CS 445 Natural Language Processing
Project 4: Named Entity Recognition

Cavit Çakır

23657

Table of Contents:

Features:

Example morphologic analysis of word “Istanbul” is:

İstanbul+Noun+Prop+A3sg+Pnon+Nom

➔ Root

I used the root of the token which is provided in morphological analysis data.
Root features detect the similarity between “Evleri” and “Evde”. Their both roots are “ev”. So our CRF can learn according to this similarity.

In our example above the root tag is: 'İstanbul'

➔ Part of Speech

I took the final(after the last derivational boundary) form of the token from morphological analysis.
For every token, a Part of Speech tag is one of the followings:

[Adj, Adv, Conj, Det, Dup, Interj, Noun, Num, Postp, Pron, Punc, Verb]

In our example above the Part of Speech is: Noun
Result:

➔ Proper Noun:

I checked the morphological analysis of the token and tagged True if “+Prop” is in the analysis and vice versa.
Prop tag is generated by analyzer. Analyzer looks to big name database and returns true if the particular token is in the database.

In our example above the Proper Noun tag is: True

➔ Noun Case:

I checked the last part of the morphological analysis and if it is a Noun Case tag I tagged as it is own value but if there is not Noun Case tag, then I tagged as 0.
Noun Case of the token is one of the followings if it is nominal;
Nominative(NOM), Accusative/Objective(ACC), Dative (DAT), Ablative(ABL), Locative(LOC), Genitive(GEN), Instrumental(INS), Equa- tive(EQU).

In our example above the Noun Case tag is: Nom

➔ Orthographic Case:

If the first letter of the token is uppercase I tagged it as UC and if it is lowercase I tagged it as LC.
This information is important for NER because most of the tagged entities are uppercased.

In our example above the Orthographic Case tag is: UC

➔ All Inflectional Features:

I took all of the tags in morphologic analysis after the post tag.

In our example above the All Inflectional Features tag is: Prop A3sg Pnon Nom

➔ Start of the Sentence:

I checked if the token is at the beginning of the sentence or not.

In our example above the Start of the Sentence tag is: False

➔ Lower version of the token:

It is basically the lowercase version of the token.

In our example above the lower tag is: istanbul

➔ Last 3 characters of the token:

I took the last 3 characters of the token.

In our example above the last 3 characters tag is: bul

➔ Last 2 characters of the token:

I took the last 3 characters of the token.

In our example above the last 2 characters tag is: ul

➔ Is word digit or not:

I checked if the token is digit or not.
In our example above the is digit tag is: False

➔ If token in Lexicon or not:

I checked my lexicons and if I found the token in one of the lexicons, I tagged it with the first 3 letters of the NER Tag.

In our example above the lexicon tag is: LOC

➔ Next Word

I added the same tags for the next word if the next word exists.

➔ Previous Word

I added the same tags for the previous word if the previous word exists.

Example Extracted Feature for the token “Şenliği“

```
{'+1:word.all_inflectional': 'Prop A3sg P3sg Dat',
 '+1:word.in_lexicon': 0,
 '+1:word.isdigit()': False,
 '+1:word.lower()': 'şenliği',
 '+1:word.noun_case': 'Dat',
 '+1:word.orthographic_case': 'UC',
 '+1:word.postag': 'Noun',
 '+1:word.prop': True,
 '+1:word.root': 'Şenlik',
 '+1:word[-2:]': 'ği',
 '+1:word[-3:]': 'iği',
 'word.BOS': True,
 'word.all_inflectional': 'A3sg Pnon Nom',
 'word.bias': 1.0,
 'word.in_lexicon': 0,
 'word.isdigit()': False,
 'word.isnotpunc()': True,
 'word.lower()': 'müzik',
 'word.noun_case': 'Nom',
 'word.orthographic_case': 'UC',
 'word.postag': 'Noun',
 'word.prop': False,
 'word.root': 'müzik',
 'word[-2:]': 'ik',
 'word[-3:]': 'zik'}
```

Results:

Average results of CRF with only root feature:

Average --> (Fold 1 + Fold 2 + Fold 3 + Fold 4 + Fold 5) / 5

	precision	recall	f1-score
B-LOC	0.925	0.817	0.868
I_LOC	0.847	0.577	0.686
B-ORG	0.907	0.709	0.796
I_ORG	0.736	0.596	0.659
B-PER	0.945	0.702	0.805
I_PER	0.884	0.672	0.763
micro avg	0.898	0.71	0.792
macro avg	0.874	0.679	0.763
weighted avg	0.898	0.71	0.792

