Person Name Recognition implemented with Maxent Model by Python

Weijian Zhang
University of Macau / Macau
db62685@um.edu.mo

Abstract

Person Name Recognition has attracted much attention in Name Entity Recognition field in recent year. Maxent Model is widely used. This project will simply perform a Maxent Models Estimation and it is implemented by python. What's more, a simple website is built for presentation. This project provides a simple view for the students who are interested in Natual Language Processing.

1 Credits

This project is based on the course, CISC3025 Natural Language Processing, taught by Dr. Derek F. Wong, and provided by University of Macau. The code is based on the work of Konfido.

2 Introduction

Name Entity Recognition (NER) is a hot research area in Natual Language Processing. One of the main problem of this area is the Person Name Recognition. The most widely used method of the NER system are the Generative Model and Discriminative Model. In this project, one of the discriminative model will be implemented. That is, Maxent Model. The Maxent Model is a logistic linear classification model. It uses convex optimization to optimize the cost function. The best advantage of Maxent Model is that the conditions can be set in a very flexible way. The tradeoff is that it needs more time to train. Due to the classification target, the person name, multiple features are designed, including the titlecase, the title, stopwords, and some verbs that are used by human. Besides, the NLTK corpus is used to optimize the program. The left parts of this report is organized as following: section 3 General Instruction, which would give instructions about how to set the environment and explain the commands to run the program; section 4 Other Issues, which states the deliverables other than this report; section 5 Implementation, which would give details about the description of the methods, implementation and additional consideration to optimize the model; section 6 Evaluation, which would show the results of the program in different settings and explain the findings; section 7 Conclusion and Future Work, which would summarise the whole project and give the guidance to future work; and last, section Acknowledge and Reference.

3 General Instructions

Basically, the main structure of the directory of the project is shown in Figure 1. The files are grouped into 2 parts: the website and the NER model. The file "model.pkl", folder "algorithm" and folder "data" belongs to NER model. The left are for the website.

There are some basic operations about the NER model. To begin with, to run the program, some packages are needed, including Django, nltk and sklearn. To train the model, make sure the command line tool is inside the directory "algorithm", and then run command "python run.py -t". Similarly, to test the model, run "python run.py -d". And to see the prediction results, run "python run.py -s". The trained model will be saved into the parent directory of the program "run.py".

The website runs in local host. To start the website, use command line tool and go to the directory same as the program "manage.py", then run command "python manage.py runserver". By default, the port 8000 will be occupied. Go to the link: http://localhost:8000/NER/. To stop the website, return to the command line tool and press CONTROL-C.

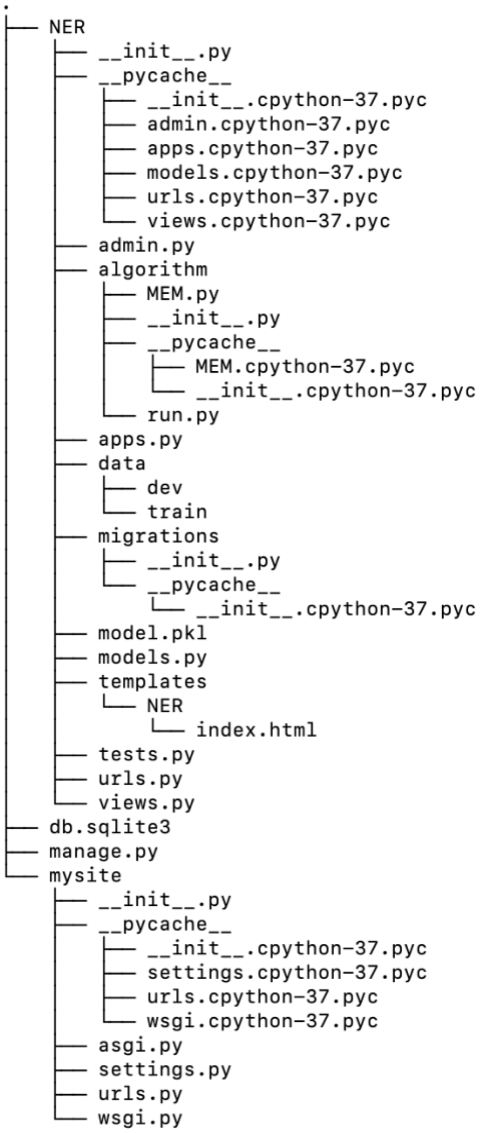

Figure 1: Project Structure

3.1 Electronically-available resources

The necessary files are available in the link, https://github.com/zhangweijian97/NERcisc3025. The Github repository includes the program files and electronic version of this report, and a presentation video.

