ECE 590.06 - TXT DATA ACQUISITION ANALYSIS - MIDTERM REPORT

Ting Chen - tc233

1. Introduction and Methods

1.1 What is part-of-speech(POS) tagger

According to what we've learnt in class, I only got a rough idea about speech tagging. Since I never touched this topic before, I would hereby summarize the main idea of it.

As we learnt in an interesting experiment in class - a sentence, or some corpora consists of different kinds of words, such as nouns, verbs, adjectives, pronouns, etc. Despite that we could tag those words with human labor, it is still important to let computer learn to categorize them, so that computer can better understand people's demands or queries.

How would computer know the part-of-speech of a word? For human beings, we highly depend on sentence structure and context. So for computer we also need to find some 'states' for it which it could depend on to infer further information. We've learnt to use Markov Model as well as Hidden Markov Model to set up some 'states' that conform to the training set. In this project, I will try to discuss the topic with different methods.

1.2 The question I wish to explore and answer

I always like to understand how signals or strings differ from each other. In the past I've done two major investigations with the knowledge of Information Theory. With numerical results of entropy, mutual information and RMSE, I evaluated the creativeness and difference between corpora as well as music notes. Now, with the tools and algorithms in NLP, I can further investigate this question: How well can a computer learn different types of corpora?

How do the types (news, fiction, etc.) of the training and testing corpora impact success?

Last time when I investigated the differences of entropy rate between genres of text, I got several interesting results. For more fictional text or those I thought to be difficult (Harry Potter, GRE readings), they do have slightly larger entropy rate. Therefore, I'm interested to find out if there exist some corpus that is also "difficult" for computer to tag. I guess scientific text could be easier than Shakespeare, or standard daily English could be more recognizable than foreign English such as Chinese English or Irish dialect, etc. The problem is that in order to explore those topics, I need some well tagged dataset. Luckily, with NTLK, we have some really handy datasets.

1.3 Dataset introduction and basic methods

I'll primarily use the corpora in the Brown Corpus in NLTK, since it is already manually tagged and recommended 'officially', which is a great data source to test my algorithms.

I'll use the in-built default tagger in NTLK, as well as an N-gram (Bigram) tagger and a Viterbi-based tagger to compare their results. I could then compare different algorithms on same set to evaluate them. Also I could test one algorithm on different corpus to understand how the genre of corpus would affect its accuracy.

On first thought, I'll just split each one piece of corpus into an 80/2 portion. 80% serves as the training set, and the 2% as the test set (memory and computing power are limited for my laptop 😕). Then I'll do repeated trials to get an average accuracy of test sets, which are randomly chosen within the same corpus I used to train my matrices.

Second, I will test the Viterbi POS tagger on different categories of corpus in the Brown. For example, I will train the matrices with news text and test it on sci-fi text. By training and testing on different genres of corpus, we could roughly see the answer to our proposed question: how do the types of corpus affect the accuracy of tagging.

Third, I want to expand to a wider range of topics, such as comparing different tagging algorithms, the effective ways of improving our Viterbi algorithm and how would some factors affect our correctness. Depending on time, I might select a few of them to discuss, and then summarize and complete the discussion after I manage to hand in my homework.

For better readability and less memory consumption, I will use the universal tags rather than the default Penn Tree. Due to shrunken size of tags, we could anticipate a higher success rate of prediction, since there are less transition paths from tag to tag.

1.4 Other related information

1.4.1 My smoothing approach

Since we do not mainly focus on smoothing technique, I just follow what I've done in homework 7 - the 'add 1' smoothing method.

- For the starting probability matrix, if a tag has not appeared at the first place of a sentence, just add 1 to its frequency number.
- For the transition probability matrix, if tag A to tag B transition never occurred, just add 1 to its frequency number.
- For the emission probability matrix, if a word is not recognized with any tag, it falls to the label '#NA' with a frequency number less than one. (Discuss later)

1.4.2 How to run the code

For each part of this homework, I created a separate file for it. In order to get the results, you can run them one by one in alphabetical order. No more dependencies required other than our course requirements.

