# CS59000

## Homework 1

**Penghao Wang**

February 26, 2016

## 1 Introduction

The aim of this project is to develop a structured learning system. The data is form the University of Carnegie Mellon. They provided a fast and robust Java-based tokenizer and part-of-speech tagger for tweets, its training data of manually labeled POS annotated tweets, a web-based annotation tool, and hierarchical word clusters from unlabeled tweets. The data is divided into three part, train, dev and test part. The challenge of this project is using the traditional NLP processing tool to work with the Twitter data.

In this project, two learning method is used. First is the MEMM, which is local learning. The second is SVM, which is global learning.

## 2 Feature Sets

I believe the feature setting is the most important part in this project. Bad feature set will lead to a terrible result. For instance, when I started the project, I wanted to do it as the same way that last semester do. That is, set all the words in the training set as the features. However, I found it is impossible to do it. Also, it is not the right way to do it. There are several reasons. First, it will be come very very slow, and it is impossible to do it in time. Second, the feature is not relevant. In this project, several features are set. I will introduce some of them.

1. The word is in the startList or not. To achieve that, a startWordList is created for it. In the function, when a word come in, it will be compared with the startWordList. If it is in the list, the value should be 1. Otherwise, it will be 0.

2. Whether the word is in the first of the last. For each generation, the first word is marked as one. The other value of the other word will be 0. If the word position is equal to the length of the sentence, it means the word is the last word. The feature position that judges the last word will be one.

3. Previous_Tag. It is very important because that is directly related to the tag we want to learn. If the word is the start of the sentence, the previous tag is "start"In this part, I use a set of 25 zeros as the original features. If it is belong to one tag, that position of the feature set will be one.

4. The word tag is similar to the Previous tag. There are['!','#','$','&',',',',A','@',',E',',D',',G',',M',',L',',O',',N',' all of the set. The way I do has some drawback. It is very hard to add tag.

5. Is the first letter is capitalized. isupper() function are used.

6. Is the word in the startWordList is capitalized or not.

## 3 Program

The are several function in the programming. First set of the functions is to use. First set of them is the read files. The second set of function is to generate the features. The third set of functions is the Learn function. The fourth function is the evaluation function.The main functions are as followings.

1. The package of numpy, io and other packages are imported.

2. readfile(filename) is to read the information in the document. There are three documents are read.

3. featureGenerate(file,n) is to generate the feature sets. In the begining, it calculate what the list is the current word in. If the word has never appear in the Taglist, it will return set of zeros.

4. startList(filename) is to build the start words list.

5. buildTaglist() is to build 25 different lists in the training set.

6. MEMM(wordList) In this function, the Stochastic gradient descent is used to calculate. Then use the Viterbi to calculated the path of the prediction. The return is the path.

7. SVM(wordList) is use SVM to learn the features. It return the resultList.

8. calTrainAcc(wordList,result) is to calculate the accuracy. It compares all the tags and the predicted tags. The function return a number of the right tags.

