Domain-Specific Entity Linking via Fake Named Entity Detection

Jiangtao Zhang[1(✉)], Juanzi Li[1], Xiao-Li Li[2], Yao Shi[1], Junpeng Li[3], and Zhigang Wang[1]

[1] Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China
zhang-jt13@mails.tsinghua.edu.cn, lijuanzi@tsinghua.edu.cn, wangzigo@gmail.com
[2] Institute for Infocomm Research, A*STAR, Singapore 138632, Singapore
xlli@i2r.a-star.edu.sg
[3] Software School, Xidian University, Xian 710126, China
jpl_xd@163.com

Abstract. The traditional named entity detection (NED) and entity linking (EL) techniques cannot be applied to domain-specific knowledge base effectively. Most of existing techniques just take extracted named entities as the input to the following EL task without considering the interdependency between the NED and EL and how to detect the Fake Named Entities (FNEs). In this paper, we propose a novel approach to jointly model NED and EL for domain-specific knowledge base, facilitating mentions extracted from unstructured data to be accurately matched to uniquely identifiable entities in the given domain-specific knowledge base. We conduct extensive experiments for movie knowledge base by a data set of real-world movie comments, and our experimental results demonstrate that our proposed approach is able to achieve 84.7 % detection precision for NED and 87.5 % linking accuracy for EL respectively, indicating its practical use for domain-specific knowledge base.

Keywords: Entity linking · Named entity detection · Fake named entity · Domain-specific knowledge base · Joint model

1 Introduction

Entity linking (EL), determining the identity of entities mentioned in text, is the key issue in bridging *unstructured* textual data with *structured* knowledge bases (KBs) [7]. It has been widely used in diverse applications such as question answering, information integration and KB construction [19]. Significant portion of recent research in this area focus on linking named entities in text to *general* knowledge bases, such as Wikipedia based KBs or WordNet based KBs.

Recently, establishing *domain-specific* KBs has been found more effective and useful to manage and query knowledge within a specific domain. For example,

S.B. Navathe et al. (Eds.): DASFAA 2016, Part I, LNCS 9642, pp. 101–116, 2016.
DOI: 10.1007/978-3-319-32025-0_7

IMDB[1] contains more concrete and comprehensive movie knowledge than *general* knowledge base Wikipedia or Baidu Baike. Therefore, *domain-specific* EL techniques become more and more important, with the increasing demand for constructing and populating *domain-specific* KBs. An Entity Discovery and Linking (EDL) task is introduced by KBP 2014[2]. In particular, given an unstructured document, the EDL task aims to automatically extract mentions (i.e. Named Entity Detection, or NED), link them to a general KB, e.g. Wikipedia (i.e. Entity Linking or EL), and identify NIL mentions that do not have corresponding KB entries [9]. However, traditional EL methods are ineffective for *domain-specific* EL tasks, due to the different characteristics between *domain-specific* area and *general* area. Specifically, we observe there are two unsolved key **challenges** in *domain-specific* EL problem:

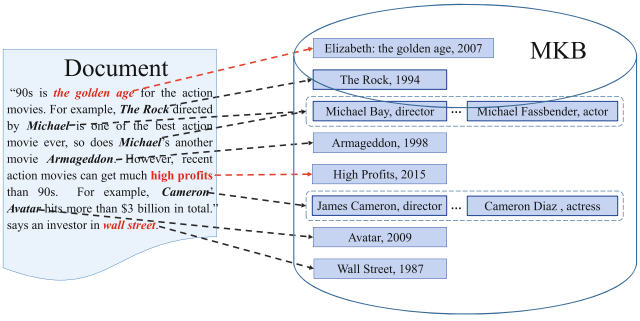

Fig. 1. An illustration for the task of domain-specific entity linking

1. Fake Named Entity: Given a document and a domain-specific KB, there exist many common phrases in the document which could likely be linked to entities in the given KB. However, not all these common phrases should be linked. As an example shown in Fig. 1, the mention "*the golden age*", "*The Rock*", "*high profits*" are all common phrases in general domain. For a *domain-specific* Movie-Knowledge-Base (MKB)[3], however, these mentions are the titles of entities/movies in MKB. As such, these mentions in the document are quite likely to be linked to MKB. Nevertheless, according to their context, except "*The Rock*" is *true* named entity, both "*the golden age*" and "*high profits*" are just common phrases that should not be recognized as named entities. We denote these mentions which should not be linked to entities in KB as **Fake Named Entities** (FNEs). Traditional methods do not consider the FNE issue and thus will not work well for the *domain-specific* EL task due to the fact that there exist many FNEs in a *domain-specific* area. In this paper, we propose a novel technique for FNE detection from the given unstructured text.

[1] http://www.imdb.com/.
[2] http://nlp.rpi.edu/kbp2014/.
[3] MKB is constructed by knowledge engineering laboratory of department of computer science and technology, Tsinghua University, Beijing.

