The ZJU_EDL System for Entity Discovery and Linking at TAC KBP 2019

Yufeng Hu[*], **Haochen Shi**[*], **Tao Chen, Siliang Tang**[†],
Quan liu, Zhigang Chen, Xiang Ren, Fei Wu, Yueting Zhuang
Zhejiang University, Hangzhou, Zhejiang, P. R. China
{xiaofeem, hcshi, tchen_ckc, siliang, wufei, yzhuang}@zju.edu.cn
iFLYTEK Research, Anhui, Hefei, P. R. China
{zgchen, quanliu}@iflytek.com
University of Southern California, Los Angeles, California, United States
xiangren@usc.edu.cn

Abstract

This report gives a detailed description of ZJU_EDL system manufacture by team ZJU_EDL, which submitted to the NIST TAC Knowledge Base Population (KBP2019) Entity Discovery and Linking Track. Our system consists of two cascaded components, use Location Model to detect the mentions from documents and Typing Model distribute a single label to the mentions with the assistant of Linking Model. Since the datasets we have are all generated from distant supervision, we also apply a series of methods to denoise our dataset.

1 Introduction

This report gives a detailed description of ZJU_EDL system manufacture by the team ZJU_EDL. As shown in figure (1), our system consists of two cascaded components, use the Location Model to detect the mentions from documents and Typing Model distribute a single label to the mentions with the assistant of the Linking Model. In the mention detection part, In the mention detection part, the system takes a document as input. It first splits the document into sentences, and further use bert extracts word-level features for each sentence, Then a deep sequence labeling model will annotate each word with I-O-B tag (Ramshaw and Marcus, 1995) to detect the span and send mention to the Typing model.

We use bert (Devlin et al., 2019) to extract the features and standard entity typing model to distribute a single type to the target mention. Since the datasets we have are all generated from distant supervision, we also apply a series of methods to denoise our dataset. simultaneously, we apply a linking system to mask the unrelated labels that generate from the typing system by a finely designed rule.

Since the datasets we have are all generated from distant supervision, we also apply a series of methods to denoise our dataset. For the noise in the typing trainset we proposed a method called recursive denoising procedure to denoise the dataset.

2 Dataset

2.1 Dataset Merge

To generate a training set, we merge our dataset from four different datasets.

- **Fine-Grained Entity Recognizer (Figer) Dataset** (Ling and Weld, 2012): Figer is a fine grain dataset that formulates the tagging problem as a multi-class, multi-label classification problem. It introduces a set of 112 overlapping entity types curated from Freebase types. However multi-label is unfit for our task, the problem happens in Figer and Ultra-fine which will be mentioned in the next item. As a result we have to refine the labels, then choose the single label instances among them. Unbalancing between types also troubles us a lot in which the top 5 types cover 80 percent of the data in Figer.

- **Ultra-fine Entity Typing Dataset** (Choi et al., 2018): The ultra-fine dataset is much more diverse and fine grained when compared to most existing datasets. The entity is divided into NOM and NAM, and the Ultra-fine dataset generated from head-word supervision is full of NOM, we use the Entity-Linking supervision dataset only.

- **Wikipedia Dataset**: In order to avoid the multi-type problem, We construct a dataset from single linking from Wikipedia, and we choose the single type instances among it.

[*]These authors contributed equally to this work and share first authorship.

[†]Corresponding author.

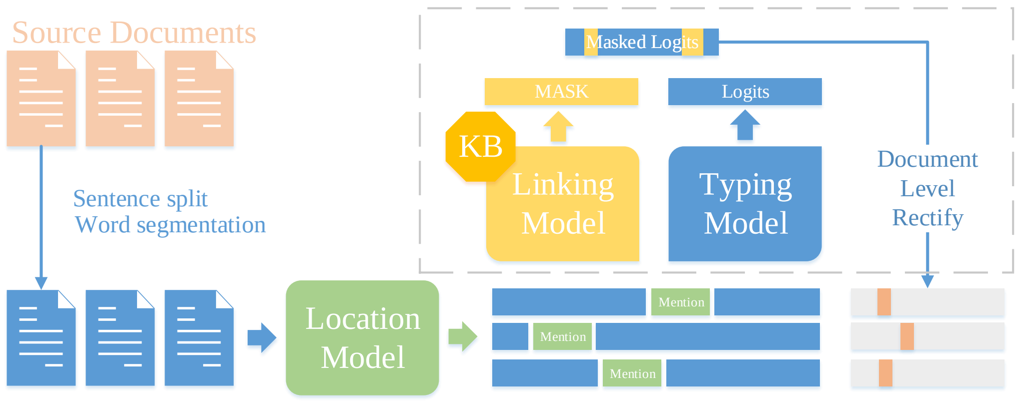

Figure 1: ZJU-EDL System Overview.