2. Results and Discussion

2.1 General Viterbi POS-tagging result on the Brown Corpus

Following are the results for 10 repetitive times of POS tagging with Viterbi algorithm on the Brown Corpus. The training set is the first 80% of the whole 57340 sentences in it, and the testing set are randomly selected within the corpus, each with a length of 2% the whole corpus.

0.940	0.962	0.863	0.913	0.963	0.855	0.960	0.954	0.961	0.948

Form 1. Viterbi POS tagging on the whole Brown Corpus

The 10-time average of this test is 93.2%, which is better than my expectation. Another repetitive trial of the same program gave me a 95.0% accuracy rate. Given that the training set is extremely big and each of the testing sets is rather small, a higher accuracy rate does make sense. However, I thought it could also introduce overfitting with such a large training set.

Another topic that I want to discuss is that how well we could tag a corpus other than the Brown. Following is the result of Penn Treebank testing set. This time I used 100% of the Brown Corpus as training set, and also 10 x 2% Penn Treebank as test sets.

0.676	0.727	0.673	0.710	0.784	0.740	0.692	0.732	0.775	0.760

Form2. VITERBI POS TAGGING with training set Brown Corpus and testing set Penn Treebank corpus

The 10-time average of this test is 72.7%, which is drastically lower than before.

0.720	0.646	0.458	0.726	0.845	0.744	0.732	0.832	0.744	0.778

Form3. VITERBI POS TAGGING with training set Brown Corpus and testing set MASC corpus

The 10-time average of this test is 72.2%, similar to the above Penn Treebank corpus. Overall our tagger works fine both on same corpus and on different test sets. Still, there are more that we can achieve to improve this score.

2.2 Training and testing with different genres of sources

According to the documentation of NLTK, there are 15 categories in the Brown Corpus. Therefore, I want to investigate the effectiveness of training and testing on different genres. By training on 100% full set and testing on 2% of each genres, I plotted the following correlation graph.

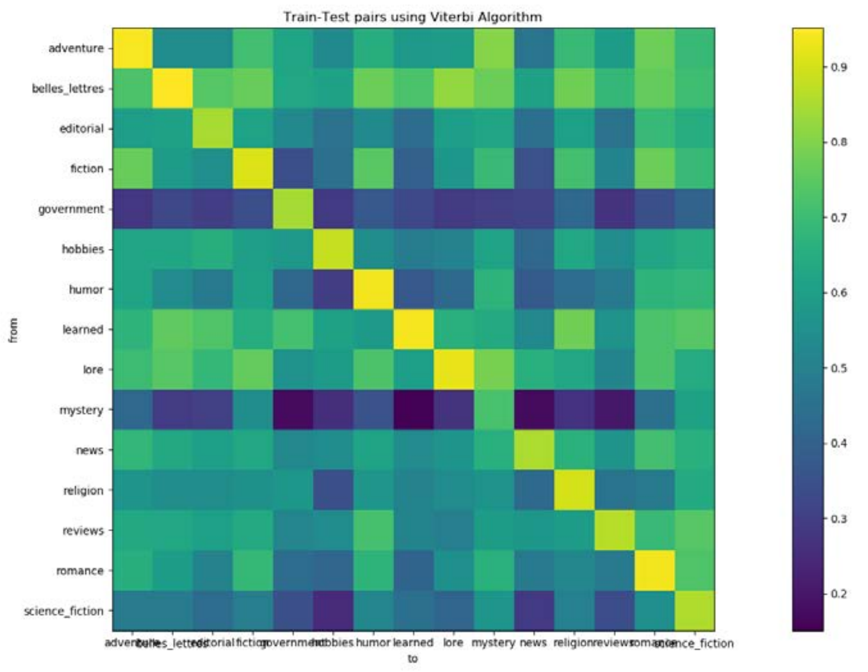

Figure 1. Training-Testing set pairs with Viterbi algorithm

As expected, it's obvious to find that for each genre, the best prediction accuracy falls on itself, since the testing set has exactly the same sentences that are included in the training set. A little bit surprising though, I found that the predicting accuracy is not 100% which I thought to be. After some analysis, I understood that the overall probability depends on not only a single sentence, but the whole set. Thus it is unlikely to have a 100% prediction rate, unless the testing set is full of very short sentences or the training set is rather small, where we can map each sentence exactly to its original state.

