High Quality Candidate Generation and Sequential Graph Attention Network for Entity Linking

Zheng Fang
Institute of Information Engineering, Chinese Academy of Sciences & University of Chinese Academy of Sciences
fangzheng@iie.ac.cn

Yanan Cao*
Institute of Information Engineering, Chinese Academy of Sciences
caoyanan@iie.ac.cn

Ren Li
Institute of Information Engineering, Chinese Academy of Sciences & University of Chinese Academy of Sciences
liren@iie.ac.cn

Zhenyu Zhang
Institute of Information Engineering, Chinese Academy of Sciences
zhangzhenyu1996@iie.ac.cn

Yanbing Liu
Institute of Information Engineering, Chinese Academy of Sciences
liuyanbing@iie.ac.cn

Shi Wang
Institute of Computing Technology, Chinese Academy of Sciences
wangshi@ict.ac.cn

ABSTRACT

Entity Linking (EL) is a task for mapping mentions in text to corresponding entities in knowledge base (KB). This task usually includes candidate generation (CG) and entity disambiguation (ED) stages. Recent EL systems based on neural network models have achieved good performance, but they still face two challenges: (i) Previous studies evaluate their models without considering the differences between candidate entities. In fact, the quality (gold recall in particular) of candidate sets has an effect on the EL results. So, how to promote the quality of candidates needs more attention. (ii) In order to utilize the topical coherence among the referred entities, many graph and sequence models are proposed for collective ED. However, graph-based models treat all candidate entities equally which may introduce much noise information. On the contrary, sequence models can only observe previous referred entities, ignoring the relevance between the current mention and its subsequent entities. To address the first problem, we propose a multi-strategy based CG method to generate high recall candidate sets. For the second problem, we design a Sequential Graph Attention Network (SeqGAT) which combines the advantages of graph and sequence methods. In our model, mentions are dealt with in a sequence manner. Given the current mention, SeqGAT dynamically encodes both its previous referred entities and subsequent ones, and assign different importance to these entities. In this way, it not only makes full use of the topical consistency, but also reduce noise interference. We conduct experiments on different types of datasets and compare our method with previous EL system on the open evaluation platform. The comparison results show that our model achieves significant improvements over the state-of-the-art methods.

CCS CONCEPTS

• **Information systems** → **Information extraction**;

KEYWORDS

Entity linking, BERT, graph attention network, candidate generation

ACM Reference Format:
Zheng Fang, Yanan Cao, Ren Li, Zhenyu Zhang, Yanbing Liu, and Shi Wang. 2020. High Quality Candidate Generation and Sequential Graph Attention Network for Entity Linking. In *Proceedings of The Web Conference 2020 (WWW '20), April 20–24, 2020, Taipei, Taiwan.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3366423.3380146

1 INTRODUCTION

Entity Linking (EL) is the task which aims at aligning mentions in natural-language text to corresponding entities in a Knowledge Base (KB). This task is challenging because mentions are usually ambiguous. For example, different named entities may share the same surface form and the same entity may have multiple aliases. EL alleviates this problem by bridging unstructured text and structured KB. It is a fundamental task in the field of natural language processing (NLP), which can facilitate many other tasks, such as semantic search, question answering, information integration, and text understanding.

Typically, the entity linking task consists of three stages [36]: Mention Detection (MD), Candidate Generation (CG), and Entity Disambiguation (ED). Because MD is usually studied as an independent task and there are many publicly available MD tools such as Stanford NER[1] and OpenNLP[2], this paper focuses on the latter two stages. In the CG stage, EL system aims to retrieve a candidate set which contains possible entities that mention may refer to. To achieve this goal, search engines and name dictionaries are widely used in previous systems [25, 31, 36]. In the ED stage, many state-of-the-art models [8, 10, 42] prefer to combine local and global contextual information to disambiguate mentions: local information based on words that occur in mention context window is employed

[*] Corresponding author.

WWW '20, April 20–24, 2020, Taipei, Taiwan

ACM ISBN 978-1-4503-7023-3/20/04.
https://doi.org/10.1145/3366423.3380146

[1] https://nlp.stanford.edu/ner/
[2] https://opennlp.apache.org/

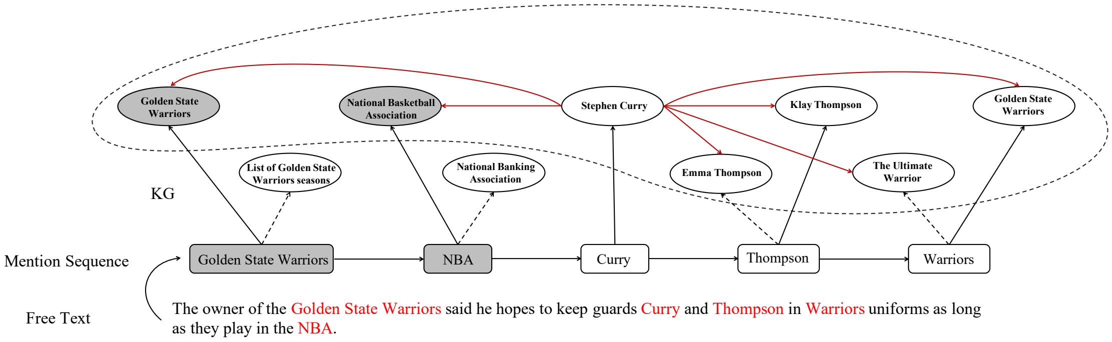

Golden State Warriors
National Basketball Association
Stephen Curry
Klay Thompson
Golden State Warriors
List of Golden State Warriors seasons
National Banking Association
Emma Thompson
The Ultimate Warrior
KG
Mention Sequence
Golden State Warriors
NBA
Curry
Thompson
Warriors
Free Text
The owner of the Golden State Warriors said he hopes to keep guards Curry and Thompson in Warriors uniforms as long as they play in the NBA.

Figure 1: An illustration of disambiguating mentions in text. We first utilize local features to rank mentions, and then deal with them in a sequence manner. To select the target entity for "Curry", we construct a graph for it. Solid black lines point to the correct target entities corresponding to the mentions and gray nodes represent previously referred entities and disambiguated mentions.

to capture the textual and semantical features, and global information that reflects the topical coherence among the referred entities is utilized to model the interdependence between mentions. Although previous EL models have achieved significant improvements, there are still many problems in the above two stages.

For the CG stage, the quality of the candidate set is often ignored by previous studies. As the final results in EL are only generated from candidate sets and the corresponding gold recall is an upper bound on ED accuracy, a good CG module with high gold recall is very important for EL system. After analyzing the candidate sets constructed by [1, 10, 17, 42], we find that the gold recall of some candidate sets is only about 90%, which may make a large number of mentions unable to refer to the right entities. Moreover, the semantic relevance, the number and the page view of candidates may also have effect on EL result. However, previous works never evaluate the CG methods from these respects. So, how to generate a high quality candidate set and evaluate them is the first problem we need to solve in EL.

For the ED stage, many graph-based collective entity linking methods [1, 11, 12, 42] are proposed to capture topic consistency between mentions in text. But there are a large number of non-target candidates in their graphs, which may introduce noise when performing collective disambiguation. To address this problem, some recent works [8, 23] rank mentions based on their ambiguity degree and deal with them in a sequence manner. According to the study, starting with mentions that are easier to disambiguate and utilizing information provided by previously referred entities will be effective to reduce the interference of noise data. However, these sequential models can only observe previously referred entities, without considering the subsequent ones which may also provide useful clues for disambiguation. For example, in Figure1, except from previously referred entity "Golden State Warriors" and "National Basketball Association", the subsequent candidates "klay Thompson", "Golden State Warriors" can also provide useful information for disambiguating "Curry". So, how to combine the advantages of graph and sequence models to make full use of the topical consistency and reduce noise interference is the second problem that needs to be solved.