4 Other Issues

Besides this report, the project includes a program that implement the NER model, and a simple website running in local computer to present the basic functions of the NER model. A presentation video is submitted together.

5 Implementation

5.1 Maxent Model

A feature is a function with a bounded real value:

$$f: C \times D \rightarrow \mathbb{R}$$

A positive weight votes that this configuration is likely correct. A negative weight votes that this configuration is likely incorrect.

Empirical count of a feature:

$$emperical\ E(f_i) = \sum_{(c,d) \in observed\,(C,D)} f_i(c,d)$$

Model expectation of a feature:

$$E(f_i) = \sum_{(c,d) \in (C,D)} P(c,d) f_i(c,d)$$

In practice, nltk.classify.maxent.MaxentClassifier is used.

5.2 Local Website

The website is implemented by Django framework.

5.3 Feature Design

For every word in a sentence, basically, the word ifseft, the previous one word and the after one word are considered.

Then, the feature “previous_label” is used. If the word is the first one for the whole string, the previous_label will be set to ‘O’.

Then, I consider all name’s first letter should be a capital letter. The corresponding feature is called “Titlecase”.

It’s frequently seen that a name is surrounded by stopwords like “.” or “,”.

Human usually have titles, such as Mr., Miss, Dr., President, manager. The titles are also considered.

Some verbs are probably used by a person, like say, disscuss, speak, talk, etc. They are considered.

6 Evaluation

TABLE I
Train Case and Results

Case	Features and Params	Results
1	features['has_(%s)' % current_word] features['prev_label'] features['Titlecase'] MAX_ITER = 5	f_score= 0.8715 accuracy= 0.9641 recall= 0.7143 precision= 0.9642
2	(same as case 1) features ['in_nltk_names'] MAX_ITER = 5	f_score=0.9355 accuracy=0.9778 recall=0.8261 precision=0.9775
3	(same as case 1) features['before_is_stopword'] MAX_ITER = 5	f_score=0.8720 accuracy=0.9641 recall=0.7140 precision=0.9655
4	(union of case 2 and 3) MAX_ITER = 5	f_score=0.9308 accuracy=0.9764 recall=0.8141 precision=0.9775
5	(same as case 2) features['before_is_title'] MAX_ITER = 5	f_score= 0.9350 accuracy= 0.9777 recall= 0.8250 precision= 0.9774
6	(same as case 2) features['somebody_say'] MAX_ITER = 5	f_score= 0.9353 accuracy= 0.9777 recall= 0.8256 precision= 0.9775
7	(same as case 2) features ['titlecase_and_say'] MAX_ITER = 5	f_score= 0.9368 accuracy= 0.9781 recall= 0.8284 precision= 0.9782
8	(same as case 2) features ['titlecae_after_stopword'] MAX_ITER = 5	f_score= 0.9338 accuracy= 0.9773 recall= 0.8220 precision= 0.9773
9	(save as case 8) features ['titlecae_with_title] MAX_ITER = 5	f_score= 0.9370 accuracy= 0.9782 recall= 0.8285 precision= 0.9784
10	(union of case 8 and 9) features['in_gazetteers'] MAX_ITER = 5	f_score= 0.8767 accuracy= 0.9650 recall= 0.7222 precision= 0.9643
11	(same as case 10) MAX_ITER = 20	f_score= 0.9621 accuracy= 0.9864 recall= 0.8982 precision= 0.9820

In mose of the cases, the precision is high but the recall is relative low. That means lots of words which are the label “PERSON” won’t predict the label to be person.
It could be seen that the word itself forms a baseline. The nltk corpus data improves the performance a lot. The titlecase contributes for some percents. Stopwords, titles and gazetteers do not make significant improvement.

To be mention, theoretically, the features for the “PERSON” are helpful to recall but may lose presicion. The features shared by PERSON and other words like geometric name would harm the recall but increase the precision score.

7 Conclusion and Future Work

This project is a good practice on the study for Name Entity Recognition. It includes the study of Maxent Model, implementation and evaluation of the model, a simple web application for presentation, and a trial on writing academic report. The feature design for this project is reasonable but far from practical. The future work is to learn how to design better features for PERSON, and other tags. A improvement of the web application is expected.

Acknowledgments

Thanks professor Derek F. Wong and 2 TA, Du Haihua and Li Hongyan.

References

Lecture Notes for CISC3025