information

MDPI

Article

# A Two-Stage Joint Model for Domain-Specific Entity Detection and Linking Leveraging an Unlabeled Corpus

**Hongzhi Zhang [1,2], Weili Zhang [1,2], Tinglei Huang [1,*], Xiao Liang [1] and Kun Fu [1]**

1 Institute of Electronics, Chinese Academy of Sciences, Beijing 100190, China; zhanghongzhi14@mails.ucas.ac.cn (H.Z.); zhangweili14@mails.ucas.ac.cn (W.Z.); xliang@mail.ie.ac.cn (X.L.); kunfuiecas@gmail.com (K.F.)
2 University of Chinese Academy of Sciences, Beijing 100049, China
* Correspondence: tlhuang@mail.ie.ac.cn; Tel.: +86-10-5888-7208

Academic Editor: Willy Susilo
Received: 6 March 2017; Accepted: 18 May 2017; Published: 22 May 2017

**Abstract:** The intensive construction of domain-specific knowledge bases (DSKB) has posed an urgent demand for researches about domain-specific entity detection and linking (DSEDL). Joint models are usually adopted in DSEDL tasks, but data imbalance and high computational complexity exist in these models. Besides, traditional feature representation methods are insufficient for domain-specific tasks, due to problems such as lack of labeled data, link sparseness in DSKBs, and so on. In this paper, a two-stage joint (TSJ) model is proposed to solve the data imbalance problem by discriminatively processing entity mentions with different degrees of ambiguity. In addition, three novel methods are put forward to generate effective features by incorporating an unlabeled corpus. One crucial feature involving entity detection is the mention type, extracted by a long short-term memory (LSTM) model trained on automatically annotated data. The other two types of features mainly involve entity linking, including the inner-document topical coherence, which is measured based on entity co-occurring relationships in the corpus, and the cross-document entity coherence evaluated using similar documents. An overall 74.26% F1 value is obtained on a dataset of real-world movie comments, demonstrating the effectiveness of the proposed approach and indicating its potentiality to be used in real-world domain-specific applications.

**Keywords:** entity detection; entity linking; domain-specific knowledge base; LSTM; topical coherence

## 1. Introduction

The prosperity of the web has greatly facilitated the dissemination of information. However, most of the web contents are still in the form of unstructured free texts, which are less machine-understandable than structured data, such as the data in knowledge bases (KB). To bridge this gap, one critical task is entity detection and linking (EDL), which aims to recognize entity mentions in free texts and determine their corresponding entities in the knowledge base. EDL is important for many applications, such as text disambiguation, information retrieval, question answering (QA) and information integration [1].

Many efforts on EDL [2–6] link entities to general KBs like Wikipedia and Freebase, while work about domain-specific EDL (DSEDL) is relatively less. However, the DSEDL task is also important and challenging. On the one hand, many domain-specific knowledge bases (DSKB) have been constructed in real-world projects, such as IMDB (http://www.imdb.com/), MusicBrainz (http://musicbrainz.org/) and Amazon (https://www.amazon.com/) products' KB, and related domain-specific tasks (e.g., customer service QA, in-context advertising) have put forward demands for effective DSEDL techniques. On the other hand, methods for the general EDL task may not be suitable for DSEDL, because they cannot handle problems like the increase of fake named entities [7] in texts and the sparsity of links in DSKBs [8]. In a word, DSEDL is a meaningful and challenging task that deserves further research.

Traditional methods for EDL conduct entity detection and entity linking separately in sequence, but the performance of entity detection tends to become the bottleneck in this pipeline architecture. As a result, in some recent works [7,9,10], entity detection and entity linking are jointly modeled to reinforce each other by leveraging their interdependency. A general idea of these joint models is that, for every n-gram in the input text, if it is the anchor phrase for at least one target entity, it will be selected as a candidate mention. This idea is designed to improve the recall rate of EDL, but it also generates a large number of fake mentions, resulting in much more negative candidate samples than positive ones in either the binary classification model [7] or the structure learning model [9]. Too many candidates means high computational complexity, while data imbalance also impairs the final performance.

Besides, some existing feature representation methods are inefficient for the DSEDL task:

(1) The textual context is not well utilized. In the pipeline architecture, textual context is used in entity detection models to determine mention boundaries and infer mention types. However, the labeled data are often insufficient to train such entity detection models in specific domains, so the bag of words representation, which neglects the syntax and word orders, is adopted in some joint models [7,9,10].
(2) The existing evaluation methods of entity coherence are not applicable for DSEDL. In general EDL tasks [11–14], semantic relatedness between entities is estimated relying on the massive hyperlinks in Wikipedia. However, this does not perform well in DSEDL because associations between entities are sparse in many DSKBs.
(3) Generally, only entity coherence within the input document is considered. Nevertheless, co-occurring entities can be very few in short input texts, and in this case, the inner-document coherence is not discriminative.

In summary, challenges still exist in DSEDL tasks, including the problems of unbalanced data and high computational complexity in joint models, as well as some drawbacks of the feature representation methods. Therefore, we propose a new joint model framework for DSEDL and explore more effective representation of features leveraging an unlabeled corpus. The main contributions of this paper include the following aspects:

(1) A two-stage joint (TSJ) model is proposed for the DSEDL task, shedding light on the problems of data imbalance and computational complexity.
(2) Several critical features are generated effectively by leveraging the unlabeled domain corpus. Specifically:
    (a) An LSTM-based [15] model is proposed to infer types of mentions, by exploiting the textual context. More importantly, a novel method is presented to automatically annotate training data for this model.
    (b) A corpus-based topical coherence measurement is given. To be specific, a pre-trained EDL model is used to tag the unlabeled data, then topical coherence is evaluated based on entity co-occurring relationships in the pseudo-labeled corpus.
    (c) The cross-document entity coherence is explored to solve the entity sparsity problem in short texts. Documents used here are selected from the pseudo-labeled corpus by a retrieval-based method.

The remainder of the paper is organized as follows. Section 2 reviews related work about EDL. Section 3 demonstrates the details of the proposed approach. Experimental results are presented and discussed in Section 4. Finally, we conclude the paper in Section 5.

## 2. Related Work

In this section, we briefly review some common methods for EDL, including the pipeline methods and the joint models, and some critical features used in these methods are listed and analyzed.

### 2.1. Pipeline Architecture for Entity Detection and Linking

In pipeline methods, entity detection and entity linking are processed sequentially.

Entity detection: Named entity detection (NER) models such as the hidden Markov model (HMM) [16], conditional random fields (CRF) [17], maximum entropy (ME) [18] and LSTM [19] are adopted in entity detection, and some mature NER tools are available, including Stanford NLP (http://nlp.stanford.edu/ner/), OpenNLP (http://opennlp.apache.org/) and LingPipe (http://alias-i.com/lingpipe/) [1]. Traditional research about NER mainly focuses on the recognition of persons, organizations, locations and numeric expressions [20], while efforts in specific domains are relatively few. However, a significant difference exists between NER and domain-specific entity detection, in that the former aims to recognize the entity names in common sense, while the latter detects domain entity mentions.

Entity linking: This is also known as entity disambiguation or entity normalization. Existing efforts for entity linking are based on the assumption that entity mentions are already recognized by the preceding entity detection module. Generally, crucial procedures of entity linking include candidate entity generation, candidate ranking and unlinkable mention prediction [1]. Candidate entities are usually generated using a name dictionary that is built offline [21], while candidate ranking mainly exploits supervised learning methods, such as binary classification [6,22,23], learning to rank [24–26], structure learning, graph-based methods [11,27–29] and probabilistic methods [3,30]. Among these methods, binary classification is a simple and natural choice, but it suffers from the data imbalance problem.

The bottleneck of the pipeline method lies in the performance of NER, and this problem is especially evident in specific domains due to lacking of labeled data. Besides, traditional NER models trained on formal texts often perform poorly on noisy texts such as tweets [9] or casually-written comments.

### 2.2. Joint Models for Entity Detection and Linking

Joint models are proposed to improve the overall performance by leveraging the mutual dependency between both tasks. Guo et al. [9] simply take every n-gram of input tweets as a candidate mention and then perform entity linking, and if the candidate mention cannot be linked to any entity by the model, then the mention is rejected. A re-ranking-based model suitable for longer documents is introduced in [10], in which joint EDL is preceded by a pipeline EDL model to generate high-quality candidate mentions and entity links. Zhang et al. [7] propose a joint model for DSEDL, where features of entity linking and NER models are updated iteratively until convergence, and an iterative graph-based algorithm is further introduced in [31] to capture entity-entity and entity-mention dependency.

### 2.3. Feature Representation

To ensure effective operation of EDL models, various features are explored and extracted mainly from the following aspects: Mention, candidate entity, textual similarity between the mention context and entity description text, consistency between the type of mentions and candidate entities, as well as topical coherence between mapping entities. Extraction methods for the first three types of features are similar in different papers, while methods for the other two kinds vary greatly, as is explained below.

In pipeline methods, entity types are provided by the entity detection model, while in joint models, these features are implicit, as there is no independent entity detection module. Li et al. [32] mines additional contextual words for each concerned entity type, but the context is limited to words before and after the mention. A CRF model is utilized to infer the entity type in [33]; but training data is insufficient for the complex model, and feature engineering is inevitable.