This paper aims to address these two issues mentioned above. In the CG stage, to improve the gold recall and the quality of candidate set, we retrieve candidate entities from three aspects: (i) retrieving candidates with a similar surface form of the mentions, such as the mention's abbreviations, alias name, full name and other variants, by utilizing online dictionary and Wikipedia Search Engine. (ii) retrieving semantic relevant candidates by expanding the queries with the mention's context keywords or other relevant mentions. (iii) recalling candidates using Google Search Engine when above two mechanisms do not work well. These approaches are complementary. For example, when one mention is mis-spelled or it consists of multiple words, it is difficult to directly retrieve candidates from the dictionary and Wikipedia, while the Google Search Engine could effectively recall more candidates. In experiments, we propose five metrics to evaluate different candidate sets on four datasets, and results show that the average recall of our CG method achieves 98%.

In the ED stage, to make better use of topic consistency among mentions, we propose a novel Sequential Graph Attention Network (SeqGAT) to encode the global information of entities. In our model, the mentions are disambiguated in sequence according to their ambiguity degree. Unlike traditional graph models which encode all relations between entities and their candidates, and unlike sequential models which just encodes previously referred entities, our model dynamically changes the input nodes and relations according to the current state. For each mention, except for its candidates, the previously referred entities and the subsequent mention's candidates are also encoded by the graph model. In this way, our model not only utilizes more related entities, but also reduces the noisy information. Besides, in order to pay more attention to the previous and subsequent target entities, we use attention mechanism to assign different importance to different adjacent nodes and capture the relevance between the target candidates. This mechanism could avoid the error propagation to some extent if the previous decision is wrong. Using SeqGAT as the global encoder, our model SGEL (SeqGAT based EL model) combines the local information, global information and statistical features of mentions to conduct collective decisions. In particular, we use Bidirectional Encoder Representation from Transformers (BERT) as the local encoder, which can well learn the semantic-level language representation, to encode the mention context and entity description. We conduct comparative experiments with previous EL systems and experimental results show the effectiveness of our model.

In summary, the main contributions of this paper are listed as follows:

- We propose a high quality candidate generation method and propose novel metrics to evaluate the candidate sets. In particular, we provide an easy-to-use API to facilitate the application of CG module to other datasets.
- We propose a novel SGEL framework which combine the advantages of graph models and sequence ones. Specifically, we utilize BERT to extract local features and introduce sequential GAT to capture the topical coherence of mentions.
- We evaluate our model on 8 different datasets and compare SGEL with other EL system on the open evaluation platform. The results show that our model achieves significant improvements over the state-of-the-art methods.

2 PRELIMINARIES

Formally, given a knowledge base containing a set of entities $E = \{e_1, e_2, ..., e_n\}$ and a text collection in which a set of mentions $M = \{m_1, m_2, ..., m_q\}$ are identified in advance, the task of entity linking is to find a mapping $M \mapsto E$ that links each mention to a target entity in the KB. Because there are too many entities in E, it is necessary to filter out irrelevant ones and retrieve a candidate set $C_{m_i} = \{e_1, e_2, ..., e_k\}$ which contains possible entities that mention m_i may refer to. After generating corresponding candidates C_{m_i}, the disambiguation model will be used to find the the most likely entity $e_t \in C_{m_i}$ for m_i. Specially, "NIL" will be returned if there is no correct entity in the knowledge base. Similar to most recent studies [8, 10, 25], we do not address the issue of unlinkable mentions in our paper.

2.1 Candidate Generation

Before entity disambiguation, candidate entities which related to mention will be retrieved from the knowledge base. As the final results in EL are only generated from candidate sets, we will recall candidate entities as comprehensively as possible to ensure that the target entity can appear in the candidate set. To achieve this goal, we construct candidate sets from following three aspects:

2.1.1 The surface form of mention. We first utilize Wikipedia dictionary to recall entities that have a similar form with mention. Different from the previous systems [1, 8–10, 17, 42] which use static dictionaries [13, 34], our CG module employs a online Wikipedia dictionary which is able to recall new and long-tailed entities. Specifically, we utilize exact and partial matching (i.e., entity name that shares several common words with the mention, entity name exactly matches the first letters of all words in the mention, etc.) between the entity name and the mention's surface form to recall candidates. Like previous systems [8, 25], entities in mention's redirect and disambiguation pages are also recalled by our CG module. Moreover, [32, 39] have found that entity description in Wikipedia article also provides useful source of alias names and variations of the pointed entity. Inspired by them, we exploit Wikipedia Search Engine to retrieve entities whose articles contain mention many times. Except that, we also consider the information in local document. Specifically, if current mention is an abbreviation or substring of other mentions in the same document, we merge the candidate sets of the original and extended mentions. For example, in Figure 2, "Golden State Warriors" contains mention "Warriors" as a substring, then we consider "Golden State Warriors" as an expanded form of "Warriors", and the candidates of "Golden State Warriors" will be also added to the candidate set of "Warriors".

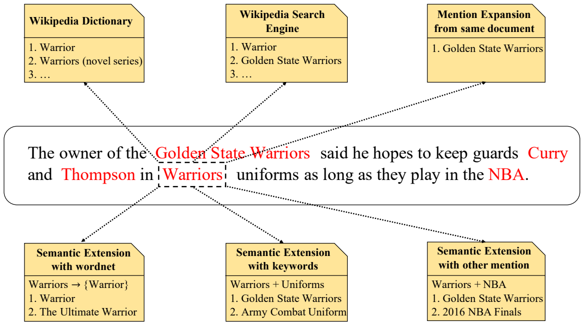

Wikipedia Dictionary
1. Warrior
2. Warriors (novel series)
3. …
Wikipedia Search Engine
1. Warrior
2. Golden State Warriors
3. …
Mention Expansion from same document
1. Golden State Warriors
The owner of the Golden State Warriors said he hopes to keep guards Curry and Thompson in Warriors uniforms as long as they play in the NBA.
Semantic Extension with wordnet
Warriors → {Warrior}
1. Warrior
2. The Ultimate Warrior
Semantic Extension with keywords
Warriors + Uniforms
1. Golden State Warriors
2. Army Combat Uniform
Semantic Extension with other mention
Warriors + NBA
1. Golden State Warriors
2. 2016 NBA Finals

Figure 2: Generating candidate entities corresponding to mention "Warriors" from multiple aspects.

2.1.2 The semantic extension. Except from utilizing the surface form of mention, we also retrieve semantic relevant candidates by expanding the mention query with other information. At first, we utilize WordNet [20] which labels the semantic relations among words to obtain synonyms of mention. Then the candidate entities corresponding to synonyms will be added to the current mention's candidate set. To ensure that candidate entities are context-dependent, the context keywords and other mentions in the same document are also used to recall candidate entities. On the one hand, we input both local keywords and mention into the Wikipedia Search Engine. In this way, the entity related to local context will be retrieved. On the other hand, we submit current mention together with another adjacent mention to the Wikipedia Search Engine, which can exploit co-occurrence information to recall entities.

2.1.3 The exception handling. In practice, we notice that neither of the above two methods can recall entities when mention names are misspelled. In addition, when a mention is composed of multiple words, the above methods can not effectively recall the target entity. Therefore, we additionally use Google Search Engine to recall candidate entities when the mention name contains more than three words or the candidate set constructed by the above methods is empty. Specifically, we submit the mention together with "Wikipedia" to the Google API and obtain identified candidate entities whose Wikipedia pages appear in the top 10 Google search results.

