symmetry

MDPI

Article

Entity Linking via Symmetrical Attention-Based Neural Network and Entity Structural Features

Shengze Hu, Zhen Tan *, Weixin Zeng, Bin Ge and Weidong Xiao

Science and Technology on Information Systems Engineering Laboratory, National University of Defense Technology, Changsha 410073, China; springsun.hu@gmail.com (S.H.); zengweixin13@nudt.edu.cn (W.Z.); gebin@nudt.edu.cn (B.G.); wdxiao@nudt.edu.cn (W.X.)

* Correspondence: tanzhen08a@nudt.edu.cn; Tel.: +86-151-1144-0303

Received: 26 February 2019; Accepted: 24 March 2019 ; Published: 1 April 2019

Abstract: In the process of knowledge graph construction, entity linking is a pivotal step, which maps mentions in text to a knowledge base. Existing models only utilize individual information to represent their latent features and ignore the correlation between entities and their mentions. Besides, in the process of entity feature extraction, only partial latent features, i.e., context features, are leveraged to extract latent features, and the pivotal entity structural features are ignored. In this paper, we propose SA-ESF, which leverages the symmetrical Bi-LSTM neural network with the double attention mechanism to calculate the correlation between mentions and entities in two aspects: (1) entity embeddings and mention context features; (2) mention embeddings and entity description features; furthermore, the context features, structural features, and entity ID feature are integrated to represent entity embeddings jointly. Finally, we leverage (1) the similarity score between each mention and its candidate entities and (2) the prior probability to calculate the final ranking results. The experimental results on nine benchmark dataset validate the performance of SA-ESF where the average F1 score is up to 0.866.

Keywords: symmetrical neural network; entity linking; entity structural features; prior probability; information integration

1. Introduction

In the era of big data, there are massive and continuously growing data in people's lives. Hence, automatically extracting effective structural information becomes more and more pivotal. Under these circumstances, knowledge bases (KB), an effective tool to store and represent large-scale data with a structural form, have been widely leveraged and studied in various domains, e.g., information retrieval and extraction, question answer system, and text mining. Currently, almost all knowledge bases are fairly sparse and far from complete. Entity linking (EL), as a vital procedure in the knowledge base construction, links mentions in unstructured texts with the structural knowledge base.

Formally, given a document d that contains mention set $\mathcal{M} = \{m_1, m_2, \ldots, m_L\}$ and a knowledge base $\mathcal{K}$ that contains an entity set $\mathcal{E} = \{e_1, e_2, \ldots, e_N\}$, the task of entity linking is to choose a $< m_i, e_j >$ pair that maps each mention to a corresponding entity in the knowledge base. Take as an example in Figure 1 the sentence from Wikipedia that includes four mentions, *Michael Joseph Jackson*, *Gary*, *Indiana*, and *Chicago metropolitan area*. Entity linking aims at leveraging known and contextual information to find the correct entity for each mention, e.g., $<$ *Michael Joseph Jackson*, *[Michael_Jackson]*$_e$ $>$.

At the present stage, the entity linking methods can be divided into two classes: local methods and global methods. Local methods mainly utilize the semantic features, e.g., the context of each mention, character similarity between mentions and candidate entities, and so on [1–3]. They choose the candidate entities for each mention independently. These models work well when the corpus includes rich context information. Global methods aim at leveraging document features to extract and represent mention features, which consider that the mentions in the same document have correlations with each other [4,5]. Recently, several models have combined local and global information to jointly represent mention and entity features, which has achieved state-of-the-art results [6,7]. Especially, Phan et al. [6] concatenated entity context and entity ID features to obtain entity features, and Guo et al. [8] utilized *random work* to extract the correlation between different entities. To the best of our knowledge, all models above omit three crucial points: (1) entities and mentions obviously have a correlation, especially in semantic features; (2) entity and relation are basic units in the knowledge base that are equally important and have strong coherence between them; and (3) the attention mechanism can filter out pointless information in context, but it has also been omitted in entity embedding construction.

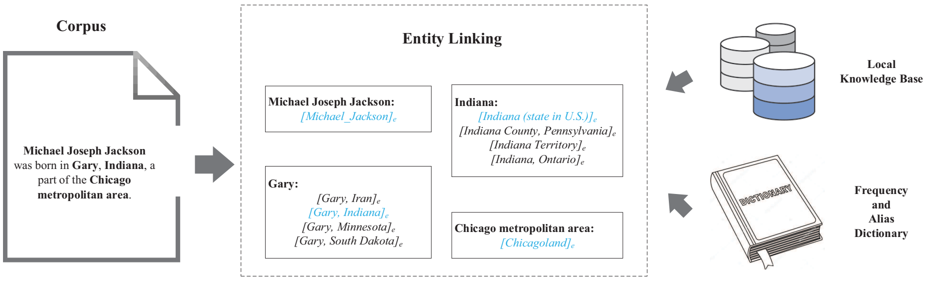


Figure 1. Linking a corpus to a local knowledge graph.

In this paper, we propose a novel entity linking model utilizing symmetrical Bi-LSTM with the attention mechanism and entity structural feature, namely SA-ESF, to address above disadvantages. Specifically, SA-ESF involves three steps:

- Candidate entities' generation: We utilize four strategies to generate candidate entities that can filter noise and improve the upper bound of linking accuracy.
- Joint words and entity embedding learning: In this step, we use the skip-gram model to map tokens' (entity ID) features into low-dimensional continuous embeddings. As a result, the token embeddings can capture the semantic similarity between different tokens effectively and can be leveraged as the input of Bi-LSTM.
- Entity disambiguation via symmetrical Bi-LSTM with the attention mechanism: We harness symmetrical Bi-LSTM with the attention mechanism to generate embeddings of each mention and its candidate entities; and then calculate the semantic similarity between mentions and their candidate entities; finally, we integrate semantic similarity and prior probability with different weights to obtain the correct $< m, e >$ pair.

On different datasets, the experimental results validate the effectiveness of SA-ESF, and in-depth analysis shows that compared to the existing entity linking methods, our model achieves better performance by providing more accurate semantics.

Contributions. In short, the major contributions of the paper can be summarized into the following four ingredients:

- We integrate multiple strategies to improve the recall of candidate entity generations, which is vital to the following process.
- We concatenate context features, entity ID features, and structural features to represent entity embeddings that can describe entity semantic features more comprehensively.
- We propose an improved symmetrical Bi-LSTM framework with the double attention mechanism to model abstract embeddings of each mention and its entities, which can extract semantic correlative features of mention and entity.
- We validate our model on nine datasets, and the results show that SA-ESF achieves state-of-the-art performance on various text corpora.

Organization. This paper is organized as follows. Section 2 introduces several related works, and then, Section 3 introduces the proposed model. Afterwards, experiments and results' analysis are reported in Section 4. Finally, conclusions are drawn in Section 5.

2. Related Work

Traditional entity linking systems tend to hand-design a set of useful features to calculate the similarities between different mentions and entities, as well as correlations between entities. We summarize those research works as entity linking models with feature engineering, which also can be divided into two general methods: independent and collective methods.

In the former method, the context features are only leveraged to disambiguate mentions, and the task is transformed into candidate entities' ranking based on mention-entity similarity. Various hand-designed features are utilized to calculate the similarity, e.g., name string similarity, prior probability, context similarity (skip-gram), and so on. The unsupervised methods [1] and supervised [9,10] approaches can be leveraged to rank the candidate entities, and in unsupervised methods, the similarity of feature vectors is calculated, while classifiers are trained in supervised approaches. Despite that these models achieve acceptable results, they ignore the correlation between different entities in the same document, which can improve the entire performance.

