# The ZHI-EDL System for Entity Discovery and Linking at TAC KBP 2017

**Siliang Tang[1], Yankun Ren[1], Xiyuan Yang[1],**
**Dan Liu[2], Guoping Hu[2], Fei Wu[1], and Yueting Zhuang[1]**
1. College of Computer Science, Zhejiang University, Hangzhou, Zhejiang, P. R. China
2. iFLYTEK Research, Anhui, Hefei, P. R. China
{siliang, renyk, yangxiyuan}@zju.edu.cn, {danliu, gphu}@iflytek.com,
{yzhuang, wufei}@zju.edu.cn

## Abstract

This report gives a detailed description of ZHI(知)-EDL system of team 'rise_dcd_zju', which submitted to the TAC KBP 2017 Trilingual Entity Detection and Linking (EDL) track. Our system consists of two cascaded components, one for mention detection, another for entity linking. Our best system has achieved an overall F1 score of 0.685 for trilingual in term of the CEAFmC metric this year.

## 1 Introduction

This report gives a detailed description of ZHI-EDL system of team 'rise_dcd_zju'. As shown in Figure (1), our system consists of two cascaded components, one for mention detection, another for entity linking. In the mention detection part, system takes a document as input. It first splits the document into sentences, and further extracts word level features for each sentence, such as position and POS. Then a deep sequence labelling model will annotate each word with I-O-B tag to detect the span and type of mentions in the given sentence. In addition to this deep model based procedure, we also designed some rule-based mention detectors to extract nested mentions, e.g., 欧盟, 欧 is always regard as GPE NAM. The following stage is entity linking. It takes detected mentions as input, and contains two steps, i.e., candidate generation and candidate ranking. For the candidate generation, each detected mention is sent to three parallel branches respectively with following strategies: 1) Query a database with all Freebase entities; 2) Query a database with Wikipedia redirection and disambiguation information; (3) Lucene fuzzy search on Wikipedia title, first paragraph, and document context, as well as on Freebase object name. Generated candidate entities are further ranked by our ranking model. It is a MLP model with three fully-connected layers. The ranking model outputs ranking score for each candidate, and we choose the candidate with the highest score as our linking result to input mention.

## 2 Mention Detection

Mention detection is considered to be one of the crucial steps toward natural language understanding. Its goal is to identify entity mentions and classify them into predefined categories. One of the typical ways of solving this problem is using sequence labelling models such as Conditional Random Fields (CRFs) (Lafferty et al., 2001). The performance of CRF models heavily relied on handcrafted features (e.g., whether a word is capitalized) and language-specific resources (e.g., gazetteers), which makes them domain or language dependent. To overcome this problem especially to make our model language independent, we designed a deep recurrent neural networks (RNN) (Mikolov et al., 2010) based mention detection model, which can automatically extract word-level and character-level features. To detect nominal mentions, we treat them as entity types just like other named entity types, and the model jointly detect both named and nominal mentions together. Figure 2 shows the structure of our mention detection model, and we will discuss it in the following

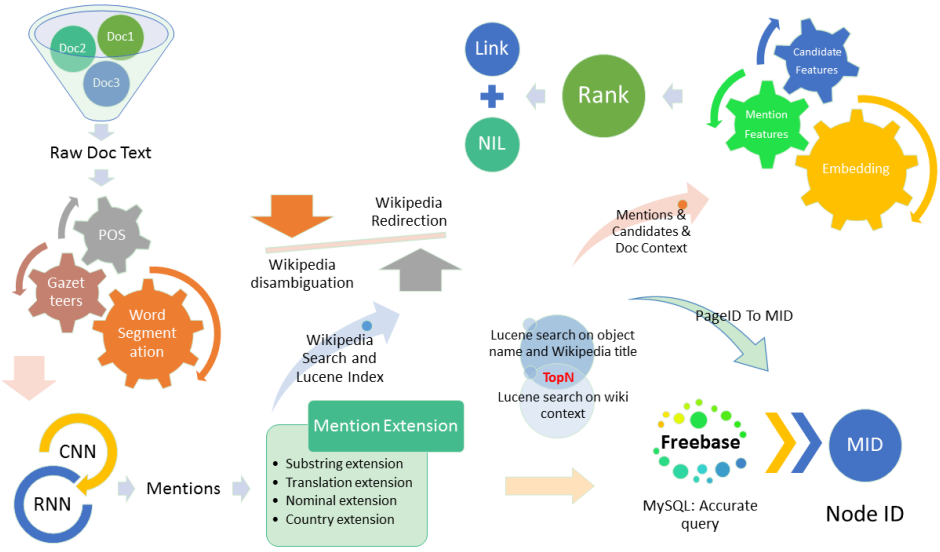


Figure 1: ZHI-EDL System Overview.