In EDL tasks for general KBs, topical coherence between mapping entities is measured based on the hyperlinks in articles of encyclopedias. However, links in some DSKBs are very sparse. The works in [7,33] incorporate the link structure of encyclopedias by mapping entities in the DSKB to encyclopedias. Li et al. [8] mines word-level disambiguation evidence from the entity description text in the KB. These efforts focus on topical coherence between entities within the input text, while coherence across documents is explored in [14,26,34], which perform entity linking in short texts like tweets.

## 3. Method

In this section, the formal definition of the DSEDL task is first given. Then, we illustrate the framework and details of the TSJ model in Section 3.1. Finally, all critical features used in the model are fully discussed in Section 3.2.

Task definition: Suppose $K_D$ is a domain-specific knowledge base and $d$ is an unstructured document. Then, the objective of DSEDL is to recognize all of the textual entity mentions $M$ from $d$ and determine the corresponding entity set $E_L = \{e_i \,|e_i \in K_D \bigwedge m \to e_i\}$ for each mention $m \in M$, where $m \to e_i$ means $e_i$ is a mapping entity of $m$ and $e_i \in K_D$ means that $e_i$ is an entity recorded in $K_D$.

Note that a mention $m$ can be linked to more than one entity. For example, the mention *Coen Brothers* should be linked to *Joel Coen* and *Ethan Coen*. Mentions whose target entity sets are empty will be regarded as fake mentions on the assumption that interested entities are all included in $K_D$.

### *3.1. Two Stage Joint Model*

#### 3.1.1. Framework

The data imbalance and high computational complexity in traditional joint EDL models are mainly caused by the mentions with high ambiguity. According to observation, highly ambiguous mentions often appear in two scenarios: (1) the ambiguous mentions of a certain entity are preceded by unambiguous ones; (2) the ambiguous mentions are used when their target entity appears frequently in the current topic.

The main idea of the proposed TSJ model is inspired by the above observation. The first stage aims to process the less ambiguous mentions in the input text. Then, the second stage mainly focuses on finding and linking the highly ambiguous mentions, by restricting candidate entities to those that are either mentioned unambiguously in the input text or frequently referred to under the topic (i.e., topic of the input text). Specifically, these candidate entities are collected by merging both the target entities recognized in Stage 1 and frequent entities in the topic-similar documents. This solution greatly reduces the number of false candidates, so TSJ can tackle the aforementioned two problems to some extent.

Figure 1 shows the overall framework of TSJ. In general, TSJ is the cascade of two similar, but differentiated modules (i.e., stages). Each module consists of three main parts, including the name dictionary, the candidate generation block and the binary classification block. The roles of these three parts are listed as follows.

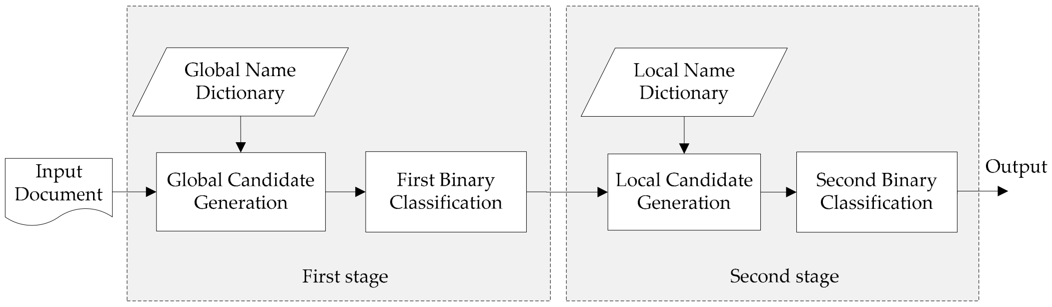
Global Name Dictionary
Input Document
Global Candidate Generation
First Binary Classification
First stage
Local Name Dictionary
Local Candidate Generation
Second Binary Classification
Output
Second stage

**Figure 1.** Overall framework of the two-stage joint (TSJ) model.

Name dictionary: This is a mapping dictionary, of which the keys are mentions, and the values are the candidate entity sets of these mentions. Formally, given a mention $m$ and its candidate entity set $E_C = \{e_1, e_2, \ldots, e_{N_C}\}$, where $N_C$ is the number of its possible target entities, then we can get $\mathcal{D}[m] = E_C$ from the name dictionary $\mathcal{D}$ if $m$ is recorded in $\mathcal{D}$. The name dictionary is used to guide the candidate generation step, so it directly affects the performance of TSJ. Specifically, a global name dictionary built offline is used in the first stage, while the local name dictionary that works in the second stage is built online for every input document. Details about the global and local name dictionaries will be presented in Section 3.1.2.

Candidate generation: Given an input document and all of its n-grams, this step aims to select valid <n-gram, entity> candidate pairs and filter out impossible or highly ambiguous ones as much as possible. The method is described in Section 3.1.3.

Binary classification: The binary classifier serves to decide whether the input candidate pair <n-gram, entity> is positive or negative based on features extracted for this pair. The output of the first classifier is utilized in the construction of the local name dictionary, while the output of the second one presents the final results. Details are illustrated in Section 3.1.4.

Suppose the input document is: *I watched I am legend the film majored by Will Smith my favorite actor. Although the film is somewhat disappointing, it wins in the charm of WS … LOVE Smith.* The whole procedure of the TSJ model is illustrated in Figure 2. Note that *WS* and *Smith* are excluded from the global name dictionary because they have too many candidate entities and are quite ambiguous. However, they are added into the local name dictionary, with only one candidate entity that is confirmed by the first classifier. Finally, all four possible mentions are linked correctly to their target entities in the DSKB.

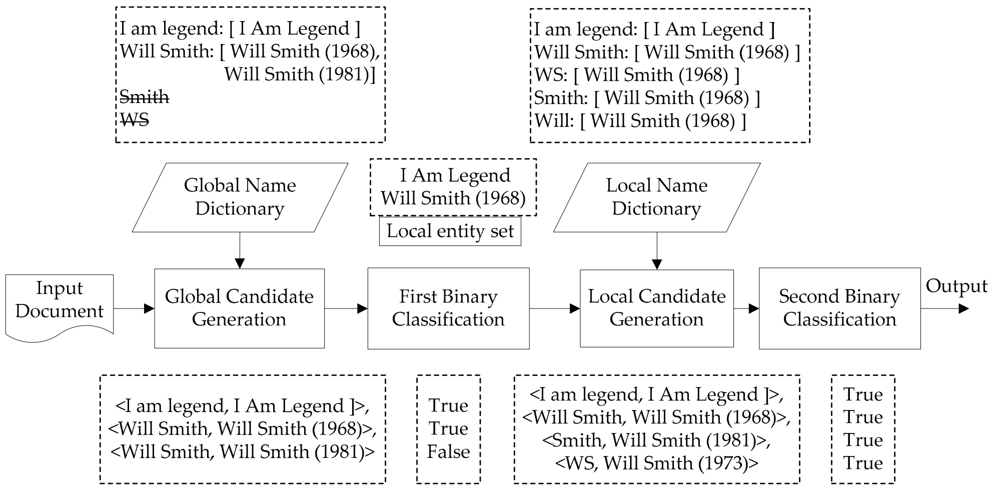
I am legend: [ I Am Legend ]
Will Smith: [ Will Smith (1968), Will Smith (1981)]
~~Smith~~
~~WS~~
I am legend: [ I Am Legend ]
Will Smith: [ Will Smith (1968) ]
WS: [ Will Smith (1968) ]
Smith: [ Will Smith (1968) ]
Will: [ Will Smith (1968) ]
Global Name Dictionary
I Am Legend
Will Smith (1968)
Local entity set
Local Name Dictionary
Input Document
Global Candidate Generation
First Binary Classification
Local Candidate Generation
Second Binary Classification
Output
<I am legend, I Am Legend ]>, <Will Smith, Will Smith (1968)>, <Will Smith, Will Smith (1981)>
True True False
<I am legend, I Am Legend ]>, <Will Smith, Will Smith (1968)>, <Smith, Will Smith (1981)>, <WS, Will Smith (1973)>
True True True True

**Figure 2.** The illustration of the processing procedure of TSJ.

It can be seen that the combination of the two stages introduces more flexibility in performing the EDL task. On the one hand, the separation of the two name dictionaries yields a more effective generation of candidates. On the other hand, the cascaded structure of TSJ allows the second stage to run on the results of the first stage, thus boosting the overall performance.

#### 3.1.2. Name Dictionary Construction

A name dictionary records the mappings between name strings (mentions) and their candidate entities. Since name dictionaries are used to determine the <n-gram, entity> pairs in the candidate generation step, so they directly influence the model's computational complexity and the balance of data. Construction methods of the global and local name dictionaries are described in the following.

The global name dictionary is built offline, using names and the alias of entities recorded in the DSKB. Some transformations are made to the names of entities to cover the common name variants. However, in order to exclude the highly ambiguous mentions, only simple transformations are conducted, and meanwhile, variants that have too many candidate entities are not added to the dictionary. For example, partial names of *Leonardo DiCaprio*, i.e. *Leonardo* and *DiCaprio*, are recorded in the dictionary as variants, but for the mention *Will Smith*, the variant *Smith* is not considered because it is too frequently used to be easily disambiguated, and neither is the abbreviation *WS*. The respectively simple transformation of entity names, as well as the removal of ambiguous names, can prevent the generation of massive negative candidates in the global candidate generation procedure.

