Insert Your Title Here∗

Insert Subtitle Here

FirstName Surname†  
 Department Name  
 Institution/University Name  
 City State Country  
 email@email.com

FirstName Surname  
 Department Name  
 Institution/University Name  
 City State Country  
 [email@email.com](mailto:email@email.com)

FirstName Surname  
 Department Name  
 Institution/University Name  
 City State Country  
 email@email.com

2 POS Tagging

In this part, we try POS tagging. POS tagging is a challenging task due to the ambiguity of a word in different context. Here two different methods of pos tagging are implemented and test on the reviews randomly chosen. The first one is combining tagger and the other is the regular expression tagger. Specifically, the combining tagger is a combination of bigram tagger, unigram tagger and default tagger. These are implemented by using NLTK.

2.1 Combining Tagger

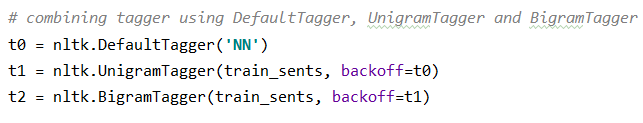
This combining tagger is mainly based on the use of N-gram tagger. N-gram is a common model used in text management. We can apply it to do some prediction. It is derived from probability theory and based on a corpus to provide the data underpinning the calculation. In this way, the size of corpus is a significant factor of the effect. Similarly, we can apply n-gram model to POS tagging. To simplify the process, here unigram and bigram are used.

*2.1.1 Unigram Tagger.* Unigram tagger is a tagger that only used the current word to decide the corresponding tag. It is an obvious and simple way, we just need to find the most frequent use of this word in corpus as the tag. It could get a quite good tagging result in major situation. However, due to the ambiguity of the tagging of some words, there would be definitely unavoidable mistakes using this tagging method.

*2.1.2 Bigram Tagger.* Bigram tagger is a simple N-gram tagger where N equals 2. It uses the tag of the previous one word to predict the tag of the current word. Bayes role is applied, formulated as follow.

By using Bayes role, we can get the tag transition probability. Then we pick making this probability maximum. With N increasing, the accuracy should be improved; however, due to the limit of corpus size, that is, the training data amount, the effect of bigram and other N-grams tagger is not satisfactory. Specifically, there are some situations that we cannot find the corresponding sequence in the training so that we cannot tag the current word. Taking bigram tagger as an example, if the previous one word is unseen in the training, the tagger would not know the tag of it, so it cannot calculate the tag of the current word correctly.

*2.1.3 Combining Tagger for accuracy and coverage balance.* As we mentioned before, bigram and other N-grams tagger would have a higher accuracy during predicting but that depends on the amount of training data a lot. Unigram, on the other hand, may not predict accurately but it can be applied in most cases. Therefore, we can combine them together to make a balance between accuracy and coverage. At the same time, we combine a default tagger as a backoff, which would directly assign the same tag to every token. The strategy is that we first choose to try a specific tagger, then try a more general one as backoff. The code implementation is shown as figure below.

Figure 1: Combining Tagger Code Implementation

2.2 Regular Expression Tagger

Obviously, this tagging method is based on regular expression. All this tagger needs is setting accurate pattern to sift the characteristic of different words in order to apply tagging. It does not rely on any corpus data. From this perspective, the regular expression tagger can be used easily and universally, however, we should also realize that tagging task can hardly be done by using regular expression tagger solely, since words are complicated and even some common words cannot be tagged correctly by using regular expression. The pattern of the regular expression tagger is set as shown below.