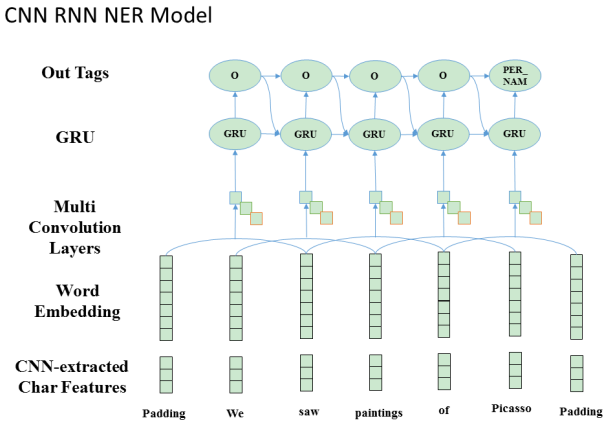


Figure 2: The architecture of mention detection model.

subsections:

### 2.1 Feature Extraction Layer

Character-level features have been empirically verified to be helpful in numerous sequence labeling tasks (Peters et al., 2017). Therefore in the Feature Extraction Layer of our mention detection model, we obtain the character-level features of each word using CNN (Kim, 2014). In this CNN, chars vectors are considered as 1-dimension inputs and fed into a multi-layer CNN. The outputs of the CNN are max-pooled over the entire width to obtain a fixed-size vector for each word. We then use pre-trained word vectors by GloVe (Pennington et al., 2014), to obtain the fixed word embedding of each word. A concatenation of the character vectors and word vector is then passed to another CNN that is stacked with several 1-dimension convolutional layers to generate the representation for the entire input sequence. We do not use any pooling layers, and zero-padding is used in each layer. Therefore, the length in each convolutional layer remain the same as the input sequence.

### 2.2 RNN for Sequence Labeling

We believe that long-term dependency is important for mention detection, especially for capturing long rage relation, and for long mentions that contain surface name of other entities. Therefore we adopt recurrent neural networks as our sequence labelling model instead of traditional CRFs.

If we denote an input sequence $X = (x_1, x_2, \ldots, x_N)$ and the corresponding output labels $Y = (y_1, y_2, \ldots, y_N)$, then our sequence labeling model is aim to maximize following

equations:

$$Pr(Y|X) = \prod_{i=1}^{N} P(y_i|X, y_i-1, y_i-2, \ldots, y_1) \quad (1)$$

The input at each time step of RNN is the output of feature extraction layer, also the label of a time step is passed to the next time step. In consequence, $P(y_i|X, y_i-1, y_i-2, \ldots, y_1)$ can be computed at each time step of recurrent neural networks.

For simplicity, we use one layer of gated recurrent units (GRU) (Cho et al., 2014), which essentially computes all conditional probabilities in Eq. (1) one by one sequentially, each of which conditions on the CNN-generated representation of X and the preceding partial output labels.

In the training stage, we minimize the cross-entropy based on all collected sequence pairs in the training set, $\{X_i, Y_i\}$. In the test stage, the learned hybrid model of CNNs and GRU-based RNN is used to compute the conditional probability of every possible label of each word, and we use Viterbi decoding algorithm to get the label $Y$ with the highest probability for each input sentence $X$.

### 2.3 Model Configurations

We use one 1-dimension convolutional layer to extract character-level features. The filter size is set to 3 and the number of out-channel is set to 50. And we use a stack of five 1-dimension convolutional layers for the input word sequences. We set the filter size to 3 and set the number of feature maps to 512. In this way, the outputs of the CNN provide a vector representation of every word in the input sequence. Parameter optimization of all models are performed using AdaDelta (Zeiler, 2012) and early stopping is also used by monitoring a small held-out development set. Similar to all neural networks, the performance of our proposed models relies on the amount of the training data. However, there is not too much matched in-domain training data for KBP mention detection tasks. Therefore, for English and Chinese languages, we have used some inhouse data annotated by iFLYTEK research, which consists of about 10,000 Chinese and English documents downloaded from the web. These documents are internally labelled using some annotation rules similar to the KBP guidelines. For Spanish, there is no extra annotated training data.

| Language | Precision | Recall | F1 |
|---|---|---|---|
| Chinese | 0.88 | 0.673 | 0.75 |
| Spanish | 0.858 | 0.672 | 0.754 |
| English | 0.853 | 0.719 | 0.762 |
| Trilingual | 0.87 | 0.663 | 0.752 |

Table 1: The Official Results of ZHI-Entity Discovery System in 2017 TAC-KBP EDL Evaluation.

### 2.4 Performance

The official entity discovery performance from the EDL evaluation in 2017 for NERC is summarized in Table 1.

## 3 Entity Linking

The Tri-lingual Entity Linking task at NIST TAC-KBP2017 aims to link entity mentions that extracted in the Tri-lingual Entity Discovery task, to an existing and pointed Knowledge Base (i.e., Freebase). The Entity Linking system is required to obtain Freebase MIDs for those entity mentions that have corresponding Freebase Topic, and cluster mentions for those NIL entities that do not. Our Entity Linking System development work can be split into three stages. The first stage is data pre-processing operations on Freebase and Wikipedia data dumps, including data extraction, clean, storage and indexing. The second stage is developing candidate generation subsystem, responsible for generating candidate Freebase MIDs for each discovered entity mentions, mainly using complicated rule-based search methods. The third stage is implementing candidate rank subsystem, in charge of scoring and ranking all candidate Freebase MIDs belonging to each discovered entity mention, mainly using neural network model.

### 3.1 Data Pre-Processing

#### 3.1.1 Freebase

We use regular expression based extractors to extract five types resources from Freebase data dump. The schema of these resources are presented in Table 2 to 7. We will describe these resources in detail, and utilized them later in our candidate generation subsystems.

| ID | Name |
|---|---|
| m.0_0002 | Invocation of Apocalyptic Evil |
| m.0__0003 | M.A.R.L.E.Y. (skit) |
| m.0_0009_ | Annette Ventura-Glenn |
| m.0_0009 | Valley of the Damned |
| m.0_000dm | Anthony Glenn |

Table 2: Freebase ⟨MID, Object Name⟩

Since Freebase is organised in RDF format, for Table ⟨MID, Object Name⟩ 2, we extract Freebase triples whose object contain "@en/@es/@zh" substring.