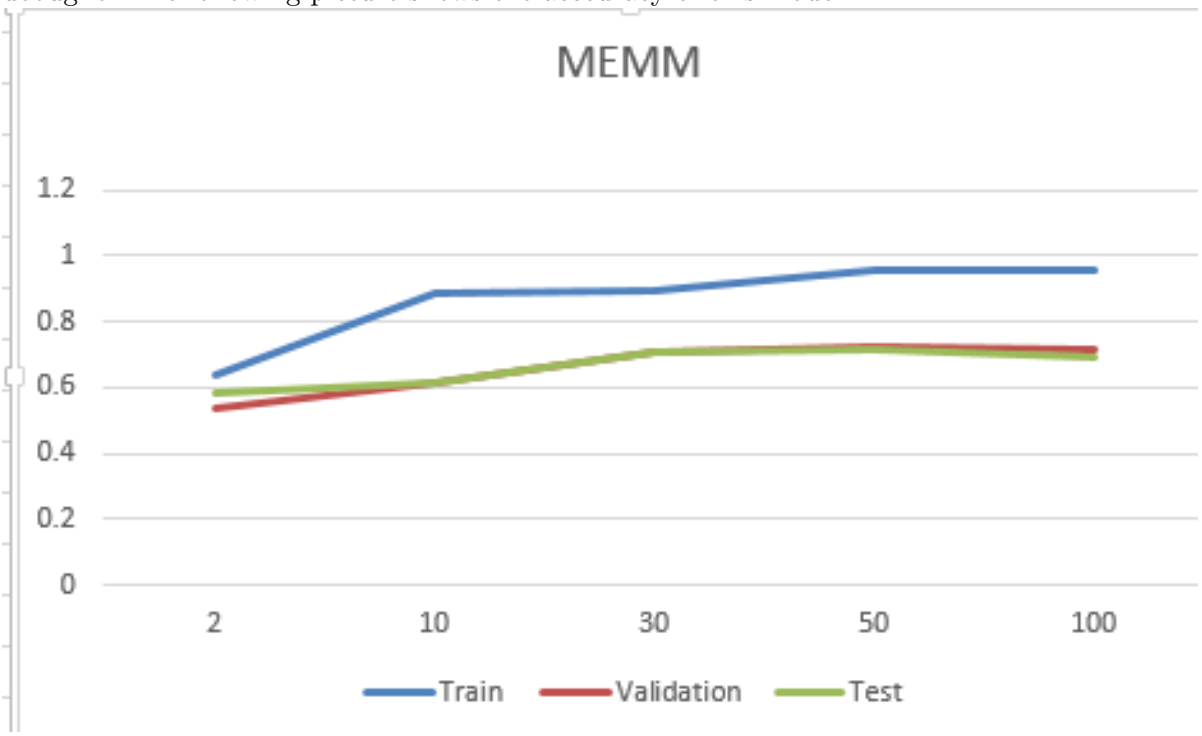
The following picture shows the prediction and the original word list part.

```
['I', 'predict', 'I', "won't", 'win', 'a', 'single', 'game', 'I', 'bet', 'on', '.', 'Got', 'Cliff', 'Lee', 'today', ',', 'so', 'if', 'he', 'loses', 'its', 'on', 'me', 'RT', '@e_one', ':', 'Texas
['O', 'V', 'O', 'V', 'V', 'D', 'A', 'N', 'O', 'V', 'P', ',', 'V', '^', '^', 'N', ',', 'R', 'P', 'O', 'V', 'D', 'N', ',', '~', '@', '~', '^', ',', 'N', ',', 'U']
Processing
['RT', '@DjBlack_Pearl', ':', 'wat', 'muhfuckaz', 'wearin', '4', 'the', 'lingerie', 'party', '?????']
['~', '@', '~', 'O', 'V', 'V', 'P', 'D', 'N', 'N', ',']
Processing
['Wednesday', '27th', 'october', '2010', '.', '\xe3\x80\x8bhave', 'a', 'nice', 'day', ':)']
['O', 'V', 'D', 'N', ',', 'V', 'D', 'A', 'N', ',']
Processing
['RT', '@ddlovato', ':', '@joejonas', 'oh', ',', 'hey', 'THANKS', 'jerk', '!']
['~', '@', '~', '@', '!', ',', 'N', 'N', 'N', ',']
Processing
['@thecamion', 'I', 'like', 'monkeys', ',', 'but', 'I', 'still', 'hate', 'COSTCO', 'parking', 'lots', '..']
['@', 'O', 'V', 'N', ',', 'V', 'O', 'R', 'V', '^', 'N', 'N', ',']
Processing
['@DDaimaru', 'I', 'may', 'have', 'to', 'get', 'minecraft', 'after', 'watching', 'videos', 'of', 'it']
['@', 'O', 'V', 'V', 'P', 'V', 'V', 'P', 'V', 'N', ',', 'U']
Processing
['RT', '@eye_ee_duh_Esq', ':', 'LMBO', '!', 'This', 'man', 'filed', 'an', 'EMERGENCY', 'Motion', 'for', 'Continuance', 'on', 'account', 'of', 'the', 'Rangers', 'game', 'tonight', '!', '\xc2\xab'
['O', 'V', ',', '!', ',', 'D', 'N', 'V', 'D', 'N', 'N', 'P', 'N', 'P', 'N', 'P', 'D', '^', 'N', 'N', ',', '~', '!', ',']
Processing
['RT', '@musicdenver', ':', 'Lady', 'Gaga', '-', 'Bad', 'Romance', 'http://dld.bz/n6Xv']
['~', '@', '~', '^', '^', ',', 'A', 'N', 'U']
```

