Assignment 1: Fine-grained Entity Typing

Last Modified: 09/25/2020

In this assignment, you’ll be implementing a fine-grained entity typing model.

**Data**

<https://drive.google.com/file/d/1sx3d9J5hFha_o7xWxw08EixracAAoc5g/view?usp=sharing>

**Format**

The data files are in JSON format where each line represents a JSON object.

Example:

|  |
| --- |
| {"tokens": ["It", "was", "created", "from", "part", "of", "the", "province", "of", "Normandie", "."], "annotations": [{"mention": "Normandie", "mention\_id": "90497-0", "start": 9, "end": 10, "labels": ["Country108544813", "Region108630039", "State108654360", "Region108630985"]}]} |

tokens: a list of token strings.

annotations: a list of mentions, where each item has the following fields:

* mention: mention text string
* mention\_id: a unique ID assigned to each mention
* start: start token index (inclusive)
* end: end token index (exclusive)
* labels: a list of fine-grained entity labels

**Task**

1. (8-12 pt) Implement a fine-grained entity typing model based on previous work or new ideas. Train your model on en.train.json, select the best model based on scores on en.dev.json, and test the model on en.test.json. Report macro-F1, micro-F1.
   1. As the training set is large, you may downsample the training set.
   2. To avoid out-of-memory errors, you can implement a “lazy” dataloader (load the dataset in a streaming way).

We will rank all submitted systems based on F-score, and assign points between 8-12 based on the rank.

1. (3 pt) Analyze and categorize remaining errors produced by the model and propose at least three possible solutions.
2. (3 pt) Write a clear and informative written report about your methods, results, and findings.

**Evaluation Metric:**

def calculate\_macro\_fscore(golds: List[List[int]],

preds: List[List[int]]

) -> Tuple[float, float, float]:

"""Calculate Macro F-score.

Args:

golds (List[List[int]]): Ground truth. The j-th element in the i-th

list indicates whether the j-th label is associated with the i-th

entity or not. If it is 1, the entity is annotated with the j-th

label. If it is 0, the j-th label is not assigned to the entity.

preds (List[List[int]]): Prediction. The j-th element in the i-th

list indicates whether the j-th label is predicted for the i-th

entity or not.

Returns:

Tuple[float, float, float]: Precision, recall, and F-score.

"""

total\_gold\_num = total\_pred\_num = 0

precision = recall = 0

for gold, pred in zip(golds, preds):

gold\_num = sum(gold)

pred\_num = sum(pred)

total\_gold\_num += (1 if gold\_num > 0 else 0)

total\_pred\_num += (1 if pred\_num > 0 else 0)

overlap = sum([i and j for i, j in zip(gold, pred)])

precision += (0 if pred\_num == 0 else overlap / pred\_num)

recall += (0 if gold\_num == 0 else overlap / gold\_num)

precision = precision / total\_pred\_num if total\_pred\_num else 0

recall = recall / total\_gold\_num if total\_gold\_num else 0

fscore = 0 if precision + recall == 0 else \

2.0 \* (precision \* recall) / (precision + recall)

return precision \* 100.0, recall \* 100.0, fscore \* 100.0

def calculate\_micro\_fscore(golds: List[List[int]],

preds: List[List[int]]

):

# Calculate Micro F-score.

total\_gold\_num = total\_pred\_num = overlap = 0

for gold, pred in zip(golds, preds):

total\_gold\_num += sum(gold)

total\_pred\_num += sum(pred)

overlap += sum([i and j for i, j in zip(gold, pred)])

precision = 0 if total\_pred\_num == 0 else overlap / total\_pred\_num

recall = 0 if total\_gold\_num == 0 else overlap / total\_gold\_num

fscore = 0 if precision + recall == 0 else \

2.0 \* (precision \* recall) / (precision + recall)

return fscore \* 100.0

**Additional Data:**

KBP: <https://drive.google.com/file/d/143SmLZCagojkSVD3qOg6z4wkk-cxLOrO/view?usp=sharing>

Original data (no downsampling): <https://drive.google.com/file/d/1xstlGlkv9arrBzmWjVx4cplR8RQ8N5fp/view?usp=sharing>

**Tools:**

* Fine-grained entity recognition: <https://github.com/xiaoling/figer>
* Ultra-fine grained entity typing: <https://homes.cs.washington.edu/~eunsol/open_entity.html>
* Neural Entity Typing with Knowledge Attention: <https://github.com/thunlp/KNET>
* <https://github.com/guillaumegenthial/sequence_tagging> (Tensorflow)
* <https://github.com/threelittlemonkeys/lstm-crf-pytorch>  
  <https://github.com/LiyuanLucasLiu/LM-LSTM-CRF>

**References**

1. Xiao Ling and Daniel S Weld. 2012. Fine-grained entity recognition. In Proceedings of the 26th AAAI Conference on Artificial Intelligence (AAAI 2012).
2. Dan Gillick, Nevena Lazic, Kuzman Ganchev, Jesse Kirchner, and David Huynh. 2014. Context-dependent fine-grained entity type tagging. arXiv preprint arXiv:1412.1820.
3. Dani Yogatama, Daniel Gillick, and Nevena Lazic. 2015. Embedding methods for fine grained entity type classification. In Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (ACL 2015).
4. Xiang Ren, Wenqi He, Meng Qu, Lifu Huang, Heng Ji, and Jiawei Han. 2016a. AFET: Automatic fine-grained entity typing by hierarchical partial-label embedding. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP 2016).
5. Xiang Ren, Wenqi He, Meng Qu, Clare R Voss, Heng Ji, and Jiawei Han. 2016b. Label noise reduction in entity typing by heterogeneous partial-label embedding. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (SIGKDD 2016).
6. Sonse Shimaoka, Pontus Stenetorp, Kentaro Inui, and Sebastian Riedel. 2016. An attentive neural architecture for fine-grained entity type classification. In Proceedings of the 5th Workshop on Automated Knowledge Base Construction.
7. Sonse Shimaoka, Pontus Stenetorp, Kentaro Inui, and Sebastian Riedel. 2017. Neural architectures for fine-grained entity type classification. In Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2017).
8. Eunsol Choi, Omer Levy, Yejin Choi, and Luke Zettlemoyer. 2018. Ultra-fine entity typing. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL 2018).
9. Ji Xin, Yankai Lin, Zhiyuan Liu, and Maosong Sun. 2018. Improving neural fine-grained entity typing with knowledge attention. In Proceedings of the 32nd AAAI Conference on Artificial Intelligence (AAAI 2018).
10. Peng Xu and Denilson Barbosa. 2018. Neural fine-grained entity type classification with hierarchy-aware loss. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL HLT 2018).