For Table ⟨MID, Wikipedia Page ID⟩ 3, we focus on Freebase triples whose predicate contain "Wikipedia.en_id/es_id/zh-cn_id/zh-tw_id" substring.

| ID | Page ID |
|---|---|
| m.01009ly3 | 42210866 |
| m.0100bkts | 42213667 |
| m.0100cyq6 | 36040425 |
| m.0100d0nq | 42223807 |
| m.0100fxk0 | 42110086 |

Table 3: Freebase ⟨MID, Wikipedia Page ID⟩.

For Table ⟨MID, Freebase type⟩ 4, we fill it with Freebase triples whose predicate is end up with "type.object.type" substring.

| ID | Type |
|---|---|
| m.010006m6 | music.single |
| m.010006m6 | base.type_ontology.non_agent |
| m.010006m6 | base.type_ontology.abstract |
| m.010006m6 | common.topic |
| m.010006m6 | music.recording |

Table 4: Freebase ⟨MID, Object Type⟩.

The contents of Table ⟨MID, Wikipedia Redirect⟩ 5 are collected from both Freebase and Wikipedia data dumps. As we all know, even a Wikipedia page has a "real" title, it may appear under different names because of redirect records that point to the real page. In Freebase data dump, the real title is encoded in the \wikipedia\{lang}_title\ namespaces, whereas the titles that redirect to the real title are encoded in the \wikipedia\{lang}\ namespaces, where {lang} is an ISO 639-1 or a variation of an ISO code.

| ID | Title |
|---|---|
| m.0_018 | Coalmont |
| m.0_018 | Coalmont, PA |
| m.0_018 | Coalmont, Pennsylvania |
| m.0_029 | Dudley |
| m.0_029 | Dudley, PA |
| m.0_029 | Dudley, Pennsylvania |

Table 5: Freebase ⟨MID, Wikipedia Redirect⟩.

When derived from Wikipedia titles, Wikipedia keys replace the space character with an underscore, and escape punctuation and non-ASCII characters with the $, see detail in Table 6.

When extracting ⟨MID, Wikipedia Redirect⟩ triples, we first scan the whole Freebase data dump to locate triples whose predicate contains "wikipedia.{lang}" and "wikipedia.{lang}_title", where {lang} is one of "en/es/zh-ch/zh-tw". Then, we develop a decoding algorithm to decode the $ sequences in Freebase keys.

Wikipedia keys in Freebase is derived from Wikipedia data dumps. However, Google stopped updating Freebase website and its APIs since June 30, 2015, while Wikipedia data dumps are continuous updated monthly. This means we can use redirection resources from latest Wikipedia dumps to update ⟨MID, Wikipedia Redirect⟩ tables. To achieve this, as shown in Figure 3, we first transfer Freebase MID to the related Wikipedia page ID via ⟨MID, Wikipedia Page ID⟩ tables, then retrieve the attached Wikipedia Title as well as Wikipedia Redirects by invoking JWPL (Java Wikipedia Library). Finally, we merge Freebase Wikipedia Keys, Wikipedia Title and Wikipedia Redirects into a unified Wikipedia Redirection Resource.

For Table 7 ⟨MID, Relationship, MID⟩ , we import Freebase triples whose subject and object are both MIDs, then we count the number

| ID | Encoded Title | Decoded Title |
|---|---|---|
| m.0__00py | Sylvan_Esso_$0028album$0029 | Sylvan Esso (album) |
| m.0_00r03 | Woman$002C_Man$002C_Life | Woman, Man, Life |
| m.0_01m | Cromwell_Township$002C_PA | Cromwell Township, PA |
| m.0_018 | Coalmont$002C_Pennsylvania | Coalmont, Pennsylvania |
| m.0_029 | Dudley$002C_Pennsylvania | Dudley, Pennsylvania |

Table 6: Freebase ⟨MID, Encoded Title-Decoded Title⟩.

| Subject ID | Relation_Type | Object ID |
|---|---|---|
| m.010006m6 | Invocation of Apocalyptic Evil | m.0___0943 |
| m.010006m6 | music.recording.artist | m.0w2l6q2 |
| m.010006m6 | common.topic.notable_types | m.0kpv11 |
| m.01000_yw | music.release_track.recording | m.0___7x8f |
| m.01000_yw | music.release_track.release | m.0___7rh7 |

| ID | Hot |
|---|---|
| m.0yc670s | 4 |
| m.0dnt8s1 | 11 |
| m.0v__8fj | 4 |
| m.03531xp | 1 |
| m.0t7g3g | 13 |

Table 7: Freebase ⟨MID-Relation, MID⟩ and ⟨MID, Hot⟩.

of MID to obtain Table ⟨MID, Hot Value⟩ as Node Hot table.

#### 3.1.2 Wikipedia

We use JWPL (Java Wikipedia Library) (Zesch et al., 2008) to process Wikipedia data dumps, which is a free, Java-based application programming interface that allows to access all information in Wikipedia. The Wikipedia data dumps for each language at least contain the following three archives:

- wiki-[DATE]-pages-articles.xml.bz2
- wiki-[DATE]-pagelinks.sql.gz
- wiki-[DATE]-categorylinks.sql.gz

### 3.2 Chinese and Spanish Candidate Generation Subsystem

#### 3.2.1 Subsystem Overview

Candidate generation subsystem plays a vital role in the overall linking performance, for that the final accuracy is decided by the quality of generated candidate list. Therefore, we designed a complicated rule-based system as our candidate generation module to generate candidates for each detected mention, which is illustrated in Figure 4.