For collective methods that consider the mentions in the same document have a strong correlation: they regard that the corresponding correct entities also have a correlation with each other. Recently, the most popular method has been to construct a undirected graph of different candidate entities and leverage various graph mining algorithms to find final correct entities. Several models [5,8,11–13] design and improve the graph mining models to extract correlation features between different graph nodes, and based on the correlation, correct entities are obtained. Generally, the performance of collective methods is better than the traditional models with feature engineering.

In recent years, entity linking has also been applied on other related domains, e.g., target disambiguation tasks [14,15], named entity disambiguation with linkless KBs [16], the list-only ELtask [17,18], and so forth. Most of them leverage various features to obtain competitive entity linking results.

Entity Linking via Deep Neural Networks. Currently, with the development of deep neural networks, most of the natural language understanding (NLU) research has imploded. DNNs can extract latent semantic features effectively and avert intricate hand-designed feature engineering. He et al. [19] were the first to introduce deep neural networks into the entity linking framework by using an auto-encoder to represent mention and entity embeddings, which can remove the noise. For calculating the similarity score between mention and entity context embeddings, Zwicklbauer et al. [20] leveraged both word2vec and doc2vec to learn the features on a corpus created from Wikipedia pages and Google hyperlinks. Besides, aimed at obtaining the token and entity features simultaneously, several works [21,22] replaced tokens with corresponding entities and utilized word2vec to extract entity and token features. The trained embeddings enable the accurate similarity calculation between mention and entities. Based on the embeddings obtained by word2vec or doc2vec, convolutional neural network (CNN) [23–27], recurrent neural network (RNN) [6,23] and the attention mechanism [6,7] are also leveraged to extract high-level features to represent mentions and entities.

Compared with the above models, SA-ESF has three main different points:

* Considering that only utilizing LSTM does not capture the features accurately, we use Bi-LSTM with the attention mechanism to obtain the context features, which can filter the low value features and represent positional features simultaneously.
* We concatenate entity ID features, context features, and structural features to represent entity embeddings, which can capture the interactions among them.
* We leverage the entity-feature-based attention mechanism to capture the mention features, which can preserve the correlation between mention and entity effectively.

3. Methodology

The procedure of the entity linking system can be divided into two steps: candidate entities' generation and candidate entities' ranking. In the first step, we used two dictionaries (alias and frequency) and a knowledge graph to extract the entities features. Then, the well-trained symmetrical Bi-LSTM with the attention mechanism was used to embed the features' mentions and candidate entities into a continuous dense space. Finally, the similarity scores between each entity mention and its candidate entities were used to rank candidate entities and generate results. The detailed approaches are shown as follows.

3.1. Candidate Entities' Generation and Filter

The corpus containing entities' mentions may have different styles, such as long texts (news) and short texts (Twitter). Additionally, for the input sentence, it should be judged whether the words are mentions to be disambiguated or not. Hence, we should use the named entity recognition (NER) techniques to detect the mentions in each sentence. In our paper, we used the Stanford NER tools to detect the mentions with high efficiency.

After detecting mentions in each sentence, we used the local knowledge bases to extract candidate entities. For the English entity linking task, some large-scale knowledge bases, such as YAGO, DBpedia, and Wikipedia, can be downloaded and utilized. If we use mention without any modification to query the candidate entities in the knowledge base, it is hard to get the results, because each mention has many surface forms. Hence, to improve the recall of candidate entities' generation, we utilized the following four strategies.

3.1.1. Strategy 1: Noisy Removal

Firstly, we removed the noise in each mention. As is described in Table 1, several candidate mentions included extra adherent adjuncts, punctuation, and so on. The non-uniform description made mentions hard to detect during mention detection. Besides, if a mention consisted of a prefix (suffix) and another mention name, we removed the prefix or suffix to obtain its original form. Moreover, if mention m_1 appeared around m_2 and m_1 contained m_2 as a substring, we considered m_1 as an expanded form of m_2.

Table 1. Candidate mention regularization.

Original Candidate Mentions	Processed Candidate Mentions	Type
President Trump	Trump	Removing adherent adjunct
An apple	apple	Removing preposition
Job's	Job	Removing 's

3.1.2. Strategy 2: The Frequency Dictionary

Akin to other excellent candidate entity generation models [6,7,23], we also leveraged a frequency dictionary that uses $< key, value >$ pairs to unify the irregular surface forms of each mention. Specifically, the elements (mentions and entities) were collected from the home pages of Wikipedia. Apart from normal words, the pages also contained several original texts, which were hyperlinks and could link to the definition pages of other entities. Since the anchor text and corresponding entity title after redirection were a direct $< mention, candidateentity >$ pair, each anchor text can be deemed as a surface form of the redirect entity. Along these lines, it is easy to obtain a dictionary composed of the surface form and their possible referential entities.

Moreover, using anchor text information, it is easy to count the frequencies of each $< mention, candidateentity >$ pair, which can be used to calculate the prior probability. In general, the frequency dictionary uses the anchor text to construct the $< mention, candidateentity >$ pair, which can both generate candidate entities of each mention and calculate the prior probability in the following candidate entities' ranking. Several examples of entries in the frequency dictionary are shown in Table 2.

Table 2. Frequency dictionary.

Original Mentions	Possible Linking Entities	The Frequencies
Donald Trump	Donald John Trump (President of the United States)	137
	Donald Trump Jr. (American businessman)	3
	Donald L. Trump (American oncologist)	2
	...	...

3.1.3. Strategy 3: Wikipedia Functional Pages

Apart from normal words and anchor texts, Wikipedia pages also embody entity titles, redirect pages (synonym), and disambiguation pages (polysemy) [28], which help to distinguish the uncertain candidate mentions. We use Example 1 to further explain this method.

Example 1. *For the candidate mention Shaquille, we could not find a direct page matching to this surface form. However, by using the redirect page, Shaquille was redirected to entity Shaquille O'Neal; thus, we can generate the* $< Shaquille, ShaquilleO'Neal >$ *pair.*

Additionally, querying the mention Apple in Wikipedia, one would be directly redirected to the entity Apple (fruit) page. Nonetheless, in disambiguation pages of Apple, there are also other meanings, such as entity Apple Inc., entity The Apple (1980 film), entity Apple (album), and some others. The disambiguation pages can be used to enhance the recall of the candidate mentions and candidate entities.

3.1.4. Strategy 4: Alias Dictionary

Using the frequency dictionary and Wikipedia function pages, we could obtain most of the $< mention, candidateentity >$ pairs. However, some candidate mentions are aliases of the linked entities, which are hard to infer from the surface forms. To obtain the alias mentions, we constructed a map function (dictionary) to accomplish this goal. The components of the map function (alias dictionary) were entities' names and their potential aliases, which can be obtained from the info-box of each entity's Wikipedia page. The example is shown in Table 3.

Table 3. Alias dictionary.

Formal Entity Name	Possible Aliases
Rene Liu	milk tea, Liu Jo-ying ...
Jessica Ellen Cornish	Jessie J ...
LeBron Raymone James	Emperor JAMES; LeBron ...
Michael Joseph Jackson	King of Pop, Michael Joe Jackson ...
...	...

