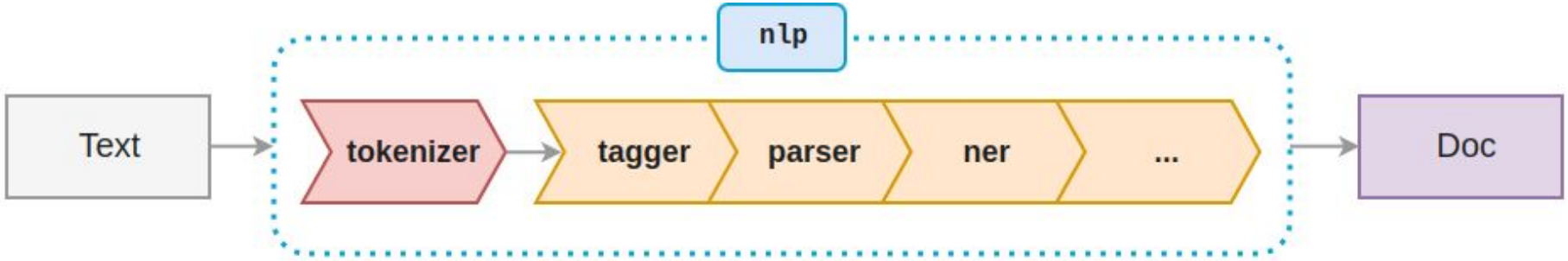
Train Spacy NER model using the GermaNER dataset

Shang-Ching Liu
Lijunnan Bai
Xiaoqui Qin

Contents

What the Task is?[1]

- Train a new German Spacy NER model
- Fine tune the pretrained German NER model in Spacy
- Evaluate Fine-tuning model

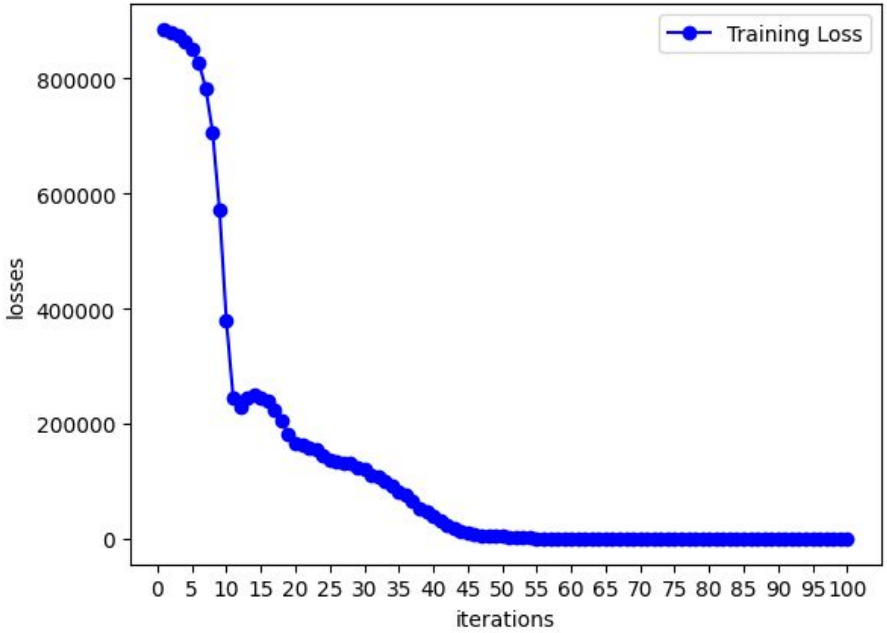


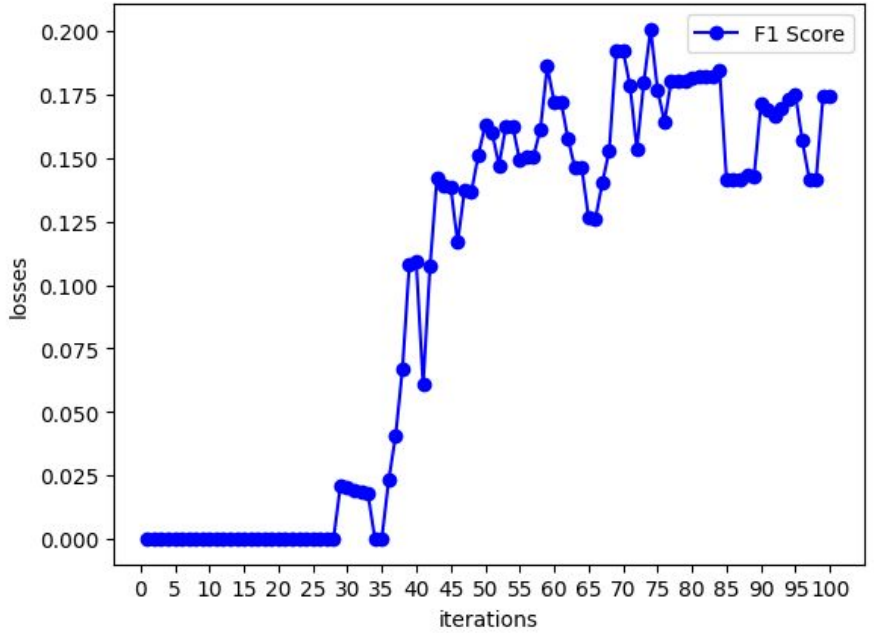
- Note Device:
  - AMD Ryzen 9 5950X 16-Core Processor
  - NVIDIA GeForce RTX 3090