The local name dictionary, however, is built online for every input document based on the local entity set, the entities in which are selected in the following two ways:

(1) Entities that are linked to in the first stage. That is, if an <n-gram, entity> candidate pair is labeled as positive by the first binary classifier, then this entity is added into the local entity set. The entities *Will Smith* and *I am legend* in Figure 2 are examples.
(2) Entities that frequently appear in the extension document set. Here, the extension document set is a collection of documents that are selected from the pseudo-labeled corpus and have similar topics as the input text (see Section 3.2.4 for details). For an entity $e$, if its document frequency exceeds the threshold $\theta_F$, then it is included in the local entity set.

After determining the local entity set, corresponding mentions are generated by performing string transformations for the original surface name of each entity. For example, *Smith* and *WS* are abbreviations of *Will Smith*, so entries {*Smith*: *Will Smith*} and {*WS*: *Will Smith*} are included in the local name dictionary.

In summary, by covering the highly ambiguous mentions in the local name dictionary, the size of fake mentions and negative candidate pairs can be reduced greatly without decreasing the recall, thus shedding light on the settlement of data imbalance and high complexity problems.

#### 3.1.3. Candidate Generation

The objective of candidate generation is to find out possible mentions from n-grams of the input text and then output the candidate <mention, entity> pairs. Specifically, the input text is traversed to match the mentions recorded in the name dictionary, and the detailed algorithm is presented below.

Formally, given an input text $d = w_1 w_2 \cdots w_{N_d}$ with $N_d$ words $w_i (1 \le i \le N_d)$ (or $N_d$ characters for Chinese) and a name dictionary $\mathcal{D}$, let $s = w_i \cdots w_j,\ 1 \le i \le j \le N_d$ denote a substring of the input text. If $s$ satisfies the following conditions, then it is selected as a possible mention.

$$s \in \mathcal{D}$$

$$w_k \cdots w_l \notin \mathcal{D},\ \forall\, k,\ l,\ 1 \le k < i < l \le |d|\ \vee k = i \le j < l \le |d|$$

where $s \in \mathcal{D}$ means that this substring is one of the keys in $\mathcal{D}$. The second formula ensures there is no overlapping boundaries for any two mentions.

Since the input text needs to be traversed to find all mentions, a prefix dictionary $\mathcal{D}_p$ is built to reduce the search space. For every mention $w_1 \cdots w_{N_m}$ in the name dictionary, its prefixes $w_1,\ w_1 w_2,\ \cdots,\ w_1 \cdots w_{N_m}$ are added to $\mathcal{D}_p$. The pseudo code of the candidate generation step is given in Algorithm 1.

**Algorithm 1.** Candidate generation algorithm.

```
Input: text $d = w_1 w_2 \cdots w_{N_d}$, name dictionary $\mathcal{D}$
Output: set of candidate pairs $C_p$

Initialize $\mathcal{D}_p, C_p = \{\}, i = 0$;
while $i < N_d$:
    $temp = NULL,\ j = i + 1$:
    while $j < N_d$:
        $s = w_i, \ldots, w_j$;
        if $s \in \mathcal{D}_p$:
            if $s \in \mathcal{D}$:
                $temp = \{\langle s, e \rangle$ for each $e \in \mathcal{D}[s]\}$;
        else:
            if $temp \neq NULL$:
                $C_p = C_p \cup temp$;
                $i = j$;
            else:
                $i = i + 1$;
                break;
    end
end
return $C_p$;
```

### 3.1.4. Binary Classification

The binary classifier takes in all candidate pairs generated by the previous step and determines whether each pair is positive or negative. For the first binary classifier, if a pair is tagged positive, then the entity in this pair will be added to the local entity set in Stage 2. While in the second binary classification, a positive pair means that a link can be established between the corresponding mention and entity, and this entity is added into the target entity set of this mention. A simple example about the binary classification is given in Figure 3.

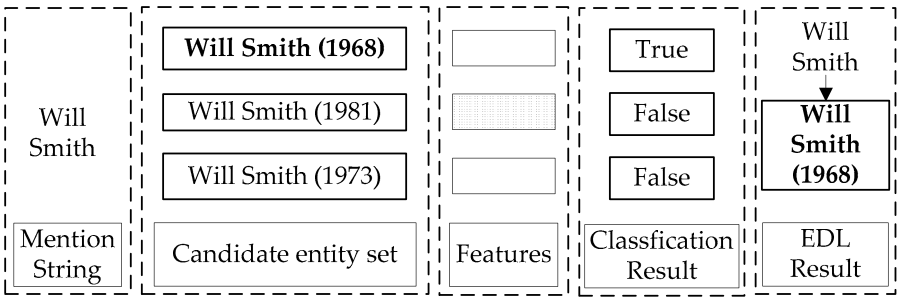

**Figure 3.** A demo of the binary classification process. EDL, entity detection and linking.

Besides the model structure and the quality of input data, the features are also crucial for a classifier. Due to the importance, the details of feature generation and representation are specially discussed in the next section.

### *3.2. Feature Representation Leveraging an Unlabeled Corpora*

Features used in the binary classifiers of TSJ are presented in this section. A brief overview of all features is given in Section 3.2.1, and detailed explanations about three kinds of crucial features are given in the remaining sub-sections.

#### 3.2.1. Features' Overview

Features used in TSJ can be divided into two types: Context independent features and context-dependent ones [1]. Context independent features reflect the static characteristics of mentions and entities, while context-dependent features are extracted from the context of mentions. Table 1 gives an overview of all of the features. Among these features, we specially study the representation methods for three groups of context independent features, including the mention type, corpus-based inner-document entity coherence and topical coherence across documents, and the details are introduced in the following three sub-sections. The rest of the features compose the basic feature set; the corresponding details are explained in the Appendix due to the limit of space.

**Table 1.** An overview of all of the features used in the TSJ model. DSKB, domain-specific knowledge base.

| Features | | Descriptions |
|---|---|---|
| Context independent | Mention string | Textual and statistical characteristics of the mention string |
| | Entity | Entity popularity and entity type |
| Context-dependent | Textual context similarity | Textual similarity between context of mentions and descriptions of entities |
| | Mention type (Section 3.2.2) | The probability of the mention being a certain type of entity |
| | DSKB-based entity coherence | Topical coherence between entities within the input document based on the link structures of the DSKB |
| | Corpus-based inner-document entity coherence (Section 3.2.3) | Topical coherence between entities within the input document based on the pseudo-labeled corpus |
| | Topical coherence across documents (Section 3.2.4) | Entity coherence across similar documents |
| | Other | Length of the input document |

#### 3.2.2. Mention Type

An LSTM-based model is proposed to infer the type information, namely the probability of a mention belonging to a certain type. Besides, an automatic data labeling method is introduced to generate training data for the model from unlabeled corpora.

##### The LSTM-Based Mention Type Classification Model

The model aims to infer the type of a mention from its textual context. To be specific, given a mention and the sentence it appears in, the model generates the probability of the mention belonging to each type $t \in T$, where $T$ is the pre-defined entity type set. Note that for consistency, the *fake mention type* is also included, because a mention may not belong to any of the meaningful pre-defined types, and in this case, this mention is likely to be a fake mention.

The structure of the model is shown in Figure 4. The model consists of three layers: Word representation layer, sentence representation layer and output layer. The word representation layer transforms words in the sentence into the corresponding vectors. In the sentence representation layer, the word vectors are feeding to LSTM sequentially to get vector representation of the sentence. Finally, a soft-max layer (the output layer) is used to generate the classification result. The dimension of the output is $N+1$, where $N$ denotes the number of meaningful pre-defined types, and the additional dimension is corresponding to the fake mention type.

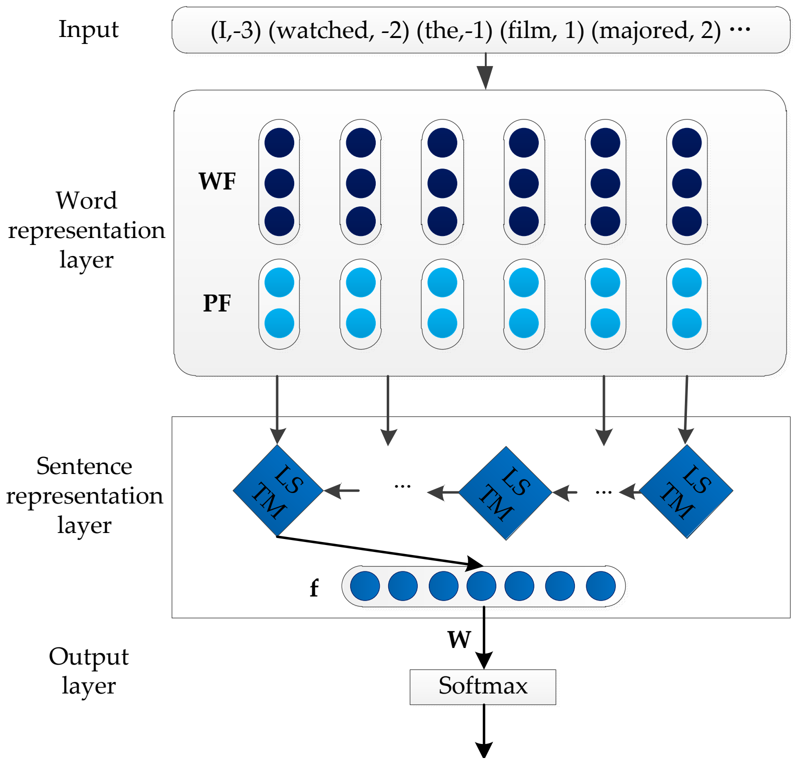
Input
(I,-3) (watched, -2) (the,-1) (film, 1) (majored, 2) ···
Word representation layer
WF
PF
Sentence representation layer
LS TM
···
LS TM
···
LS TM
f
Output layer
W
Softmax