3.2. Joint Feature Embedding

The key problem, after generating candidate entities, is to extract one candidate entity as the correct one. In our model, we used deep neural network (DNN) to extract the features of each mention and its candidate entities. Before training the DNN, we first needed to obtain the embeddings of mentions and entities, which are pivotal to the final results. Mention and entity embeddings were leveraged to describe potential features of each mention and the entities. By using embeddings, we can calculate the similarity between different entities (mentions) with most similarity calculation methods (e.g., cosine, Euclidean distance) [20]. In fact, an entity is a superficial combination of words, so an intuitive way to represent the entity is by simply adding the word embeddings together. Nevertheless, obviously, each entity has its distinct features, and it would be more reasonable to represent them individually.

In this work, similar to recent entity linking models [6,22], we propose an embedding model that maps each token into a low-dimensional continuous vector space and satisfies Hypothesis 1, where the context is determined by the surrounding words or entities.

Hypothesis 1. *Two tokens tend to share similar types if they share a similar context in the corpus, and the converse also holds.*

In our model, we used *word2vec (skip-grim)* [29] to learn the embedding of each token where the token is used to predict context tokens. Formally, let $\mathcal{X}$ denote the set of tokens in each input sentence, Token $x \in \mathcal{X}$ can be either a word (e.g., Donald, Trump) or an entity (e.g., [*Donald_Trump*]$_e$). Suppose $\{x_1, x_2, \dots x_n\}$ is a given sentence; the model tries to maximize the following loss function:

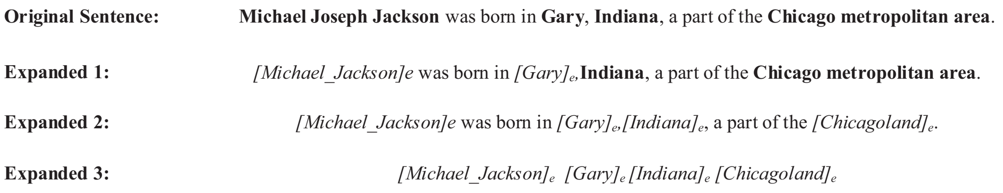
$$\mathcal{L} = \frac{1}{N} \sum_{i=1}^{N} \sum_{-w \leq j \leq w, j \neq 0} \log P(x_{i+j} | x_i). \tag{1}$$

where w denotes the size of the context window, x_i represents the target token, and x_{i+j} is a context token. The conditional probability $P(x_{t+j}|x_t)$ is defined as follows:

$$P(x_{t+j}|x_t) = \frac{\exp({v'}_{x_{t+j}}^{\top} v_{x_t})}{\sum_{x \in \mathcal{X}_w} \exp({v'}_x^{\top} v_{x_t})}. \tag{2}$$

where $\mathcal{X}_w$ denotes the set of all context words in each sentence and ${v'}_{x_{t+j}}^{\top}$ and v_{x_t} denote the token and its context embeddings, respectively. After training, we used the context embeddings for each token.

In our model, the token in each sentence was a word or an entity. For training the embedding more comprehensively, we trained the word and entity embeddings simultaneously. Specifically, for each input sentence that contained at least one mention, we constructed additional sentences by replacing any number or all of the anchor texts with their entities' names. Then, we also constructed an order-based entity list to describe the correlation between different mentions. As is displayed in Figure 2, replacing any number of or all anchor texts with entity identifiers, the expanded texts (Expanded 1, 2) for training embeddings can thus be generated. Besides, we also leveraged the entity ID to train the features, which can better capture the correlation between different mentions (Expanded 3).



Original Sentence:	**Michael Joseph Jackson** was born in **Gary**, **Indiana**, a part of the **Chicago metropolitan area**.
Expanded 1:	*[Michael_Jackson]e* was born in *[Gary]$_e$,***Indiana**, a part of the **Chicago metropolitan area**.
Expanded 2:	*[Michael_Jackson]e* was born in *[Gary]$_e$,[Indiana]$_e$*, a part of the *[Chicagoland]$_e$*.
Expanded 3:	*[Michael_Jackson]$_e$ [Gary]$_e$ [Indiana]$_e$ [Chicagoland]$_e$*

Figure 2. Corpus expansion.

The advantages of the embedding model are three-fold:

- Compared with other embedding models, the joint embedding model can describe the mention and entities' features more accurately, because using entities to replace separate words can present the integral entities' features rather than single word features.
- Reserving part of the word features can train the mentions and entities' features simultaneously and obtains the correlation between them, which describes the mentions and entities' features more comprehensively.
- Since mentions and entities are represented in the same embedding space, the same features have the same embeddings, which makes it easier to calculate the similarities between words and entities.

3.3. Entity Disambiguation via DNN

Conventional models leverage diverse discrete and manual-designed features to calculate the similarities between mentions and candidate entities for candidate entities' ranking. However, it is hard to exhaust all features, design the intrinsic unseen features, and calibrate the effectiveness of manual features [24].

Recently, with the extension of deep neural network, DNN has been utilized in many entity linking models [24–26], which can extract the intrinsic features in text and calculate the similarities between mentions and candidate entities more accurately. Nevertheless, the existing solutions with DNN have obvious shortcomings for entity features' extraction:

* The crucial entities' description information is ignored. The position information of mentions was overlooked in previous models. As a consequence, the different mentions in the same context are considered as unanimous, which is undoubtedly illogical.
* Existing DNN models omit the word order in mention and candidate entities' contexts, which has proven to be pivotal for natural language understanding [30].
* The structural information of entities is missing. Existing approaches [6,26] only used the entity description to extract candidate entities' features, which is clearly impractical; because, in knowledge bases, the correlation between different entities is equally important to describe entity features.

In this work, we put forward SA-ESF, which outperforms previous models by:

* Integrating symmetrical Bi-LSTM with the attention mechanism into both mention and candidate entities' features extraction;
* Optimizing the inputs of symmetrical Bi-LSTM; and
* Simultaneously using entities' description and knowledge graph structure to extract entity features.

The framework of our model is shown in Figure 3, which consists of three basic units: candidate entity features extraction, mention features extraction, and similarity calculation. Firstly, the entity description and structure information were used to extract description embedding and structure embedding, which were integrated to obtain entity embedding. Then, word embedding, entity embedding,and entity ID embedding were integrated to extract the mention embedding with symmetrical Bi-LSTM. Finally, the mention and candidate embeddings were input to two hidden layers, and the similarity between the mention and candidate entity was generated by the sigmoid function, which calculates the output of hidden layers. The final ranking score was composed of the similarity score and prior probability. The max-pooling and double-attention mechanism were embedded in the framework to alleviate the negative effect imposed by noisy information and improve model performance.

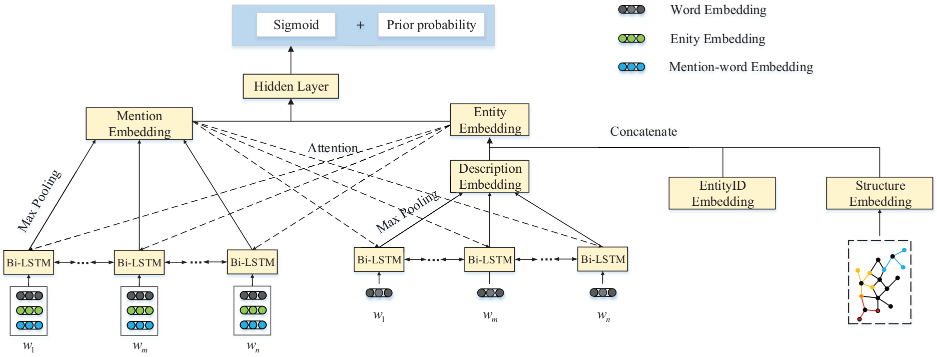


Figure 3. The framework of our model.