This graph yields some really interesting results. Apart from the distinct yellow diagonal, we can also find two horizontal deep blue rows. One is government and the other is mystery.

The degree of likelihood of government training set to other testing sets are somehow uniform, which could be interpreted as that government corpus or documents always do not resemble our daily language. That could be intuitive, e.g. every time when I try to read the documents or archives of the United Nations, I become sleepy and cannot keep focus on those ‘dry’ and ‘rigid’ words…

The darkest blue coordinate lies on the cross of news and mystery. Considering the fact that news text are committed to report the truth, while mystery literature pieces tend to convey the illusionary stories, this could be an interesting metric that tells us how different they are even in verbal expressions. Though we can still see that those stories which involve more fictional part do share some similarity with mysteries - look at adventure, fiction and sci-fi columns in the figure. Not a sharp contrast, but still recognizable, especially comparing with government or news. We can consequently say that at least people could distinguish the differences between reality and imagination, and those fictional text are more unpredictable given our common sense daily language 😊. That is also a thought provoking result considering that fictional works have larger entropy rate. Maybe we can come to a conclusion that fictional literature represented the creativeness and imagination of human kind.

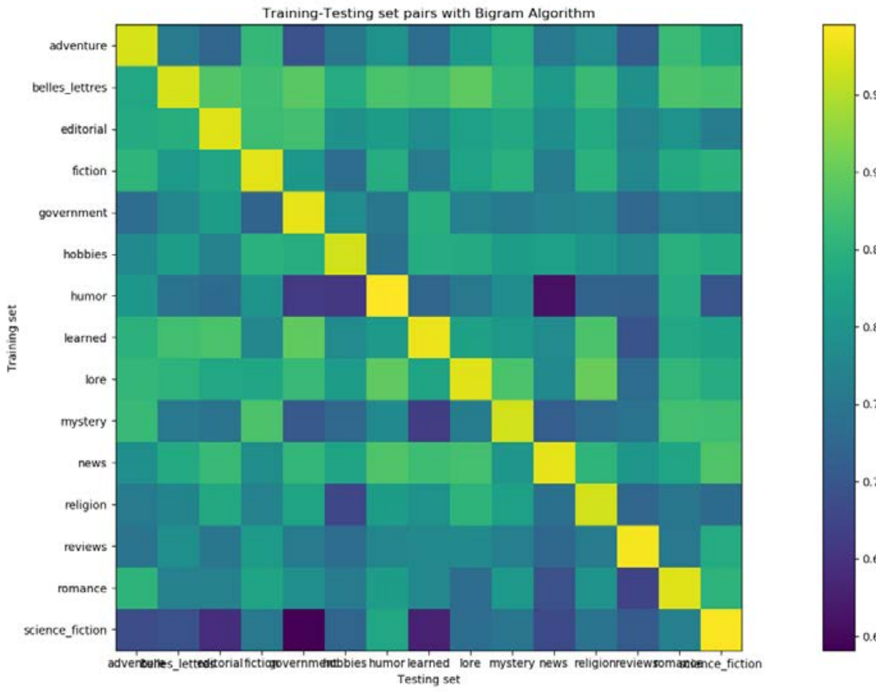

FIGURE 2. TRAINING-TESTING SET PAIRS WITH BIGRAM ALGORITHM

Also, I’ve tested the Bigram model which based on the assumption of Markov chain. That is - each word depends on its previous word. With the built-in system Bigram tagger and unigram fallback, I got the figure above with 100% training set and 2% of each genre. As we can see in this figure, apart from the distinct diagonal which is intuitive, the other tagging accuracy are also pretty good. It turns out that the simpler approach might even work better than the delicate Viterbi algorithm, and it is even faster in training than Viterbi’s! With this algorithm, the worst cases come with sci-fi, humor and mystery. We could derive some more imaginative conjectures, but I’ll just stop here.

2.3 Comparing multiple tagging method on different genres

Furthermore, I’m interested in finding the differences between algorithms, and how do they differ from each other on tagging various genres/types of corpus.

The training set was 80% of each genre and testing sets are 3 randomly chosen 2% sets in same corpus. The result was plotted below.

