University of Macau

CISC3025 - Natural Language Processing

Project#3 - Report

Li Jialin
Zhang Huakang D-B9-2760-6

1. Introduction

In this project, we build a maximum entropy model (MEM) for identifying person names ('Named Entity', NER) in newswire texts and it achieves a very high performance by a set of features of the input words. We notice that whether a word is a name not only depends on itself, but also its neighbors. Based on this observation, we choose the part of speech of the word and its neighbors as the main features in our model. We also built a front-end website[1] for this project and make the source code public on GitHub[2].

2. Methods

For our approach, we directly use the part of speech of the input word and its neighbors in a sentence, $w_{n-2}^{n+2} = \{w_{n-2}, w_{n-1}, w_n, w_{n+1}, w_{n+2}\}$. The basic feature list contains:

- w_n
- $lable(w_{n-1})$
- $isUpper(w_n[0])$

The features we add:

- $isAlpha(w_n)$
- $isPeriod(w_n)$
- w_{n-2}
- $pos(w_{n-2})$
- w_{n-1}
- $pos(w_{n-1})$
- w_{n+1}
- $pos(w_{n+1})$
- w_{n+2}
- $pos(w_{n+2})$

where $pos(\cdot)$ is the function that get the part of speech of the word.
But only the part of speech is not enough for NER since the name word can be replaced with any nouns. For example, '*President Biden today agrees to send weapons to Ukraine*' and '*US Congress today agrees to send weapons to Ukraine*' have same sentence structure. If we only use the part of speech of the target word and its neighbors, this model will become noun recognition instead of the person's name recognition.

[1] https://nlpproject.boxz.dev

[2] https://github.com/BoxMars/NLP_Project/tree/master/Project3

Thus, we consider using `nltk.corpus.name` to check if the word is a name word to enhance our model. The name feature list contains:

- $isInNameCorpus(w_n)$
- $isInNameCorpus(w_{n-2})$
- $isInNameCorpus(w_{n-1})$
- $isInNameCorpus(w_{n+1})$
- $isInNameCorpus(w_{n+1})$

3. Implementation

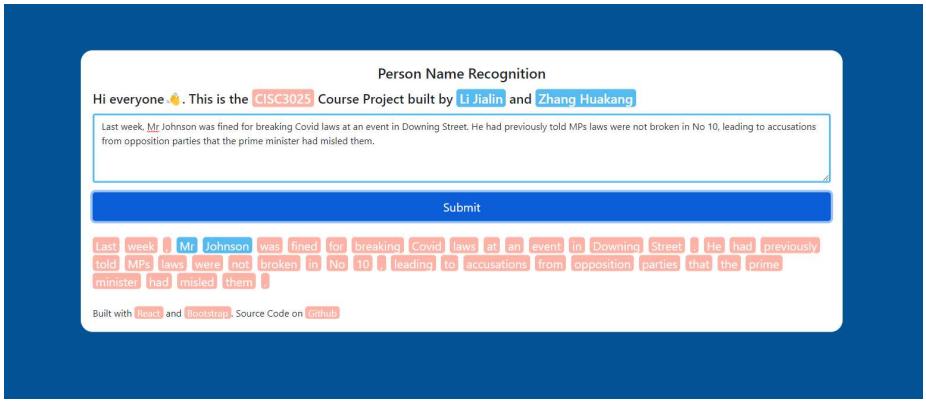
3.1. NER Model

For the MEM features, we use NLTK toolkit to analyze part of speech. The following code is our implementation:

```
features = {}
#===== Baseline Features =======#
current_word = words[position]
features['has_(%s)' % current_word] = 1
features['prev_label'] = 0 if previous_label=='O' else 1
if current_word[0].isupper():
    features['Titlecase'] = 1

#===== TODO: Add your features here =======#

features['is_all_letters']=current_word.isalpha()
features['previous_.'] = words[position-1]=='.' or position==0
try:
    if words[position-1].isalpha():
        features['previous_tag']=nltk.pos_tag([words[position-1]])[0][1]
        features['previous'] = words[position - 1]
        features['p_name'] = words[position - 1] in self.name_lsit
except Exception:
    pass
try:
    if words[position+1].isalpha():
        features['next_tag']=nltk.pos_tag([words[position+1]])[0][1]
        features['next'] = words[position + 1]
        features['n_name'] = words[position + 1] in self.name_lsit
except Exception:
    pass
if current_word.isalpha():
    features['tag']=nltk.pos_tag([current_word])[0][1]
    features['name'] = current_word in self.name_lsit
try:
    if words[position-2].isalpha():
        features['previous_2_tag']=nltk.pos_tag([words[position-2]])[0][1]
        features['previous_2'] = words[position - 2]
        features['p_2_name'] = words[position - 2] in self.name_lsit
except Exception:
    pass
try:
    if words[position+2].isalpha():
        features['next_2_tag']=nltk.pos_tag([words[position+2]])[0][1]
        features['next_2'] = words[position + 2]
        features['n_2_name'] = words[position + 2] in self.name_lsit
except Exception:
    pass

#============== TODO: Done ===============#
```