3.3.1. Symmetrical Bi-LSTM with the Attention Mechanism

Recurrent neural network (RNN) is a typical feed-forward DNN, which extracts the features with time-series information, yet it is hard to reserve the effective features in a long sequence. To tackle these issues, *long short-term memory network* (LSTM), an extension of RNN, is proposed to extract the features with variable-length sequences. Specifically, given input sequence $S = \{x_1, x_2, ..., x_n\}$, where x_t is a token embedding to be passed as input at position t, the LSTM unit will output h_t for each position t. The hidden feature h_t can be calculated as follows:

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i) \tag{3}$$

$$f_t = \sigma(W_{fi}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f) \tag{4}$$

$$c_t = f_t c_{t-1} + i_t \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \tag{5}$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_{t-1} + b_o) \tag{6}$$

$$h_t = o_t \tanh(c_t) \tag{7}$$

where i_t, f_t, o_t, and c_t are the input gate, forget gate, output gate, and cell memory at position t, respectively. σ denotes the sigmoid function. W_x, W_h, and W_c are weighted matrices, and b_i, b_f, b_c, and b_o represent the biases of the LSTM network. All the above parameters need to be learned during training.

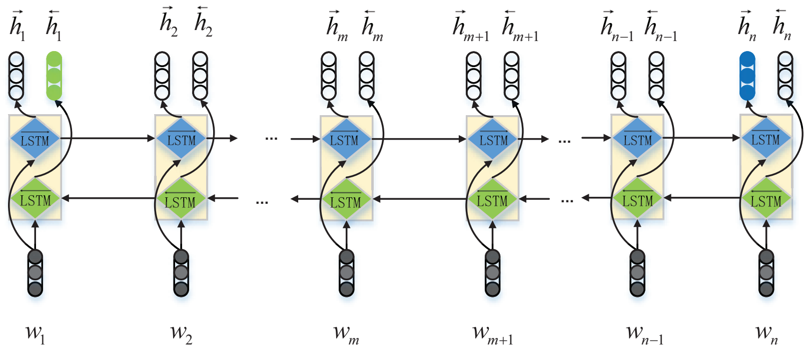
Bi-LSTM. Due to the fact that LSTM only extracts single-directional features in each sentence, in this paper, we used Bi-LSTM to extract bi-directional features. Specifically, in *Bi-LSTM*, which is comprised of two LSTMs, one extracts forward hidden features $\overrightarrow{h_t}$, and the other extracts the backward hidden features $\overleftarrow{h_t}$. The token embedding sequence can be described as $S = \{x_1, x_2, ..., x_n\}$, and n is the length of the sentence. The outputs $\overrightarrow{h}_t$ and $\overleftarrow{h}_t$ are computed as follows:

$$\overrightarrow{h}_t, \overrightarrow{c}_t = LSTM(x_t, \overrightarrow{h}_{t-1}, \overrightarrow{c}_{t-1}) \tag{8}$$

$$\overleftarrow{h}_t, \overleftarrow{c}_t = LSTM(x_t, \overleftarrow{h}_{t+1}, \overleftarrow{c}_{t+1}) \tag{9}$$

The output of the Bi-LSTM layer can be described as $\overline{h_{\mathrm{t}}} = [\overrightarrow{h}_t, \overleftarrow{h}_t]$, and the basic structure can be observed in Figure 4. Due to the dimension of $\overline{h_{\mathrm{t}}}$ being $2k$, we used an activation function to project $\overline{h_{\mathrm{t}}}$ into the k-dimensional space. The function is denoted as:

$$h_t = \tanh(W_t\overline{h}_t + b_t) \tag{10}$$



$\overrightarrow{h}_1$ $\overleftarrow{h}_1$
$\overrightarrow{h}_2$ $\overleftarrow{h}_2$
$\overrightarrow{h}_m$ $\overleftarrow{h}_m$
$\overrightarrow{h}_{m+1}$ $\overleftarrow{h}_{m+1}$
$\overrightarrow{h}_{n-1}$ $\overleftarrow{h}_{n-1}$
$\overrightarrow{h}_n$ $\overleftarrow{h}_n$
LSTM
LSTM
...
w_1
w_2
w_m
w_{m+1}
w_{n-1}
w_n

Figure 4. Basic Bi-LSTM structure.

Attention Mechanism. It is hard for standard Bi-LSTM to judge whether the components in the input sequence are meaningful or not, which introduces noise into embeddings and reduces the reliability of mention/entity representations. In order to overcome this issue, the attention mechanism was leveraged to capture the key units of the input sentence in response to a given mention or entity embeddings.

Specifically, suppose $H \in R^{d \times n}$ is a matrix comprised of hidden vectors $(h_1, ..., h_n)$ produced by Bi-LSTMs (mention embedding or entity description embeddings), where d is the dimension of the hidden layers and n is the length of the input sentence. The attention mechanism will produce the weighted hidden representation $\bar{H}$.

$$Z = \tanh \begin{bmatrix} W_h H \\ W_a v_a \otimes e_n \end{bmatrix} \tag{11}$$

$$\alpha = softmax(w^{\mathrm{T}} Z) \tag{12}$$

$$\bar{H} = H \alpha^{\mathrm{T}} \tag{13}$$

where v_a represents the attention vector (mention or entity embedding), e_n is the vector of ones, and α represents the weight vector of attention. W_h and W_a are matrices to be learned, and $v_a \otimes e_n = [v_a; v_a; ...; v_A]$. Then, the re-weighted $\bar{H}$ is forwarded to the max-pooling process.

Max-Pooling. The standard Bi-LSTM network regards the last forward and backward hidden vectors $\overrightarrow{h_n}$ and $\overleftarrow{h_1}$ as the final representation of the sequence, as is highlighted by different colors in Figure 4. To utilize fully the information in the sequence, we followed [6] to generate a fixed-length final representation via max-pooling all hidden states over time steps, i.e., $(h_1, ..., h_n)$.

3.3.2. Entity Features' Extraction

For entity features' extraction, all the recent models only used entity description to extract entities' features and completely ignored the structural information, which is common and pivotal in knowledge bases. We present the entity features' extraction method, which considers both the structural and text-description features to construct entity embedding.

Entity Structure Embedding. By utilizing candidate entities in the entity linking system, it is easy to construct a knowledge base. In the knowledge base, the triplet is the basic unit, which can be described as (e_h, e_t, r), where e_h and e_t denote the head and tail entities and r denotes the relations. For extracting the structural information in each entity, we used a state-of-the-art model CombinE [31] to embed the entity structural features. In CombinE, plus and minus combinations are used to describe the intrinsic and extrinsic features of entities. For plus combination ($r \approx e_h + e_t$), the relation embeddings can represent the shared features of entities' pairs, and for minus combination ($r \approx e_h - e_t$), the relation embeddings can represent the individual features of head and tail entities. As a consequence, the overall score function of the integrated model is:

$$\begin{aligned} f(e_h, e_t, r; \Theta) = & \|\mathbf{h_p} + \mathbf{t_p} - \mathbf{r_p}\|^2_{L_1/L_2} \\ & + \|\mathbf{h_m} - \mathbf{t_m} - \mathbf{r_m}\|^2_{L_1/L_2}. \end{aligned} \tag{14}$$

where Θ is the set of parameters, $\mathbf{h_p}$, $\mathbf{t_p}$, $\mathbf{h_m}$, and $\mathbf{t_m}$ denote the head and tail entity structural embeddings, and $\mathbf{r_p}$ and $\mathbf{r_m}$ denote the relation embeddings. The subscripts p and m represent plus and minus combination, respectively. After training, we concatenated plus and minus combination features and obtained the structural embedding of each entity $\mathbf{e}_s = [\mathbf{e_p}, \mathbf{e_m}]$.