2. Interdependency: Existing techniques typically treat NED and EL as two separated tasks and use a pipeline/sequencial architecture [10–12,17,21] that simply takes extracted named entities as the input to the following EL task, without considering the interdependency between NED and EL tasks. As such, the errors, i.e. FNEs, occurred in the NED task will inevitably affect the performance of the subsequent EL task. For example in Fig. 1, we could mistakenly treat FNEs "*the golden age*", "*wall street*" and "*high profits*" as true mentions and link them to entities in a domain-specific knowledge base (e.g. MKB). However, if we consider NED and EL tasks jointly, such FNE errors could be fixed because we can update the confidence of mentions iteratively. For example, the linked entities/movies of above FNEs are not action movies while the main thread of the text is talking about action movies and all other TNEs in the text is related to action movies. Such context information is the result of the EL and thus can in turn be used to lower the confidence of these FNEs. Furthermore, such information is also useful for the ranking of "*Michael*" and "*Cameron*", which are both famous directors of action movies. Therefore, the errors of FNEs can be fixed because the two tasks EL and NED are inherently coupled. Different from traditional methods, our proposed technique will leverage their interdependency iteratively making both NED and EL tasks more robust.

In summary, detecting the FNEs and linking the true/correct mentions are two significant challenges, because textual mentions in a specific domain could be potentially far more ambiguous than those in general domain. Therefore, we propose a new technique to detect FNEs in a specific domain via jointly modeling named entity detection and entity linking.

Contributions. The main contributions of this paper are summarized as follows.

- We are among the first to explore the problem of joint NED and EL with the domain-specific knowledge base. To the best of our knowledge, our research is the first to define the important concept Fake Named Entities (FNEs), which is critical for *domain-specific* EL task.
- We proposed an effective technique that jointly models NED and EL by iteratively enhancing the confidence of entity extraction and certainty of entity linking. Particularly, we leverage the entity linking result to increase the confidence of true named entities (TNEs) and thus lower the confidence of FNEs. Conversely, this enhancement of extraction/detection confidence improves the performance of the entity linking/disambiguation. This process can be repeated iteratively until convergence, as long as there is an improvement in the extraction and disambiguation.
- To evaluate the effectiveness of our proposed approach, we conducted extensive experiments on a manually annotated data set of real world movie comments and a real *domain-specific* knowledge base. The experimental results show that our proposed approach outperforms baseline methods significantly.

2 Preliminaries

In this section, we first introduce some fundamental concepts for our problem and subsequently define the task of linking named entities in a specific domain.

Domain-Specific Knowledge Base. A domain-specific knowledge base defines a set of representational primitives to model domain knowledge from different perspectives, which can be defined as DSKB= $\{C, E, P, R\}$, where C represents a set of *concepts* in the domain such as actors, movies and producers; $E = \{e_1, e_2, ...e_{|E|}\}$ is the *entities* of concepts such as Steven Spielberg – a movie director; P denotes a set of *properties* to describe attributes of concepts or entities such as actor names, movies' production time; R means the set of triples, each of them describes the *relation* between entities *or* between entity and concept, which can be defined as $\{s, p, o\}$, where $s \in E \cup C, p \in P$, $o \in E \cup C \cup L$, and L is the set of literals. In this paper, we choose domain-specific Movie-Knowledge-Base (MKB) as the target DSKB for our task. The MKB is a high quality knowledge base about movies, TV series and celebrities which integrates several English and Chinese movie data sources from Baidu Baike and Douban, and it contains 23 concepts, 91 properties, more than 700,000 entities and 10 million triples.

Mentions and Linked Entities. We define a *mention* as a textual phrase (e.g., the "*the golden age*" in Fig. 1) which can potentially be linked to some entities in DSKB. We consider every possible n-gram (e.g. $n \leq 5$) as a candidate mention. Given a document d, we define $M = \{m_1, m_2, ..., m_{|M|}\}$ as the set of candidate mentions. In addition, let $E(m) = \{e_1, e_2, ..., e_{|E(m)|}\} \subseteq E$ denote the set of candidate entities which a candidate mention $m \in M$ might be linked to. For a mention m, we define the correct entity $e_m \in E(m)$ which m should actually be linked to as *linked entity* (i.e. ground-truth mapping entity). For example, in Fig. 1, the set of entities that mention "*Cameron*" could be linked to is $E($"*Cameron*"$) = \{$*"James Cameron"*, *"Cameron Diaz"*$\}$ and the *linked entity* is *"James Cameron"*.

Fake Named Entity. We define $M_F = \{m_{f1}, m_{f2}, ..., m_{|M_F|}\} \subseteq M$ that should not be linked to any entity in E, which should only be treated as common textual phrases as *Fake Named Entities*(FNEs). We also define the *True Named Entities* (TNEs) as $M_T = \{m_{t1}, m_{t2}, ..., m_{|M_T|}\} \subseteq M$, denoting the mentions that should be linked to entities in E. Obviously, $M_F \cup M_T = M$. As the example shown in Fig. 1, the set TNEs is $M_T = \{$"*The Rock*", "*Michael*", "*Armageddon*", "*Cameron*", "*Avatar*"$\}$, while the set FNEs is $M_F = \{$"*the golden age*", "*high profits*", "*wall street*"$\}$.

