HITS-UKP at TAC KBP 2019: Entity Discovery and Linking Track

Jan-Christoph Klie[1*] **Federico López**[2*] **Iryna Gurevych**[1] **Michael Strube**[2]
[1]Ubiquitous Knowledge Processing Lab [2]Research Training Group AIPHES
Technische Universität Darmstadt Heidelberg Institute for Theoretical Studies

Abstract

HITS and the UKP Lab participated in the Entity Discovery and Linking Track at TAC KBP 2019. The main tasks were Named Entity Detection and Fine Grained Entity Typing, with a large inventory of entity types provided by DARPA AIDA (Active Interpretation of Disparate Alternatives). To address the task we apply a pipeline setup of named entity detection followed by entity typing, which aims to profit from the hierarchical relations present on the type inventory. We provide a description of our modular system and preliminary results on silver data.

1 System Overview

The tasks of named entity detection (NED) and entity typing can be approached separately, therefore we opted for a two-step pipeline architecture instead of an end-to-end system. In this way, we are able to exploit large NER-annotated corpora to train a mention detector and then use a different dataset for the entity typing module for which data is scarcer.

2 Mention Detection Module

For mention detection, we train a sequence tagger based on a BiLSTM with a Conditional Random Field classifier (Huang et al., 2015) using Glove (Pennington et al., 2014) and character embeddings.

2.1 Implementation and Resources

For training, we use OntoNotes (Pradhan et al., 2013), as it is a large corpus and contains documents of diverse genres and domains. We only keep entity types that we expect to be in the target data, therefore annotations for `TIME`, `ORDINAL`, `CARDINAL`, `MONEY`, `QUANTITY`, `DATE`, and `PERCENT` are removed.

*Authors contributed equally

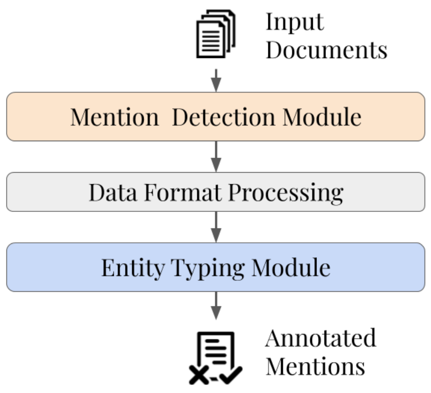

Figure 1: Overview of the pipeline setup of the system.

2.2 Data Format Processing

As we were only interested in detecting mentions and not specific types, we only train on the coarse `BIO` tags, e.g. replacing `B-LOC` with `B` and `I-LOC` with `I`.

The entity typing module is designed to assign a type to a single mention. Therefore, if a sentence contains N mentions, we split it into N times the same sentence with one different annotated mention each. We filter out sentences without any mentions.

2.3 Results

Results on the OntoNotes dataset can be seen on Table 1.

Splits	Acc	P	R	F1
Train	0.995	0.970	0.969	0.970
Dev	0.986	0.927	0.921	0.924

Table 1: Performance of the mention detection module on the OntoNotes dataset.

3 Entity Typing Module

The task under consideration for the entity typing module is, given a context sentence c containing an entity mention m, predict the correct type labels t_m that describe m from a type inventory T.

For TAC EDL 2019 the defined type inventory comes from the DARPA AIDA program. This inventory is organized as a hierarchy, ranging from general, *coarse* types such as “person” near the top, to more specific, *fine* types such as “politician” in the middle, to even more specific, *ultrafine* entity types such as “governor” at the bottom. The hierarchy has multiple roots such as ”person”, ”facility”, ”organization” and more.

Although the track guidelines specify that only one label per mention is required, we cast the task as a multi-class multi-label classification problem, in order to take advantage from the hierarchical relations present in the type inventory T.

3.1 Model

The developed entity typing module is based on BERT (Devlin et al., 2019). We apply BERT in an analogous way to the usage for sentence pair classification. Following the approach of Onoe and Durrett (2019), we adapt the typing task to BERT by forming an input sequence "`[CLS] sentence [SEP] mention [SEP]`" and assign the segment embedding A to the sentence and B to the mention span. Then, we take the output vector at the position of the `[CLS]` token (i.e., the first token) as the feature vector $\mathbf{v}$ to feed the classifier.