EntityID Embedding. *EntityID* embedding $\mathbf{e}_i$ is generated by the *skip-grim* method where the Wikipedia entity ID is utilized to replace mention in the text corpus. The *EntityID* Embedding can describe the overall entity features rather than single word embedding.

Entity Description Embedding. For intrinsic entity features, we used Bi-LSTM to extract entity description features. From bottom to top, the framework consisted of four components, embedding layer, Bi-LSTM layer, attention mechanism, and max-pooling. Firstly, we set a window size of c to extract the entity description from its Wikipedia page. Then, the embedding sequences were utilized as inputs, which were forwarded to a Bi-LSTM to extract latent features of each embedding. Third, the attention mechanism with *mention features* was used to judge the importance of each word. Finally, max-pooling was leveraged to improve the performance and obtain the entity description embedding $\mathbf{e}_d$. We integrated structure and description embeddings and obtained the final entity embedding $\mathbf{e} = [\mathbf{e}_s, \mathbf{e}_i, \mathbf{e}_d]$.

3.3.3. Mention Features' Extraction

Similar to the entity features' extraction, we used the embedding layer, Bi-LSTM layer, attention mechanism, and max-pooling to extract the mention features. Different from entity description embedding, in the embedding layer, we integrated word embedding, entity embedding, and mention-word embedding to represent input features jointly.

Word Embedding. Word embedding $\mathbf{x}$ was constructed by the joint embedding model described in Section 3.2, where the embedding can capture the semantic and syntactic similarity between different words.

Entity Embedding. Entity embedding $\mathbf{e}$ can be obtained via Section 3.3.2, where both description and structure information were leveraged to represent entity embedding.

Mention-word Embedding. For each mention m constructed by a word list $\{x_{m_1}, x_{m_2}, ..., x_{m_s}\}$, s is the length of mention words. We used a linear combination of each word in the mention to represent the mention-word embedding, which can be denoted as $\mathbf{m}_w = \frac{1}{s}\sum_{i=1}^{s} \mathbf{x}_{m_i}$.

For the words in the mention context, the mention-word embedding was concatenated with its word and entity embeddings, which can describe the correlation between mention and context inputs. Hence, the final concatenated input can be shown as $[\mathbf{x}, \mathbf{e}, \mathbf{m}_w]$.

After concatenating the input embedding, we used Bi-LSTM to extract the latent features of each word. Then, the attention mechanism was used to extract crucial units in context. Similar to entity features' extraction, which uses *mention embedding* to adjust context weight, we used *entity embedding* to calculate the weight of the mention context, which can capture the useful mention features more effectively. Final, we leveraged max-pooling to choose pivotal features that were used to represent the mention embedding $\mathbf{m}$.

Discussion. Traditional entity linking models [6,7] only leverage entity embedding to constrain the feature weights of the mention context word, which omits the correlation between mention and entity description features. Hence, we used symmetrical Bi-LSTM with the attention mechanism to consider simultaneously the correlation in two aspects: (1) the correlation between mention embeddings and entity description features; and (2) the correlation between entity embeddings and mention context features.

3.3.4. Similarity Calculation

After obtaining mention and candidate entities' embeddings, they were concatenated and forwarded to two hidden layers and a sigmoid function to calculate the similarity score between mention and entity embedding. Then, the results of a linear combination between the similarity score and prior probability were leveraged to obtain final candidate entities' ranking.

Hidden Layer. The mention and entity embeddings were concatenated and then forwarded to the two-layer neural network. The output was a single representation vector, denoting the similarity score after being processed by a sigmoid function. Suppose s is the final similarity score and t represents whether the entity is true or not. The loss function $\mathcal{L}(s,t)$ is described as follows:

$$\mathcal{L}(s,t) = t\log(s) + (1-t)\log(1-s) \tag{15}$$

Candidate Entities' Ranking. It is inappropriate to rank candidate entities solely based on context features. Instead, the final similarity score between mention and entity is a linear weighted combination of the similarity score $sim(m, e)$ and prior probability $p(e|m)$. We used the frequency dictionary to get the specific values of prior probabilities, and the frequency value was assigned as zero when entities were missing in the frequency dictionary. Formally, the ranking score $r(m, e)$ of the $< mention, entity >$ pair is:

$$r(m, e) = \alpha p(e|m) + \beta sim(m, e) \tag{16}$$

where α and β are coefficients balancing the weights of similarity and prior probability, and $\alpha + \beta = 1$.

4. Experiments and Results' Analysis

Our model SA-ESF, as well as several variants, were evaluated and compared against several competing models, which have been shown to achieve state-of-the-art performance. In this section, we first introduce the experiment settings, which consisted of constructing the local knowledge base, creating the frequency and alias dictionary, and generating the input embedding. Then, the datasets, along with baselines in comparison, are introduced, followed by results' analysis.

4.1. Experimental Settings

For constructing the local knowledge base, Wikipedia dumps were utilized. Specifically, as for English datasets, we leveraged the English Wikipedia dump on 1 July 2016 to construct the local knowledge base, which included 5,187,458 entities; and for Chinese datasets, the Chinese Wikipedia dump on 1 December 2017 was utilized. Besides, we utilized MySQL, a popular and simple database, to store the knowledge base.

Frequency Dictionary. We also utilized the Wikipedia dump to construct the frequency dictionary where the Wikipedia Extractor was adopted to extract anchor texts and their *hyperlinks*. Then, the *hyperlinks* were replaced by the entity names of their corresponding Wikipedia pages. Finally, we recorded the co-occurrence of candidate entities' names and mentions and obtained the frequency dictionary. Note that the process of constructing the alias dictionary is described in Section 3.

Joint Pre-trained Embeddings. As is described in Section 3, the pre-trained corpus contained four units: the original form and its three expanded forms. Since there is no space between words in Chinese, Jieba was leveraged as a segmentation tool to pre-process Chinese sentence. Then, both Chinese and English corpus were forwarded to Gensim to train and obtain the embeddings of each word and *entityID*. The embedding dimension was set to 200; the window size was five; and the number of training iteration was 10. We finally attained 6,363,417,735 and 5,553,238 effective items on the Chinese and English corpus, respectively.

Settings for the Neural Network. The specific hyperparameters in the deep neural network are shown in Table 4. For each mention, we extracted the context with a window size of 10, and the entity description text included the first 150 words in its Wikipedia page. The numbers in the sentence were removed, and all words in the English corpus were lowercased. The word embeddings in the neural network were stable during the model training, and only the parameters in the neural network were learned.

The Wikipedia text and hyperlinks were leveraged as training sources. Due to the limitation of computational resources, for each entity, we randomly picked 100 $< mention, entity >$ pairs with their context as positive training samples, and we randomly generated five negative samples by means of replacing the entity in the $< mention, entity >$ pair with a random entity.

Table 4. Neural network parameter settings.