Context Mention and Entity. For a given mention m in a document d, We define all the other candidate mentions $C_M(m) = \{m_{c1}, m_{c2}, ..., m_{|C_M(m)|}\} \subseteq M$

in the same document or in a certain size window as *Context Mentions*. In our experiments, we employ the window size, which is set as 50, following the experimental setting in literature [15]. Notice that each context mention $m_c \in C_M(m)$ could be ambiguous as we do not know its *linked entity* in E. As such, we define *Context Entities* $C_E(m) = \{e_{c1}, e_{c2}, ..., e_{|C_E(m)|}\} \subseteq E$ as the set of most possible *linked entities* for each context mention for the time being.

Task Definition. Given an unstructured document d in a specific domain and a DSKB pertaining to the same domain, our task is to extract TNEs and filter out FNEs in d, and to develop a function $\sigma : M \rightarrow E$ which maps each extracted TNE $m \in M_T$ to its *linked entity* $e \in E(m)$ in DSKB (e.g., MKB). Specifically, our task consists of two parts, namely *Named Entity Detection* (NED) (i.e. *Mention Extraction*) and *Entity Linking* (EL) (i,e. *Disambiguation*). NED is the task of detecting FNEs and extracting TNEs. EL is the task of linking an extracted TNE to a specific definition or instance of an entity in DSKB. The output of our task is the set of *mention* and *entity* mapping pairs: $\{\langle m, \sigma(m)\rangle | \ \forall m \in M_T\}$.

3 Our Proposed Approach

In this section, we propose a novel approach that jointly models NED and EL iteratively to link all the TNEs in an unstructured document to uniquely identifiable entities in DSKB. The main idea of our approach is as follows: in each iterative step, we gradually improve the confidence of TNEs while reduce the confidence of FNEs. Specifically, by leveraging the interdependency of NED and EL, we use the results of EL (linking certainty) to provide the feedback for NED and thus could potentially improve the performance of NED via updating the weights of some features in NED. On the other hand, the results of NED (detection confidence) could also enhance EL process via updating the weights of some features in EL.

3.1 Framework Overview

The framework of our proposed model is shown in Fig. 2. From the figure, we can see that first we train EL and NED models independently based on a manually annotated data set (refer to experiment section) to learn two weighted vectors $\overrightarrow{W}_{el} = \{w_1^{el}, w_2^{el}, w_3^{el}, w_4^{el}\}$ and $\overrightarrow{W}_{ned} = \{w_1^{ned}, w_2^{ned}, w_3^{ned}, w_4^{ned}\}$ for two constructed feature vectors $\overrightarrow{F}_m(e) = \{f_1^{el}, f_2^{el}, f_3^{el}, f_4^{el}\}$ and $\overrightarrow{F}(m) = \{f_1^{ned}, f_2^{ned}, f_3^{ned}, f_4^{ned}\}$ respectively. Next, we apply the two learned models on an input document iteratively to predict TNEs and their *linked entities* in DSKB. The results of NED model $Ned(m)$ show the confidence of a candidate mention m being a TNE while the results of EL model $El_m(e)$ indicate the confidence of a mention m being linked to a candidate entity e. In addition, we take the results of each model to update the weights of some features of the other. That is to say, we can apply the two models mutually and iteratively. With the increase of the number of iterations, the detection confidence and linking certainty of TNEs will increase.

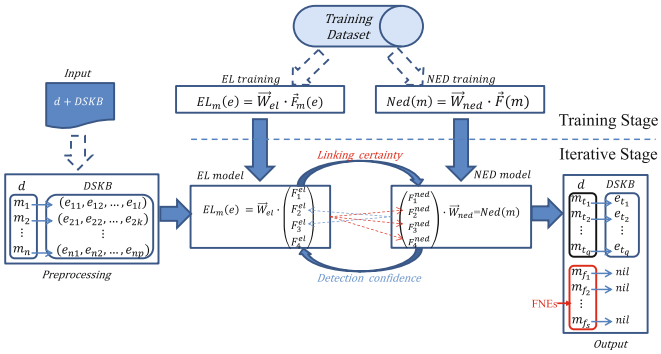

Fig. 2. Framework of our proposed iteratively joint model

Preprocessing. In this subsection, we briefly present how to generate the candidate mentions and entities for a given document d.

First, we build a dictionary D that contains various surface forms of the named entities. The detailed construction method is introduced in [19]. The dictionary D is in the form of $\langle key, value \rangle$ mapping, where the column of the key is a list of surface forms and the column of the mapping $value$ is the set of entities which can be referred to by the key. Next, we consider every possible n-gram (e.g. $n \leq 5$) in d existing in the key column of dictionary D to generate a high-recall candidate mentions M, to avoid missing possible real mentions. Then, For each mention $m \in M$, we search for m in the column of key in D and add the set of entities $value$ to the candidate entities $E(m)$.

3.2 Model Training

As the strategy used by existing studies [13,20,24], we can model NED and EL into two binary classifiers. First we need to construct the training set for these two classifiers based on a manually annotated data set. In EL model, the sample in the training set is a pair $(m, e), e \in E(m)$. Let $label(m, e) \in \{0, 1\}$ indicate positive sample or negative sample, which is determined as follows:

1. For each TNE m, if its true *linked entity* in the candidate entity set $E(m)$ is e_m, then (m, e_m) is a *positive* sample. For all other entities $\forall e_i \in E(m), e_i \neq e_m$, the (m, e_i) are regarded as *negative* samples.
2. For each FNE m, $\forall e \in E(m)$, (m, e) is treated as *negative* sample as well.