3.2 Exploiting Hierarchical Information

As noted before, the type inventory is organized as a hierarchy. Following López et al. (2019), we hypothesize that by virtue of such a hierarchy, a model learning about ”governors” will be able to transfer this knowledge to related entities such as ”politicians”. Thus, we aim to predict not only the most fine-grained type but the complete path from the root to the bottom of the hierarchy, casting the task as a multi-class multi-label classification problem.

We do not set an upper bound on the number of predicted types, hence, for every instance, the model returns a reduced set of candidate labels. This set tends to contain three elements given that most of the training data is annotated in accordance with the AIDA ontology, which has three levels of granularity. From this set we shall choose only one label for the final prediction, since TAC EDL 2019 guidelines stipulate one predicted label per mention. This offers a wide range of possible criteria to be applied in order to select the final label that is assigned to the mention.

Due to time constraints, we only experiment with two:

Most fine-grained prediction: the simplest option is to take the most fine-grained label, since according to the guidelines of the task that is the preferred granularity.

Most specific common superclass: in this case, we focus on coherent predictions of the classifier. Therefore, we retrieve the most fine-grained label of the most coherent prediction. This is:

- If all the labels belong to the same *path* of the hierarchy (*i.e.* they belong to the same branch of the hierarchy tree), we take the most fine-grained one. Ex: if the types predicted are $\{$*org, org.association, org.association.club*$\}$, we select *org.association.club*.

- In case of different paths being predicted, we choose the most fine-grained type of the most coherent predicted path. Ex: if the prediction is $\{$*fac, fac.building, org.association.club*$\}$, we select *fac.building*.

- When different paths (belonging to different roots) of the same length are predicted, the selection is random among the longest paths.

3.3 Data

To train the model we used a dataset derived from Wikipedia markups collected by Pan et al. (2017). The type ontology used to annotate this dataset comes from YAGO (Suchanek et al., 2007). To obtain data annotated with the target type hierarchy we convert YAGO types to AIDA types following the mapping provided by the task organizers[1].

From this annotated data, we randomly sample 5,000 sentences per type as train set. To generate development and test sets, we only incorporate sentences whose mention is not present on the train set. This is, if a sentence on the train set contains the mention ”Picasso”, then no instance in the development or test set will contain this exact same word as mention. In this way, we are able to evaluate the

[1]http://nlp.cs.rpi.edu/kbp/2019/YAGO_AIDA_mapping.xlsx

Granularity	P	R	F1
Coarse	90.4	90.7	90.6
Fine	83.9	79.7	81.7
UltraFine	60.8	75.3	67.3
Total	78.5	83.1	80.7

Table 2: Performance of the typing module over our test dataset.

generalization of the model beyond its ability to memorize instances.

3.4 Experiments

Since the system is build as a modular pipeline, it allows us to easily experiment with the Entity Typing module by itself, detached from the Mention Detection component. We train and evaluate over the typing data described in the preceding section.

We use the pre-trained BERT-Base, uncased model[2]. We trained the model for 50 epochs with a learning rate of $2e-5$ and batch size 128. We optimize the total F1-score on the validation set, and evaluate on the test set.

3.5 Preliminary Results on Team Data

The AIDA typing hierarchy has three levels of granularity, which we denominate, from general to specific, as *Coarse*, *Fine* and *UltraFine*. The task evaluation assigns a partial score for missed annotations that belong to the same branch of the hierarchy. Therefore, it could be more beneficial to increase the amount of "safe" predictions over the *coarse* types than over the most fine-grained ones.

We evaluate the capacity of the model to predict the complete set of labels when the data is annotated using the full path of the AIDA hierarchy. In Table 2, we report the performance of our model over the test set for each granularity. Given this results, we consider the Macro-F1 score for the *UltraFine* types appropriate to aim for this level of prediction, instead of falling back to upper levels of the hierarchy.