For Chinese candidate generation, the first step is to extend each detected mention by substring extension, translation extension, country extension and nominal extension respectively. After that, all the extensions are sent to three parallel branches respectively with following strategies: 1) Query a database with all Freebase entities; 2) Query a database with Wikipedia redirection and disambiguation information; (3) Lucene fuzzy search on Wikipedia title, first paragraph, and document context, as well as on Freebase object name and Wikipedia redirection resource. Figure 4 shows the framework of our Chinese candidate generation subsystem.

However, for Spanish, Lucene fuzzy search shows some side effects, resulting in performance degradation. Therefore we remove Lucene fuzzy search strategy in Spanish candidate generation system.

#### 3.2.2 Mention Extension

Mention Extension can improve the link performance. For example, given a detected mention "Washington", it would be much easier for the system to discover its true Freebase entry "Washington DC", if the system can extend it to "Washington DC". One possible solution is to define a series of extension methods for each detected mention under the constrains of corresponding entity type. This will help system to generate more candidates, and further improve the overall candidate coverage.

We have designed five types of mention extension methods:

**Substring Extension:** For a particular mention, we will go through its context document, and select all the recognized named enti-

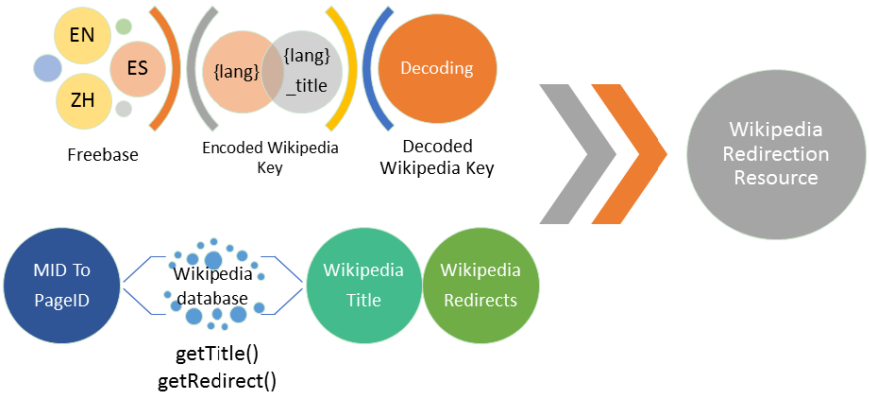


Figure 3: Wikipedia Redirection Resource Merge.

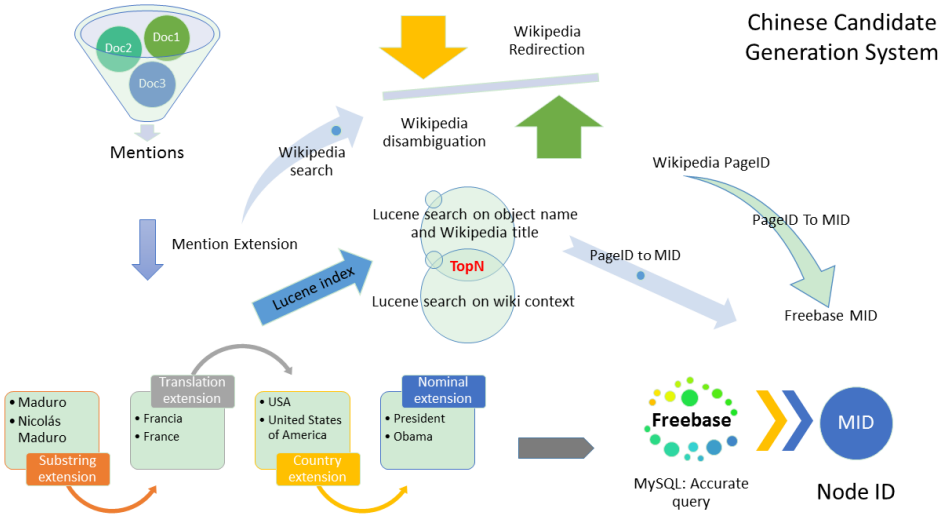


Figure 4: Chinese Candidate Generation Subsystem.

ties which contain that mention. For instance, given the mention "Maduro" in document $d$, we will select "Nicolás Maduro" as its substring extension if named entity "Nicolás Maduro" is found in $d$.

**Translation Extension:** For the languages with low resources than English in Freebase or Wikipedia (such as Chinese and Spanish), we invoke Google Translate to get their translations as Translation Extension.

**Country Extension:** the abbreviation of country names or nationalities can be extended to a more concrete one. For example, "USA" can be extended to "United States of America" when its entity type is "GPE".

**Nominal Extension:** to extend nominal mentions, we adopt greedy search, which selects the nearest recognized entity with the same entity type as its nominal extension.

**Traditional Chinese Extension:** in order to utilize the rich traditional Chinese resources, we also transfer the simplified Chinese mentions into traditional Chinese mentions.

### 3.2.3 Candidate Query

After mention extension, all the extensions are sent to three parallel branches respectively with following strategies:

1. Query a database with all Freebase entities (by using Table 2).

2. Query a database with Wikipedia redirection and disambiguation information (by using Table 5 and Wikipedia disambiguation resource).