Task 1: Train new “NER” model

- Settings
  - Epoch: 100
  - Batch size : 50
- Data Preprocessing
  - Input: #\n1\t”word1”\t’O’\t’O’\n\n2\t”word1”\t’O’\t’O’\n…etc
  - Output: [(“Sentence”, “entities”:[(char_start, char_end, pattern)])]
  - Note: Don’t use CSV reader to read TSV file in python, directly open it up and using split()
- GPU Prefer
  - *spacy.prefer_gpu()*
- Training[4]
  - Create blank model: *nlp = spacy.blank('de')*
  - Using optimizer: *optimizer = nlp.initialize()* or *optimizer = nlp.resume_training()*
  - Frozen other pipes: *with nlp.disable_pipes(*other_pipes):*
  - Update model: nlp.update(examples,dropout, optimizer, losses)

Task 1 Result





Task 1 Result

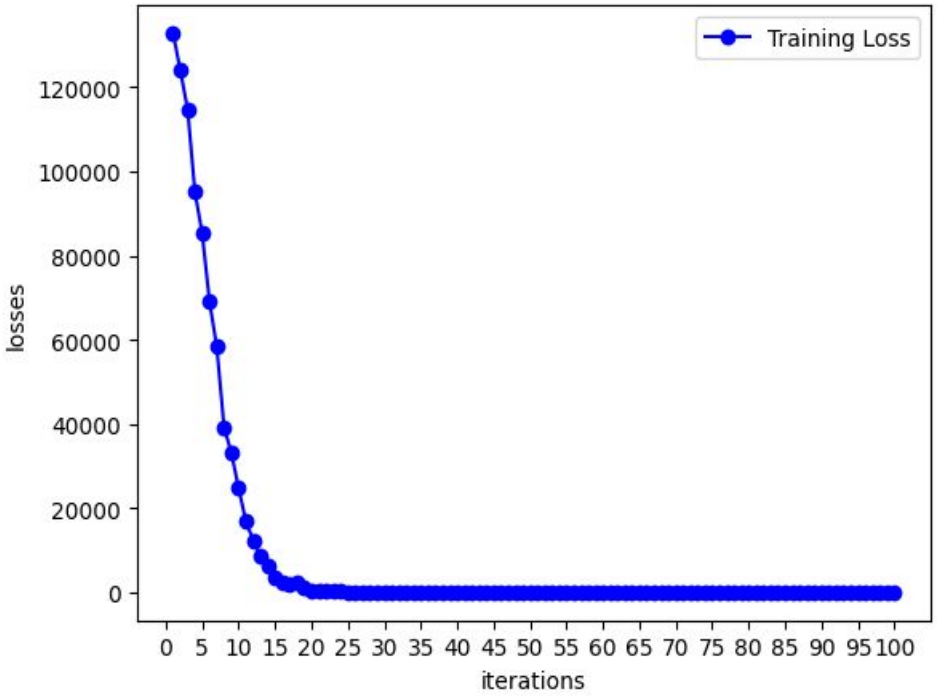
```
### Start training ###
losses {'ner': 0.05734510908575951}
### Start validation ###
f1 score 0.1741424802110818
 99%|                                                                                                  | 99/100 [1:43:30<01:02, 62.46s/it]
epoch:100
### Start training ###
losses {'ner': 0.00019589857507981102}
### Start validation ###
f1 score 0.1741424802110818
100%|                                                                                                 | 100/100 [1:44:33<00:00, 62.73s/it]
### Writting loss data ###
### Writting score data ###
### Start testing ###
```