2.2 Candidate Pruning

After generating the original candidate sets, we hope to reduce the number of noise candidates and optimize for memory and running time of ED module. Consequently, we further prune unrelated

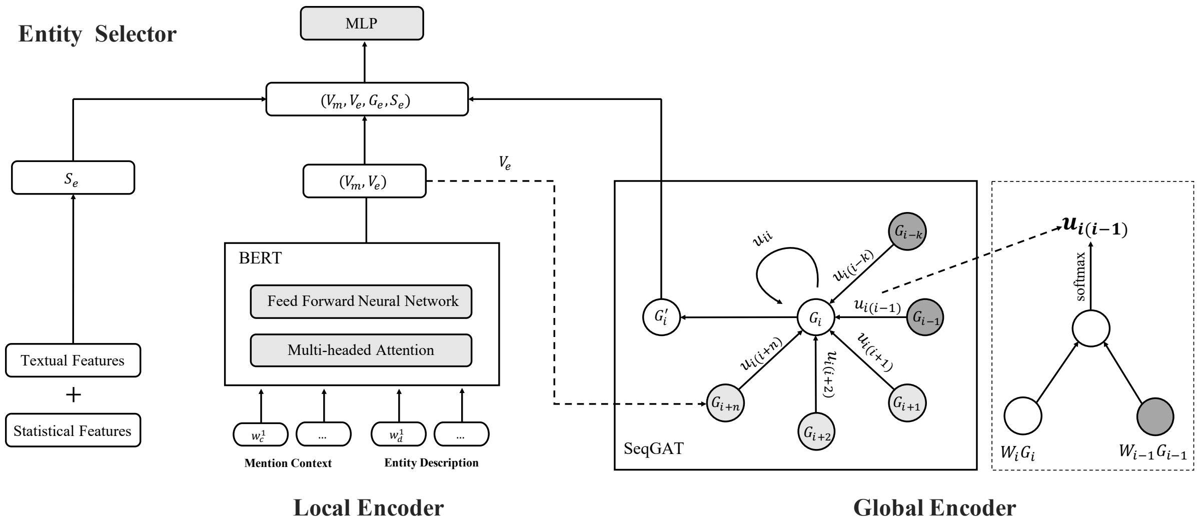


Figure 3: The overall structure of our SGEL model. It contains three parts: Local Encoder with BERT, Global Encoder with SeqGAT, Entity Selector with multiple information. Specifically, in Local Encoder, V_m and V_e denote the mention context vector, entity description vector respectively. In Global Encoder, $G_i^{'}$ represents the global vector of e_i, $u_{i(i-1)}$ indicates the importance of e_{i-1} to e_i. In Entity Selector, S_e is the manual features extracted from mention and entity. Dark grey and light grey nodes represent the previously referred entities and subsequent mention's candidate, respectively.

candidates and retain entities that are highly relevant to mention. To ensure that pruning strategies are as simple as possible, we use the following statistical and textual features to rank mention-to-candidate pairs. For convenient using, we concatenate these features as a vector S_e and input it into a ranking model. Similar to previous works [8, 25, 26], a Gradient Boosted Tree model (XGBoost) [3] is trained as candidate ranker to reduce the size of candidate set. Finally, these pruning heuristics result in a significantly improved running time at a slight accuracy loss.

- Textual Features. We use several string similarity metrics in our model, which includes (1) Levenshtein ratio and jaro distance between the mention's form and entity name; (2) the number of identical words between the mention's form and entity name; (3) the number of identical words between the mention context and the entity name. (4) the number of identical words between the mention context and the entity category. (5) whether mention is an abbreviation of the entity name;
- Statistical Features. We firstly use the page views of the candidate entity as the prior feature. Like [8], these values are downloaded from the Wikipedia Tool Labs[3] which count the number of visits on each entity page in Wikipedia. Because the ranking result of Wikipedia search engine reflects the prior correlation between the candidate and mention, we also add the position of entity in search results to prior features.

3 ENTITY DISAMBIGUATION

Our entity disambiguation system mainly includes three modules: Local Encoder with BERT, Global Encoder with sequential GAT, Entity Selector with multiple information. More concretely, the Local Encoder utilize BERT to encode mention context and entity description. According to BERT results, the semantic relevance between mention and candidate entity can be effectively identified. In the Global Encoder, the global interdependence between mentions will be captured by a graph attention network. In this process, mentions will be ranked based on the local features and disambiguated in a sequential way. In the Entity Selector, some lexical and statistical evidences will be combined with local and global representations to select the target entity. The overall structure of our ED model is shown in Figure 3.

3.1 Local Encoder with BERT

For each mention, we use the complete sentence in which it is located as the context. As for the candidate entities, we select the summary keywords in corresponding Wikipedia pages as their description. To alleviate the long-term dependency problem and extract the worthy local information, we use BERT which composed of deep bidirectional Transformer [40] to encode mention context and entity description.

BERT is a multi-layer bidirectional Transformer encoder, which is pre-trained with two strategies on large-scale unlabeled text, i.e., masked language model and next sentence prediction. Based on the BERT structure, it is able to unambiguously represent both a single text sentence and a pair of text sentences. In order to obtain the local representations of mention and entity, and capture the

[3] The url of the website is: https://tools.wmflabs.org/pageviews/

semantic relevance between them, we fine-tune the BERT and apply it to EL task.

During the fine-tuning process, the mention context T_m and entity description T_e are packed together into a single sequence. To get their respective encodings at BERT's output layer, we record their lengths in advance. Based on the original structure of BERT, we introduce a fully-connected layer over the final hidden state to calculate the local similarity between Tm and Te. For model fitting, we utilize a max-margin loss that ranks ground truth entities higher than other candidate entities. The loss function is defined as follows:

$$L_{local} = max(0, \gamma - \rho(m_i, e_i^+) + \rho(m_i, e_i^-)) \tag{1}$$

where $\rho(m_i, e_i)$ represents the local similarity between m_i and its candidate e_i, $\gamma > 0$ is a margin parameter. We aim to find a function ρ such that the score of the positive target entity e_i^+ is at least a margin γ higher than that of negative candidate entity e_i^-.

After fine-tuning the BERT, we obtain the mention context vector V_m and entity description vector V_e via average-pooling over the hidden states of the corresponding tokens in the last BERT layer. These vectors will be used as input of the graph attention network and Entity Selector. In addition, the similarity scores calculated by fully-connected layer will be utilized for ranking mentions in the next section.

3.2 Global Encoder with Sequential GAT

In this section, we aim to capture the interdependence among mentions and obtain global representation for each candidate entity. To achieve this goal, we construct a graph and apply a Sequential GAT (SeqGAT) to encode entities. The graph structure enables our model to make better use of consistency among target entities, while sequential operation allows us to make full use of previously decisions.

3.2.1 Ranking Mention. In the SeqGAT model, candidate entities are input in a fixed order. As illustrated in the [8, 12], mentions in text usually have different ambiguity degrees according to the quality of contextual information and prior knowledge. In order to start with mentions that are easier to disambiguate and gain correct results, we rank mentions based on manual features and semantic representations. As stated in section 2.2 and 3.1, XGBoost prunes candidates according to statistical and textual features, while BERT distinguishes candidates based on the semantic information. Combining results of the above two aspects, we rank mentions as follows:

$$\Phi(m_i, e_i^*) = max(\frac{1}{R_{e_i}^b} \cdot \frac{1}{R_{e_i}^x}) \tag{2}$$

where $R_{e_i}^b$ and $R_{e_i}^x$ represent the ranking position of entity e_i in BERT and XGBoost respectively, e_i^* denotes the entity with the highest overall score. According to the value of $\Phi(m_i, e_i^*)$, m_i will be placed before m_j in the sequence if $\Phi(m_i, e_i^*) > \Phi(m_j, e_j^*)$.