- **RPI Dataset** (Pan et al., 2017): RPI created silver-standard annotation derived from Wikipedia markups for 16K+ YAGO (Suchanek et al., 2007) entity types. Although the RPI is one of the most complete data we got, the noise in the RPI dataset is so vast that an entity could gain a lot of different kinds of labels in the dataset. In this paper, we gather the information of all RPI mention to types count, and choose the top one type as mention's type.

2.2 Type Mapping

Since the task was change from YAGO types to Active Interpretation of Disparate Alternatives (AIDA) types, we have to make a type mapping to make our dataset still usable, To convert the label space, we manually map a single label from our AIDA vocabulary to each formal-language type in the YAGO, Figer and Ultra-fine ontology. We first add up the types that appear in the dataset, and sort to get top use type that appears in the dataset, and we collect the description sentence to generate vector then use dot product similarity match as a candidate recommend type. With the information and helper of similarity score, we ask crowd annotators to annotate top types map. We make two assumptions:

Assumption 2.1 *If type A is an ancestor of type B, then the type map of A is also usable for type B. Under the guidance of this assumption, we finally map 4000 Yago types to AIDA types.*

To solve the shortage of some specific types data, we finally have to face the serious multi-type problem, for example, 'country' is always co-occur with 'government' in Figer, we have to force the type to single country for whose frequency is more often. As a result, our Map gain more than 82 percent of labels appear in AIDA labels and we finally own our new AIDA dataset with amount of 8 million.

3 Data Processing

Since the dataset is constructed through distance supervision, there is a lot of noise in the dataset for entity typing. The noise in the constructed dataset mainly consists of the following three different types of noise:

- **Wrong Entity Types**: the entities are assigned with wrong entity types as labels. For example, there is a large percentage of mention ”U.S.” in the constructed dataset is assigned the label ”Cash”, but in the context, its true label should be ”Country”.
- **Wrong Entity Name Extent**: the extent of entity names labeled in the dataset is incorrect. For example, ”the Eiffel Tower” is labeled as an entity name in the dataset, however, the correct extent should be ”Eiffel Tower”.
- **Misspelling of Entity Names**: the names of entities is misspelled.

We focus on the Wrong Entity Types noise which is the dominant one among the above three types of noise. In order to reduce the impact of this kind of noise, we proposed a method called recursive denoising procedure to denoise the dataset, which will be described in the next subsection.

3.1 Recursive Denoising Procedure

Although a certain percentage of mentions in the dataset are labeled with incorrect entity types, there

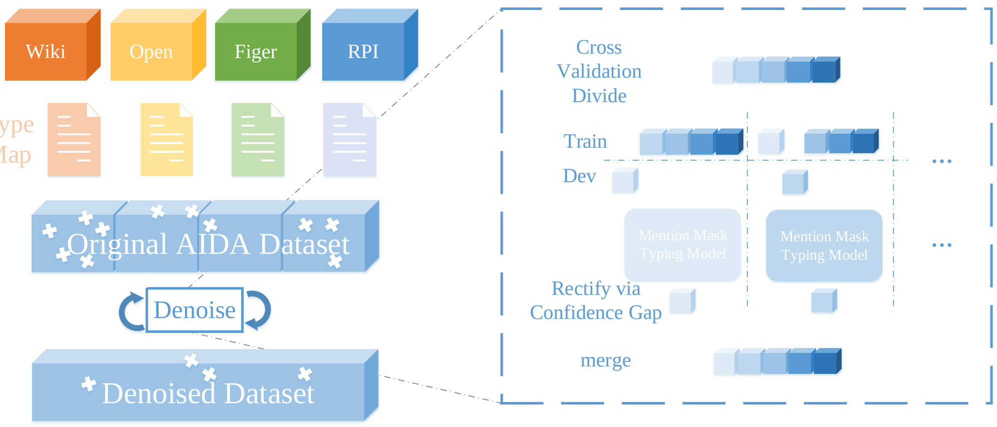

Figure 2: Dataset and Denoise.

are still a large number of correctly labeled examples for each entity type. Named entities of the same entity type often appear in similar contexts and act as homogeneous semantic roles, so we can use the context of these entities to correct the labels of examples that are mislabeled. For instance, there are many instances of ”U.S.” in the dataset that should be labeled ”Country” being incorrectly labeled ”Cash”, but there are more mentions with context similar to ”U.S.” correctly labeled as ”Country”. To do so, we trained an entity typing model by masking the mentions with a high probability to force the model to make predictions using the context, so that the model can be used to correct the incorrect labels of the examples. Figure 2 illustrates the process, which is described as follows:

1. For every example e_i in an entity typing dataset D, the sentence, mention string and labeled entity type of e_i are denoted as s_i, x_i, y_{label_i} respectively. Let current dataset D_c be the origin entity typing dataset D_s and n be the number of all possible entity types.

