Tagging Parts of Speech

Timothy Liu
University of California at Berkeley
CS294-19
timothytliu@berkeley.edu

Dounan Shi
University of California at Berkeley
CS294-19
dounan@berkeley.edu

ABSTRACT
This paper focuses on the task of tagging text with their parts of speech. The methodology chosen for this task is the Maximum Entropy based Model and although complex will only be explained briefly. More importantly, the focus will center on the differences in performance of the maxent model with varying feature sets compared to the baseline model. One problem highlighted in part-of-speech tagging is that of ambiguity between different tags. Finally, statistical and lexical error analysis will further characterize the task of tagging part-of-speech.

General Terms
Maximum Entropy, Viterbi Algorithm, Part-of-Speech-Tagging

Keywords
Accuracy, Local Trigram Context

1. INTRODUCTION
Part of Speech Tagging is simply defined as marking up words in text with its corresponding part of speech. These include the major categories of: noun, pronoun, adjective, verb, adverb, preposition, conjunction, interjection, and determiner, which are later differentiated to produce 46 different categories. Several purposes exist for part of speech tagging such as pre-processing for parsing and lemmatization. In Natural Language Processing, as opposed to more traditional and manual methods, statistical methods are adopted for efficiency and higher accuracies. The basic statistic is the probability of producing the correct tag given the word or other variables, i.e. $P(t|w)$.

2. MAXIMUM ENTROPY PART OF SPEECH TAGGER
The Maximum Entropy model was chosen for higher accuracies as well as more flexibility. The features in the feature vectors can focus on the recurrent problem of *ambiguity* of words and their parts of speech such as *position* being both a **verb** and a **noun**. Only the context of the word will help differentiate between the tags, which leads to how the word will be tagged.

Each word is used to form a local trigram context, which consists of the current word, position, previous tag, and previous-previous tag. However, the entire sentence can be extracted from the context. A trigram model was used to provide the most relevant data as a balance between fitting the overfitting and generality. After scoring each local trigram context, these scores form a probability distribution of how likely a specific tag fits the word. The probability model is as follows:

$$P(t|w) = \prod_{i=1} P_{MAXENT}(t_i|w, t_{i-1}, t_{i-2})$$

A Maximum Entropy model is used to score transitions and emissions from one tag to the next. This model utilizes feature and weight vectors to extract information about a certain state.

$$score(t_i) = \log P(t_i|t_{i-1}, t_{i-2}, \vec{w}, i)$$

The next step required is how to choose among the scored tags. This requires using the Viterbi method to find the highest scoring sequence of tags rather than greedily taking the highest tag like the Greedy Decoder. The Viterbi Decoder is illustrated here:

$$\delta_i(s) = max_{s_0 \ldots s_{i-1} s} P(s_0 \ldots s_{i-1} s, w_1 \ldots w_{i-1})$$

$$\delta_0(s) = \begin{cases} 1 & if\ s =< \bullet, \bullet > \\ 0 & otherwise \end{cases}$$

The Viterbi Decoder functions on the principles of dynamic programming; therefore, to backtrace and extract the best states, the previous states are kept track of.

$$\psi_i(s) = argmax_{s'} P(s|s')P(w|s')\delta_{i-1}(s')$$

2.1 Baseline Tagger
The provided baseline model used a simple method of tagging words. The baseline model scored local trigram contexts based on the tag counts for every word. The tag counts were simply how frequent the tags appeared with each word. Therefore, the tag that occurred the most often would be applied.

Unknown words are handled with a separate count distribution over tags. Initially, all words are unknown. During training, when an unknown word is seen, the count of its corresponding tag in the unknown word count distribution is incremented and the word now becomes known. When tagging an unknown word during testing, the most frequent label for the unknown word is applied. The accuracy for known words is 92% and 40% for unknown words.

2.2 Maximum Entropy Features
The Maximum Entropy Features used are intentionally split into several categories. The baseline category features are

Table 1: Core Features

Feature Name	String	Count
Prev-Prev + Prev Tag	"PrevPrev+Prev-"	1.0
Previous Tag	"Prev-"	1.0
Word	"Word-"	1.0
Position	"Position-"	1.0

Table 2: All Word Features

Feature Name	String	Count
Prev-Prev Word	"PrevPrevWord-"	1.0
Previous Word	"PrevWord-"	1.0
Next Word	"NextWord-"	1.0
Next Next Word	"NextNextWord-"	1.0

mandatory with every test and consist of four features: previous tag, previous-previous tag, current position, and the word at the current position. This is used to provide a general harness that does not exploit too many patterns within parts of speech. Furthermore, these features do not address the topic of unknown words. The results of using just these features provided 80.88% known word accuracy and 44.89% unknown word accuracy. See **Table 1**.

These core features still do not provide enough context as often times the previous word as well as following word give clues to the part of speech of the current word. Therefore, in all circumstances, a new category of features called all words arose. These features included the previous-previous word, the previous word, the next word, and the next-next word. These can be seen in **Table 2**. This had an increase in both overall and unknown accuracy showing 84.24% overall accuracy and 57.12% unknown accuracy.

Often times hidden traits are seen inside the words. The next category of features deal explicitly with prefix and suffix characteristics of the current word. These features can address unknown words, target specific types of parts-of-speech, and even group together words that are similar. Of these three, targeting specific tags show the greatest promise especially with unknown words. However, some of these features still might be too broad as there are numerous subcategories of parts-of-speech. The features used can be seen in **Table 3**. These features improved the unknown accuracy greatly; however, decreased the overall accuracy. The statistics were 64.41% known accuracy and 64.02% unknown accuracy.