**Figure 4.** The network structure of the LSTM model.

(1) Word representation layer:

For a given mention and its sentence, this layer learns the vector representation for each word in the sentence. The word representation consists of two components: The word itself and its relative position to the mention; both are represented with the corresponding vectors. Suppose $\mathbf{w_i} \in \mathbb{R}^{n_{e1}\times 1}$ is the word embedding of the $i$-th word in the sentence and $\mathbf{p_i} \in \mathbb{R}^{n_{e2}\times 1}$ is the position embedding, then the word feature and position feature of the sentence are $\mathbf{WF} = [\mathbf{w}_0, \mathbf{w}_1, \ldots, \mathbf{w_{n}}_{-1}]$ and $\mathbf{PF} = [\mathbf{p}_0, \mathbf{p}, \ldots, \mathbf{p_n}]$, where $n_{e1}$ and $n_{e2}$ are hyper parameters denoting the dimensions of the embedding, and $n$ is the length of the sentence. Combining **WF** and **PF**, the matrix $\left[\mathbf{WF}^T, \mathbf{PF}^T\right]^T \in \mathbb{R}^{(n_{e1}+n_{e2})\times n}$ represents all of the words in the sentence. Word embeddings are initialized with weights pre-trained on the unlabeled corpus, while position embeddings are randomly initialized.

Note that the given mention is removed from the sequence, in order to force the model to capture the information of textual context, rather than remember the limited mentions that are used to generate training data (see Section Automatic Generation of Training Data for details).

(2) Sentence representation layer:

By representing the word in a low-dimensional vector space, the word representation layer can capture the similarity between words, so the model is more robust to the variability of linguistic expression at the lexical level compared with models based on the bag of words representation. For example, the model can capture the similarity between words 'watch' and 'see' by representing them with adjacent low-dimensional vectors.

LSTM is a recurrent neural network (RNN) model [35] with a long-short term memory unit, which can handle the gradient explosion and vanishing problems of the traditional RNN model.

The LSTM model is adopted in a variety of tasks, such as the language model, the neural translation model and speech recognition and outperforms traditional methods that rely on feature engineering. Since LSTM has the potential to capture the long-term dependence in sequence, it is employed to extract context features that are crucial for mention type identification from the input word sequence. Feature representation is automatically extracted without feature engineering, so the proposed method can be feasibly adopted in different domains.

(3) Output layer:

Vectors from the matrix $\left[\mathbf{WF}^T, \mathbf{PF}^T\right]^T$ are fed to LSTM sequentially. The LSTM layer then outputs the feature vector $\mathbf{f} \in \mathbb{R}^{n_s \times 1}$, where $n_s$ is the dimension of the sentence feature. The sentence-level feature vector $\mathbf{f}$ is first processed by a linear transformation:

$$\mathbf{o} = \mathbf{Wf} \tag{1}$$

where $\mathbf{W} \in \mathbb{R}^{(N+1)\times n_s}$ is the transformation matrix and $\mathbf{o} \in \mathbb{R}^{(N+1)\times 1}$ whose $i$-th component $o_i$ denotes the score of the $i$-th pre-defined mention type (the last dimension is the score of the fake mention type). Finally, a soft-max operation is applied, to convert the scores into the probabilistic form:

$$p_i = \frac{e^{o_i}}{\sum_{k=1}^{N+1} e^{o_k}} \tag{2}$$

Additionally, these probabilities will be fed to the binary classifiers of the TSJ model as the mention type features.

#### Automatic Generation of Training Data

Generally, some unambiguous mentions can be found in the specific domain. One can distinguish the types of these mentions with certainty. For example, *The Twilight Saga: Breaking Daw—Part 1* can be linked to a movie entity, and *Barack Hussein Obama* is no doubt a person. Therefore, each time these mentions appear in a sentence, this sentence can be tagged as a positive example of the corresponding type. By contrast, if a word is randomly chosen from the text, there is little chance that this word is an entity mention, so the sentence it appears in is labeled as negative.

Based on the above idea, the training data can be generated automatically as follows. For every entity type, a set of unambiguous mentions is selected, by rules (e.g., very long names) or by experience (e.g., the formal names of well-known entities in the current domain). Then, the unlabeled domain-specific corpus is traversed to find sentences containing these mentions as positive samples, while the negative samples are generated by sampling all sentences of the corpus and avoiding the possible positions of unambiguous mentions.

Using this method, one only needs to collect a small number of unambiguous mentions for each entity type, and then, massive weakly supervised training data can be generated for the representation learning-based model, which relies heavily on labeled data, so the data annotation is quite efficient.

### 3.2.3. Corpus-Based Inner-Document Entity Coherence

Generally, entities co-occurring in one document are about the same or related topics. This topical coherence is widely considered in many existing entity linking systems [11–14]. In the general EDL tasks, topical coherence is usually estimated based on the hyperlinks between entity pages in Wikipedia, using measures like the Wikipedia link-based measure (WLM) [36,37], Jaccard distance [9] and point-wise mutual information (PMI-like) [3]; while in DSEDL tasks, relationships between entities in the DSKB are utilized, by assuming that two entities directly linking to each other tend to share similar topics. This DSKB-based coherence is also adopted in our method, and details are given in the Appendix. However, this measurement suffers from the relationship sparsity of some DSKBs, so the novel corpus-based metrics are proposed in this sub-section to solve this problem.

The proposed metrics measure the topical similarity based on the distribution of entities in the corpus. Here, the corpus is composed of massive unlabeled documents accumulated in domain-specific applications, such as product comments in Amazon or movie reviews in IMDB. A preprocessing step is to label all documents in the corpus with a TSJ model pre-trained on the training dataset, using only the aforementioned features. After this step, we get a *pseudo-labeled corpus*. Though the model is imperfect, its annotation result can still reflect the distribution of topic-coherent entities, so semantic similarity can be measured on the pseudo-labeled corpus. Specifically, given two entities $e_i$ and $e_j$, their corpus-based topical coherence can be computed by:

$$Coh_{ee}(e_i,\ e_j) = \begin{cases} \frac{|D_i \cap D_j|}{|D_i \cup D_j|} & e_i \neq e_j \\ 0 & e_i = e_j \end{cases} \tag{3}$$

where $D_i$ and $D_j$ are the sets of documents in the pseudo-labeled corpus that contain $e_i$ and $e_j$, respectively. If regarding every word as a latent entity, the coherence between an entity and a contextual word is defined as:

$$Coh_{ew}(e_i, w_j) = \frac{\left|D_i \cap D'_j\right|}{\left|D_i \cup D'_j\right|} \tag{4}$$

where $D'_j$ is the set of documents containing the word $w_j$.

Note that the pseudo-labeled documents can be accumulated by online DSEDL systems in real-world applications, and the above values can be updated incrementally in real time.

Based on the above two metrics, three features are incorporated to measure the confidence of a mention $m$ being linked to its candidate entity $e_i$. Suppose $M$ is the set of mentions in the input document and $W$ is the set of words, then the definitions of these features are as follows.

(1) Coherence between the candidate entity and all of the contextual words:

$$Coh_1(m, e_i) = \sum_{w_j \in W} Coh_{ew}(e_i,\ w_j) \tag{5}$$

(2) Coherence between candidate entities:

$$Coh_2(m,\ e_i) = \sum_{m_k \in M} \sum_{e_j \in E_{Ck}} Coh_{ee}(e_i,\ e_j) C_b(m_k,\ e_j) \tag{6}$$

Here, $E_{Ck}$ is the candidate entity set of $m_k$, and $C_b(m_k,\ e_j)$ is the classification result of a binary classifier using features that are introduced in the above sections. That is, $C_b(m_k,\ e_j) = 1$ if $m_k$ is predicted to be linked to $e_j$ by the classifier; otherwise, it is zero.

(3) Distance-weighted coherence between entities:

$$Coh_3(m, e_i) = \sum_{m_k \in M} \sum_{e_j \in E_{Ck}} Coh_{ee}(e_i,\ e_j) C_b(m_k,\ e_j) \frac{1}{1 + d_k} \tag{7}$$

where $d_k$ is the distance between $m$ and $m_k$ in the text. The distance factor is introduced based on the idea that topical relatedness between entities usually decreases with the increase of their distance.

### 3.2.4. Topical Coherence between Entities across Documents

The previous sub-section proposes metrics for topical coherence between entities based on the pseudo-labeled corpora, so as to solve the link sparsity problem in DSKB. Those features focus on semantic cohesiveness between the candidate entity and contextual components within the input text, so they are less discriminative for the documents with few entities (i.e., the short texts). Thus, the topical coherence across documents is further explored.