2. Split D_c into k-folds (k=5 in our experiments) sets $\{D_1, \ldots, D_k\}$ by tuple (x_i, y_{label_i}), so that there is no identical (x_i, y_{label_i}) in each fold.

3. Do k-fold cross-validation on $\{D_1, \ldots, D_k\}$ by using each fold D_i $(i = 1, \ldots, k)$ once as validation set while the k-1 remaining folds form the training set. During the training and validation stages of each round of the k-fold cross-validation, the mention x of every example e is masked. For each example e in D_i with label $y_{label_j}(j = 1, \ldots, n)$, the trained entity typing model will give the probability of each entity type. If the probability p_{pred} of predicted entity type y_{pred} and the probability p_{label} of labeled entity type y_{label} satisfy:

$$p_{pred} - p_{label} >= g_j$$

where $g_j(j = 1, \ldots, n)$ is a predefined gap for each entity type, we change the label of e to y_{pred}. After doing so, we could get new k-folds sets $\{D'_1, \ldots, D'_k\}$. let $D_c = \sum_{i=1}^{k} D'_i$

4. Repeat step2, 3 for 2 times. Let the finall denoised dataset D_d be D_c.

3.2 Entity Typing Model

Our entity typing model aimed to generate classification results from the combined input of mention and its context. For an instance, ’China’ is the detected mention with context ’He is from China’, model will tell us ’China’ is a ’Country’ based on specific context.

In this sense, our model needs to pay more attention to construct the representations of mention and context.

Bert model is responsible for generating origin simple representations of mention and context independently, and then more complicate mechanisms like attention (Vaswani et al., 2017), fusion (Xiong et al., 2019) will be introduced to form more accurate representations.

Once we get the attentive representation, a simple fully connected layer will be adopted as classification decoder, and some hierarchy loss (Xu and Barbosa, 2018) can be added to model complex hierarchy structure.

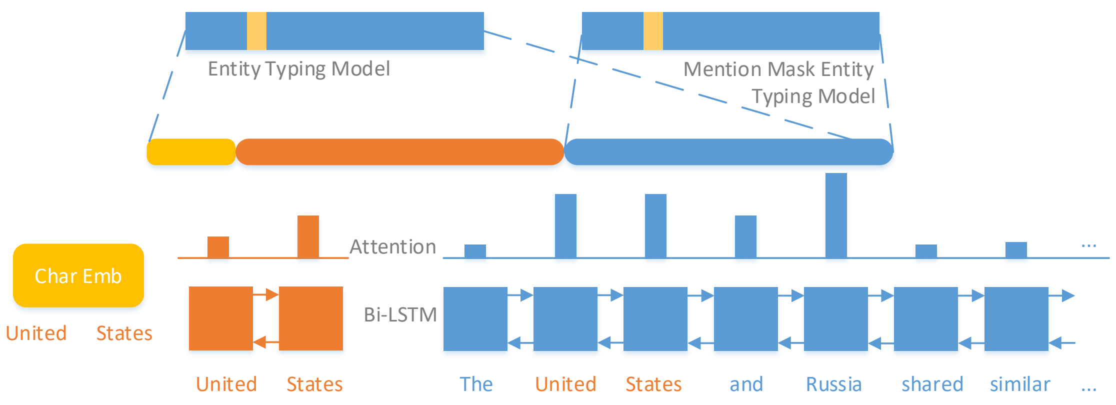

Figure 3: Entity Typing Model and Mention Mask Entity Typing Model.

Similar training strategies are adopted like previous Bert entity recognition model.

Besides, the entity model also plays the role of data denoising model. During the denoising process, the only difference is that the mention is masked, more details can be refered to sec 3.1.

4 Mention Detection and Denoising

Bert Entity Recognition: our entity recognition model inherits the classic named entity recognition architecture which uses BiLSTM (Schuster and Paliwal, 1997) CRF (conditional random field) (Lafferty et al., 2001) to generate sequence output of ”BIO” tags from origin word sequence input. With regard to this entity recognition task, we only need to find the boundaries of the entity without having to judge the specific type of the entity. For example, if origin word sequence is ”He is from China”, model will generate an output like ”O O O B”, which pointed out the word ”China” is a named entity. In order to introduce the powerful representation ability of pretrained model, we use Bert as our word representation instead of simple word vector, and followed by the classic Bi-LSTM CRF architecture.