# 4 Evaluation

## 4.1 MEMM

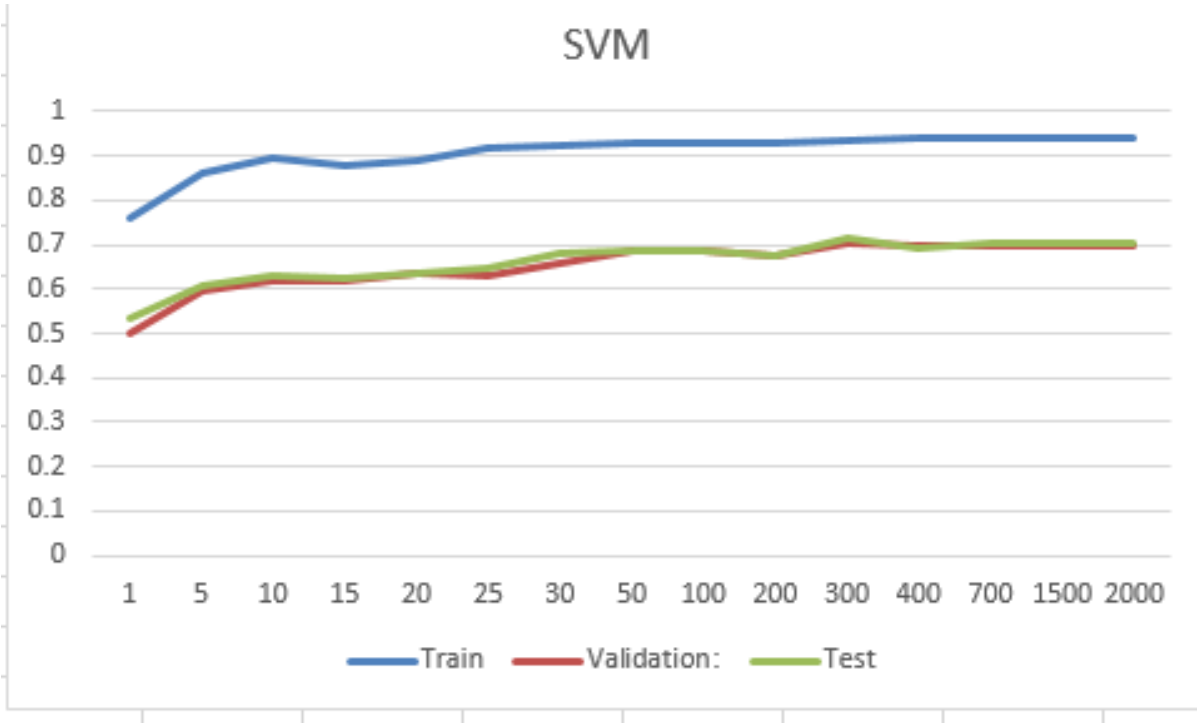
MEMM is a local learning model. Memm is combination of hidden Markov models and maximum entropy models. I use Stochastic gradient descent in this project. Then use the Viterbi to solve it. This Viterbi is different from the Viterbi to solve the HMM. It took me a lot of time to debug it. The following picture shows the acccuracy of this model.




Because of my computer, it took very long time to train the model. I don't make to much iterations. But I believe it will converge soon after the 100 iterations. As the iteration increase, the accuracy of the training increase. However, the accuracy of validation and the test set decrease after 30 iterations. It could be overfitting. The best iteration should be 50.

## 4.2 SVM

In this project, I use SVM as the global training method. The following picture shows the accuracy of the model.




The blue line the the training accuracy, the red and green line is stand for the accuracy of the validation and the test. The accuracy of the training increase as the iteration goes up. These are the accuracy of the tag but not the entire tweets. However, the accuracy of the test is going down. It means overfiting. The best iteration should be 300 iteration. The overall accuracy is OK for the training part when compared with the original source.

## 5 Future work

There are several things should be improved. The first one is the feature set. There should be more feature that I create. However, because of the time limit, I don't have the time to do it. The second is the space of the program. It is kind of slow to train the model. More than that, I should do analysis about what tags that are classified wrong the most and improve the program.