In NED model, on the other hand, the sample is m. Also let $label(m) \in \{0, 1\}$ be the indicator of positive sample and negative sample. Obviously, the set of TNEs M_T which are manually annotated is the set of *positive* samples while M_F is the set of *negative* samples.

Next, we learn two weight vectors $\overrightarrow{W}_{el}$ and $\overrightarrow{W}_{ned}$ for two constructed feature vectors $\overrightarrow{F}_m(e)$ and $\overrightarrow{F}(m)$ respectively which will be elaborated in next

subsection by supervised machine learning technique on training data set – in our experiments, we employ state-of-art classification model SVM due to its good performance. Then these two models can be formulated as: $El_m(e) = \overrightarrow{W}_{el} \cdot \overrightarrow{F}_m(e)$, $Ned(m) = \overrightarrow{W}_{ned} \cdot \overrightarrow{F}(m)$. Specifically, for EL model, we will rank all entities in the candidate entity set $E(m)$ and select the entity with highest score $El_m(e)$ as the most possible *linked entity* in DSKB, and for NED model we classify a mention m into a TNE or FNE based on the result score $Ned(m)$.

Obviously the results of EL $El_m(e)$ show the certainty (strength) of m linking to e, while the results of NED $Ned(m)$ indicate the confidence level of m being a TNE. Therefore, we can use the results of each model to calculate features of the other iteratively, which will benefit the performance of both two models. In next section, we will introduce our constructed features of our two models and explain how these features are used to interact between two models in an iterative manner to improve performance.

3.3 Features in EL Model

Popularity. In the domain-specific area, taking movie comments as an example, people tend to review more popular and classic movies. Therefore, we choose *popularity* as an important context-free feature. We define the popularity via leveraging the count information from Baidu Baike as follows:

$$Pri_m(e) = \frac{count_m(e)}{\sum_{e \in E(m)} count_m(e)} \tag{1}$$

where $count_m(e)$ is defined as the number of times that mention m links to entity e in Baidu Baike. Notice that we can calculate this feature in advance.

Context Relatedness. Intuitively, one would expect that *mentions* which co-occur in the same document are related to one or a few topics, or have certain semantic relatedness [19]. In Fig. 1 both "*The Rock*" and "*Armageddon*" are similar action movies and thus people tend to talk/compare them together. Therefore, we propose a second feature: the *context relatedness*. Specifically, we calculate the average value of the semantic relatedness between all context entities $e_c \in C_E$ and the candidate entity $e \in E(m)$ to get the context relatedness.

$$ConRel_m(e) = \frac{\sum_{e_c \in C_E} SmtRel(e_c, e)}{|C_E|} \tag{2}$$

where $SmtRel(e_c, e)$ is semantic relatedness of a context entity e_c and the candidate entity e. However, there are two problems of above calculation:

1. Context mention $m_c \in C_M$ could also be ambiguous when performing the linking of current mention m. Thus, its corresponding context entity e_c is also unknown in current stage. As such, in the follow-up iteration, we use the results of EL model in last iteration to choose the best context entity with highest score, denoted as $e_{top}(m_c)$, for the relatedness calculation.

2. FNEs in the Context Mentions C_M could damage the performance of the following entity linking task. As mentioned above, the results of NED model $Ned(m)$ indicate the true confidence level of a mention m being a TNE. Therefore, we use $Ned(m_c)$ to denote the confidence level of m_c being a FNE. In other words, if a context mention m_c is a FNE with high probability, then the value of $Ned(m_c)$ will thus be small and has less impact to EL tasks.

Therefore, the context relatedness will be the weighted average value of the semantic relatedness of all Context Entities $e_c \in C_E$ and the candidate entity e, i.e.,

$$ConRel_m(e) = \frac{\sum_{m_c \in C_M} Ned(m_c) * SmtRel(e_{top}(m_c), e)}{|C_M|} \tag{3}$$
$$e_{top}(m_c) = \arg\max_{e_c \in E(m_c)} (El_m(e_c))$$

Next, we adopt two techniques: Wikipedia Link-based Measure (WLM) and Jaccard distance to calculate the semantic relatedness to get two kinds of context relatedness, denoted as $ConRel1_m(e)$ and $ConRel2_m(e)$, respectively.

WLM: The WLM is based on the Wikipedia hyperlink structure [13]. Given two entity e_i and e_j, we define the semantic similarity between them as $WLM(e_i, e_j) = \frac{\log(\max(|E_i|,|E_j|)) - \log(|E_i \cap E_j|)}{\log(|W|) - \log(\min(|E_i|,|E_j|))}$, Where E_i and E_j are the sets of entities that link to e_i and e_j respectively in MKB, and W is the set of all entities in MKB.

Jaccard Distance: We first extract the content of two entity e_i and e_j to compose two bag-of-words representations S_i and S_j respectively, and subsequently calculate Jaccard distance between S_i and S_j.