3. Lucene fuzzy search on Wikipedia title, first paragraph, and document context, as well as on Freebase object name and Wikipedia redirection resource. (by using Table 2, Table 5 and Wikipedia database separately according to the corresponding language).

### 3.3 Query Optimization

The first strategy usually yields a fairly large set of candidate entities. For example, when we query a mention "Washington", we will get about 293 results, some results are locations, while others are person names. To address this problem, we proposed a query optimization method to put constraints on the query results, the optimization process is illustrated in Figure 5.

First of all, we realize that different entity type maps to different Freebase types, i.e. "GPE" often maps to "location.location", "location.region" and other similar Freebase types, while "PER" often maps to Freebase types like "people.person" and "religion.deity". So the first thing we can do is to divide the training data (2015-2016 LDC TAC KBP-EDL data) into five sets, according to their entity type (GPE, FAC, LOC, ORG, PER).

Secondly, every entity mention can have several Freebase types at the same time, which is so called Freebase type block. For example, "Vietnam" is a GPE entity, and its Freebase types contain: location.region, location.location, common.topic, government.governmental_jurisdiction and so on, about 45 Freebase types in total. So we need to query ⟨MID, Freebase type⟩ table, and collect the Freebase type package filled with Freebase type blocks for each entity mention set obtained in the first step.

Thirdly, every Freebase type package contains some "core Freebase types", which occur more frequently among all the other Freebase type blocks. For example, "Vietnam" belongs to "location.location | location.country | ⋯" Freebase type, while "Wisconsin" belongs to "location.location | location.us_state | ⋯" Freebase type. Then we can define "location.location" is the core Freebase type, finally we can summarize a core Freebase type set for each entity type by counting.

The advantage of core Freebase type set lies in that it could help us shrink the query scope, thereby narrow the result set size, and further save computational time.

### 3.4 Chinese and Spanish Candidate Ranking Subsystem

In our rank module, we aim to find the most relevant candidate of given mention. Therefore, we proposed a multi-layers neural network model to compute the probability for each candidate entities. Our model accepts following features from both the mention and candidate entities as input:

1. **Word Embedding**: it contains mention string embedding and candidate string embedding, each word in the string is projected into 200-dimension word vector, and the string vector is represented as the average of all the word vectors. This feature is trained by word2vec and is fixed in our NN-ranking model.

2. **Node-hot Value**: node-hot represents the frequency of an entity in the Freebase dump. It is computed by the numbers of links with other Freebase node. In our model, we map the node-hot values into a 10-dimension one-hot vector.

3. **Tf-Idf Cosine Similarity**: the tf-idf cosine similarity measures the similarity between the Wiki text and the KBP corpus, and the idf is computed based on the KBP corpus. In our model, we also map it into a 10-dimension one-hot vector.

4. **String Commonness**: String commonness shows the similarity between the mention string and the candidate string. In

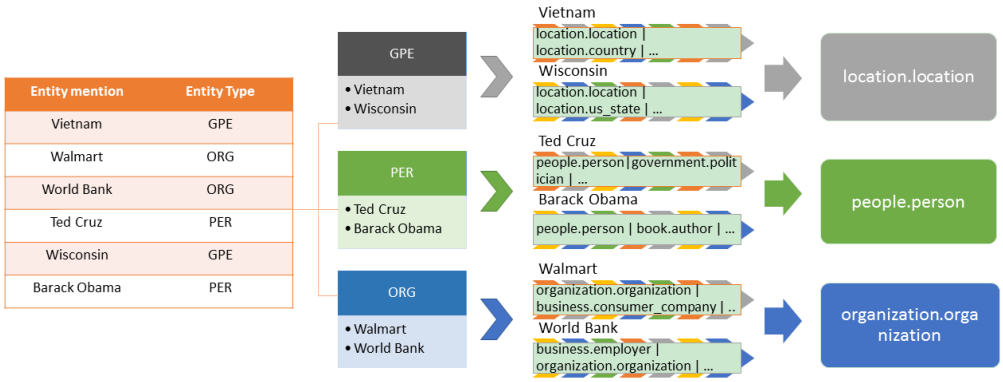


Figure 5: Freebase Type Selection Process.

our system, we use the longest common sequence(LCS), that means, if we have mention and entity like "Barack Obama" and "Obama", we may think the common string length is 5, and the commonness can compute by $l_c/l_m$, where $l_c$ means the common string length and $l_m$ means the mention string length.

5. **Document Domains**: the KBP data corpus are collected from two domains, news and forum. Thus we add a binary feature to represent their domains.

6. **Entity type**: this model contains two kind of entity type, Freebase node type and the KBP entity type. The KBP entity has five types, that is, "PER, ORG, GPE, LOC and FAC". The Freebase node type we used for this task contains mainly four types, which include "people.person, organization.organization, geographg.geographical_feature and location.location". Then we use 10-dimension dense vector to represent these types, and these vectors are updated during the model training.

The model structure is presented in Figure 6. We use a simply three-layer fully-connected neural network as our ranking model. The first hidden layer and the second hidden layer of our model contain 256 and 128 neurons respectively. We use ReLU as model activation function. Through the Neural Network, we can assign a score for every candidate. Finally, we can compute the posterior by a Softmax function, then choose the entity with the highest probability as linking output.

