leveraging Automatically Generated Annotation*. In: Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010).

Zheng, Z., Li, F., Huang, M. & Zhu, X. 2010. *Learning to Link Entities with Knowledge Base*. In: The Proceedings of the Annual Conference of the North American Chapter of the ACL.

Zhou, Y., Nie, L., Rouhani-Kalleh, O., Vasile, F. & Gaffney, S. 2010. *Resolving Surface Forms to Wikipedia Topics*. In: Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pp. 1335-1343.