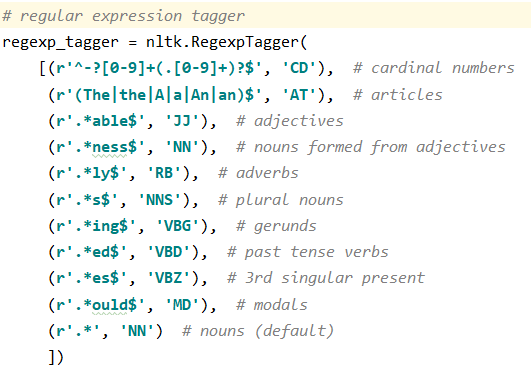


Figure 2: Regular expression Tagger Code Implementation

2.3 Tagging Result

*2.3.1 Results of Combining Tagger.* In this project, we divide Brown corpus into training set and test set. We evaluated that picking one category as training data has a low accuracy, so nearly the whole corpus is used as training data. The accuracy evaluated on the text set is nearly 91%. Then we test the tagger on the reviews randomly selected. One result is shown as the figure below. We can check manually and find that it has quite a good accuracy for tagging the review.

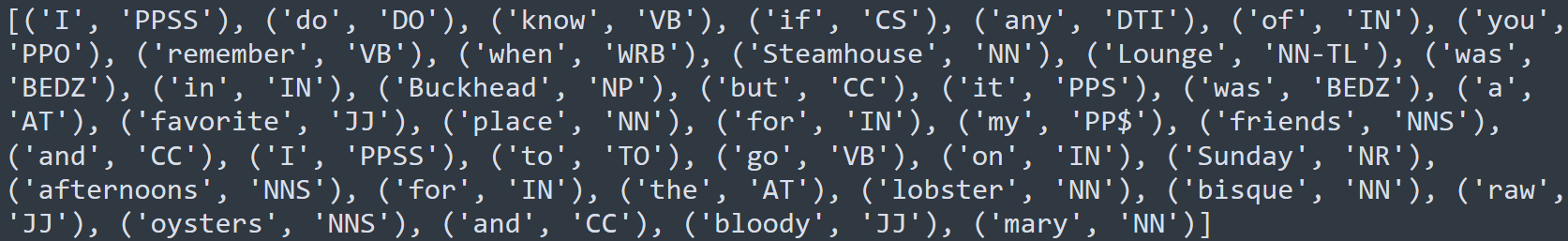


Figure 3: Result of Combining Tagger

*2.3.2 Result of Regular Expression Tagger.* Similarly, we test the same reviews with regular expression tagger. One result is shown as the figure below. We can see that due to the limit of the pattern set here, the result is not good, only few of the words can be tagged correctly. This also proves that tagging with regular expression is hard and the regular expression tagger can only be used as a backup.

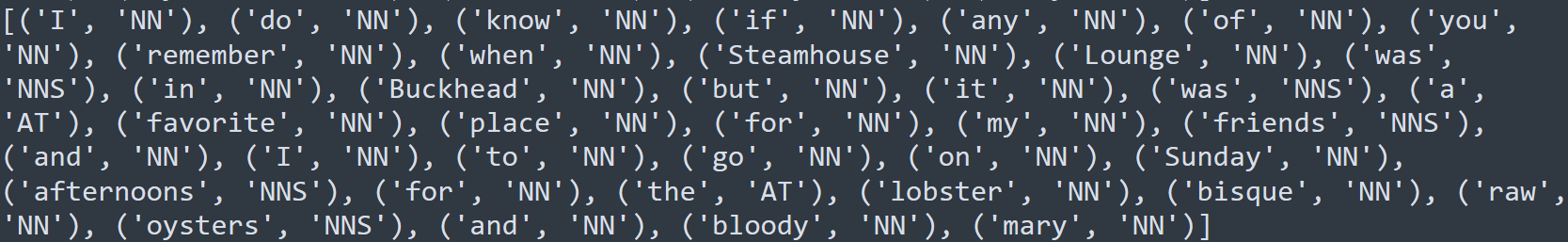


Figure 4: Result of Regular Expression Tagger

Conference Short Name:WOODSTOCK’18

Conference Location:El Paso, Texas USA

ISBN:978-1-4503-0000-0/18/06

Year:2018

Date:June

Copyright Year:2018

Copyright Statement:rightsretained

DOI:10.1145/1234567890

RRH: F. Surname et al.

Price:$15.00