Content Similarity. It has been an effective way to use the context information to perform entity disambiguation. Therefore, we introduce out last feature for the EL model: *content similarity*. We define the content similarity as the similarity between the context around a candidate mention m and its candidate entity e, i.e., $Consim_m(e)$, which also calculated by Jaccard distance.

3.4 Features in NED Model

Link Probability. Link probability introduced in [12] is a proven feature which indicates how often a mention links to an entity in a knowledge base. For example shown in Fig. 1, for the mention "*wall street*", the probability that it links to a certain entity is much less than that just treat it as a common phrase, because in most cases when people mention "*wall street*", they refer it to a location rather than the movie in 1987. Therefore, the link probability is an important feature and is helpful to filter out FNEs. We define the link probability $LP(m)$ as follows.

$$LP(m) = \frac{\sum_{e \in E(m)} count_m(e)}{count(m)} \tag{4}$$

where $count_m(e)$ is defined as the number of times that mention m links to entity e and $count(m)$ is the total occurrence number of m in Baidu Baike. We can also calculate this feature in advance.

Linking Certainty. The results of EL model $El_m(e)$ provide a certainty score whether a mention m correctly links to an entity e. Therefore, we use it as a feature of NED model, to indicate the linking certainty level of a mention. Obviously, the higher the linking certainty that a mention m links to an entity e, the higher the probability that the mention m is not a FNE. We use the value of the highest score entity $e_{top}(m)$, denoted as $El_m(e_{top}(m))$, as the value of linking certainty $LC(m)$ of the mention m. That is to say, we use the results of the EL model as a feature of NED model.

$$LC(m) = El_m(e_{top}(m)) = \max\{El_m(e)|e \in E(m)\}. \tag{5}$$

Coherence. As mentioned above, entities occurring in a given document d are likely to be topically coherent, i.e. they are semantic related. So, we can exploit this topic *coherence* between entities in the document d to define the coherence feature for each candidate mention m, which is defined as the average semantic relatedness between m and all context mentions $m_c \in C_M$. Nevertheless, we still need to know the *linked entity* e for m to calculate the semantic relatedness. Therefore, for each m (and m_c), we also choose the highest score entity $e_{top}(m)$ (and $e_{top}(m_c)$) returned by the EL model as the *linked entity*.

$$Coh(m) = \frac{\sum_{m_c \in C_M} Ned(m_c) * SmtRel(e_{top}(m_c), e_{top}(m))}{|C_M|} \tag{6}$$

We can also calculate two kinds of semantic relatedness for two coherence features $Coh1(m)$ and $Coh2(m)$ as depicted in the Context Relatedness subsection.

3.5 Iterative Process

After training stage, we have learned two weight vectors $\overrightarrow{W}_{el}$ and $\overrightarrow{W}_{ned}$ for our two models. Here, we illustrate our iterative process as follows.

$$\begin{aligned}
\mathbf{El_m(e)} &= \overrightarrow{W}_{el} \cdot \overrightarrow{F}_m(e) \\
&= w_1^{el} * Pri_m(e) + w_2^{el} * ConRel1_m(e) \\
&\quad + w_3^{el} * ConRel2_m(e) + w_4^{el} * ConSim_m(e) \\
&= w_1^{el} * Pri_m(e) + w_2^{el} * \frac{\sum_{m_c \in C_M} \mathbf{Ned(m_c)} * WLM(\mathbf{e_{top}(m_c)}, e)}{|C_M|} \\
&\quad + w_3^{el} * \frac{\sum_{m_c \in C_M} \mathbf{Ned(m_c)} * Jac(\mathbf{e_{top}(m_c)}, e)}{|C_M|} + w_4^{el} * ConSim_m(e)
\end{aligned} \tag{7}$$

$$\begin{aligned}\mathbf{Ned(m)} &= \overrightarrow{W}_{ned} \cdot \overrightarrow{F}(m) \\ &= w_1^{ned} * LP(m) + w_2^{ned} * LC(m) + w_3^{ned} * Coh1(m) + w_4^{ned} * Coh2(m) \\ &= w_1^{ned} * LP(m) + w_2^{ned} * El_m(\mathbf{e_{top}(m)}) + \\ &\quad w_3^{ned} * \frac{\sum_{m_c \in C_M} \mathbf{Ned(m_c)} * WLM(\mathbf{e_{top}(m_c)}, \mathbf{e_{top}(m)})}{|C_M|} \\ &\quad + w_4^{ned} * \frac{\sum_{m_c \in C_M} \mathbf{Ned(m_c)} * Jac(\mathbf{e_{top}(m_c)}, \mathbf{e_{top}(m)})}{|C_M|}\end{aligned} \tag{8}$$

For any iterative process, one of the most important issue is the convergence. Here, we define *iteration deviation* as the maximal value of difference of $Ned(m)$ of two successive iterations for all $m \in M$. Then we set the condition to complete the iteration is that *iteration deviation* is less than a predefined threshold ε, namely *iteration deviation threshold*, i.e.,

$$\max_{m_i \in M} (Ned(m_i^{(j)}) - Ned(m_i^{(j-1)}) \leq \varepsilon \tag{9}$$

From extensive experiments, we found that with the increase of the number of iterations, *iteration deviation* gradually decreases and iterative process usually stops after some iterations. We will discuss the convergence issue in experiment section.