Preprocessing is needed, as well. A new pseudo-labeled corpus is generated, using a pre-trained TSJ model based on features introduced in the above sections. Then, for each input document $d$, all documents that have similar topics with $d$ are retrieved from this corpus using the vector space model [38], and these documents are named *extension documents* of $d$. Based on the collection of extension documents, topical coherence across documents can be measured from the following three aspects.

(1) Mention-entity linking rate:

This feature is extracted based on the observation that the same mentions in topic-similar documents tend to refer to the same entity. Given a mention $m$ and its candidate entity $e_i$, the mention linking rate evaluates the rate that $m$ is linked to $e_i$ in the extension documents:

$$MLR(m, e_i) = \frac{\Gamma(m, e_i)}{\sum_{e_j \in E_D} \Gamma(m, e_j)} \tag{8}$$

where $\Gamma(m, e_i)$ is the number of $m$ being linked to $e_i$ in the extension documents and $E_D$ denotes the collection of entities that are recognized in the extension documents.

(2) Entity linked frequency:

Heuristically, if a candidate entity is frequently mentioned in the extension documents, there is high prior probability that this entity is referred to in the input document. The frequency of $e_i$ being linked to in the extension documents is used as the prior confidence:

$$ELF(e_i) = \sum_{m_d \in M_D} \Gamma(m_d, e_i) \tag{9}$$

Here, $M_D$ is the set of mentions appearing in the extension documents.

(3) Coherence between entities across documents:

Similarly, entities in extension documents also have semantic relatedness with entities in the input text. According to this idea, a cross-document coherence measurement is given:

$$Coh_4(m, e_i) = \sum_{e_j \in E_D} rel_1(e_i, e_j) \tag{10}$$

where $rel_1$ is the entity relatedness based on relationships in the DSKB,

$$rel_1(e_i, e_j) = \begin{cases} 1 & if\ i \neq j\ and\ there\ are\ reltionships\ between\ e_i\ and\ e_j\ in\ DSKB \\ 0 & otherwise \end{cases} \tag{11}$$

Furthermore, note that the pseudo-labeled documents can be accumulated in real-world applications, so our method has the potential to capture the time-varying topical transformation by giving priority to the latest documents in the document retrieval step.

## 4. Experiments

In this section, a series of experiments are conducted to validate the effectiveness of the proposed TSJ model and the features extracted utilizing the unlabeled corpus.

### *4.1. Dataset Preparation*

A benchmark dataset is needed to verify the performance of our method. Besides, a domain-specific unlabeled corpus is crucial throughout the experiments. Details about these two datasets are demonstrated below.

(1) The benchmark dataset:

The dataset provided in the DSEDL evaluation task at China Conference on Knowledge Graph and Semantic Computing (CCKS2016) is used as the benchmark. A movie knowledge base (MKB) and a collection of labeled documents in the movie domain are provided. Entities recorded in MKB mainly belong to two categories: artists such as actors, directors, screenwriters and hosts, as well as films and television works, such as movies, television series, TV shows, etc. The labeled documents' collection includes unstructured texts like movie reviews, short reviews, comments of movie reviews, group topics and synthetic reviews obtained from Douban (https://www.douban.com/), a website focusing on publishing movie-related information. Statistics of this document collection are given in Table 2. As can be seen from the table, movie reviews and synthetic reviews are generally long texts while the other three kinds of documents are relatively short.

**Table 2.** Statistics about the labeled documents in the benchmark dataset.

| **Document Category** | **$\lvert$Documents$\rvert$** | **Train/Test** | **$\overline{\text{Text length}}$** | **$\lvert$Mentions$\rvert$** | **$\overline{\lvert\text{Mentions}\rvert}$** |
|---|---|---|---|---|---|
| Movie review | 225 | 150/75 | 2781 | 4085 | 18.16 |
| Short review | 480 | 320/160 | 72 | 814 | 1.70 |
| Comments of movie review | 299 | 200/99 | 129 | 478 | 1.60 |
| Group topic | 298 | 199/99 | 429 | 1801 | 6.04 |
| Synthetic review | 7 | 5/2 | 5460 | 713 | 101.86 |
| Total | 1309 | 874/435 | 661 | 7891 | 6.03 |

(2) The unlabeled corpus:

As Douban contains rich textual data about millions of movies, it is capable of providing high quality corpora for the EDL task in the movie field. About 1.09 million movie reviews are crawled from Douban during October 2016, with the help of Amazon Web Service (AWS), composing the raw unlabeled corpus, which is used intensively to produce effective features.

### 4.2. Evaluation Metrics

Measurements used in the CCKS2016 EDL evaluation task are adopted to assess the performance of our method from all of the following three aspects.

(1) Entity detection evaluation:

The precision, recall and F1-measure are used as the assessment measures for entity detection, defined as:

$$P_{ED} = \frac{|\{correctly\ recognized\ entity\ mentions\}|}{|\{recognized\ entity\ mentions\ by\ the\ method\}|}$$

$$R_{ED} = \frac{|\{correctly\ recognized\ entity\ mentions\}|}{|\{all\ entity\ mentions\ tagged\ in\ the\ labeled\ document\ collection\}|}$$

$$F1_{ED} = \frac{2 \cdot P_{ED} \cdot R_{ED}}{P_{ED} + R_{ED}}$$

(2) Entity linking accuracy:

The result of entity linking is evaluated based on mentions that are correctly detected, so we have:

$$A_{EL} = P_{EL} = R_{EL} = F1_{EL} = \frac{|\{correctly\ linked\ entity\ mentions\}|}{|\{correctly\ recognized\ entity\ mentions\}|}$$

where $A_{EL}$ is the accuracy of entity linking.

(3) Overall evaluation:

Similarly, the precision, recall and F1 value are used, denoted by $P_{EDL}$, $R_{EDL}$ and $F1_{EDL}$, respectively. Additionally, we have $P_{EDL} = P_{ED} \cdot A_{EL}$ and $R_{EDL} = R_{ED} \cdot A_{EL}$.

## *4.3. Experimental Setup*

Some tools used in the experiments, as well as the values of hyper parameters and some details in the models are introduced in this sub-section.

(1) The threshold in the local name dictionary:

The threshold $\theta_F$ used in the local name dictionary construction is set by conducting experiments on a randomly-selected validation set (10%) from the training set. The best performance of the model is obtained when $\theta_F = 5$, so this is chosen as the best threshold.

(2) The binary classification model of TSJ:

As for the binary classification block, the GBDT (gradient boosting decision tree) model implemented in eXtreme Gradient Boosting package (XGBoost) (https://github.com/dmlc/xgboost) is exploited as the classifier. The reason to choose this model is that, GBDT outperforms many other classifiers in its efficiency and classification accuracy, and meanwhile, it is less likely to over-fit; and features can be sent to the model without much pre-processing.

(3) The LSTM-based model:

The LSTM-based model proposed in Section The LSTM-Based Mention Type Classification Model is implemented utilizing Keras (https://github.com/fchollet/keras) with Tensorflow (https://www.tensorflow.org/) as the backend.

The hyper parameters in this model are set according to the performance on validation data (10%), and their values are listed in Table 3. Since it is sufficient to consider two types of entities, the artist and the movie, so we have two meaningful pre-defined types, and $N = 2$ is used in the output layer. Besides, the initial weights of word embeddings are trained on the climbed Douban movie reviews (i.e., the unlabeled corpus), using the word2vec toolkit provided by Gensim (http://radimrehurek.com/gensim/), a python package for natural language processing, while weights of the position embeddings are initialized randomly.

**Table 3.** Hyper parameters used in the experiment.

| Hyper Parameter | Word Embedding Dim. | Position Embedding Dim. | Sentence Feature Dim. | Meaningful Pre-defined Types |
|---|---|---|---|---|
| Value | $n_{e1} = 50$ | $n_{e2} = 10$ | $n_s = 60$ | $N = 2$ |

Note: "dim." means dimension.

The training data for this model are obtained using the approach introduced in Section Automatic Generation of Training Data, We select 716 and 665 unambiguous mentions from artists and film and TV programs, respectively, and the statistics of the generated training samples are given in Table 4. Using a small number of unambiguous mentions and the unlabeled corpus, massive labeled data can be automatically generated, so the approach is quite applicable.

**Table 4.** Statistics about the automatically-generated training data for mention type recognition.

| Positive Samples for Movie Type | Positive Samples for Artist Type | Negative Samples |
|---|---|---|
| 1,839,893 | 1,359,595 | 5,172,380 |

### *4.4. Experimental Results*

We illustrate the effectiveness of our TSJ model and analyze the effects of the three kinds of features. A comparison with other methods is also presented to prove the advantages of our approach.

#### 4.4.1. Performance of the Proposed Method

Five groups of experiments with different settings are carried out to validate the effectiveness of the proposed TSJ model and features extracted leveraging the corpus. Details of the experimental settings are explained as follows.