### 3.5 English Candidate Generation Subsystem

We adopted different strategies for English candidate generation and ranking, As shown in Figure 7, for English mention detection, instead of using the system we mentioned early in section 2, we also use Stanfold CoreNLP/NER (Manning et al., 2014) and yields an ensemble NER results .

We created an alias dictionary with disambiguation pages, redirect pages and anchor texts of Wikipedia (Cucerzan, 2007) for candidate generation. The "also known as" values of Freebase are also added into the dictionary. In general, the alias dictionary may product too many candidates for a single mention. This behavior lead to much noise, and slow down the whole system. Therefore we ranked the candidates of each mention based on their commonness of the name-string of the mention, and keep the top 30 candidates for each mention.

### 3.6 English Candidate Ranking Subsystem

After candidate generation, we use commonness (Medelyan and Legg, 2008), cosine TF-IDF similarity, and IWHR to rank the candidates.

Commonness reflect the popularity of the name-string of the mention used as an alias

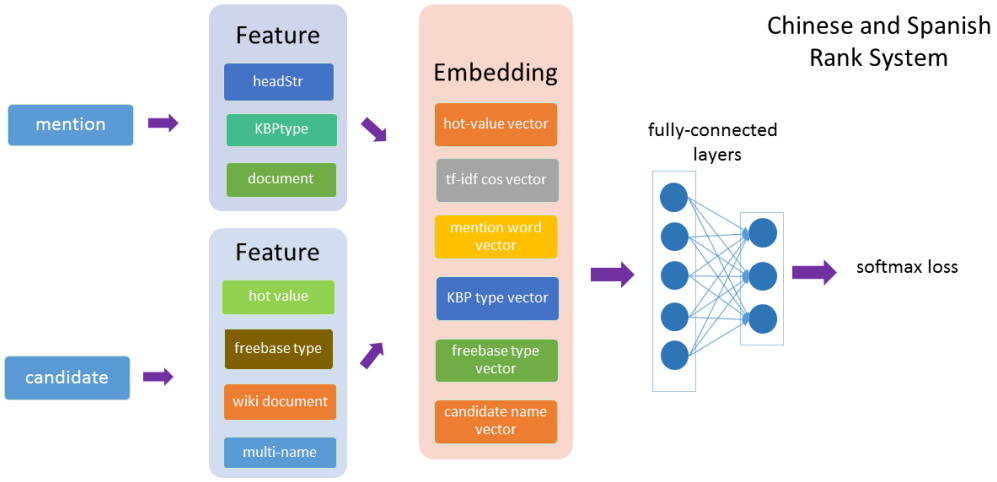


Figure 6: Chinese and Spanish Rank System.

for the entity. For example, in the sentence "Obama likes running", we cannot determine who "Obama" is by text, but generally it refer to the former US president.

TF-IDF similarity is computed between the input document and the Wikipedia article of the candidate entity. It gives an evaluation on how similar two documents are in general. But for Entity Linking, it cares many trivial words even we use the stop words-table. Therefore we used IWHR to capture the similarity between the input document and the Wikipedia article of the candidate entity. IWHR, which is the abbreviation of "important word hit rate", is defined as follows:

$$f(e,m) = \frac{\sum_{w \in W_d \cap W_e, idf(w) > T} idf(w)}{\sum_{w \in W_d, idf(w) > T} idf(w)} \qquad (2)$$

where, $m$ is the notation of the mention, $e$ represents the candidate entity. $T$ is a threshold given manually.

We adjust the weights of all the three feature manually based on the TAC2016 EDL dataset.

### 3.7 NIL Clustering

We group the different NIL mentions into one cluster only when they have the same string and the same type.

## 4 Submission Strategy

We designed five submission strategies, which is presented in Figure 8, and the corresponding results are shown in Figure 9. The best results of ZHI-EDL System can also be found in Table 9 and 8.

| Language | Precision | Recall | F1 |
|---|---|---|---|
| Chinese | 0.813 | 0.611 | 0.698 |
| English | 0.781 | 0.603 | 0.680 |
| Spanish | 0.802 | 0.523 | 0.633 |
| Trilingual | 0.812 | 0.575 | 0.673 |

Table 8: The Official Trilingual Linking Results of ZHI-EDL System in 2017 TAC-KBP EDL Evaluation. (in terms of strong_typed_all_match)

| Language | Precision | Recall | F1 |
|---|---|---|---|
| Chinese | 0.830 | 0.624 | 0.712 |
| English | 0.788 | 0.608 | 0.686 |
| Spanish | 0.847 | 0.552 | 0.669 |
| Trilingual | 0.827 | 0.585 | 0.685 |

Table 9: The Official Trilingual Linking Results of ZHI-EDL System in 2017 TAC-KBP EDL Evaluation. (in terms of typed_mention_ceaf)

## Acknowledgments

This work was supported in part by Chinese Knowledge Center of Engineering Science and Technology (CKCEST), NSFC (No. U1611461,