The detailed iterative algorithm is given in Algorithm 1.

Input: $M; \forall m \in M, E(m); \overrightarrow{W}_{el}; \overrightarrow{W}_{ned}$
Output: M_T; $\forall m \in M_T, \langle m, Ned(m)\rangle, \langle m, e_{top}(m), El_m(e_{top}(m))\rangle$
repeat
  for each $m \in M$ **do**
    for each $e \in E(m)$ **do**
      $El_m(e) = \overrightarrow{W}_{el} \cdot \overrightarrow{F}_m(e)$;
    end
    $e_{top}(m) = \arg\max_{e \in E(m)}(El_m(e))$;
  end
  for each $m \in M$ **do**
    $Ned(m) = \overrightarrow{W}_{ned} \cdot \overrightarrow{F}(m)$
  end
until convergence;

Algorithm 1. Algorithm of the iterative process

4 Experiments and Evaluation

To fairly evaluate the effectiveness of our proposed approach, we have conducted extensive experimental studies to compare it with existing methods. All the programs were implemented in Python and all the experiments were conducted on a server (with four 2.7 GHz CPU cores, 1024 GB memory, Ubuntu 13.10).

Table 1. Statistical data of the user data set

Documents	$\|FNEs\|$	$\|TNEs\|$	CEs	$\overline{\|M\|}$	$\overline{\|E(m)\|}$
843	2529	11848	42105	17.05	2.92

4.1 Data Preparation

To the best of our knowledge, there is no publicly available benchmark data set for the domain-specific EL task. Thus, we manually create a first-of-its-kind gold-standard data set for our task. Since there could be subjective in the manual annotation process, in order to avoid introducing bias in the annotation task, we organized annotators into three groups and each group has several members. The first and second groups annotate the same data set *independently*, while the third group checks the annotation results and annotates those inconsistent named entities. The final results are determined by majority voting. Obviously, the annotation task is very time consuming and labour intensive.

In this paper, we focus to perform the entity linking from user movie comments/documents to the knowledge base MKB (i.e. Movie-Knowledge-Base). Specifically, we crawl user comments on movies from established Websites in China, including Sina, Sohu, 163, and Tianya, with 1 year time span, from 1/1/2014 to 12/31/2014. Finally, we obtained 843 documents forming the gold-standard data set. Table 1 lists the size of the data set and some statistical information about the data set, where $|FNEs|$ and $|TNEs|$ denote the numbers of Fake/True named entities respectively, CEs means the total number of candidate entities, $\overline{|M|}$ represents the average number of mentions per document, and $\overline{|E(m)|}$ shows average number of candidate entities per mention.

4.2 Experimental Results

In this subsection, we study the effectiveness of our proposed approach under different configurations, and compare them with some baseline methods.

Baseline Methods. Since most of joint NED and EL frameworks deal with short text linking to general KB based on high complexity algorithms, which could not apply directly on MKB and our data set, and furthermore these works are lack of publicly available APIs, we created two baselines in this paper, both of which employed the traditional pipeline architecture that takes extracted named entities as the input to the following EL task.

1. Prior Probability-based method (POP). First, in NED process, we only used the link probability for detection. Particularly, we set a threshold as 0.2 (which produces the best performance for POP method) and retain the mentions whose link probabilities are higher than the pre-set threshold. In the EL process, on the other hand, we used entity popularity for ranking. In other words, the entities with the highest popularity among all the candidate entities is considered as the *linked entity* for this mention.

2. Vector Similarity-based method ($VSim$). We constructed a context vector for each extracted mention and a profile vector for each candidate entity based on standard TF-IDF representation. Then we measure the similarity between these two vectors by computing their *Cosine* distance. Finally, the entity with the highest similarity is considered as the *linked entity* for the mention. For the NED process, we set a threshold as 0.087 (which gives the best results) and only retain the mention whose highest score of vector similarity is larger than it.

Parameter Setting. In our approach, there exists an important parameter, i.e. *iteration deviation threshold*, which needs to be determined.

Iteration Deviation Threshold. Figure 3 shows the curve of the *iteration deviation* versus the number of iterations for eight documents randomly chosen from our data set. From the results we observe that for each document the *iteration deviation* gradually decreases with the increase of the number of iterations. When the number of iterations exceeds 10, the *iteration deviation* flattens out gradually (≤ 0.001). As such, we can set the *iteration deviation threshold* $\varepsilon = 0.001$. This empirically proves that our iterative process converges, and also verifies our proposed approach that NED and EL contribute to each other within limited iterations.