Parameters	Values
window size of mention context	10
window size of entity description	150
the output size of Bi-LSTM for mention context	288
the output size of Bi-LSTM for entity description	96
output size for hidden layer	200
activation function for hidden layer	tanh
number of epochs	50
batch size	128
optimizer	Adam

4.2. Datasets and Baselines

SA-ESF was evaluated on nine benchmark datasets, including short and long text, formal and informal text, and English and Chinese text. The statistic information is shown in Table 5. Note that we only considered the mentions with linked entities present in the Wikipedia dump, using the same settings as in [20,22,32,33]. Each dataset is described as follows:

* **Reuters128** extracts 128 news articles in the economic domain from the Reuters-21587 corpus, and 111 of them had $< mention, entity >$ pairs with the English Wikipedia dump.
* **ACE2004** was constructed from ACE2004, which is a co-reference corpus annotated by Amazon. The dataset on average contained 7.4 mentions in each document.
* **MSNBC** extracts news from the MSNBC dataset and had 20 documents and 658 mentions.
* **DBpedia** leverages the knowledge base DBpedia to construct the corpus, which contained 57 documents, each document of which on average had 5.81 mentions.
* **RSS500** collects short text from different domains, and each short text contained 1.51 mentions on average.
* **KORE50** is composed of 50 short texts, which include various domains, e.g., music and business.
* **Microposts2014** extracts short text from tweets. Each short text contained 2.09 mentions on average.
* **NLPCC2013** extracts short text from a Chinese micro-blog, which is a Chinese text source with much noise. The dataset had 441 sentences and 1123 mentions.
* **NLPCC2014** also extracts short text from a Chinese micro-blog. Each sentence in the dataset had 1.30 mentions on average.

Table 5. Statistic of datasets. #Doc, #Men, and #Avg_m are the number of documents, number of mentions, and average number of mentions per document, respectively.

Dataset	Type	#Doc	#Men	#Avg_m
Reuters128	news	111	637	5.74
ACE2004	news	35	257	7.34
MSNBC	news	20	658	32.90
DBpedia	news	57	331	5.81
RSS500	RSS-feeds	343	518	1.51
KORE50	short sentences	50	144	2.88
Micro2014	tweets	696	1457	2.09
NLPCC2013	Microblogs	441	637	1.44
NLPCC2014	Microblogs	263	343	1.30

We compared SA-ESF with nine state-of-the-art entity linking models. AIDA [5], Kea [34], WAT [35], PBoH [32], and DoSeR [20] were implemented on the English datasets; and CBWV [36], MSCM [37], MKCM [38], and SCWE [39] were utilized to be compared with SA-ESF on Chinese datasets. The details are as follows:

* AIDA [5] constructs a dense subgraph to describe the features between different candidate entities.

- Kea [34] leverages a fine-grained model to analyze the mention context and uses context similarity to extract the $< mention, entity >$ pair.
- WAT [35], an improved version of TagMe [40], utilizes a graph-based and vote-based algorithm to calculate the correlation and rank candidate entities.
- PBoH [32] leverages Wikipedia pages to obtain the prior distribution and designs a probabilistic graphical model to calculate the similarity between different candidate entities.
- The DoSeR [20] system builds a text analysis model to extract context features and runs the PageRank algorithm on constructed candidate-entities' graphs, which can use entities' correlation to extract correct entities.
- CBWV [36], first, extends the knowledge base and constructs the synonym dictionary, then leverages the word vector to extract features, and finally calculates the semantic similarity between mention and entities.
- MSCM [37] utilizes TF-IDF and the fast-Newman clustering algorithm to extract mention and entity features and rank candidate entities.
- MKCM [38] extracts features from the encyclopedia and synonym dictionary and then uses SVM to rank candidate entity features.
- SCWE [39] uses the neural network and cluster algorithm to extract entity features and then leverages multiple classifiers to categorize candidate entities.
- Ppo [20] only leverages the prior probability to link the candidate entities.

4.3. Experimental Results and Analyses

In this paper, we omit the NILsituation and guarantee that each mention has at least one entity in the knowledge base [6,22,32,41].

Candidate Generation Strategy Analysis. The higher *recall* of candidate entity generation CEG_{Recall} means a higher probability of finding the correct entity for each mention, which is pivotal for improving the accuracy of entity linking tasks. The definition of CEG_{Recall} is detailed in Equation (17).

$$CEG_{Recall} = \frac{\#mentions\ (candidates\ containing\ true\ entity)}{\#all\ mentions} \tag{17}$$

The CEG_{Recall} of different candidate generation strategies for different datasets is shown in Table 6, where S denotes the strategy described in 3.1 and **original** denotes only matching the mention name with the candidate entity. From Table 6, we can conclude that:

- All strategies can improve the recall of candidate entity generation, and **S2** (frequency dictionary) had the largest increment compared with other strategies.
- Compared with news articles, the short sentences and tweets' contents were hard to recognize. Especially, for the KORE50 dataset, the value of *recall* was only 0.899.
- Compared with first seven English datasets, NLPCC2013 and NLPCC2014, two Chinese datasets, had lower results when it came to being **original**. The direct reason is that Chinese texts need to be segmented, which will bring a large amount of noise.

Table 6. *Recall* with respect to different strategies.

Dataset	Original	+ S1	+ S1 + S2	+ S1 + S2 + S3	+ S1 + S2 + S3 + S4
Reuters128	0.533	0.611	0.813	0.899	0.967
ACE2004	0.553	0.633	0.841	0.922	0.989
MSNBC	0.521	0.602	0.814	0.913	0.974
DBpedia	0.499	0.564	0.788	0.876	0.951
RSS500	0.489	0.562	0.768	0.879	0.934
KORE50	0.455	0.534	0.732	0.839	0.899
Micro2014	0.467	0.564	0.799	0.879	0.913
NLPCC2013	0.423	0.571	0.821	0.886	0.968
NLPCC2014	0.369	0.567	0.801	0.880	0.978

Comparisons with Other Systems. We used the *precision*, *recall*, and micro-average *F1* (aggregated across mentions) as metrics, which are shown as follows:

$$Precision = \frac{|T \cap O|}{|O|} \tag{18}$$

$$Recall = \frac{|T \cap O|}{|T|} \tag{19}$$

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall} \tag{20}$$

where T and O are the sets of mentions linking to entities. The results are shown in Tables 7 and 8. From the tables, we can conclude that

- Except MSNBC, our model achieved state-of-the-art results on almost all datasets, and the average value of F1 was up to 0.858. On KORE50, SA-ESF largely improved the F1 value, and the increment was 0.227. Besides, on the Chinese datasets NLPCC2013 and NLPCC2014, the F1 values were up to 0.923 and 0.872, respectively. The results can prove the robustness and effectiveness of our model.
- DoSeR was the second best model and uses doc2vec to extract the context features. In comparison, SA-ESF used Bi-LSTM with the attention mechanism to capture the context features, which can model the positional information effectively.
- Compared with PBoH, which uses the probabilistic graphical model to calculate the similarity between different candidate entities, SA-ESF uses CombinE to extract the correlation between different entities and achieved state-of-the-art results.

Table 7. Micro-averaged F1 on the English datasets.

System	Reuters128	ACE2004	MSNBC	DBpedia	RSS500	KORE50	Micro2014	Average
AIDA	0.599	0.820	0.759	0.249	0.722	0.660	0.433	0.606
Kea	0.654	0.796	0.854	0.736	0.709	0.620	0.639	0.715
WAT	0.660	0.809	0.795	0.671	0.700	0.599	0.604	0.691
PBoH	0.759	0.876	0.897	0.791	0.711	0.646	0.725	0.772
DoSeR	0.873	0.921	**0.912**	0.816	0.762	0.550	0.756	0.798
Ppo	0.513	0.670	0.741	0.359	0.689	0.772	0.574	0.691
SA-ESF	**0.914**	**0.923**	0.911	**0.831**	**0.811**	**0.799**	**0.814**	**0.858**

Table 8. Micro-averaged F1 on the Chinese datasets.

