Diffbot's Fine-Grained Entity Typing System for TAC KBP 2019

Zhaochen Guo and **Chunliang Lyu**

Diffbot Inc.

{zhaochen,chunliang}@diffbot.com

Abstract

In this report, we give a detailed description of Diffbot's system for the TAC-KBP 2019 Fine-Grained Entity Typing track. Fine-grained entity typing is a task to extract mentions of pre-defined types and assign fine-grained types to those mentions. Instead of modeling the task as a multi-label classification problem, we tackle it using Entity Linking (EL) by first linking mentions to their referent entities in a Knowledge Base (KB) and then inferring the fine-grained types of mentions based on the linking results and type ontology.

1 Entity Linking

Our Entity Linking component is to detect mentions from given texts, and then disambiguate them to their referent entities in a KB. It mainly consists of two steps: *Mention Detection* and *Mention Disambiguation*

1.1 Mention Detection

Our Mention Detection uses a Bi-LSTM-CRF based model for Named Entity Recognition (NER) (Huang et al., 2015), which detects named entities and assigns them with coarse types such as *Person*, *Organization*, *GPE*, and *Misc*. To address the name variation issue and increase the recall of mention detection, we also exploit a mention dictionary collected from various sources in Wikipedia (mainly Wikipedia title and wikilinks), with which we do an exhaustive search for names in the dictionary. Our mention detection can achieve a recall of 99% on some datasets such as TAC-KBP 2010[1], with some noisy mentions introduced.

[1] https://tac.nist.gov/2010/

1.2 Mention Disambiguation

For the disambiguation part, we employ Gradient Boosting Decision Tree (GDBT) to incorporate features from different categories. Figure 1 lists the major features used in our disambiguation system.

Lexical	Name related similarity
Statistical	$Prior(e\|m)$ $InLinks(e)$ $OutLinks(e)$
Contextual	Word context similarity Category similarity
Semantic	Word-Word embedding similarity Word-Entity embedding similarity Entity-Entity embedding similarity

Table 1: Features for mention disambiguation

For the embedding based similarity, we use the embedding generated from Wikipedia2Vec (Yamada et al., 2018) which maps the representation of words and entities into the same vector space. Our system can achieve 89% accuracy on the TAC-KBP 2010 EDL dataset, on par with the state-of-the-art.

2 Entity Typing

2.1 Type Ontology

To infer the types of mentions, we will need to map each entity in the KB to their types in the AIDA ontology. We organize the type ontology in a tree structure, with the top nodes being the most generic types such as *PER*, *ORG*, or *FAC*. and the leaf nodes are the entities. We use the YAGO AIDA type mapping to map each entity to their AIDA types.

Figure 1 gives an example type ontology. As shown, some entities (e.g. *New York Freedom*) belong to multiple types, which posits a challenge for the type inference.

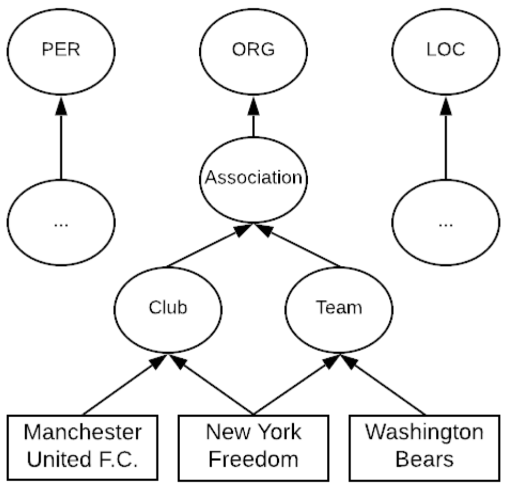

Figure 1: An example type ontology.

2.2 Type Inference

There are mainly three cases for each mention during type inference. The first one is when a mention is linked to an Out-of-KB entity (**NIL**), in which no type is available from the KB. In this case, we use the coarse type obtained from *Mention Detection*. The second case is the referent entity of a mention has only one type, for which we use the type as the inferred type for the mention. The third case is to choose the most relevant type among the multiple types of the referent entity.

Our first approach was to pick the type with the largest number of entities; however, it made too many false positive errors since it doesn't consider the semantics of types. Instead, we choose k entities [2] for each type to capture the semantics of the type, assuming that the probability an entity belongs to a type can be approximated by its semantic similarity to the type's representative entities.

For each type, let $E(type)$ be the set of representative entities.

$$E(type) = \{e_i\}_{1 \leq i \leq k}$$

We then measure the semantic similarity $sem\_sim(e_m, type)$ between the referent entity e_m and $type$ as follows:

$$sem\_sim(e_m, type) = \frac{\sum_{e_i \in E} sem\_sim(e_m, e_i)}{k}$$

Here the semantic similarity $sem\_sim(e_m, e_i)$ between entities is measured by the dot product of entity embeddings of e_m and e_i.

[2]k is set to 10 experimentally

The criteria to pick the k entities for each type is based on popularity, measured by their PageRank value computed on the Wikipedia link graph [3]. Our results indicate that this approach can greatly improve the accuracy of entity typing. To alleviate the Out-of-KB issue and infer the fine-grained types of mentions, an idea we would like to explore is to measure the semantic similarity between the contextual words of a mention and the representative entities of a type, and select the type with the highest similarity.

3 Results

We submitted 1 run for both the feedback and the final evaluation. Our system gets 0.422 precision, 0.358 recall, and 0.388 F1 on the final evaluation.

3.1 Error Analysis

We manually inspected the results and found a few issues in our entity typing system. The biggest one is from the mention detection component which introduced many extraneous mentions. This is a common issue in most entity typing systems, and can be further improved by better mention detection approaches. Regarding our EL-based typing approach, there are mainly two challenges: 1) errors propagated from wrong EL results, and 2) the types of mentions that are linked to Out-of-KB entities, in which case we only report the coarse types inferred from NER. When mentions are correctly linked to their referent entities, our system can accurately infer the fine-grained types of mentions.

4 Conclusion

This report described our fine-grained entity typing system and analyzed the advantages and challenges of our system. We found that a highly accurate entity linking system could help with the entity typing, with some performance sacrifice from wrong EL results and missing types of the Out-of-KB entities. On the other hand, we believe that entity linking systems can be used to generate high quality of training datasets for the entity typing task, especially with the high cost of data annotation.

[3]`https://github.com/athalhammer/danker`

References

Ikuya Yamada, Akari Asai, Hiroyuki Shindo, Hideaki Takeda, and Yoshiyasu Takefuji 2018. *Wikipedia2Vec: An Optimized Tool for Learning Embeddings of Words and Entities from Wikipedia*. arXiv preprint 1812.06280

Zhiheng Huang, Wei Xu, and Kai Yu 2015. *Bidirectional LSTM-CRF Models for Sequence Tagging*. arXiv:1508.01991