- Baseline (Base 1): only one joint model based on the binary classification, using the basic feature set. Partial names and abbreviation names are included in the name dictionary to cover the ambiguous mentions.
- The first stage of the TSJ model (FTSJ): The difference between Base 1 and FTSJ lies in that abbreviated names and ambiguous partial names that have more than 15 candidate entities are excluded from the name dictionary.
- The basic TSJ model (TSJ): the proposed two-stage architecture utilizing the basic feature set.
- + Mention type (+ MT): using the TSJ model and adding mention type features generated by the LSTM-based model to the feature set.
- + Extra inner-document topical coherence (+ EIC): on the basis of +MT, adding the corpus-based topical coherence features, namely $Coh_1(m, e_i)$, $Coh_2(m, e_i)$ and $Coh_3(m, e_i)$ introduced in Section 3.2.3.
- + Cross-document coherence (+ CC): on the basis of +EIC, incorporating topical coherence across extension documents, namely $MLR(m, e_i)$, $ELF(e_i)$ and $Coh_4(m, e_i)$, introduced in Section 3.2.4.

The experimental results of different configurations are listed in Table 5, and more illustrations and discussions about the effect of each elements are given subsequently.

**Table 5.** Performance comparison of different settings. FTSJ, first TSJ; MT, mention type; EIC, extra inner-document topical coherence; CC, cross-document coherence.

| Configuration | Entity Detection | | | Entity Linking | Overall EDL | | |
|---|---|---|---|---|---|---|---|
| | $P_{ED}$ | $R_{ED}$ | $F1_{ED}$ | $A_{EL}$ | $P_{EDL}$ | $R_{EDL}$ | $F1_{EDL}$ |
| Base | 77.01 | 64.94 | 70.46 | 88.67 | 68.29 | 57.58 | 62.48 |
| FTSJ | 80.57 | 65.29 | 72.13 | 89.96 | 72.48 | 58.64 | 64.89 |
| TSJ | 83.64 | 68.74 | 75.46 | 91.03 | 76.13 | 62.57 | 68.69 |
| + MT | 85.64 | 72.87 | 78.74 | 89.48 | 76.63 | 65.21 | 70.46 |
| + EIC | 85.74 | 72.80 | 78.74 | 90.63 | 77.71 | 65.98 | 71.36 |
| + CC | 87.05 | 73.37 | 79.63 | 93.26 | 81.18 | 68.43 | 74.26 |

Note: The percent signs (%) behind the numbers are omitted.

(1) The TSJ model:

Compared with the baseline method, the proposed two-stage joint model reports a 5%, 2.46% and 6.41% increase on $F1_{ED}$, $A_{EL}$ and $F1_{EDL}$, respectively. Statistics of the candidate pairs generated at the two stages are listed in Table 6. Massive negative samples are generated to cover the ambiguous mentions such as partial names and abbreviated names. Though more positive samples are recalled in the candidates, the overall result is poor because of the data imbalance problem. Compared with the baseline, the candidate pairs are more balanced in the first stage of TSJ, and samples at the second stage are well balanced, indicating the effectiveness of the TSJ architecture. Besides, the number of candidates decreases notably, thus reducing the computational complexity.

**Table 6.** Statistics of candidate pairs generated by the baseline method and TSJ.

| Method & Stage | Positive | Negative | Total | Positive/Negative |
|---|---|---|---|---|
| Baseline | 7666 | 789,378 | 797,044 | 1/102 |
| FTSJ | 6848 | 95,988 | 102,834 | 1/25 |
| TSJ | 5833 | 1938 | 7771 | 3/1 |

(2) The mention type features:

The model reports a 3.28% and 1.77% increase on $F1_{ED}$. and $F1_{EDL}$ by adding the mention type features. The improvement shows the effectiveness of the LSTM-based mention type extraction model and the automatically generated training data. The representation learning approach contributes to the above improvement, and the independence of feature engineering, as well as automatic generation of training data also indicate the generality of this method. As for the slight decrease on $A_{EL}$, the reasons lie in the following two aspects. First, more true mentions are recalled after considering the type information, so the denominator of $A_{EL}$ grows; second, as for a true mention, if its candidate entity set contains false entities, namely $E_C - E_L \neq \emptyset$, and it happens that some of these false entities have the same type as the target entities. In this case, the high confidence on type consistency results in false positives, so the current mention is not correctly linked. However, in general, the features from textual context captured by the LSTM-based model bring improvement to the overall performance, proving its importance in the DSEDL task.

(3) Corpus-based inner-document entity coherence:

This group of features bring a 1.15% increase on $A_{EL}$ and a 0.9% increase on $F1_{EDL}$. The overall EDL F1-measure on different kinds of documents is illustrated in Figure 5. As the figure shows, the most significant increase appears in the movie reviews, because longer documents often contain more entities and, thus, benefit more from the topical coherence feature. However, there is little improvement on synthetic reviews. It is discovered that the synthetic reviews are more formally written and the original $A_{EL}$ already reaches 94.5%, so it is hard to obtain noticeable improvement on this category. For short documents, $A_{EL}$ decreases because of the sparseness of entities, and this problem can be solved by exploiting the following cross-document coherence features. Generally, by considering the corpus-based inner-document semantic relatedness, the TSJ model performs better, especially for DSEDL on long texts.

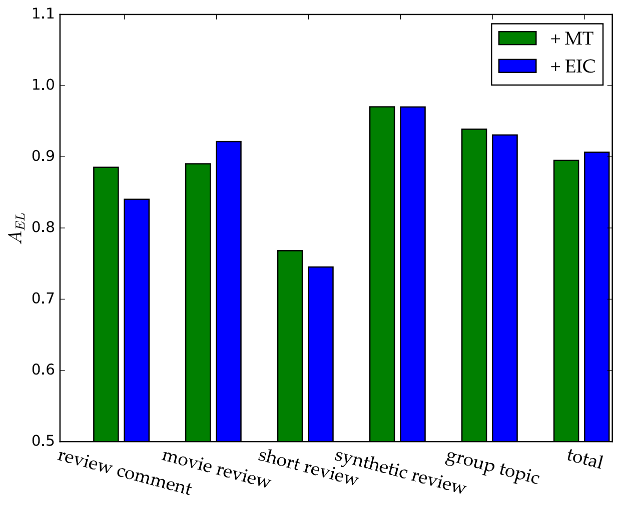

**Figure 5.** The impact of the corpus-based inner-document entity coherence on different types of documents.

(4) Cross-document coherence:

Significant improvements are observed by incorporating the cross-document cohesiveness extracted from the extension documents. $F1_{ED}$, $A_{EL}$ and $F1_{EDL}$ increase 0.89%, 2.63% and 2.9%, respectively. Similarly, we examine the impacts of this feature group on different documents, and the result is given in the (a–c) sub-plots of Figure 6. As is expected, almost all performances involving the short documents are improved, and meanwhile, remarkable gains are also observed for long texts, providing powerful evidence that our method has great advantages to capture the semantic correlation that implicitly exists in the unlabeled, but topic-similar documents.

Besides, we investigate the influence of the number of the extension documents on the final performance, and the result is given in Figure 6d. As the figure shows, with the increase of the number, the curve of overall $F1_{EDL}$ rises steeply at first and then flattens after a certain point (in the figure, when the number reaches 30). If the number keeps growing, $F1_{EDL}$ will decrease slightly, because the topics of the extension documents become less cohesive. In our experiment, the number is set to be 50.

In summary, both the TSJ model and the proposed features contribute to improve the performance from different aspects. The TSJ model contributes to balance the candidate pairs, and the three types of features incorporate evidence that is mined from the textual context or unlabeled corpora, and especially, the cross-document features bring the most noteworthy promotion, for both long and short texts, because the coherence and prior knowledge hidden in the topic-similar unlabeled documents are well utilized.

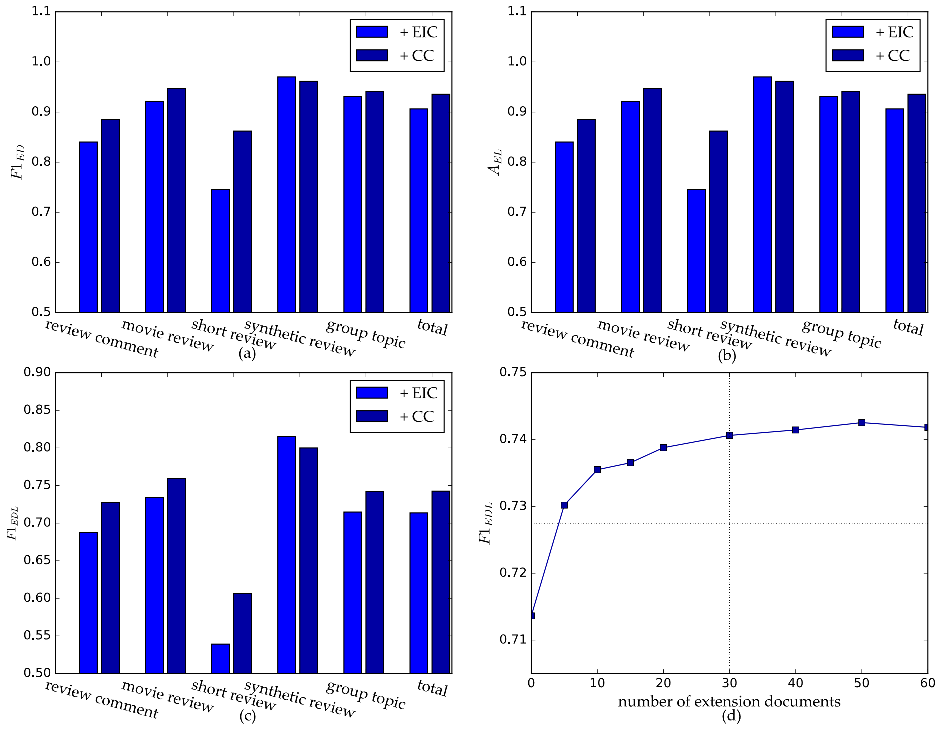

**Figure 6.** The impact of cross-document entity coherence. (**a**) $F1_{ED}$ on different types of documents; (**b**) $A_{EL}$ on different types of documents; (**c**) $F1_{EDL}$ on different types of documents; (**d**) the trend of overall $F1_{EDL}$ changing with the number of extension documents.