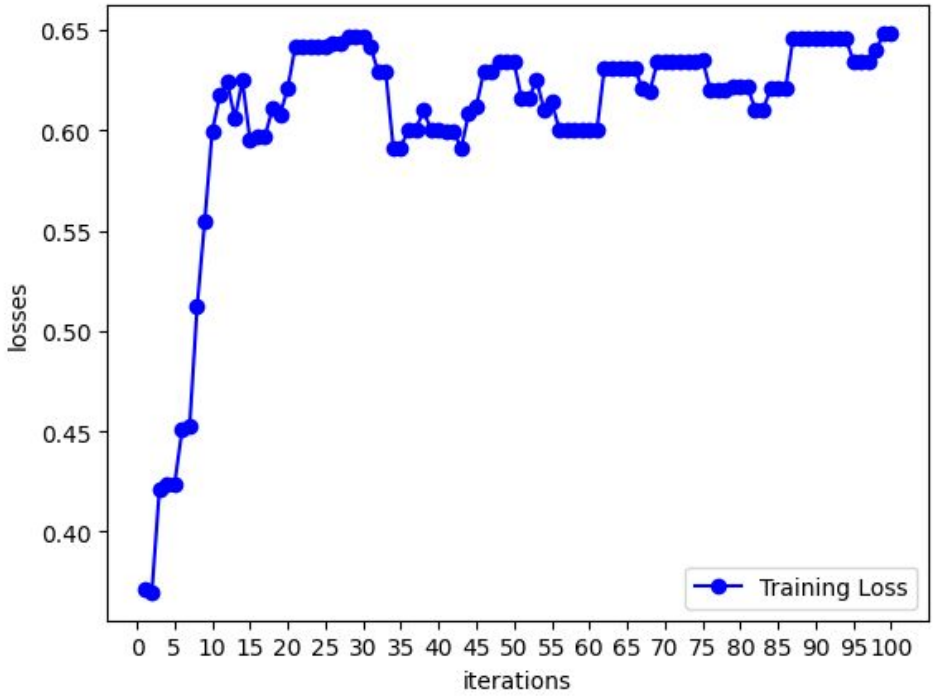
- Train in 100 epoch and batch_size = 100 within the first 10000 lines in training data.
- Result loss: 0.000196
- Validation F1 score: 0.17 (First 10000 lines)
- Note: First 30 epoch the validation F1 score is always 0.00

Task 2: Retrain from Spacy pretrained model

- Settings
  - Epoch: 100
  - Batch size : 50
- Data Preprocessing
  - Input: #\n1\t”word1”\t’O’\t’O’\n\n2\t”word1”\t’O’\t’O’\n…etc
  - Output: [(“Sentence”, “entities”:[(char_start, char_end, pattern)])]
  - Remove suffix: “deriv”, “part”
  - Mapping: {"B-LOC":"LOC", "B-PER":"PER", "B-ORG":"ORG","B-OTH":"MISC"}
- Training[4]
  - Load pretrained model: *nlp = spacy.load(old_model_file)*
  - Using optimizer: optimizer =optimizer = nlp.resume_training()
  - Frozen other pipes: with nlp.disable_pipes(*other_pipes):
  - Update model: nlp.update(examples,dropout, optimizer, losses)

Task 2 Result





Task 2 Result

```
### Start training ###
losses {'ner': 0.0022551773696288645}
### Start validation ###
f1 score 0.6484641638225256
 99%|                                                                                                  | 99/100 [1:46:26<01:04, 64.05s/it]
epoch:100
### Start training ###
losses {'ner': 7.859566651635935e-06}
### Start validation ###
f1 score 0.6484641638225256
100%|                                                                                                 | 100/100 [1:47:30<00:00, 64.50s/it]
### Writting loss data ###
### Writting score data ###
### Start testing ###
```

- Train in 100 epoch and batch_size = 100 within the first 10000 lines in training data.
- Result loss: 7.86 * 10^-6
- Validation F1 score: 0.65 (First 10000 lines)
- Note: First epoch the validation F1 score start around 0.35

Task 3: Evaluation Task 2 model

- Get result from loaded model: *nlp = spacy.load(evaluate_model_file)*
- Write function to transform result to the evaluation TSV file
  - Remapping: {
  - "LOC":"B-LOC",
  - "PER":"B-PER",
  - "ORG":"B-ORG",
  - "MISC":"B-OTH"}
- Load the test file[3] and predict the result to *eval.tsv*
- Run the perl script[2]: *perl nereval.perl < eval.tsv > evaluation_result.txt*

Task 3 Result

For the first 10000 lines in test Data

Category\Metrics	Accuracy	Precision	Recall	FB1
Strict	94.43%	58.64%	32.24%	41.60
Loose	94.75%	66.29%	36.44%	47.03
Per-Level Evaluation	88.86%	58.64%	32.24%	41.60
Per-Level Global Evaluation	100.00%	0.00%	0.00%	0.00

Reference

- [1] Ch6 Class materials from Natural Language Processing and the Web WS22/23
- [2] option1-evaluation-script from Natural Language Processing and the Web WS22/23
- [3] option1-test from Natural Language Processing and the Web WS22/23
- [4] Spacy models Documentation https://spacy.io/usage/models

Thank you for your attention