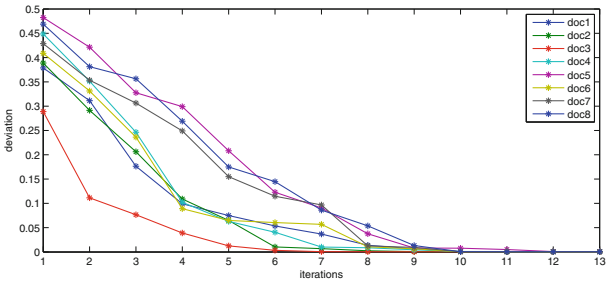

Fig. 3. Convergence of iteration

Our Model and Evaluation Metrics. Now we study the effectiveness of our proposed approach, which is configured into 4 different settings:

- *No Training + No Iterating (NoT+NoI):* we do not use the machine learning method to train the weight of the NED and EL. We assume that all features have the same weight, that is, for NED and EL models, the weight of all features is 0.25. Furthermore, we do not perform the iteration.
- *Training + No Iterating ($T + NoI$):* we use the machine learning method to learn the weight of the NED and EL, but we do not perform the iteration. Notice that $T + NoI$ is actually the traditional machine learning based approach.
- *No Training + Iterating ($NoT + I$):* we set the weight of NED and EL manually, but we perform the iteration.
- *Training + Iterating ($T + I$):* we use the machine learning method to train the NED and EL models and we perform the iteration.

In our proposed approach, EL is performed over noisy NED output and participates to the final decisions about extractions. Therefore, we evaluate NED, EL and their combination by employing the following evaluation metrics:

- **NED:** precision, recall and F1-measure;
- **EL:** accuracy over correctly recognized named entities. Notice here we do not use precision, recall and F1-measure due to the fact that if the correct extracted mentions are given, then precision=recall=F1-measure=accuracy [19] and most EL systems simply use accuracy to assess their performance.
- **Overall NED+EL:** precision, recall and F1-measure, where precision/recall is computed as the product of the NED precision/recall by the EL accuracy.

Table 2. Comparison of experiment results

Approach	NED			EL	Overall NED + EL		
	Precision	Recall	F1	Accuracy	Precision	Recall	F1
POP	0.776	0.643	0.703	0.792	0.615	0.509	0.557
VSim	0.724	0.715	0.719	0.825	0.597	0.590	0.594
NoT+NoI	0.761	0.738	0.749	0.849	0.646	0.627	0.636
T+NoI	0.808	0.754	0.780	0.864	0.698	0.651	0.674
NoT+I	0.826	0.748	0.785	0.852	0.704	0.637	0.669
T+I	0.847	0.788	0.816	0.875	0.741	0.690	0.714

Result and Analysis. Table 2 shows the comparison of our proposed approach and the other two baseline methods. From the results, we can see that 4 different configurations of our proposed approach all significantly outperform the two baseline methods, which demonstrates the effectiveness of our proposed approach.

In general, the performance of EL is higher than NED, for both our approach and baseline methods. That is because the calculation of the EL accuracy is based on the correct results of the NED, i.e., it doesn't take FNEs into consideration. Clearly, we can see that our proposed approach achieves 5.7−8.3 % higher accuracy across all configurations, indicating our approach is very effective for EL task.

Further, for the assessment of the *POP* baseline, obviously, for a mention with high prior probability, the probability of it being a TNE is high. However, due to the fact that *POP* uses the method of simply setting a threshold to exclude the mention with small prior probability, it gets a high precision but low recall. For the *Vsim* baseline, on the other hand, because it considers context rather than prior probability, it is able to get higher recall but lower precision (as it also introduces the FNEs) than *POP*.

Additionally, for our proposed approach with the configuration of $NoT+NoI$, we observe that both NED and EL outperform the two baseline methods because

four features are considered. The performance of $T + NoI$ improves as it introduces the machine learning method that takes the importance of different features into consideration. Meanwhile, key point of the $NoT+I$ is to investigate the influence of the iteration to the FNEs. The results indicate that the precision has been further improved due to the fact that iterations exclude FNEs effectively. However, because there is no training in this configuration, the performance of recall falls as all features are treated equally.

Moreover, we can see that the EL accuracy of $T + NoI$ is higher than that of $NoT + I$, which demonstrates that the contribution of iterations to the EL is small as it does not take FNEs into consideration, while the contribution of training to the EL is bigger as training considers the importance of different features.

Finally, as expected, because both the feature importance and the iterations are included in the $T + I$, it is able to achieve the highest performance both for NED precision and EL accuracy, which is consistent with our intuition, since our proposed approach can obtain more related knowledge about candidate entities.

5 Related Work

The problem of EL and NED has been addressed by many researchers starting from papers [1,5]. However, most of existing approaches [3,4,7,8,23,24] focus on the general-purpose knowledge bases and cannot be applied to the domain-specific knowledge base, as we have discussed before.

In addition, many previous systems have employed a pipeline frameworks [10–12,17,21]. Our work is different as we provide high-quality candidate mentions and entity links by jointly modeling NED and EL tasks iteratively. Recently, work [6,16,22] were proposed to perform named entity detection and entity linking jointly to make these two tasks reinforce each other. But their techniques are best-suited for short microblog text (e.g., tweets), while our techniques are better suited for longer documents. In addition, a key difference is that they link mentions to general knowledge base, while our technique links mentions to domain-specific knowledge bases that become more crucial for many domain specific real-world applications.