Systems	NLPCC 2013	NLPCC 2014
CBWV	0.902	0.833
MSCM	0.896	0.829
MKCM	0.873	0.828
SCWE	0.885	0.835
Ppo	0.754	0.713
SA-ESF	**0.923**	**0.872**

Hyperparameters' Analysis. Both context similarity and prior probability are equally important to the entity link model, and to achieve state-of-the-art results, we should choose suitable α and β for each dataset. The results are displayed in Table 9. From Table 9, we can conclude that:

- Prior probability is a vital element for the entity linking model where α was larger than β in all the datasets; especially on ACE2014, α was up to 0.80.
- The first four datasets extracted from news had smaller β than other datasets, which can be attributed to the fact that the mentions from news were usually easy to recognize because the mentions were more accurate and less ambiguous than short text.

- * On the Chinese datasets (NLPCC2013, NLPCC2014), the prior probability is also important, and the values of α were 0.60 and 0.55, respectively.

Table 9. Micro-averaged precision, recall, the F1 score of SA-ESF, and the optimal settings on each dataset.

Data Set	Precision	Recall	F1	α	β
Reuters128	0.917	0.911	0.914	0.75	0.25
ACE2004	0.927	0.919	0.923	0.80	0.20
MSNBC	0.913	0.909	0.911	0.70	0.30
DBpedia	0.834	0.828	0.831	0.70	0.30
RSS500	0.813	0.809	0.811	0.60	0.40
KORE50	0.800	0.798	0.799	0.55	0.45
Micro2014	0.815	0.813	0.814	0.65	0.35
NLPCC2013	0.925	0.921	0.923	0.60	0.40
NLPCC2014	0.874	0.870	0.872	0.55	0.45

Neural Network Variants' Analysis. We further compared SA-ESF with its variants, which can evaluate each component in our model. The results are shown in Table 10, where LSdenotes only using the single LSTM model to replace Bi-LSTM; NAdenotes removing the attention mechanism; MAdenotes only using the attention mechanism on mention features' extraction; EArepresents only using the attention mechanism on entity features' extraction; and ESdenotes omitting the entity structural features. From Table 10, we can conclude that:

- * All the components can bring improvement to our model. Concretely, if we remove the entity structural embeddings (ES), the corresponding result (0.822) was far away from the best result (0.866), which can prove the effectiveness of the ES component.
- * Using the attention mechanism on both mention and entity contexts can extract latent features effectively, i.e., on NLPCC, the value of NA was 0.880, which was lower than the value of SA-ESF. Besides, the average values on both MA and EA were 0.848, which were lower than SA-ESF (0.866) and larger than NA (0.824). The results can prove that the symmetrical attention mechanism has effectively extracted latent contexts, and using the double attention mechanism can obtain both mention and entity features simultaneously.
- * The LS results were also smaller than SA-ESF on all datasets, which can prove that the Bi-LSTM can capture the context features more accurately.

Table 10. F1 value of different neural network variants.

Data Set	SA-ESF	LS	NA	MA	EA	ES
Reuters128	**0.914**	0.909	0.877	0.902	0.900	0.879
ACE2004	**0.923**	0.919	0.881	0.909	0.905	0.878
MSNBC	**0.911**	0.901	0.858	0.895	0.891	0.859
DBpedia	**0.831**	0.815	0.782	0.809	0.807	0.776
RSS500	**0.811**	0.809	0.776	0.803	0.805	0.771
KORE50	**0.799**	0.788	0.749	0.774	0.781	0.751
Micro2014	**0.814**	0.800	0.766	0.794	0.792	0.765
NLPCC2013	**0.923**	0.899	0.880	0.895	0.896	0.878
NLPCC2014	**0.872**	0.860	0.848	0.853	0.855	0.841
Average	**0.866**	0.856	0.824	0.848	0.848	0.822

5. Conclusions

In this paper, we first leveraged multiple strategies to improve the recall of candidate entity generation; then, we integrated context features, entity ID features, and structural features to represent entity features, which can describe the entity semantic features more comprehensively; and then, the improved Bi-LSTM framework with the attention mechanism was utilized to model abstract embeddings of each mention and its candidate entities, which can extract semantic features of the mention and entity; finally, the experimental results validated the effectiveness of SA-ESF. Compared with other models, our model leveraged structural information and attention mechanism to extract entity features, which can represent entity features more comprehensively.

Author Contributions: conceptualization, Z.T.; methodology, W.Z.; software, Z.T.; validation, B.G.; formal analysis, S.H; investigation, Z.T.; resources, S.H; data curation, S.H.; writing—original draft preparation, W.Z.; writing—review and editing, W.X.; visualization, W.Z.; supervision, Z.T.; project administration, Z.T.; funding acquisition, S.H.

Funding: This research was funded by NSFC grant number 61872466, 71690233, 71331008.

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

References

1. Bunescu, R.C.; Pasca, M. Using Encyclopedic Knowledge for Named entity Disambiguation. In Proceedings of the EACL 2006, 11st Conference of the European Chapter of the Association for Computational Linguistics, Trento, Italy, 3–7 April 2006.
2. Bagga, A.; Baldwin, B. Entity-Based Cross-Document Coreferencing Using the Vector Space Model. In Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Montreal, QC, Canada, 10–14 August 1998; pp. 79–85.
3. Zhao, X.; Xiao, C.; Lin, X.; Zhang, W.; Wang, Y. Efficient structure similarity searches: A partition-based approach. *VLDB J.* **2018**, *27*, 53–78. [CrossRef]
4. Cucerzan, S. Large-Scale Named Entity Disambiguation Based on Wikipedia Data. In Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, Prague, Czech Republic, 28–30 June 2007; pp. 708–716.
5. Hoffart, J.; Yosef, M.A.; Bordino, I.; Fürstenau, 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, John McIntyre Conference Centre, Edinburgh, UK, 27–31 July 2011; pp. 782–792.
6. Phan, M.C.; Sun, A.; Tay, Y.; Han, J.; Li, C. NeuPL: Attention-based Semantic Matching and Pair-Linking for Entity Disambiguation. In Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, Singapore, 6–10 November 2017; pp. 1667–1676. [CrossRef]
7. Ganea, O.; Hofmann, T. Deep Joint Entity Disambiguation with Local Neural Attention. In Proceedings of the EMNLP, Copenhagen, Denmark, 9–11 September 2017; pp. 2619–2629.
8. Guo, Z.; Barbosa, D. Robust Entity Linking via Random Walks. In Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management, Shanghai, China, 3–7 November 2014; pp. 499–508. [CrossRef]
9. Dredze, M.; McNamee, P.; Rao, D.; Gerber, A.; Finin, T. Entity Disambiguation for Knowledge Base Population. In Proceedings of the 23rd International Conference on Computational Linguistics. Association for Computational Linguistics, Beijing, China, 23–27 August 2010; pp. 277–285.
10. Mihalcea, R.; Csomai, A. Wikify!: Linking documents to encyclopedic knowledge. In Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge Management, Lisbon, Portugal, 6–10 November 2007; pp. 233–242. [CrossRef]
11. Alhelbawy, A.; Gaizauskas, R.J. Graph Ranking for Collective Named Entity Disambiguation. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, Baltimore, MD, USA, 22–27 June 2014; pp. 75–80.
12. Pershina, M.; He, Y.; Grishman, R. Personalized Page Rank for Named Entity Disambiguation. In Proceedings of the NAACL HLT 2015, The 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Denver, Colorado, USA, 31 May– 5 June 2015; pp. 238–243.