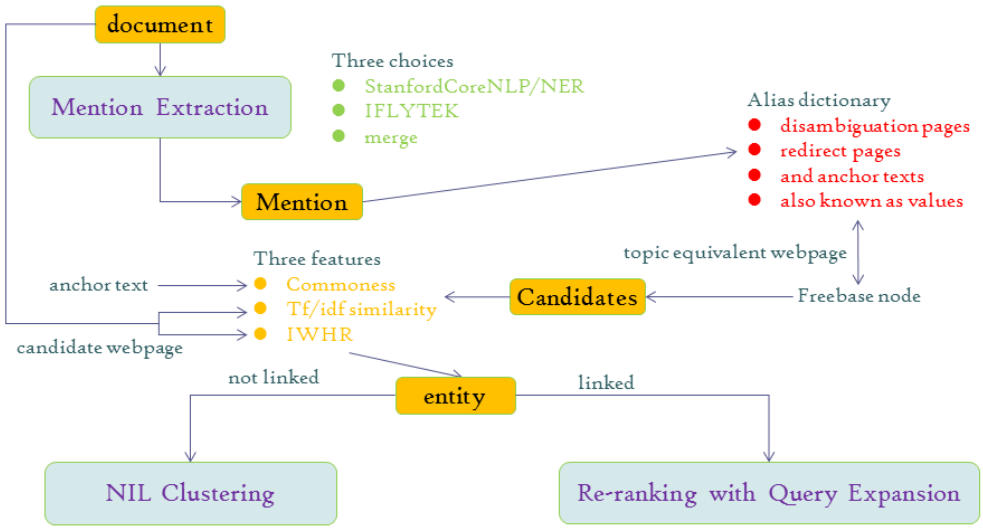


Figure 7: English Entity Linking System.

61402401), 973 program (No. 2015CB352302), Qianjiang Talents Program of Zhejiang Province 2015. Finally we would like to thank the following contributors from ZHI for their work on data collecting, transforming, rule constructing and system debugging and testing: Sheng Lin, Bo Chen, and Kai Wang.

## References

Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259* .

Silviu Cucerzan. 2007. 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 (EMNLP-CoNLL)*.

Google. 2015. Freebase data dumps. https://developers.google.com/freebase/data.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *In EMNLP*. Citeseer.

John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data .

Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *ACL (System Demonstrations)*. pages 55–60.

Olena Medelyan and Catherine Legg. 2008. Integrating cyc and wikipedia: Folksonomy meets rigorously defined common-sense .

Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Interspeech*. volume 2, page 3.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. pages 1532–1543.

Matthew E Peters, Waleed Ammar, Chandra Bhagavatula, and Russell Power. 2017. Semi-supervised sequence tagging with bidirectional language models. *arXiv preprint arXiv:1705.00108* .

Matthew D Zeiler. 2012. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701* .

Torsten Zesch, Christof Müller, and Iryna Gurevych. 2008. Extracting lexical semantic knowledge from wikipedia and wiktionary. In *LREC*. volume 8, pages 1646–1652.

| Run | | Chinese | Spanish | English |
|---|---|---|---|---|
| zju1 | Entity Discovery | Mentions detected by our model and named mentions detected by regex but part of nominal mentions are excluded by regex | Mentions detected by our model and named mentions detected by regex but all nominal mentions are excluded | Mentions detected by our model but part of nominal mentions are excluded by regex |
| | Entity Linking | Retain all nominal entity | Retain all nominal entity | Drop some NOM |
| zju2 | Entity Discovery | Mentions detected by our model and named mentions detected by regex but part of nominal mentions are excluded by regex | Mentions detected by our model and named mentions detected by regex but all nominal mentions are excluded | Mentions detected by our model and named mentions detected by regex but all nominal mentions are excluded |
| | Entity Linking | Remove all nominal entity | Retain all nominal entity and richer country extension resource | Drop some NOM |
| zju3 | Entity Discovery | Mentions detected by our model and named mentions detected by regex but all nominal mentions are excluded | Mentions detected by our model and named mentions detected by regex | Mentions detected by our model and named mentions detected by regex but all nominal mentions are excluded |
| | Entity Linking | Remove all nominal entity | Retain all nominal entity and richer country extension resource and richer Wikipedia redirection resource | Retain all nominal entity |
| zju4 | Entity Discovery | Mentions detected by our model and named mentions detected by regex but all nominal mentions are excluded | Mentions detected by our model and named mentions detected by regex | Mentions detected by our model |
| | Entity Linking | Only retain a few nominal entities which are close to the named entity and link a few mentions to a nodeID directly according to past ground-truth result. | Remove all nominal entity and richer country extension resource | Drop some NOM |
| zju5 | Entity Discovery | Mentions detected by our model | Mentions detected by our model | Mentions detected by our model but part of named and nominal mentions are excluded |
| | Entity Linking | only retain a few nominal entities which are close to the named entity and link a few mentions to a nodeID directly according to redirect table. | Remove all nominal entity and richer country extension resource and richer Wikipedia redirection resource and apply more complicated rules | Drop some NOM |

Figure 8: The ZHI-EDL System Submission Strategies.