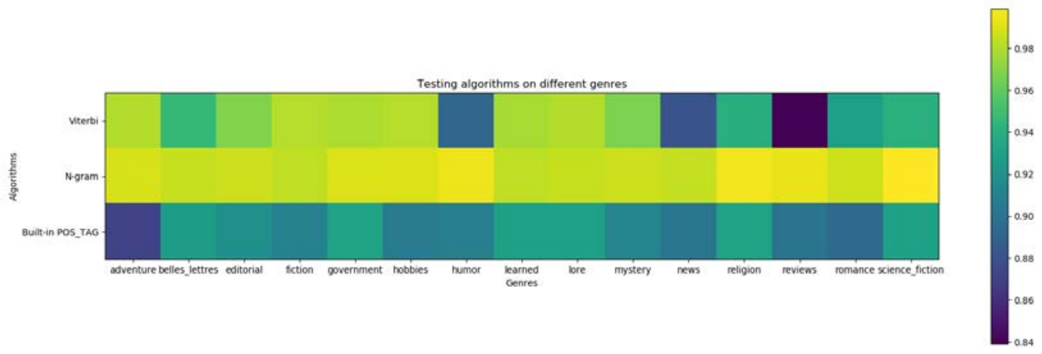

FIGURE 3. DIFFERENT ALGORITHMS ON DIFFERENT GENRES OF CORPUS, TRAINING SET 80%

As we can see in the figure, all of the 3 algorithms have done well in tagging the corpus. The N-gram method win without doubt on most of cases, given that the training set is large enough for it to over fit. To my surprise, ‘naïve’ Viterbi algorithm overall performs better than the built-in POS tagger in NTLK, but the margin is small. With some further tests, I found that the built-in tagger has a more stable performance over various sets.

2.4 How would * affect the result of tagging

2.4.1 Training set size

It depends on other conditions, but generally a larger training set could give a better HMM.

- If the training set and testing set are of same type of corpus, then a larger training set size could yield better result in accuracy. However, that could cause overfitting problem.
- If the training and testing sets are different, enlarging the size of training set can improve result when it is small, but there seems to have some asymptotic boundary when the training set size becomes sufficiently large.

Thus, in order to balance the computational consumption and accuracy, we need both a suitable type of training corpus, as well as a reasonable size of it.

2.4.2 Testing set size

Ideally, if an algorithm is stable, it should not fluctuate in performance with any input.

In my experiments,

- ➢ For the same type of corpus, the testing set size hardly affect average accuracy. With Penn Treebank Corpus, given a 50% training set size, the accuracy rate is around 92~93% for most of the testing set size.
- ➢ For Penn Treebank training and Brown testing, I got 79.2% and 79.6% respectively on 50% and 5% testing set size.

Therefore, testing set size does not substantially affect the result.

2.4.3 Emission matrix unknown word fallback probability

When I was dealing with the smoothing process of matrices, I found that if I assign a 1 to the non-exist element, it could somehow affect the final result, since sometimes there is only one word or tag for a certain transition in a large corpus.

80.9% - 0.01	79.2% - 0.1	76.3% - 0.5	73.2% - 1

FORM4. SMOOTHING FACTOR WILL AFFECT ACCURACY OF TAGGING

For optimal case, this number should be as small as possible, since it represents “non-exist” transition. However, too small a number could cause numerical precision issue when being computed. In case that we do not set a bar for tagging accuracy, I just set it a convenient 0.01. With same training code and different settings of this smoothing factor, we got the corresponding list above.

2.4.4 Other fallback methods

- ➢ DefaultTagger - this could be a last but faithful backing for all sort of taggers, since there will always be some unrecognized words. For convenience, we can just let unknown words to have the tag of ‘NN’ or noun, since nouns are greatest in amount of English language.
- ➢ UnigramTagger - If there are no transition available, assign the most popular tag of context to a word. For example, when tagging a fishing article, ‘bass’ could be given a ‘NN’ tag, while an ‘ADJ’ tag for music related text.

2.5 How to improve the tagging accuracy

In order to improve our tagger, we need to first analyze what could be its drawbacks and then try to solve them with other approaches.

2.5.1 Some low accuracy examples and possible explanations

In order to investigate possible enhancement of the Viterbi POS tagger, I trained it with 80% of Penn Treebank Corpus as well as Brown Corpus. Then I randomly select some sentences which have accuracy less than 80% in its tagging. Then we analyze why our tagger performed bad in these cases.