Average of 5 Fold:

Average --> (Fold 1 + Fold 2 + Fold 3 + Fold 4 + Fold 5) / 5

	precision	recall	f1-score
B-LOC	0.949	0.937	0.943
I_LOC	0.824	0.722	0.768
B-ORG	0.923	0.912	0.917
I_ORG	0.868	0.860	0.863
B-PER	0.933	0.937	0.935
I_PER	0.909	0.921	0.915
micro avg	0.921	0.915	0.918
macro avg	0.901	0.882	0.890
weighted avg	0.921	0.915	0.918

Fold 1:

Fold 1 --> (0, 2000)

	precision	recall	f1-score	support
B-LOC	0.945	0.938	0.942	850
I-LOC	0.849	0.646	0.734	113
B-ORG	0.909	0.902	0.905	650
I-ORG	0.852	0.828	0.840	424
B-PER	0.932	0.942	0.937	1055
I-PER	0.905	0.910	0.908	458
micro avg	0.916	0.906	0.911	3550
macro avg	0.899	0.861	0.877	3550
weighted avg	0.915	0.906	0.910	3550

Fold 2:

```
Fold 2 --> (2000, 4000)
              precision    recall  f1-score   support

       B-LOC      0.943     0.953     0.948       762
       I-LOC      0.793     0.793     0.793        92
       B-ORG      0.928     0.903     0.915       568
       I-ORG      0.879     0.830     0.854       395
       B-PER      0.946     0.943     0.944      1064
       I-PER      0.933     0.946     0.939       498

   micro avg      0.929     0.922     0.925      3379
   macro avg      0.904     0.895     0.899      3379
weighted avg      0.928     0.922     0.925      3379
```

Fold 3:

```
Fold 3 --> (4000, 6000)
              precision    recall  f1-score   support

       B-LOC      0.958     0.932     0.945       811
       I-LOC      0.884     0.731     0.800       104
       B-ORG      0.931     0.936     0.934       594
       I-ORG      0.893     0.909     0.901       396
       B-PER      0.927     0.944     0.936      1109
       I-PER      0.907     0.924     0.916       488

   micro avg      0.927     0.927     0.927      3502
   macro avg      0.917     0.896     0.905      3502
weighted avg      0.927     0.927     0.927      3502
```

Fold 4:

```
Fold 4 --> (6000, 8000)
              precision    recall  f1-score   support

       B-LOC      0.953     0.932     0.943       899
       I-LOC      0.817     0.742     0.777       120
       B-ORG      0.937     0.901     0.919       646
       I-ORG      0.851     0.845     0.848       438
       B-PER      0.931     0.929     0.930      1141
       I-PER      0.893     0.920     0.906       498

   micro avg      0.919     0.908     0.913      3742
   macro avg      0.897     0.878     0.887      3742
weighted avg      0.919     0.908     0.913      3742
```

Fold 5:

Fold 5 --> (8000, 10000)

	precision	recall	f1-score	support
B-LOC	0.948	0.932	0.940	859
I-LOC	0.780	0.702	0.739	121
B-ORG	0.914	0.918	0.916	588
I-ORG	0.865	0.889	0.877	388
B-PER	0.933	0.930	0.932	1175
I-PER	0.910	0.907	0.909	549
micro avg	0.918	0.914	0.916	3680
macro avg	0.892	0.880	0.885	3680
weighted avg	0.918	0.914	0.916	3680

Summarize

I combined the morphological analysis and the NER tagged data. If i cannot find the morphological analysis of any token, I written *UNKNOWN* to its morphological analysis. After the data preparation, I reconstructed the dataset according to 5 Fold which was described in project documentation. So, I put the first sentence to first fold, second sentence to second fold and so on. Afterwards, I divided the data with respect to folds and featurized them. I used the sklearn wrapper of the crf-suite module in order to calculate the result of folds. After calculating the result for each fold, I calculated the precision, recall and f1-score average.

The results showed that my results are very close to the papers of Reyyan and Gulsen Hoca. Baseline model with only root feature scores 0.71 and after adding all the features we got an average of 0.91 which shows the importance of the features. Also we can say that my model is really generic because the variance of the results in each fold is really small and very close to the average.

We could add word embeddings and some keywords like “caddesi”, “hanim” in order to increase accuracy and f1-score.