| Run | NERLC | | | NERC | | | KBIDs | | | NER | | | CEAFmC | | | CEAFm | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | P | R | F1 | P | R | F1 | P | R | F1 | P | R | F1 | P | R | F1 | P | R | F1 |
| **TRILINGUAL** | | | | | | | | | | | | | | | | | | |
| 1 | **0.812** | 0.575 | **0.673** | **0.882** | 0.624 | 0.731 | 0.805 | **0.701** | **0.750** | **0.919** | 0.650 | 0.761 | **0.827** | 0.585 | **0.685** | **0.851** | 0.602 | **0.705** |
| 2 | 0.790 | 0.575 | 0.665 | 0.871 | 0.634 | 0.734 | 0.805 | 0.694 | 0.745 | 0.909 | 0.661 | 0.766 | 0.805 | 0.586 | 0.678 | 0.830 | 0.604 | 0.699 |
| 3 | 0.767 | **0.584** | 0.663 | 0.870 | **0.663** | **0.752** | **0.814** | 0.691 | 0.747 | 0.910 | **0.694** | **0.788** | 0.778 | **0.593** | 0.673 | 0.801 | **0.610** | 0.693 |
| 4 | 0.768 | 0.578 | 0.659 | 0.871 | 0.655 | 0.748 | 0.805 | 0.692 | 0.744 | 0.909 | 0.684 | 0.781 | 0.782 | 0.588 | 0.671 | 0.806 | 0.606 | 0.692 |
| 5 | 0.733 | 0.572 | 0.643 | 0.847 | 0.661 | 0.742 | 0.802 | 0.677 | 0.734 | 0.890 | 0.694 | 0.780 | 0.749 | 0.584 | 0.657 | 0.773 | 0.603 | 0.677 |
| **CHINESE** | | | | | | | | | | | | | | | | | | |
| 1 | 0.813 | **0.611** | **0.698** | 0.874 | 0.657 | **0.75** | **0.821** | **0.748** | **0.783** | 0.906 | **0.681** | 0.778 | 0.83 | **0.624** | **0.712** | 0.853 | **0.641** | **0.732** |
| 2 | 0.813 | 0.611 | 0.698 | 0.874 | 0.657 | 0.75 | 0.821 | 0.748 | 0.783 | 0.906 | 0.681 | 0.778 | 0.83 | 0.624 | 0.712 | 0.853 | 0.641 | 0.732 |
| 3 | **0.831** | 0.596 | 0.694 | **0.891** | 0.639 | 0.744 | 0.821 | 0.741 | 0.779 | **0.924** | 0.663 | 0.772 | **0.849** | 0.609 | 0.709 | **0.872** | 0.626 | 0.729 |
| 4 | 0.831 | 0.596 | 0.694 | 0.891 | 0.639 | 0.744 | 0.821 | 0.741 | 0.779 | 0.924 | 0.663 | 0.772 | 0.849 | 0.609 | 0.709 | 0.872 | 0.626 | 0.729 |
| 5 | 0.736 | 0.599 | 0.66 | 0.828 | **0.673** | 0.742 | 0.796 | 0.736 | 0.765 | 0.872 | 0.709 | **0.782** | 0.763 | 0.62 | 0.684 | 0.786 | 0.639 | 0.705 |
| **ENGLISH** | | | | | | | | | | | | | | | | | | |
| 1 | **0.819** | 0.574 | 0.675 | **0.894** | 0.627 | 0.737 | 0.773 | 0.723 | **0.747** | **0.934** | 0.655 | 0.77 | **0.822** | 0.576 | 0.677 | **0.848** | 0.595 | 0.699 |
| 2 | 0.781 | **0.603** | **0.68** | 0.856 | 0.66 | 0.746 | 0.729 | **0.762** | 0.745 | 0.9 | 0.694 | 0.784 | 0.788 | **0.608** | **0.686** | 0.817 | **0.63** | **0.711** |
| 3 | 0.741 | 0.599 | 0.663 | 0.853 | **0.689** | **0.762** | **0.773** | 0.723 | 0.747 | 0.905 | **0.731** | **0.809** | 0.741 | 0.598 | 0.662 | 0.768 | 0.62 | 0.686 |
| 4 | 0.781 | 0.603 | 0.68 | 0.856 | 0.66 | 0.746 | 0.729 | 0.762 | 0.745 | 0.9 | 0.694 | 0.784 | 0.788 | 0.608 | 0.686 | 0.817 | 0.63 | 0.711 |
| 5 | 0.798 | 0.581 | 0.672 | 0.867 | 0.631 | 0.73 | 0.75 | 0.726 | 0.738 | 0.91 | 0.663 | 0.767 | 0.799 | 0.582 | 0.673 | 0.829 | 0.603 | 0.698 |
| **SPAINISH** | | | | | | | | | | | | | | | | | | |
| 1 | **0.802** | 0.523 | **0.633** | **0.884** | 0.576 | 0.697 | 0.816 | **0.633** | **0.713** | **0.923** | 0.601 | 0.728 | **0.847** | 0.552 | **0.669** | **0.875** | 0.57 | **0.691** |
| 2 | 0.761 | 0.496 | 0.6 | 0.884 | 0.576 | 0.697 | **0.882** | 0.579 | 0.699 | 0.923 | 0.601 | 0.728 | 0.808 | 0.526 | 0.637 | 0.835 | 0.544 | 0.659 |
| 3 | 0.708 | **0.555** | 0.622 | 0.858 | **0.672** | **0.754** | 0.849 | 0.609 | 0.71 | 0.898 | **0.703** | **0.789** | 0.741 | **0.58** | 0.651 | 0.763 | **0.598** | 0.671 |
| 4 | 0.674 | 0.528 | 0.592 | 0.858 | 0.672 | 0.754 | 0.882 | 0.579 | 0.699 | 0.898 | 0.703 | 0.789 | 0.71 | 0.556 | 0.624 | 0.733 | 0.574 | 0.644 |
| 5 | 0.671 | 0.525 | 0.589 | 0.858 | 0.672 | 0.753 | 0.876 | 0.573 | 0.693 | 0.897 | 0.702 | 0.788 | 0.71 | 0.555 | 0.623 | 0.732 | 0.573 | 0.643 |

Figure 9: Overall Entity Discovery and Linking Performance of ZHI-EDL System on the TAC 2017 Evaluation dataset.