3.2.2 Building Sequential Graph Network. After sorting mentions, we encode their candidate entities in a sequential way. Particularly, we construct a subgraph $\mathcal{G}_{m_i}$ for the i-th mention m_i. The formulation of $\mathcal{G}_{m_i}$ is:

$$\mathcal{G}_{m_i} = (\mathcal{V}_{m_i}, \mathcal{E}_{m_i}) \tag{3}$$

$$\begin{aligned}\mathcal{V}_{m_i} = \{e \mid e \in \{e_p^*, e_i, e_q\},\ e_p^* \in C_{m_p},\ e_i \in C_{m_i},\\ e_q \in C_{m_q}, m_p \in \mathcal{S}_{[1:i-1]}, m_q \in \mathcal{S}_{[i+1:n]}\}\end{aligned} \tag{4}$$

$$\begin{aligned}\mathcal{E}_{m_i} = \{(e_i, e_j) \cup (e_i, e_i) \mid e_j \in \{e_p^*, e_q\},\ e_i \in C_{m_i},\\ e_p^* \in C_{m_p}, e_q \in C_{m_q}, m_p \in \mathcal{S}_{[1:i-1]}, m_q \in \mathcal{S}_{[i+1:n]}\}\end{aligned} \tag{5}$$

$$\mathcal{S}_{[1:n]} = \{m_1, m_2, m_3 ..., m_{n-1}, m_n\} \tag{6}$$

where $\mathcal{V}_{m_i}$ stands for the set of nodes in $\mathcal{G}_{m_i}$, and $\mathcal{E}_{m_i}$ represents edges between nodes. n is the number of mentions in the sequence $\mathcal{S}$. According to the position of current mention, the sequence $\mathcal{S}$ is divided into two parts: $\mathcal{S}_{[1:i-1]}$ and $\mathcal{S}_{[i+1:n]}$. The $\mathcal{S}_{[1:i-1]}$ denotes a set of mentions that have been disambiguated, and mention in $\mathcal{S}_{[i+1:n]}$ is after m_i. The nodes in the $\mathcal{G}_{m_i}$ include three types: the previous referred entity e_p^*, current candidate entity e_i, and subsequent mention's candidate entity e_q. In order to reduce the interference of noise data and get the clues of subsequent target entities, we define three types of edges between pairs of nodes: (i) an edge between previously referred entity node and current candidate entity; (ii) an edge between subsequent mention's candidate and current candidate entity; (iii) self-connection. In this way, our model can make use of previously decisions and keep the topic consistent with subsequent mentions. Specially, to address the cold start problem, we choose the target entity e_1^* of the first mention m_1 based on BERT and XGBoost results. Moreover, to prevent previous error from propagating backwards, we use a graph attention network which can selectively use of previous information.

3.2.3 Graph Attention Network. Because the previous decisions in the sequence may be wrong and the number of non-target candidates is larger than that of target ones in the graph, we use a graph attention network which can focus on the correlation between the target entities. GAT [41] is a novel convolution-style neural networks that operate on graph-structured data, leveraging masked self-attention layers. With the help of graph attention layer, we can (implicitly) assign different importances to different adjacent nodes and capture the relevance between the target candidates.

The input to our network is a set of node features, which are represented as follows:

$$G^1 = \{V_e \mid e \in \mathcal{V}_{m_i}\} \tag{7}$$

where $\{V_e \in \mathbb{R}^F\}$ is the representation of each node, which is generated by the Local Encoder. Given the input vector of each node and the adjacent matrix A of graph, we use graph convolution network to extract features from the hidden state of the entity as well as that of its neighbors. The spectral graph convolution define as [15]:

$$G^{(l+1)} = \rho(AG^l W^l) \tag{8}$$

where G^l and W^l are the hidden states and weights in the l-th layer. And ρ is a non-linear activation function. To distinguish the relevance between different entities, we introduce a self-attention layer to GCN and get the graph attention network (GAT). In GAT, the attention coefficients are computed as follows:

$$u_{ij} = a(W_i^l G_i^l, W_j^l G_j^l) \tag{9}$$

where G_i^l and G_j^l are the hidden states of node i and j in the l-th layer. a is a single-layer feedforward neural network and u_{ij} indicates the importance of node j's features to node i. It should be emphasized that we only compute u_{ij} for nodes $j \in \mathcal{N}_i$, where $\mathcal{N}_i$ is the neighborhood of node i. As described in section 3.2.2, the $\mathcal{N}_{e_i}$ includes three parts: previously referred entities, candidates for subsequent mentions and itself. In addition, to make coefficients easily comparable across different nodes, we normalize them using a softmax function:

$$a_{ij} = softmax_j(u_{ij}) = \frac{exp(u_{ij})}{\sum_{k \in \mathcal{N}_i} exp(u_{ik})} \tag{10}$$

Next, the normalized attention coefficients will be used to assign different importances to nodes of neighborhood. And the attention layer output is defined as:

$$G_i^{'} = \rho\left(\sum_{j \in \mathcal{N}_i} a_{ij} W_j^l G_j^l\right) \tag{11}$$

where $G_i^{'}$ is the updated representation of node i and ρ is a nonlinearity activation function, such as Relu.

Before using graph attention network to generate the global representation for each entity, we pre-train the GAT using a node classification task. During the training process, the model takes a graph as an input and output labels for each node. Particularly, we add a multi-layer perceptron (MLP) after the last self-attention layer, and transform the G_i into a class vector. The parameters of network are trained to minimize cross-entropy of the predicted and ground truth:

$$L_{global} = -\sum_{i=1}^{n} y \log(P(y^{'} = e_i)) \tag{12}$$

Where $y \in \{0, 1\}$ represents the real label of the candidate entity and $y^{'} \in (0, 1)$ indicates the predicted result of our model. When y equals 1, the corresponding candidate is correct; otherwise, the candidate is not the target entity. After pre-training the graph attention network, we input G_i as the global representation of entity e_i into the Entity Selector module.

3.3 Entity Selector with Multiple Information

Aiming at combining the global interdependence between EL decisions with the local mention-to-entity compatibility, we propose an Entity Selector to collectively disambiguate mentions. For each mention m_i and its candidate entity e_i, we generate the input vector as follows:

$$V_{(m,e)} = V_{m_i} \oplus V_{e_i} \oplus G_{e_i} \oplus S_{(m_i,e_i)} \tag{13}$$

where $\oplus$ indicates vector concatenation. V_{m_i} and V_{e_i} respectively denote the local vector of m_i and e_i, G_{e_i} is the global representation output by GAT. Since $V_{m_i}, V_{e_i}, G_{e_i}$ mainly represent semantic relevance between m_i and e_i, we add vector S_{e_i} to enrich lexical and statistical evidence. Specifically, the S_{e_i} consists of features described in section 2.2. To make full use of prior knowledge, we extend S_{e_i} to the same dimension as V_{m_i}. Then we feed the concatenated vectors into a multi-layer perceptron:

$$h^l = Relu(W^l h^{l-1} + b^l) \tag{14}$$

where W^l and b^l are trainable parameters and bias, and h^{l-1} is the output of (l-1)th hidden layer. As before, we use a margin based objective to fit parameters and define a ranking loss:

$$L_{fusion} = \sum_{D \in \mathcal{D}} \sum_{m_i \in D} \sum_{e_i \in C_{m_i}} \Psi(m_i, e_i) \tag{15}$$

$$\Psi(m_i, e_i) = max(0, \gamma - \phi(m_i, e_i^+) + \phi(m_i, e_i^-)) \tag{16}$$

$$\phi(m_i, e_i) = softmax(h^l) \tag{17}$$

where $\mathcal{D}$ is the training set of entity annotated documents, and D denotes the document where m_i is located. Similar to Equation 1, $\phi(m_i, e_i)$ represents the comprehensive correlation between m_i and e_i. In particular, h^l is the output of last layer of MLP, and its dimension is 2. At test time, the candidate entity with the maximal probability $\Psi(m_i, e_i)$ will be chosen as target entity. The details of the overall training process of our model are presented in Algorithm 1.