From above discussion, we can see that considerable approaches have been proposed for general-purpose knowledge bases. Although there are several existing works [2,14,18] addressing domain-specific NED and EL, this area deserves much deeper exploration by research communities as none of above work consider the issue of FNEs, which is essential in the domain-specific area. In this paper, we jointly use the results of named entity detection and entity disambiguation to detect FNEs for domain-specific knowledge base.

6 Conclusion

The current state-of-the-art entity linking research primarily focus on *general* knowledge bases, instead of potentially very useful *domain-specific* knowledge

bases. As such, they do not consider two critical problems that we have identified for *domain-specific* knowledge bases, namely *fake named entities* and the *interdependency* between the named entity detection (NED) and entity linking (EL). In this paper, we have proposed a novel approach that dedicates to address the two issues by jointly modeling NED and EL iteratively. We observe from our experimental results that our proposed approach is highly effective comparing with existing baseline methods, indicating it is very promising to be used for many *domain-specific* real-world applications.

Acknowledgements. 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

1. Bunescu, R., Pasca, M.: Using encyclopedic knowledge for named entity disambiguation. In: Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2006), pp. 9–16 (2006)
2. Dalvi, N., Kumar, R., Pang, B.: Object matching in tweets with spatial models. In: Proceedings of the Fifth ACM International Conference on Web Search and Data Mining, pp. 43–52 (2012)
3. Finkel, J.R., Grenager, T., Manning, C.: Incorporating non-local information into information extraction systems by Gibbs sampling. In: ACL 2005, pp. 363–370 (2005)
4. Gottipati, S., Jiang, J.: Linking entities to a knowledge base with query expansion. In: EMNLP 2011, pp. 804–813 (2011)
5. Grishman, R., Sundheim, B.: Message understanding conference-6: a brief history. In: Proceedings of the 16th Conference on Computational Linguistics, vol. 1, pp. 466–471 (1996)
6. Guo, S., Chang, M.W., Kiciman, E.: To link or not to link? a study on end-to-end tweet entity linking. In: HLT-NAACL, pp. 1020–1030 (2013)
7. Han, X., Sun, L.: 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, vol. 1, pp. 945–954 (2011)
8. Han, X., Sun, L.: An entity-topic model for entity linking. In: EMNLP-CoNLL 2012, pp.105–115 (2012)
9. Heng, J., Joel, N., Ben, H.: Overview of tac-kbp2014 entity discovery and linking tasks. In: Proceedings of Text Analysis Conference (2014)
10. Lin, T., Mausam, E.O.: Entity linking at web scale. In: AKBC-WEKEX 2012, pp. 84–88 (2012)
11. Mendes, P.N., Daiber, J., Jakob, M., Bizer, C.: Evaluating dbpedia spotlight for the tac-kbp entity linking task. In: Proceedings of the TAC-KBP 2011 Workshop (2011)
12. Mihalcea, R., Csomai, A.: Wikify!: linking documents to encyclopedic knowledge. In: Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge Management, pp. 233–242 (2007)

13. Milne, D., Witten, I.H.: Learning to link with wikipedia. In: Proceedings of the 17th ACM Conference on Information and Knowledge Management, pp. 509–518 (2008)
14. Pantel, P., Fuxman, A.: Jigs and Lures: associating web queries with structured entities. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, vol. 1, pp. 83–92 (2011)
15. Pedersen, T., Purandare, A., Kulkarni, A.: Name discrimination by clustering similar contexts. In: Gelbukh, A. (ed.) CICLing 2005. LNCS, vol. 3406, pp. 226–237. Springer, Heidelberg (2005)
16. Pu, K.Q., Hassanzadeh, O., Drake, R., Miller, R.J.: Online annotation of text streams with structured entities. In: CIKM, pp. 29–38 (2010)
17. Ratinov, L., Roth, D., Downey, D., Anderson, M.: Local and global algorithms for disambiguation to wikipedia. In: HLT 2011, pp. 1375–1384 (2011)
18. Shen, W., Han, J., Wang, J.: A probabilistic model for linking named entities in web text with heterogeneous information networks. In: Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data, pp. 1199–1210 (2014)
19. Shen, W., Wang, J., Jiawei, H.: Entity linking with a knowledge base: Issues, techniques, and solutions. In: IEEE Transactions on Knowledge and Data Engineering, pp. 443–460 (2014)
20. Shen, W., Wang, J., Luo, P., Wang, M.: Linden: linking named entities with knowledge base via semantic knowledge. In: Proceedings of the 21st International Conference on World Wide Web, pp. 449–458 (2012)
21. Sil, A., Cronin, E., Nie, P., Yang, Y., Popescu, A.M., Yates, A.: Linking named entities to any database. In: EMNLP-CoNLL 2012, pp. 116–127 (2012)
22. Sil, A., Yates, A.: Re-ranking for joint named-entity recognition and linking. In: CIKM 2013, pp. 2369–2374 (2013)
23. Zhang, W., Sim, Y.C., Su, J., Tan, C.L.: Entity linking with effective acronym expansion, instance selection and topic modeling. In: IJCAI 2011, pp. 1909–1914 (2011)
24. Zhang, W., Su, J., Tan, C.L., Wang, W.T.: Entity linking leveraging: automatically generated annotation. In: COLING 2010, pp. 1290–1298 (2010)