### 4.4.2. Comparison with Other Methods

We compare our approach with three other methods. (1) The joint EDL model proposed in [7]: This method is implemented in [33] with 56 features for entity detection and 17 features for entity linking; (2) The ensemble joint EDL model introduced in [33]: They introduce ensemble classifiers into the joint architecture proposed in [7]; (3) The pipeline architecture used in [39]: A CRF model and learning to rank method is adopted in entity detection and entity linking respectively. The last two methods are the top two systems in the DSEDL evaluation task of CCKS2016. The performances of the four methods are listed in Table 7.

**Table 7.** Performance comparison with other methods.

| **Method** | **Entity Detection** | | | **Entity Linking** | **Overall EDL** | | |
|---|---|---|---|---|---|---|---|
| | $P_{ED}$ | $R_{ED}$ | $F1_{ED}$ | $A_{EL}$ | $P_{EDL}$ | $R_{EDL}$ | $F1_{EDL}$ |
| Ensemble joint EDL model | 79.33 | 81.30 | 80.30 | 93.45 | 74.13 | 75.98 | 75.43 |
| TSJ model | 87.05 | 73.37 | 79.63 | 93.26 | 81.18 | 68.43 | 74.26 |
| Joint EDL model | 76.85 | 79.21 | 78.01 | 92.41 | 71.02 | 73.18 | 72.08 |
| Pipeline model | 82.10 | 73.98 | 77.83 | 86.53 | 71.04 | 64.01 | 67.35 |

Note: The percent signs (%) behind the numbers are omitted.

The results in Table 7 indicate that, by incorporating the interdependency between entity detection and entity linking, joint models, including our proposed TSJ model, outperform the traditional pipeline architecture. Besides, our method generates a competitive result based on a relatively simple model architecture, while the iterative architecture and ensemble method in the other two joint models both involve higher complexity.

## 5. Conclusions

In this paper, a two-stage joint model is proposed to address the high computational complexity and data imbalance problems in the traditional joint models, by covering highly ambiguous mentions in the second stage. In addition, the unlabeled corpus is incorporated to promote feature representation methods from three aspects. First, the mention type features are extracted from the context, using an LSTM-based model trained on the automatic labeled data. Second, topical coherence between entities is evaluated on the pseudo-labeled corpus to overcome the link sparsity in DSKB. Finally, cross-document entity coherence is explored to improve EDL performance on both long and short texts. Experiment results prove the advantages of the TSJ model and the novelty of feature representation methods in the DSEDL task, and our method achieves competitive results using a relatively simple model structure.

In future work, we will study the extraction of the alias and incorporate the misspelling corrector, as the proposed method cannot recall the alias of entities that are not recorded in the DSKBs. Besides, the multi-view semi-supervised learning algorithm will be explored to further learn from the unlabeled corpus.

**Acknowledgments:** This work is supported by the National High-tech R&D Program of China (Nos. 2015AA7115028, 2015AA7115061).

**Author Contributions:** Hongzhi Zhang and Weili Zhang conceived of and designed the algorithms and experiments. Hongzhi Zhang and Weili Zhang performed the experiments and analyzed the results. Weili Zhang, Hongzhi Zhang and Xiao Liang wrote the manuscript. Hongzhi Zhang and Xiao Liang revised the manuscript. Tinglei Huang discussed the data and corrected the manuscript. Tinglei Huang and Kun Fu provided relevant information and instructions during the design of experiments. All authors have read and approved the final manuscript.

**Conflicts of Interest:** The authors declare no conflict of interest.

## Appendix

The details of the basic features used in our method are explained in this part.

1. Features about the mention:

    (1) Mention length: The number of characters in the mention string.
    (2) Is title: Whether the first letter of the mention string is capitalized.

(3) All capital: Whether all letters of the mention string are capitalized.
(4) Term frequency: Frequency of the mention string in the text.
(5) Document frequency: Document frequency of the mention string in the corpora.
(6) $N_C$: The number of the candidate entities.

2. Features about the candidate entity:

(1) Entity type: The type of the entity, for example, The Twilight Saga: Breaking Dawn–Part 1 is a movie, and Barack Hussein Obama is a person. The type information is represented by the one-hot encoding.
(2) Entity popularity: Entity popularity provides a prior possibility of the current entity. Traditionally, entity popularity is evaluated by hyperlinks and statistics, such as page views of the entity page within Wikipedia. The representation of entity popularity in a specific domain depends on the available data. For example, in the movie domain, the popularity of movies can be measured by the number of rating on Douban or IMDB, and the popularity of artists can be evaluated by the number of fans, while in the e-commerce domain, the popularity of products can be estimated by their sales and page views.

3. Textual context similarity:

Textual context similarity measures the textual similarity between the description of the candidate entity and the context of the mention. Textual information of the mention and the entity is converted to vectors based on the bag of words model, and the dot-product of these two vectors is used to calculate the similarity [1]. Text associated with the entity can be extracted from the description page and textual attributes of the entity.

4. DSKB-based entity coherence:

Entities mentioned within a passage are usually about the same or related topics. In DSEDL tasks, coherence between entities is evaluated based on the relationships recorded in the DSKB, and this DSKB-based coherence is also used in our method.

Before giving the coherence measures, we first introduce the one-hop relationship $rel_1$ and two-hop relationship $rel_2$ based on the link structure of DSKBs

$$rel_1(e_i, e_j) = \begin{cases} 1 & if\ i \neq j\ and\ there\ is\ reltionship\ recorded\ between\ e_i\ and\ e_j\ in\ the\ DSKB \\ 0 & otherwise \end{cases}$$

$$rel_2(e_i, e_j) = \begin{cases} 1 & if\ i \neq j\ and\ \exists e_k \in DSKB,\ rel_1(e_i,\ e_k) = 1 \bigwedge rel_1(e_k,\ e_j) = 1 \\ 0 & otherwise \end{cases}$$

Based on the above definitions, four DSKB-based features about entity coherence are given as follows.

(1) Topic coherence between mapping entities:

Given a mention $m$ and its candidate entity $e_i$ and $M$ is the set of mentions in the input document, then we have:

$$Coh_5(m, e_i) = \sum_{m_k \in M} \sum_{e_j \in E_{Ck}} rel_1(e_i,\ e_j) \quad \text{(A1)}$$

where $E_{Ck}$ is the candidate entity set of $m_k$.

(2) Coherence between mapping entities after a basic binary classification:

$$Coh_6(m, e_i) = \sum_{m_k \in M} \sum_{e_j \in E_{Ck}} rel_1(e_i,\ e_j) C_b(m_k,\ e_j) \tag{A2}$$

$$Coh_7(m, e_i) = \sum_{m_k \in M} \sum_{e_j \in E_{Ck}} rel_1(e_i,\ e_j) C_p(m_k,\ e_j) \tag{A3}$$

The definition of $C_b(m_k,\ e_j)$ can be found in Section 3.2.3, and $C_p(m_k,\ e_j)$ is the original probabilistic result of the binary classifier.