Example 1 - Penn Treebank Corpus

Sentence set: ('This', 'democracy', 'is', 'suddenly', 'a', 'little', 'more', 'democratic', '.')

```
Viterbi path: ['DET', 'NOUN', 'VERB', 'ADV', 'DET', 'ADJ', 'ADJ', 'ADJ', '.']
Original tag: ['DET', 'NOUN', 'VERB', 'ADV', 'DET', 'ADV', 'ADV', 'ADJ', '.']
```

As we can see in this example, our Viterbi algorithm made 2 wrong judgements before the final adjective. That makes sense, since the position before an adjective often reserves for adverb or adverbial phrase. In our case, ‘a little more’ is a special adverbial combination of 2 adjectives.

Example 2 - Penn Treebank Corpus

Sentence set: ('But', 'it', "'s", 'building', 'on', 'a', 'long', 'tradition', '.')

```
Viterbi path: ['CONJ', 'PRON', 'PRT', 'NOUN', 'ADP', 'DET', 'ADJ', 'NOUN', '.']
Original tag: ['CONJ', 'PRON', 'VERB', 'VERB', 'ADP', 'DET', 'ADJ', 'NOUN', '.']
```

For this example, we can see that the cause of deviation is the '['s]' in the sentence. Our model failed to recognize an abbreviated form of verb.

Example 3 - Penn Treebank Corpus

Sentence: The following *ICH*-4 were barred *-3 or , where * noted *-1 *T*-2 , suspended *-3 and consented to findings without *-3 admitting or denying wrongdoing : Edward L. Cole , Jackson , Miss. , $ 10,000 *U* fine ; Rita Rae Cross , Denver , $ 2,500 *U* fine and 30-day suspension ; Thomas Richard Meinders , Colorado Springs , Colo. , $ 2,000 *U* fine , five-day suspension…(200 more words!)

```
The tag sets are omitted due to length limit.
```

As we can see above, it's an extremely long sentence with multiple punctuations and special characters. Our model failed to recognize many of them, which results in a huge bunch of 'X's in its tag path. Considering the particularity of this case, it could be tolerated.

Example 4 - Brown Corpus

Sentence set: ('Serious', 'difficulty', 'arose', 'with', 'the', 'advent', 'of', 'Substantive', 'Due', 'Process', '.')

```
Viterbi path: ['ADJ', 'NOUN', 'VERB', 'ADP', 'DET', 'NOUN', 'ADP', 'X', 'X', 'X', '.']
Original tag: ['ADJ', 'NOUN', 'VERB', 'ADP', 'DET', 'NOUN', 'ADP', 'ADJ', 'ADJ', 'NOUN',
'.']
```

For this case we can obviously see that the capitalization caused problem in our model.

In previous classes we talked about all sort of trivial but tricky parts of NLP, including hyphens, upper-case, parenthesis, unique proper nouns, misspellings or new usage, etc. In Brown Corpus we can spot a huge lot of these cases, which by all means dragged down our accuracy rate. Since we have not talked thoroughly about the case sensitive text or those special cases, in this homework I just treat them as unrecognized '#NA's.

There are still many more cases that cause misjudgment. Due to the limit of length, I'll just put them aside right now.

2.5.2 Possible improvements

- Add certain fallbacks for special cases, e.g. if the text is not case sensitive, lowercase it; if word not recognized, treat it as a noun, etc.
- Larger training set.
- Combination of different taggers.
- More advanced techniques…

2.6 conclusion

Since it's really late in night I'll keep it in short.

- Not-so-daily genre of corpus has lower tagging correctness with Viterbi algorithm, such as government document and mysterious stories. AKA ‘unpredictable’.

- Selection of training set will affect the model accuracy when POS tagging other corpora.
- Smoothing techniques would affect accuracy.
- Sanity of corpus matters. Special symbols and letter cases can decrease tagging accuracy.
- There exist multiple ways to improve our tagger.

3. Some useful references

NLTK Doc: https://www.nltk.org/api/nltk.corpus.reader.html

Brown Corpus categories, tagged corpora and more: https://www.nltk.org/book/ch02.html