13. Han, X.; Sun, L.; Zhao, J. Collective entity linking in web text: A graph-based method. In Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval, Beijing, China, 25–29 July 2011; pp. 765–774. [CrossRef]
14. Wang, C.; Chakrabarti, K.; Cheng, T.; Chaudhuri, S. Targeted disambiguation of ad-hoc, homogeneous sets of named entities. In Proceedings of the 21st World Wide Web Conference 2012, WWW 2012, Lyon, France, 16–20 April 2012; pp. 719–728. [CrossRef]
15. Cao, Y.; Li, J.; Guo, X.; Bai, S.; Ji, H.; Tang, J. Name List Only? Target Entity Disambiguation in Short Texts. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Lisbon, Portugal, 17–21 September 2015; pp. 654–664.
16. Li, Y.; Tan, S.; Sun, H.; Han, J.; Roth, D.; Yan, X. Entity Disambiguation with Linkless Knowledge Bases. In Proceedings of the 25th International Conference on World Wide Web, WWW 2016, Montreal, QC, Canada, 11–15 April 2016; pp. 1261–1270. [CrossRef]
17. Lin, Y.; Lin, C.; Ji, H. List-only Entity Linking. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, Vancouver, Canada, July 30–August 4 2017; pp. 536–541. [CrossRef]
18. Zeng, W.; Zhao, X.; Tang, J.; Shang, H. Collective List-only Entity Linking: A Graph-based Approach. *IEEE Access* **2018**. [CrossRef]
19. He, Z.; Liu, S.; Li, M.; Zhou, M.; Zhang, L.; Wang, H. Learning Entity Representation for Entity Disambiguation. In Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, Sofia, Bulgaria, 4–9 August 2013; pp. 30–34.
20. Zwicklbauer, S.; Seifert, C.; Granitzer, M. Robust and Collective Entity Disambiguation through Semantic Embeddings. In Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval, Pisa, Italy, 17–21 July 2016; pp. 425–434. [CrossRef]
21. Fang, W.; Zhang, J.; Wang, D.; Chen, Z.; Li, M. Entity Disambiguation by Knowledge and Text Jointly Embedding. In Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, Berlin, Germany, 11–12 August 2016; pp. 260–269.
22. Yamada, I.; Shindo, H.; Takeda, H.; Takefuji, Y. Joint Learning of the Embedding of Words and Entities for Named Entity Disambiguation. In Proceedings of the SIGNLL, Berlin, Germany, 11–12 August 2016; pp. 250–259.
23. Gupta, N.; Singh, S.; Roth, D. Entity Linking via Joint Encoding of Types, Descriptions, and Context. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, Copenhagen, Denmark, 9–11 September 2017; pp. 2681–2690.
24. Nguyen, T.H.; Fauceglia, N.; Rodriguez-Muro, M.; Hassanzadeh, O.; Gliozzo, A.M.; Sadoghi, M. Joint Learning of Local and Global Features for Entity Linking via Neural Networks. In Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, Osaka, Japan, 11–16 December 2016; pp. 2310–2320.
25. Sun, Y.; Lin, L.; Tang, D.; Yang, N.; Ji, Z.; Wang, X. Modeling Mention, Context and Entity with Neural Networks for Entity Disambiguation. In Proceedings of Twenty-Fourth International Joint Conference on Artificial Intelligence, Buenos Aires, Argentina, 25–31 July 2015; pp. 1333–1339.
26. Francis-Landau, M.; Durrett, G.; Klein, D. Capturing Semantic Similarity for Entity Linking with Convolutional Neural Networks. In Proceedings of NAACL, San Diego California, USA, 12–17 June 2016; pp. 1256–1261.
27. Xu, W.; Yu, J. A novel approach to information fusion in multi-source datasets: A granular computing viewpoint. *Inf. Sci.* **2017**, *378*, 410–423. [CrossRef]
28. Han, X.; Zhao, J. Named entity disambiguation by leveraging wikipedia semantic knowledge. In Proceedings of the 18th ACM Conference on Information and Knowledge Management, Hong Kong, China, 2–6 November 2009; pp. 215–224. [CrossRef]
29. Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G.S.; Dean, J. Distributed Representations of Words and Phrases and their Compositionality. In Proceedings of the Advances in Neural Information Processing Systems, Lake Tahoe, NV, USA, 2013; pp. 3111–3119.
30. Wang, Y.; Huang, M.; Zhu, X.; Zhao, L. Attention-based LSTM for Aspect-level Sentiment Classification. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, Austin, TX, USA, 1–4 November 2016; pp. 606–615.

31. Tan, Z.; Zhao, X.; Wang, W. Representation Learning of Large-Scale Knowledge Graphs via Entity Feature Combinations. In Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, Singapore, 6–10 November 2017; pp. 1777–1786.
32. Ganea, O.; Ganea, M.; Lucchi, A.; Eickhoff, C.; Hofmann, T. Probabilistic Bag-Of-Hyperlinks Model for Entity Linking. In Proceedings of the 25th International Conference on World Wide Web, Montreal, QC, Canada, 11–15 April 2016; pp. 927–938. [CrossRef]
33. Rush, A.M.; Chopra, S.; Weston, J. A Neural Attention Model for Abstractive Sentence Summarization. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Lisbon, Portugal, 17–21 September 2015; pp. 379–389.
34. Steinmetz, N.; Sack, H. Semantic Multimedia Information Retrieval Based on Contextual Descriptions. In Proceedings of the Semantic Web: Semantics and Big Data, 10th International Conference, ESWC 2013, Montpellier, France, 26–30 May 2013; pp. 382–396.
35. Piccinno, F.; Ferragina, P. From TagME to WAT: A new entity annotator. In Proceedings of the First International Workshop on Entity Recognition & Disambiguation, Gold Coast, QL, Australia, 11 July 2014; pp. 55–62.
36. Mao, E.; Wang, B.; Tang, Y.; Liang, D. Entity linking method of chinese micro-blog based on word vector (in Chinese). *Comput. Appl. Softw.* **2017**, *5*, 75–84.
37. Xiang, Y.; Guo, Y.; Xu, X.; Zeng, W.; Li, L. Entity words disambiguation and entity linking with multi-strategy in chinese microblogs (in Chinese). *Comput. Appl. Softw.* **2016**, *4*, 12–17.
38. Wanli, C.; Hongying, Z.; Yonggang, W. Chinese Micro-blog Named Entity Linking Based on Multisource Knowledge (in Chinese). *J. Chin. Inf. Process.* **2015**, *1*, 115–121.
39. Chong, F.; Ge, S.; Yu-Hang, G.; Jing, G.; He-Yan, H. An Entity Linking Method for Microblog Based on Semantic Categorization by Word Embeddings (in Chinese). *Acta Autom. Sin.* **2016**, *42*, 915–922.
40. Ferragina, P.; Scaiella, U. Fast and Accurate Annotation of Short Texts with Wikipedia Pages. *IEEE Softw.* **2012**, *29*, 70–75. [CrossRef]
41. Pappu, A.; Blanco, R.; Mehdad, Y.; Stent, A.; Thadani, K. Lightweight Multilingual Entity Extraction and Linking. In Proceedings of the WSDM, Cambridge, United Kingdom, 6–10 February 2017; pp. 365–374.