3.2 Web Server

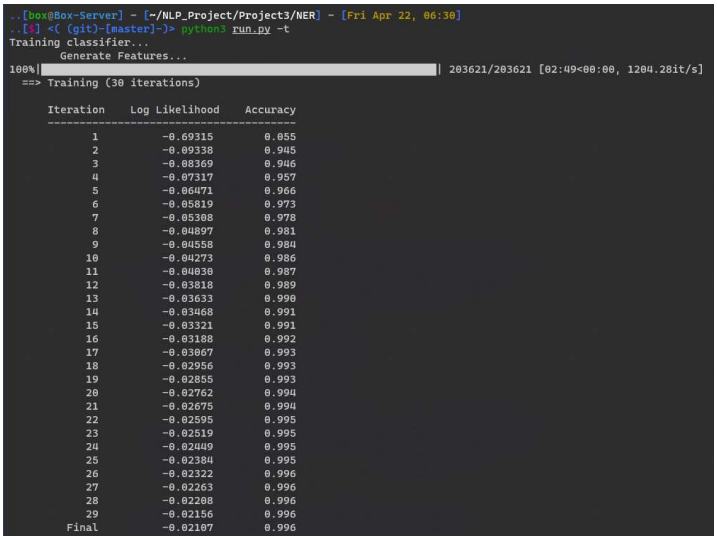
We use `flask` package to develop the API server and built a front-end website with `React` and `Bootstrap`. You can access `https://nlpproject.boxz.dev` to experience our project or access `https://nlpproject.boxz.dev/api/?text=<sentence>` to experience the back-end API.



4. Evaluations

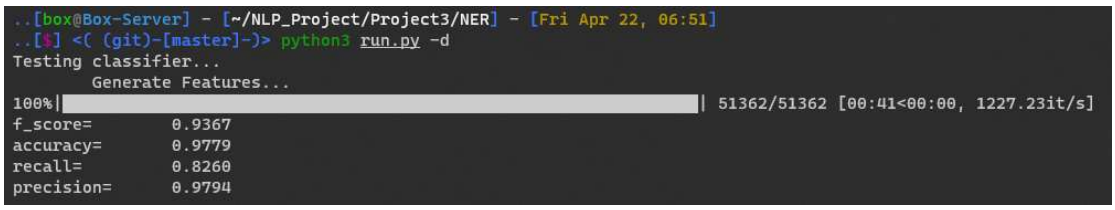
4.1. Training

The following picture shows the training process of the model with 30 iterations.



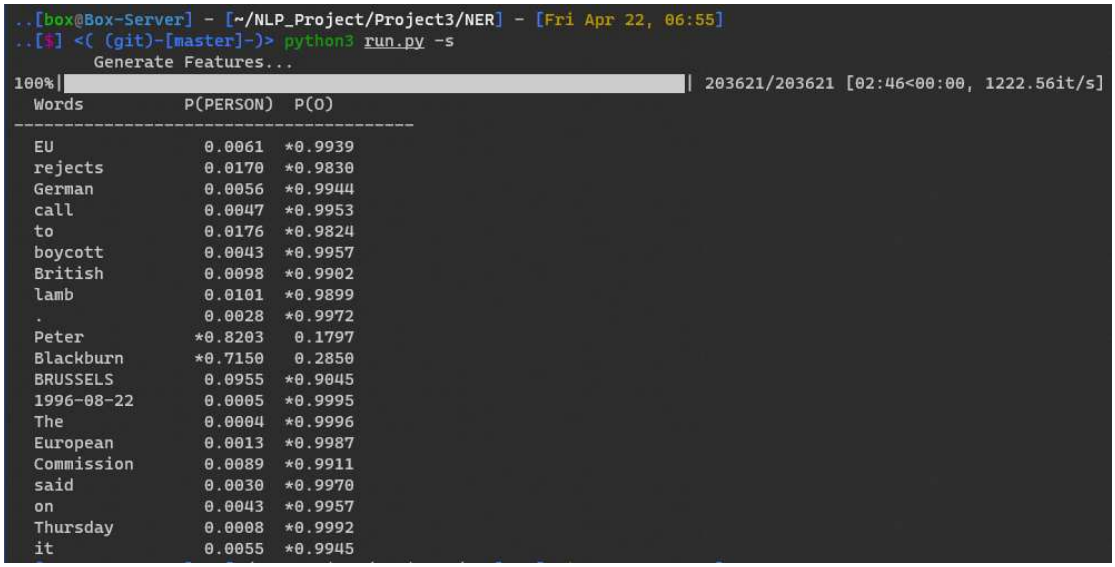
4.2. Testing

The following picture shows the testing result.



4.3. Output

The following picture shows the possibility of the first 20 words in the training set.



4.4. Discussions

We notice that when we try adding a lot of features in this model, the generating process will take a long time. It is easy to know that the generating f each is independent, actuarily it is possible that use multithreading or metaprograms to accelerate this process. But during to the Global Interpreted Lock (GIL) in python, multithreading may not work limited by the clock speed of CPU.

5. Conclusion

In this project, we build a maximum entropy model (MEM) for identifying person names ('Named Entity', NER) in newswire texts and it achieves a very high performance by a set of features of the input words. We also set up a front-end website for this model for visualization.