In the training stage, we firstly only train the Bi-LSTM CRF framework with a larger learning rate, and then train both Bert and Bi-LSTM CRF with a smaller learning rate jointly.

In the stage for practical application, python package Spacy is added to assist entity recognition. Just take the previous sentence as an example, ”He is from China” will be firstly tokenized by Spacy to generate several tokens, ’He’, ’is’, ’from’ and ’China’, both our Bert model and Spacy model will give their options about ’BIO’ tags of the tokens.

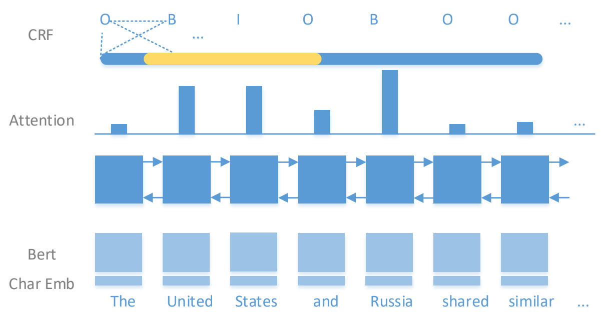

Figure 4: Mention Detection Model.

According to our observation results, Spacy (Honnibal and Montani, 2017) will have a lower recall but a higher accuracy than our Bert model. So the results generated from Spacy are used to fix Bert model’s results, some issues such as wrong entity extents can be solved in this way. After some rule-based method like this, final recognition result will be given out.

5 Linking and RPI Search Engine

Entity linking and RPI search engine also play a vital role in assisting entity typing.

Linking model tries to link the simple mention to a trusted named entity. For example, mention ’China’ will be linked to country entity ’China’. It can be seen that the linking process has considered both the mention itself and its specific context, so the linking result should have a high confidence. Once an entity is linked, the provided table(yago_at_least_10.json) which maps entity to its types will be used to get the entity’s type set mask.

Sometimes we will encounter the situation that one mention can't be linked to a specific entity, so it's necessary to get a type mask from another way. We notice a pretty large corpus called RPI was released as optional material, but this corpus is not suitable for training data because of its noise, and not reliable for using because of its data size. So we develop a search engine under these circumstances, the search engine will tell us which types can the mention be founded, in other words, this information can be used as an additional type mask as well.

6 Submission Strategy

We basically designed two submission strategies. The first submission is the best of our model, The second is based on the first one, we add document level information by electing the type with the highest score through all the same mention in the same document as mention's final type.

The best results of ZJU-EDL System of two evaluation windows can be found in Table 1.

Window	P	R	score
Window 1	0.375	0.418	0.395
Window 2	0.423	0.472	0.446

Table 1: The Official Results of ZJU-EDL System in 2019 TAC-KBP EDL Evaluation.

7 Acknowledgements

This work has been supported in part by National Key Research and Development Program of China (SQ2018AAA010010), NSFC (No.61751209, U1611461), Zhejiang University iFLYTEK Joint Research Center, Chinese Knowledge Center of Engineering Science and Technology (CKCEST), Engineering Research Center of Digital Library, Ministry of Education. FInally we would like to thank the following contributors from ZJU for their work on dataset collecting, system debugging and testing: Sheng Lin, Bo Chen and Luye Zheng.

References

Eunsol Choi, Omer Levy, Yejin Choi, and Luke Zettlemoyer. 2018. Ultra-fine entity typing. *arXiv preprint arXiv:1807.04905*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Xiao Ling and Daniel S Weld. 2012. Fine-grained entity recognition. In *Twenty-Sixth AAAI Conference on Artificial Intelligence*.

Xiaoman Pan, Boliang Zhang, Jonathan May, Joel Nothman, Kevin Knight, and Heng Ji. 2017. Cross-lingual name tagging and linking for 282 languages. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1946–1958, Vancouver, Canada. Association for Computational Linguistics.

Lance Ramshaw and Mitch Marcus. 1995. Text chunking using transformation-based learning. In *Third Workshop on Very Large Corpora*.

M. Schuster and K.K. Paliwal. 1997. Bidirectional recurrent neural networks. *Trans. Sig. Proc.*, 45(11):2673–2681.

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 '07, pages 697–706, New York, NY, USA. ACM.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Wenhan Xiong, Jiawei Wu, Deren Lei, Mo Yu, Shiyu Chang, Xiaoxiao Guo, and William Yang Wang. 2019. Imposing label-relational inductive bias for extremely fine-grained entity typing. *arXiv preprint arXiv:1903.02591*.

Peng Xu and Denilson Barbosa. 2018. Neural fine-grained entity type classification with hierarchy-aware loss. *arXiv preprint arXiv:1803.03378*.