Algorithm 1 Sequential Graph Model for EL

```
Input: Training set of entity annotated documents $\mathcal{D} = \{D_1, D_2, ..., D_X\}$
Output: The target entities $\Gamma = \{e_1, e_2, ..., e_N\}$ for mentions
Fine-tune the BERT;
Pre-train the GAT;
for $D$ in $\mathcal{D}$ do
    Detect all mentions $M = \{m_1, m_2, ..., m_q\}$ in $D$;
    Generate the candidate set $C_{m_i} = \{e_1, e_2, ..., e_k\}$ for each mention $m_i \in M$;
    Construct statistical features $S_e$ and input them into the XGBoost model to prune candidate set;
    Obtain mention context vector $V_m$ and entity vector $V_e$ via BERT;
    Rank $M$ based on the results of XGBoost and BERT;
    $i \leftarrow 0$;
    while $i < len(M)$ do
        if $i == 0$ then
            Select the target entity for $m_t$ based on XGBoost and BERT results;
        else
            Build subgraph $\mathcal{G}_{m_i}$ for $m_i$;
            Get the global representation $G_{e_i}$ of each entity in $C_{m_i}$ through SeqGAT;
            Input $V_{m_i}$, $V_{e_i}$, $G_{e_i}$, $S_{e_i}$ to the Entity Selector, and choose the target entity $e_i^*$ for $m_i$ with $\Theta$;
            Update the parameter $\Theta$ in the Entity Selector;
        end if
        $i \leftarrow i + 1$
    end while
end for
```

4 EXPERIMENT

To evaluate our model, we conduct experiments on a series of popular EL datasets which are also used by [1, 8, 9, 12, 18]. For the purpose of making a unified comparison among different EL systems, we utilize a public evaluation platform named Gerbil [38]

Table 1: The statistical results on experimental datasets. $|D|$, $|M|$ are number of documents, number of mentions, respectively.

Datasets	**Type**	$\|D\|$	$\|M\|$	**Gold Recall**
Wiki-Clueweb	wiki	639	16948	-
AIDA-A	news	216	4791	98.6%
AIDA-B	news	231	4485	98.3%
MSNBC	news	20	747	98.7%
ACE2004	news	57	257	97.7%
Reuters128	news	128	634	97.5%
AQUAINT	news	50	727	97.4%
KORE50	short sentences	50	144	97.2%
RSS500	RSS-feeds	500	523	92.9%

Table 2: Comparisons of candidate sets in our system and in [1, 10, 17, 42]. The top of each line is the result of the previous system (YCCS), and the bottom is our result.

Feature		AIDA-A	AIDA-B	AQUAINT	ACE2004
Gold Recall	YCCS	96.9%	98.2%	94.2%	90.6%
	Our	**98.6%**	**98.3%**	**97.4%**	**97.7%**
Avg Size	YCCS	51.7	53.4	**32.9**	47.9
	Our	**48.4**	**49.5**	45.9	**46.8**
Avg Name Dis	YCCS	0.64	0.64	0.63	0.60
	Our	**0.66**	**0.65**	**0.65**	**0.62**
Avg Sem Sim	YCCS	0.17	0.17	0.15	0.17
	Our	**0.19**	**0.20**	**0.17**	**0.18**
Avg PV(million)	YCCS	0.74	0.69	0.93	0.83
	Our	**2.26**	**1.34**	**1.97**	**1.30**

(version 1.2.7). The results show that our model achieves the state-of-the-art performance. Our source code and data can be found in: https://github.com/fangzheng123/SGEL.

4.1 Experiment Setup

4.1.1 Datasets. We train and evaluate our model on well-known datasets from different domains, including short and long text, formal and informal text. The statistics of the datasets are shown in Table 1. Similar to previous systems [1, 9, 12, 16, 27], we only count the number of mentions with the target entities in the KB. The datasets used in our system are described as follows:

- **Wiki-Clueweb** [12] is built from the ClueWeb and Wikipedia corpora, we remove part of the noise data and select 16948 mentions for training.
- **AIDA-CoNLL** [13] is an entity annotated corpus of Reuters news documents. It contains 1393 documents, much larger than other EL datasets. This data is divided into three parts: AIDA-Train for training, AIDA-A for validation and AIDA-B for testing.
- **AQUAINT** [21] contains news from the Xinhua News Service, the New York Times and the Associated Press.
- **ACE2004** [30] is a subset of ACE2004 co-reference documents annotated by Amazon Mechanical Turk.
- **MSNBC** [6] contains 20 news articles from 10 different topics (two articles per topic).
- **Reuters128** [38] contains 128 economic news articles taken from the Reuters-21587 corpus.
- **KORE50** [38] contains 50 short sentences, and most of mentions in which are first names of person.
- **RSS500** [38] contains feed text collected from a wide range of topics e.g., world, business, science, etc.

4.1.2 Training Details. Here, we mainly investigate systems that use Wiki data as training set and report results on Gerbil platform. To make a fair comparison with them, we train our model on Wiki-Clueweb, and evaluate it on other 8 popular test sets. In our experiment, we use a fine-tuned BERT to encode mention context and entity description. During the fine-tuning process, we adopt the pre-trained uncased BERT-Base model and train the model for 140,000 steps on same machine with batch size of 3. Following the recommended settings in the BERT code[4], we set a maximum sentence length of 128 tokens and total length of 256 tokens. The dimension of hidden representations is set to 768, and Adam with $\beta_1 = 0.9$, $\beta_2 = 0.999$ is utilized as optimizer. To get global representation for each candidate entity, we apply a two-layer GAT model. The first layer consists of 8 attention heads and outputs 256-dimension vectors for each node. In the initial stage, each node is represented by a 768 dimensional vector output by the last hidden layer of BERT. The second layer is used for classification: a single attention head that computes 2 features (where 2 is the number of classes), followed by a softmax activation. In the Entity Selector, the number of MLP layers is 3 and the learning rate is 1e-3. In addition, the rank margin γ is set to 0.3 to distinguish the positive from negative entities. All above modules are trained on Nvidia Tesla V100 GPU and implemented in the Tensorflow framework.

4.2 Analysis of Candidate Entities

As the CG results have a significant impact on ED task, we analyze the quality of our candidate entities and compare them with previous systems. Due to the limited number of open candidate sets, we mainly analyze a popular candidate set which constructed by [10] and used by [1, 17, 42]. According to our investigation, the candidate entities in their systems are mainly retrieved from YAGO dictionary [13] and Cross-Wiki dictionary [34] (YCCS). The former dictionary[5] is built from entities in YAGO [35] knowledge base, while the latter establishes a general mapping from strings to Wikipedia entities. In particular, the Cross-Wiki dictionary[6] is no longer available. Table 2 lists the statistical results of candidate sets in the following aspects:

- **Gold Recall** is the percentage of mentions where the gold entity is included in the candidate set.

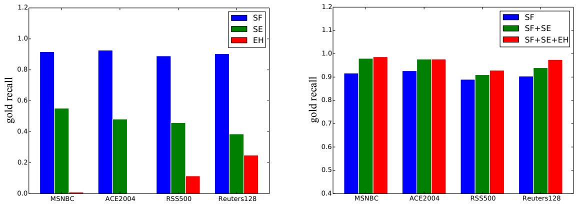
[4] https://github.com/google-research/bert

[5] https://www.mpi-inf.mpg.de/departments/databases-and-information-systems/research/ambiverse-nlu/aida/downloads/—aida_means.tsv.bz2

[6] http://downloads.cs.stanford.edu/nlp/data/crosswikis-data.tar.bz2/

Table 3: The Micro F1 results on the Gerbil platform. Particularly, we highlight the best score in bold for each dataset.