(3) Topic coherence between mapping entities based on two-jump relationships:

$$Coh_8(m, e_i) = \sum_{m_k \in M} \sum_{e_j \in E_{Ck}} rel_2(e_i,\ e_j) \tag{A4}$$

## References

1. Shen, W.; Wang, J.; Han, J. Entity linking with a knowledge base: Issues, techniques, and solutions. *IEEE Trans. Knowl. Data Eng.* **2015**, *27*, 443–460. [CrossRef]
2. Gottipati, S.; Jiang, J. Linking entities to a knowledge base with query expansion. In Proceedings of the Conference on Empirical Methods in Natural Language Processing, Edinburgh, UK, 27–31 July 2011; pp. 804–813.
3. Han, X.; Sun, L. A generative entity-mention model for linking entities with knowledge base. In Proceedings of the Meeting of the Association for Computational Linguistics, Oregon, Portland, 19–24 June 2011; Volume 1, pp. 945–954.
4. Han, X.; Sun, L. An entity-topic model for entity linking. In Proceedings of the Empirical Methods in Natural Language Processing, Jeju Island, Korea, 12–14 July 2012; pp. 105–115.
5. Zhang, W.; Sim, Y.C.; Su, J.; Tan, C.L. Entity linking with effective acronym expansion, instance selection and topic modeling. In Proceedings of the International Joint Conference on Artificial Intelligence, Barcelona, Spain, 19–22 July 2011; pp. 1909–1914.
6. Zhang, W.; Su, J.; Tan, C.L.; Wang, W.T. Entity linking leveraging: Automatically generated annotation. In Proceedings of the International Conference on Computational Linguistics, Beijing China, 23–27 August 2010; pp. 1290–1298.
7. Zhang, J.; Li, J.; Li, X.; Shi, Y.; Li, J.; Wang, Z. Domain-specific Entity Linking via Fake Named Entity Detection. In Proceedings of the Database Systems for Advanced Applications, Dallas, TX, USA, 16–19 April 2016; pp. 101–116.
8. Li, Y.; Tan, S.; Sun, H.; Han, J.; Roth, D.; Yan, X. Entity disambiguation with linkless knowledge bases. In Proceedings of the International World Wide Web Conferences, Montreal, Canada, 11–15 April 2016; pp. 1261–1270.
9. Guo, S.; Chang, M.; Kiciman, E. To link or not to link? A study on end-to-end tweet entity linking. In Proceedings of the North American Chapter of the Association for Computational Linguistics, Atlanta, GA, USA, 9–14 June 2013; pp. 1020–1030.
10. Sil, A.; Yates, A. Re-ranking for joint named-entity recognition and linking. In Proceedings of the Conference on Information and Knowledge Management, San Francisco, CA, USA, 27 October–1 November 2013; pp. 2369–2374.
11. Han, X.; Sun, L.; Zhao, J. Collective entity linking in web text: A graph-based method. In Proceedings of the International Acm Sigir Conference on Research and Development in Information Retrieval, Beijing, China, 24–28 July 2011; pp. 765–774.
12. Ratinov, L.; Roth, D.; Downey, D.; Anderson, M.R. Local and global algorithms for disambiguation to wikipedia. In Proceedings of the Meeting of the Association for Computational Linguistics, Oregon, Portland, 19–24 June 2011; pp. 1375–1384.
13. Shen, W.; Wang, J.; Luo, P.; Wang, M. Linden: Linking named entities with knowledge base via semantic knowledge. In Proceedings of the International World Wide Web Conferences, Lyon, France, 16–20 April 2012; pp. 449–458.
14. Liu, X.; Li, Y.; Wu, H.; Zhou, M.; Wei, F.; Lu, Y. Entity linking for tweets. In Proceedings of the Meeting of the Association for Computational Linguistics, Sofia, Bulgaria, 4–9 April 2013; pp. 1304–1311.

15. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
16. Zhou, G.; Su, J. Named entity recognition using an hmm-based chunk tagger. In Proceedings of the Meeting of the Association for Computational Linguistics, Philadelphia, Pennsylvania, 7–12 July 2002; pp. 473–480.
17. Mccallum, A.; Li, W. Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In Proceedings of the North American Chapter of the Association for Computational Linguistics, Edmonton, Canada, 27 May–1 June 2003; pp. 188–191.
18. Bender, O.; Och, F.J.; Ney, H. Maximum entropy models for named entity recognition. In Proceedings of the North American Chapter of the Association for Computational Linguistics, Edmonton, Canada, 27 May–1 June 2003; pp. 148–151.
19. Hammerton, J. Named entity recognition with long short-term memory. In Proceedings of the North American Chapter of the Association for Computational Linguistics, Edmonton, Canada, 27 May–1 June 2003; pp. 172–175.
20. Nadeau, D.; Sekine, S. A survey of named entity recognition and classification. *Lingvisticae Investig.* **2007**, *30*, 3–26.
21. Gattani, A.; Lamba, D.S.; Garera, N.; Tiwari, M.; Chai, X.; Das, S.; Subramaniam, S.; Rajaraman, A.; Harinarayan, V.; Doan, A. Entity extraction, linking, classification, and tagging for social media: A wikipedia-based approach. In Proceedings of the Very Large Data Base, Trento, Italy, 26–31 August 2013; Volume 6, pp. 1126–1137.
22. Varma, V.; Bharat, V.; Kovelamudi, S.; Bysani, P.; Gsk, S.; Kiran, K.N.; Reddy, K.; Kumar, K.; Maganti, N. IIIt hyderabad at TAC 2009. In Proceedings of the Text Analysis Conference, Gaithersburg, MD, USA, 16–17 November 2009; pp. 620–622.
23. Pilz, A.; Paas, G. From names to entities using thematic context distance. In Proceedings of the Conference on Information and Knowledge Management, Glasgow, UK, 24–28 October 2011; pp. 857–866.
24. Pasca, R.B.; Marius. Using encyclopedic knowledge for named entity disambiguation. In Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics, Trento, Italy, 3–7 April 2006; pp. 9–16.
25. Zhang, W.; Yan, C.; Su, S.J. Nus-i2r: Learning a combined system for entity linking. In Proceedings of the Text Analysis Conference, Gaithersburg, MD, USA, 15–16 November 2010; Available online: https://tac.nist.gov/publications/2010/participant.papers/NUSchime.proceedings.pdf (accessed on 22 May 2017).
26. Chen, Z.; Ji, H. Collaborative ranking: A case study on entity linking. In Proceedings of the Conference on Empirical Methods in Natural Language Processing, Edinburgh, UK, 27–31 July 2011; pp. 771–781.
27. Hoffart, J.; Yosef, M.A.; Bordino, I.; Furstenau, H.; Pinkal, M.; Spaniol, M.; Taneva, B.; Thater, S.; Weikum, G. Robust disambiguation of named entities in text. In Proceedings of the Conference on Empirical Methods in Natural Language Processing, Edinburgh, UK, 27–31 July 2011; pp. 782–792.
28. Shen, W.; Wang, J.; Luo, P.; Wang, M. Linking named entities in tweets with knowledge base via user interest modeling. Proceedings of 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Chicago, IL, USA, 11–14 August 2013; pp. 68–76.
29. Zwicklbauer, S.; Seifert, C.; Granitzer, M. Robust and collective entity disambiguation through semantic embeddings. In Proceedings of the International Acm Sigir Conference on Research and Development in Information Retrieval, Pisa, Italy, 17–21 July 2016; pp. 425–434.
30. Demartini, G.; Difallah, D.E.; Cudremauroux, P. Zencrowd: Leveraging probabilistic reasoning and crowdsourcing techniques for large-scale entity linking. In Proceedings of the 21st International Conference on World Wide Web, Lyon, France, 16–20 April 2012; pp. 469–478.
31. Zhang, J.; Li, J. Graph-based jointly modeling entity detection and linking in domain-specific area. In *Knowledge Graph and Semantic Computing: Semantic, Knowledge, and Linked Big Data, Proceedings of the CCKS 2016: Chinese Conference on Knowledge Graph and Semantic Computing, Beijing, China, 19–22 September 2016*; Chen, H., Ji, H., Sun, L., Wang, H., Qian, T., Ruan, T., Eds.; Springer Singapore: Singapore, 2016; pp. 146–159.
32. Li, Y.; Wang, C.; Han, F.; Han, J.; Roth, D.; Yan, X. Mining evidences for named entity disambiguation. Proceedings of 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Chicago, IL, USA, 11–14 August 2013; pp. 1070–1078.

33. Yang, T.; Zhang, F.; Li, X.; Jia, Q.; Wang, C. Domain-specific entity discovery and linking task. In *Knowledge Graph and Semantic Computing: Semantic, Knowledge, and Linked Big Data, Proceedings of the CCKS 2016: Chinese Conference on Knowledge Graph and Semantic Computing, Beijing, China, 19–22 September 2016*; Chen, H., Ji, H., Sun, L., Wang, H., Qian, T., Ruan, T., Eds.; Springer Singapore: Singapore, 2016; pp. 214–218.
34. Guo, Y.; Qin, B.; Liu, T.; Li, S. Microblog entity linking by leveraging extra posts. In Proceedings of the Conference on Empirical Methods in Natural Language Processing, Seattle, WA, USA, 18–21 October 2013; pp. 863–868.
35. Williams, R.J.; Zipser, D. A learning algorithm for continually running fully recurrent neural networks. *Neural Comput.* **1989**, *1*, 270–280. [CrossRef]
36. Milne, D.; Witten, I.H. Learning to link with wikipedia. In Proceedings of the Conference on Information and Knowledge Management, Napa Valley, CA, USA, 26–30 October 2008; pp. 509–518.
37. Milne, D.; Witten, I.H. An open-source toolkit for mining wikipedia. *Artif. Intell.* **2013**, *194*, 222–239. [CrossRef]
38. Salton, G.; Wong, A.; Yang, C.S. A vector space model for automatic indexing. *Commun. ACM* **1975**, *18*, 613–620. [CrossRef]
39. Zhao, Y.; Li, H.; Chen, Q.; Hu, J.; Zhang, G.; Huang, D.; Tang, B. Icrc-dsedl: A film named entity discovery and linking system based on knowledge bases. In *Knowledge Graph and Semantic Computing: Semantic, Knowledge, and Linked Big Data, Proceedings of the CCKS 2016: Chinese Conference on Knowledge Graph and Semantic Computing, Beijing, China, 19–22 September 2016*; Chen, H., Ji, H., Sun, L., Wang, H., Qian, T., Ruan, T., Eds.; Springer Singapore: Singapore, 2016; pp. 205–213.