4 Results on Annotated Evaluation Data

We evaluate our system predictions on the 404 documents in the 2019 EDL evaluation source corpus for which gold annotations were provided. We convert our system predictions and gold data from the LDC tab format and then run *neleval*[3] to compute the scores. These are presented in Table 3. It must be noted that this score does not include partial credit, that is if a too fine type was predicted, as the shared task scores will award. The presented scores are preliminary, the official scores will be provided once available.

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

	Mention F1	Typing F1
Run 1	60.1	16.0
Run 2	68.3	18.8
Run 3	68.3	18.4

Table 3: Results on task evaluation data for the three submissions of the HITS-UKP team.

Our first run used a mention detector trained on all mentions of OntoNotes, including numbers and dates. This resulted in very high recall but low precision since the task requires only named entities, and it also added a large amount of noise for the entity typing module. For the subsequent runs, we trained our mention detection so that it only detects named entities that were also to be expected in the evaluation data (see Sec. 2.2). This can be seen in the increase of the Mention detection F1 score by a 13.6%.

4.1 Analysis on Evaluation Data

We conduct an analysis on the predictions over the $300,000$ documents of the evaluation set provided by the task organizers. The types used to annotate the data were defined for the purpose of representing salient information over the 2014-2015 Russia-Ukraine conflict. The main genre of the texts in this dataset are news wires of a wide range of topics.

Table 4 shows the most frequently predicted types. As expected, we see that `per` (person) and `gpe.country.country` are among the top which is expected given the genre of the dataset.

Since most of the news wire texts start with the name of the news agency that is reporting it, one of the main named entities that we were able to recognize is `org.commercialorganization.newsagency`.

Furthermore, by performing a manual inspection over the source documents we found many related to sport news. We see this as the cause for the high number of predictions of the entity type `org.association.team`. For the named en-

[3]https://github.com/wikilinks/neleval

Type	%
gpe.country.country	7.94
per	7.07
per.professionalposition	5.47
fac	3.92
org	3.45
per.professionalposition.minister	2.64
org.association.team	2.61
loc.land.continent	2.52
org.commercialorganization.newsagency	2.38
org.commercialorganization.manufacturer	2.20

Table 4: Most frequent types as predicted by the entity type.

Granularity	%
Coarse	18.38
Fine	20.99
UltraFine	60.63

Table 5: Percentage of mentions annotated with a certain granularity of type.

tities related to athletes, usually the typing module fails to find a suitable annotation (beyond `person`), since there are no entity types related to sports persons.

In Table 5 we report the distribution of the labels over the three different granularities that can be predicted. As expected from the labeling strategies, the most fine-grained granularity tends to be the preferred one, and this selection criteria is reflected on the the metric.

The Entity Typing module was trained on data collected from Wikipedia, which does not fully match the writing style of news wires, which we hypothesize might have degraded the performance in many cases.

5 Conclusions

By applying a pipeline approach, we achieve a modular implementation that is easier to train and test, and which requires minor data-format processing steps between modules. The detached training capability becomes crucial when dealing with tasks where the data availability is so dissimilar. Furthermore, the model is able to capture hierarchical relations in the type inventory with a modification on the prediction schema, and different criteria to choose a final label can be seamlessly integrated on the entity typing module.

Acknowledgements

This work has been supported by the DFG-funded research training group “Adaptive Preparation of Information form Heterogeneous Sources” (AIPHES, GRK 1994/1) as well as by the German Research Foundation under grant No EC 503/1-1 and GU 798/21-1 and the Klaus Tschira Foundation, Heidelberg, Germany.

References

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.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging. *CoRR*, abs/1508.01991.

Federico López, Benjamin Heinzerling, and Michael Strube. 2019. Fine-grained entity typing in hyperbolic space. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, pages 169–180, Florence, Italy. Association for Computational Linguistics.

Yasumasa Onoe and Greg Durrett. 2019. Learning to denoise distantly-labeled data for entity typing. 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 2407–2417, Minneapolis, Minnesota. Association for Computational Linguistics.

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.

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, Doha, Qatar. Association for Computational Linguistics.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Hwee Tou Ng, Anders Björkelund, Olga Uryupina, Yuchen Zhang, and Zhi Zhong. 2013. Towards robust linguistic analysis using OntoNotes. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 143–152, Sofia, Bulgaria. Association for Computational Linguistics.

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.