System	AIDA-A	AIDA-B	AQUAINT	ACE2004	MSNBC	KORE50	RSS500	Reuters128	Avg (micro)
AIDA [13]	0.74	0.77	0.57	0.80	0.74	0.69	0.66	0.57	0.73
Babelfy [22]	0.74	0.76	0.70	0.61	0.76	**0.74**	0.64	0.55	0.73
WAT [28]	0.78	0.80	0.75	0.76	0.79	0.62	0.64	0.63	0.77
xLisa [43]	0.52	0.54	0.79	0.81	0.55	0.51	0.65	0.49	0.55
PBoH [9]	0.80	0.80	0.84	0.79	0.86	0.63	0.55	0.69	0.79
WNED [12]	0.79	0.79	0.83	0.81	**0.89**	0.56	0.65	0.62	0.78
NCEL [1]	0.79	0.80	0.87	0.88	-	-	-	-	0.80
Our SGEL	**0.85**	**0.83**	**0.88**	**0.89**	0.80	0.68	**0.68**	**0.71**	**0.83**



a. The gold recall in each part.

b. The gold recall of multiple part combinations .

Figure 4: The statistical results of three candidate generation methods. SF, SE, EH denotes surface form of mention, semantic extension and exception handling respectively.

- **Avg Size** is the average size of candidate sets. According to the previous studies [25], taking more candidates per mention will degrade disambiguation accuracy in ED stage.
- **Avg Name Dis** is the average Jaro distance [14] between entity name and mention's surface form. It reflects the string similarity between mention and candidate entities.
- **Avg Sem Sim** is the average cosine distance between mention context vector and entity description vector. Concretely, we take the average of all word vectors as the representation of sentence.
- **Avg PV (million)** is the average page views (PV) of candidate entities in Wikipedia. Specifically, PV denotes the number of times an entity page has been visited.

Generally, a good candidate set should have two characteristics: high gold recall and strong correlation between mention and candidate entities. For gold recall, we can see that our CG module achieves best performance on all four datasets, and make significant improvements on AUQINT and ACE2004. To evaluate the correlation between mention and candidates, we calculate the text and semantic similarities between them. The results show that our candidate entities are more relevant to corresponding mentions. Moreover, with the number of candidates close to us, the average PV of previous candidate set is only half that of ours. This means that many popular mention-related entities have not been recalled by prior systems.

In our system, the candidate sets are built from three aspects. To investigate the contribution of each part, we calculate the corresponding gold recall, and show the statistical results in Figure 4. We can observe that mention-based method can recall about 90% of gold entities in most datasets, and 50% of gold entities can be retrieved by adding local and global information. For RSS500 and Reuters128 datasets, Google search engine can effectively handle special mentions and recall corresponding target entities. According to the final gold recall rate, the second and third CG results can effectively complement the mention-based method and improve the recall rate by nearly 6%.

Before using entity disambiguation algorithm to select the target entity, we select top k candidate entities for each mention to reduce the number of noise data. By defining R_g to represent the recall of gold entity in test dataset, the results go as follows: when k is set to 3, 5, 8, 10, R_g is 93.4%, 96.7%, 97.4%, 97.7% respectively. Generally, taking fewer candidates per mention will lead to low recall while using more candidates degrades disambiguation accuracy in ED stage. To make a good tradeoff between recall and accuracy, we retain the top 5 candidate entities as the input of our ED module.

4.3 Results on Gerbil

Gerbil [38] is an open-source and extensible framework that allows users to evaluate their tools against other entity annotation systems by using exactly the same setting. For EL task, it provides a unified comparison among different methods across multiple datasets. Here, we compare our model with other EL systems that report the performance on Gerbil.

4.3.1 Evaluation Metric. Currently, Gerbil offers six measures and subdivides them into two groups, namely the micro- and the macro-group of precision, recall and F1-measure. The micro measures show the performance over the set of all annotations inside the dataset while macro measures show the average performance per document. Like most previous systems, we use precision, recall and F1 at mention level (micro) as the evaluation metrics and introduce the definition as [5]:

$$tp(s,g,f) = \{e \in s | \exists e^{'} \in g : f(e^{'}, e)\} \tag{18}$$

$$fp(s,g,f) = \{e \in s | \nexists e^{'} \in g : f(e^{'}, e)\} \tag{19}$$

$$tn(s,g,f) = \{e \notin s | \nexists e^{'} \in g : f(e^{'}, e)\} \tag{20}$$

$$fn(s,g,f) = \{e \in s | \nexists e^{'} \in g : f(e^{'}, e)\} \tag{21}$$

$$P_{mic}(s,g,f) = \frac{\sum_{m\in\mathcal{D}} tp(s_m,g_m,f)}{\sum_{m\in\mathcal{D}}(tp(s_m,g_m,f)+fp(s_m,g_m,f))} \quad (22)$$

$$R_{mic}(s,g,f) = \frac{\sum_{m\in\mathcal{D}} tp(s_m,g_m,f)}{\sum_{m\in\mathcal{D}}(tp(s_m,g_m,f)+fn(s_m,g_m,f))} \quad (23)$$

$$F1_{mic}(s,g,f) = \frac{2\cdot P_{mic}(s,g,f)\cdot R_{micro}(s,g,f)}{P_{mic}(s,g,f)+R_{micro}(s,g,f)} \quad (24)$$

where s denotes the linked entities output by the EL system, and g represents the gold entities. f is a binary matching function, if the position, length and selected target entity of the mention are all correct, $f(e^{'},e)=1$.

4.3.2 Baseline. For comparison, we choose a series of global EL systems which report state-of-the-art results on Gerbil. Specifically, the baseline methods we used are as follows:

- **AIDA** [13] builds a weighted graph to approximate the best joint mention-entity mapping.
- **Babelfy** [22] presents a unified graph-based approach to EL and Word Sense Disambiguation (WSD) tasks.
- **WAT** [28] uses graph-based algorithm and vote-based algorithm to approximate the global coherence.
- **xLisa** [43] applies a personalized PageRank algorithm to address the EL task.
- **PHoH** [9] proposes a probability graph model to perform collective EL.
- **WNED** [12] performs random walk on the mention-entity graph to disambiguate mentions.
- **NCEL** [1] utilizes graph convolutional network to integrate both local and global information for EL.

4.3.3 Results. The comparative results on the Gerbil platform are shown in Table 3. We observe that SGEL outperforms all baselines on both AIDA-A and AIDA-B datasets, which are the biggest EL datasets publicly available. Except for the formal text, our model also achieves the best performance on informal RSS-feeds such as RSS500. Compared with NCEL which uses GNN for collective entity linking, SGEL exceeds it a lot on each dataset, which demonstrates the effectiveness of our sequential GAT model. Although some baseline methods achieve competitive results on specific datasets, such as Babelfy and WNED respectively on KORE50 and MSNBC, they perform poorly on other datasets. Overall, SGEL performs consistently well on most of datasets, which reflects the good robustness of our model.

4.4 Impact of Different Modules

To analyze the performance of different modules and investigate their impact on the final results, we evaluate the effect of Local Encoder and Global Encoder.

4.4.1 Influence of BERT. In SGEL model, we use BERT as the local encoder to encode mention context and entity description. As we all know, in many NLP tasks, bidirectional LSTM also does well in capturing both previous and future contextual semantic information. To evaluate the effectiveness of BERT, we compare BERT with a bidirectional LSTM network (Bi-LSTM) in two ways. First, we just use local information to rank candidate entities and select the most relevant candidate as target entity. This model doesn't use manual features and global information. Second, we still use the SGEL framework but replace BERT by Bi-LSTM, named SGEL(Bi-LSTM). Comparative results in Table 5 show that our SGEL model obtains about 3% F1 improvement by using the BERT encoder, which indicates that our model can capture more valuable information between mention and entity. Besides, we find that even using a powerful language model, the F1 is just around 60% if we just utilize the local semantic information. Therefore, we combine the local information, manual features and global information in the Entity Selector.

Table 4: The comparison results using different local models.

Model	AIDA-A	AIDA-B	ACE2004	Reuters128
Bi-LSTM	55.8%	56.7%	65.4%	42.3%
BERT	61.0%	61.1%	70.0%	49.4%
SGEL(Bi-LSTM)	80.0%	81.6%	87.5%	68.1%
SGEL(BERT)	**84.8%**	**82.9%**	**88.7%**	**71.3%**

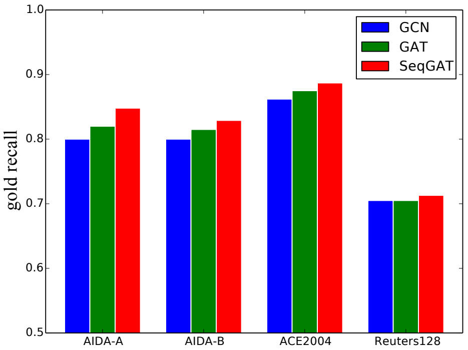


Figure 5: The comparison results using different global encoding strategies.

4.4.2 Effect of Sequential GAT. Unlike the previous graph-based models which encode the relation between mentions and all candidate entities at once, we add candidate entities in a sequential way. To evaluate the effectiveness of sequential operation, we compare SeqGAT with traditional GAT which encode links between all candidate entities. From the results in Figure 5, we can see that the model with SeqGAT achieves an improvement of 2% F1 over GAT, which demonstrates the effectiveness of introducing the sequential operation. Besides, we compare GAT with GCN to evaluate the effectiveness of self attention mechanism. Compared to GCN, GAT increases by 1.5% F1 on multiple datasets, which indicates that our model can better make use of the topical consistency between target entities. Moreover, we also notice that the effect of GAT is not obvious on the Reuters128 dataset. This is because a lot of documents in Reuters128 contain only a few mentions, which makes noisy candidates rarely affect on global encoding result.

Table 5: Results on the AIDA-B dataset to investigate the influence of different model architectures.

Model	Micro F1 (%)	
	Total	Δ
Full Model	82.9	-
– BERT	80.7	2.2
– SeqGAT	79.9	3.0
– Manual Features	64.4	18.5
– BERT, SeqGAT	78.4	4.5

4.4.3 Ablation studies. To better evaluate the contribution of different modules to the overall performance, we conduct the ablation studies on the AIDA-TestB dataset. From the results shown in table 6, we can observe that: (1) BERT is a necessary component that contributes 2.2% gain of F1 to the ultimate performance, we attribute this gain to the local semantic information. (2) Removing SeqGAT degrades the performance by 3.0% F1, which shows that the global information reflecting the interdependence between mentions is useful for ED. (3) The manual features contribute much to the overall performance, since the F1 drops drastically by 18.5% if it is removed. (4) When we remove BERT and SeqGAT, the score drops by 4.5%, which indicates that the participation of multi-aspect information is important for our model.

5 RELATED WORK

5.1 Candidate Generation

Name dictionary based techniques are the main approaches to generate candidate entities. According to our survey, existing publicly available candidate sets in [1, 10, 17, 42] are all constructed through static dictionary YAGO [13] and Cross-Wiki [34]. The former dictionary is built from entities in YAGO knowledge base, while the latter establishes a general mapping from strings to Wikipedia entities in 2012. Restricted by the dictionary size and construction time, many long-tailed and newly generated entities cannot be recalled effectively. To extend the semantics of mention, some EL systems [7, 31, 36] try to retrieve candidate entities by using web search engines. Specifically, [31] submit the mention together with its local context to the Google search engine and identify entities whose Wikipedia pages appear in the top results. In fact, they already use Google's entity disambiguation service when recalling candidates, which makes it difficult to evaluate the real performance of the ED model. [36] generates candidates by searching sentences from Wikipedia articles and directly using the human-annotated entities as the candidates. But for some entities that have no specific meanings, such as country, common noun, etc., it is difficult to recall them by sentence matching. Moreover, there are some methods [4, 33] proposed to address the misspelling problem existing in mention. For example, [4] obtains the suggested correct string by the spellchecker in Lucene[7], and [33] exploits the query spelling correction service supplied by Google search engine. Inspired by them, we also use Google search engine to solve the misspelling problem.

5.2 Entity Disambiguation

Recently, many collective ED methods [1, 2, 8, 10, 17, 25, 26, 29, 42] aiming to combine local and global contextual information are proposed. To extract local features from mention's context and entity description, CNN, LSTM based models [23, 25] are used in early studies. For the purpose of identifying the most discriminative words from local context, attention mechanism is also exploited by EL systems [10, 24, 25]. With the burgeoning popularity of BERT, there are many NLP tasks focus on fine-tuning with the pre-trained BERT model. Similarly, [19] presents a new zero-shot entity linking task and constructs a new dataset to evaluate their BERT-based model. It is worth noting that their zero-shot EL task is different from ours. Moreover, according to our experimental results, it is difficult to achieve good performance in EL task only by using BERT. To capture the interdependence among multiple mentions in a document, many graph-based models [1, 9, 12, 29, 42] are applied in EL. Assuming the topical coherence among mentions, authors in [9, 29] construct factor graph models and introduce loopy belief propagation algorithm to perform approximate inference. [12, 42] performs random walk [37] on the mention-entity graph and use the convergence score for disambiguation. Similar to us, [1] applies Graph Convolutional Network (GCN) to integrate global coherence information for EL. However, their model can not pay attention to the correlation among target entities, which may lead to a large number of noisy data in the global encoding. To reduce the impact of non-target entities on the final results, [8, 23] rank mentions and deal with them in a sequence manner. Specifically, [8] converts the global EL into a sequence decision problem and propose a reinforcement learning model. Unfortunately, their sequential models can only observe previously referred entities, making it hard to effectively utilize the consistency of subsequent ones. To address this problem, we propose a sequential GAT model in this paper.

6 CONCLUSION

In this paper, we propose a new candidate generation (CG) method and design our entity disambiguation (ED) model based on BERT and Sequential GAT (SeqGAT). We combine a variety of information to retrieve candidate entities and provide an easy-to-use API to facilitate the application of CG module to other datasets. By utilizing BERT to extract local features from mention context and entity description and introducing SeqGAT to capture the topical coherence of mentions, our ED model can disambiguate mentions from both local and global perspectives. Besides, compared with traditional graph model, our SeqGAT not only makes full use of the topical consistency, but also reduces noise interference. In experiments, we evaluate our model on 8 different datasets and the results demonstrate the competitiveness of our approach. In the future, we plan to build an end-to-end dynamic graph EL model which can automatically detect the change of association strength between entities.

ACKNOWLEDGMENTS

This research is supported by the National Key Research and Development Program of China (No. 2018YFB1004703).

[7] http://lucene.apache.org/

REFERENCES

[1] Yixin Cao, Lei Hou, Juanzi Li, and Zhiyuan Liu. 2018. Neural Collective Entity Linking. In *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018*. 675–686.

[2] Lihan Chen, Jiaqing Liang, Chenhao Xie, and Yanghua Xiao. 2018. Short Text Entity Linking with Fine-grained Topics. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM 2018*. 457–466.

[3] Tianqi Chen and Carlos Guestrin. 2016. XGBoost: A Scalable Tree Boosting System. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*. 785–794.

[4] Zheng Chen and Suzanne Tamang etc. 2010. CUNY-BLENDER TAC-KBP2010 Entity Linking and Slot Filling System Description. In *Proceedings of the Third Text Analysis Conference, TAC 2010, Gaithersburg, Maryland, USA, November 15-16, 2010*.

[5] Marco Cornolti, Paolo Ferragina, and Massimiliano Ciaramita. 2013. A framework for benchmarking entity-annotation systems. In *22nd International World Wide Web Conference, WWW '13, Rio de Janeiro, Brazil, May 13-17, 2013*. 249–260.

[6] Silviu Cucerzan. 2007. Large-Scale Named Entity Disambiguation Based on Wikipedia Data. In *EMNLP-CoNLL 2007, Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, June 28-30, 2007*. 708–716.

[7] Mark Dredze, Paul McNamee, Delip Rao, Adam Gerber, and Tim Finin. 2010. Entity Disambiguation for Knowledge Base Population. In *COLING 2010, 23rd International Conference on Computational Linguistics, Proceedings of the Conference, 23-27 August 2010, Beijing, China*. 277–285.

[8] Zheng Fang, Yanan Cao, Qian Li, Dongjie Zhang, Zhenyu Zhang, and Yanbing Liu. 2019. Joint Entity Linking with Deep Reinforcement Learning. In *The World Wide Web Conference, WWW 2019*. 438–447.

[9] Octavian-Eugen Ganea, Marina Ganea, Aurélien Lucchi, Carsten Eickhoff, and Thomas Hofmann. 2016. Probabilistic Bag-Of-Hyperlinks Model for Entity Linking. In *Proceedings of the 25th International Conference on World Wide Web, WWW 2016*. 927–938.

[10] Octavian-Eugen Ganea and Thomas Hofmann. 2017. Deep Joint Entity Disambiguation with Local Neural Attention. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017*. 2619–2629.

[11] Amir Globerson, Nevena Lazic, Soumen Chakrabarti, Amarnag Subramanya, Michael Ringgaard, and Fernando Pereira. 2016. Collective Entity Resolution with Multi-Focal Attention. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016*.

[12] Zhaochen Guo and Denilson Barbosa. 2018. Robust named entity disambiguation with random walks. *Semantic Web* 9, 4 (2018), 459–479.

[13] Johannes Hoffart and Mohamed Amir Yosef etc. 2011. Robust Disambiguation of Named Entities in Text. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, EMNLP 2011*. 782–792.

[14] Matthew A Jaro. 1995. Probabilistic linkage of large public health data files. *Statistics in medicine* 14, 5-7 (1995), 491–498.

[15] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *5th International Conference on Learning Representations, ICLR 2017*.

[16] Nikolaos Kolitsas, Octavian-Eugen Ganea, and Thomas Hofmann. 2018. End-to-End Neural Entity Linking. In *Proceedings of the 22nd Conference on Computational Natural Language Learning, CoNLL 2018*. 519–529.

[17] Phong Le and Ivan Titov. 2018. Improving Entity Linking by Modeling Latent Relations between Mentions. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018*. 1595–1604.

[18] Phong Le and Ivan Titov. 2019. Boosting Entity Linking Performance by Leveraging Unlabeled Documents. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019*. 1935–1945.

[19] Lajanugen Logeswaran, Ming-Wei Chang, Kenton Lee, Kristina Toutanova, Jacob Devlin, and Honglak Lee. 2019. Zero-Shot Entity Linking by Reading Entity Descriptions. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019*. 3449–3460.

[20] George A. Miller. 1995. WordNet: A Lexical Database for English. *Commun. ACM* 38, 11 (1995), 39–41.

[21] David N. Milne and Ian H. Witten. 2008. Learning to link with wikipedia. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management, CIKM 2008*. 509–518.

[22] Andrea Moro, Alessandro Raganato, and Roberto Navigli. 2014. Entity Linking meets Word Sense Disambiguation: a Unified Approach. *TACL* 2 (2014), 231–244.

[23] Thien Huu Nguyen, Nicolas R. Fauceglia, Mariano Rodriguez-Muro, Oktie Hassanzadeh, Alfio Massimiliano Gliozzo, and Mohammad Sadoghi. 2016. Joint Learning of Local and Global Features for Entity Linking via Neural Networks. In *COLING 2016*. 2310–2320.

[24] Feng Nie, Yunbo Cao, Jinpeng Wang, Chin-Yew Lin, and Rong Pan. 2018. Mention and Entity Description Co-Attention for Entity Disambiguation. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18)*. 5908–5915.

[25] Minh C. Phan, Aixin Sun, Yi Tay, Jialong Han, and Chenliang Li. 2017. NeuPL: Attention-based Semantic Matching and Pair-Linking for Entity Disambiguation. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, CIKM 2017*. 1667–1676.

[26] Minh C. Phan, Aixin Sun, Yi Tay, Jialong Han, and Chenliang Li. 2019. Pair-Linking for Collective Entity Disambiguation: Two Could Be Better Than All. *IEEE Trans. Knowl. Data Eng.* 31, 7 (2019), 1383–1396.

[27] Francesco Piccinno and Paolo Ferragina. 2014. From TagME to WAT: a new entity annotator. In *ERD'14, Proceedings of the First ACM International Workshop on Entity Recognition & Disambiguation, July 11, 2014*. 55–62.

[28] Francesco Piccinno and Paolo Ferragina. 2014. From TagME to WAT: a new entity annotator. In *Proceedings of the first international workshop on Entity recognition & disambiguation*. ACM, 55–62.

[29] Chenwei Ran, Wei Shen, and Jianyong Wang. 2018. An Attention Factor Graph Model for Tweet Entity Linking. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*. 1135–1144.

[30] Lev-Arie Ratinov, Dan Roth, Doug Downey, and Mike Anderson. 2011. Local and Global Algorithms for Disambiguation to Wikipedia. In *The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference, 19-24 June, 2011*. 1375–1384.

[31] Wei Shen, Jianyong Wang, and Jiawei Han. 2014. Entity linking with a knowledge base: Issues, techniques, and solutions. *IEEE Transactions on Knowledge and Data Engineering* 27, 2 (2014), 443–460.

[32] Wei Shen, Jianyong Wang, and Jiawei Han. 2015. Entity Linking with a Knowledge Base: Issues, Techniques, and Solutions. *IEEE Trans. Knowl. Data Eng.* 27, 2 (2015), 443–460.

[33] Wei Shen, Jianyong Wang, Ping Luo, and Min Wang. 2012. LINDEN: linking named entities with knowledge base via semantic knowledge. In *Proceedings of the 21st World Wide Web Conference 2012, WWW 2012*. 449–458.

[34] Valentin I. Spitkovsky and Angel X. Chang. 2012. A Cross-Lingual Dictionary for English Wikipedia Concepts. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation, LREC 2012*. 3168–3175.

[35] Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. In *Proceedings of the 16th International Conference on World Wide Web, WWW 2007*. 697–706.

[36] Chuanqi Tan, Furu Wei, Pengjie Ren, Weifeng Lv, and Ming Zhou. 2017. Entity Linking for Queries by Searching Wikipedia Sentences. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017*. 68–77.

[37] Hanghang Tong, Christos Faloutsos, and Jia-Yu Pan. 2006. Fast Random Walk with Restart and Its Applications. In *Proceedings of the 6th IEEE International Conference on Data Mining (ICDM 2006), 18-22 December 2006, Hong Kong, China*. 613–622.

[38] Ricardo Usbeck and Michael Röder etc. 2015. GERBIL: General Entity Annotator Benchmarking Framework. In *Proceedings of the 24th International Conference on World Wide Web, WWW 2015, Florence, Italy, May 18-22, 2015*. 1133–1143.

[39] Vasudeva Varma and Praveen Bysani etc. 2010. IIIT Hyderabad in Guided Summarization and Knowledge Base Population. In *Proceedings of the Third Text Analysis Conference, TAC 2010*.

[40] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017*. 5998–6008.

[41] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. In *6th International Conference on Learning Representations, ICLR 2018,*.

[42] Mengge Xue, Weiming Cai, Jinsong Su, Linfeng Song, Yubin Ge, Yubao Liu, and Bin Wang. 2019. Neural Collective Entity Linking Based on Recurrent Random Walk Network Learning. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019*. 5327–5333.

[43] Lei Zhang and Achim Rettinger. 2014. X-LiSA: Cross-lingual Semantic Annotation. *PVLDB* 7, 13 (2014), 1693–1696.