Because of the jump in unknown accuracy, an easier method of creating prefixes and suffixes arose. Prefixes and suffixes can be extrapolated from the word by having features up to four characters on either side. An example of the loop structure can be seen below.

```
for (int i = 0;i <= 4; i++) {
    features.incrementCount("Prefix-" +
            word.substring(0, i+1), 1.0);
    features.incrementCount("Suffix-" +
            word.substring(word.length()-i-1), 1.0);
}
```

Table 3: Prefix and Suffix Features

Prefix or Suffix	String	Target Tag
Prefix	"un"	ADJ
Suffix	"wh"	WH-
Suffix	"ly"	RB
Suffix	"er"	JR
Suffix	"est"	JS
Suffix	"ing"	VBG
Suffix	"ness"	JJ
Suffix	"day"	JJ
Suffix	"ion"	NN
Suffix	"ent"	NN

Table 4: Rare Word Features

Feature	String	Target Tag(s)
Prefix/Suffix	"Prefix/Suffix-"	ALL
Contains Uppercase	"capitalized"	NNP/NNPS
Contains a hyphen	"Hyphen-"	NN
Contains a number	"HasNum"	CD

However, the decrease in accuracy means that there is a fatal flaw in this. When encountering words that appear many times, the combination of previous tags and words are sufficient enough to have a high probability of displaying the correct tag. Adding these extra features will only decrease the accuracy. Therefore, prefixes and suffixes do not help in the case where words appear many times. However, there is still an increase in the unknown accuracy. The sparsity of prior history due to an unknown word would produce low probabilities and result in a bad 'tag' by invalidating the other features.

A solution is found by differentiating the current word from being known and unknown, which is termed rare. If a word is rare, or seen less than 10 times, extra features should be added to help "tag" the word. These extra features are seen in the **Table 4**, which include the suffix and prefix loop previously shown. Because of this, the unknown accuracy will increase leaving the overall accuracy to increase as well. Adding the core features to this provided accuracies of 82.59% and 63.17%.

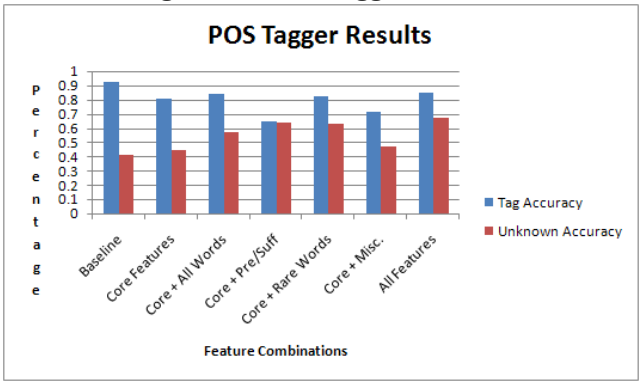
Therefore, all prefixes and suffixes that are shared among words can be grouped together instead of trying to find common prefixes and suffixes together. This helps with unknown words as rare words are often concatenations of smaller words or of different prefixes and suffixes. This also does not affect the high accuracy and tagging of known words.

The last group of features are simply miscellaneous features tried to improve accuracy by a small margin not included in previous features. There is a lack of distinction of whether to add these features to rare words or to all words. These features include length differentiation, whether the word contains a consecutive repeated letter, and the number of times the letter e shows up in the word. The new accuracies from the core features along with the miscellaneous features gave an accuracy of 71.64% and 46.90%. These results show that these features are very poorly representative of the tags.

Table 5: Accuracies of Features

Feature Combin.	Tag Accuracy	Unk Accuracy
Core Features	.809	.449
Core + AllWords	.842	.571
Core + Pre/Suff	.644	.640
Core + Rare Words	.826	.631
Core + Misc.	.716	.469
All Features	.851	.674

Figure 1: POS Tagger Results



3. SUMMARY OF FEATURES AND ERRORS

The final maxent features included the core features, the all words features and the rare words features to produce a fairly decent result of 85.13% and 67.41%. The nature of this improvement lies in dealing with known words and unknown words. Known words were dealt with by including more context clues in the training data. Unknown words were dealt with by finding the unique characteristics and trends between tags.

The trends for accuracies improve with more features especially when targeting unknown words separately. See **Figure 1**.

3.1 Errors

However, there are still large flaws in the Maximum Entropy Tagger. First, there is the trend that when one faulty tag pops up, multiple errors will follow. This is because many later tags depend on the context of the previous tags, which will obviously be wrong if the beginning tags are incorrect. This same trend also shows that if the first tag is correct, then it will be more likely to correctly tag the successive words. Thus, clustering arises as clumps of correct then incorrect phrases appear. Two examples show this from the best tagger.

Words: ...stocks of U.S. wheat to be carried over into the...
Gold Tags: NN VBN RP
Guessed Tags: NNP NN IN

Words: That figure climbs to about 47%...
Gold Tags: NN VBZ RB
Guessed Tags:JJS NNS IN

Furthermore, ambiguity is not truly solved as various adjectives, verbs, and nouns are all confused with each other. Also, it is extremely difficult to determine plurality because many things end in s, which are not nouns but verbs. In the example shown below, "growing" is a gerundive and "demands" is a noun whereas they were both mixed up. There is still room to work on for differentiating between these tags.

Words: ...trying to blunt growing demands...
Gold Tags: JJ VBG NNS
Guessed Tags: NNP NNP NNP

4. CONCLUSION

Part-of-speech tagging is integral to any kind of parsing by differentiating more general tags into specific tags for better parses. The Maximum Entropy Model is just one of many methods in how to approach part-of-speech tagging. The key issues that all models must address are the contexts of the sentence, the ambiguity of the words, and the unknown words. Therefore, the Maximum Entropy Model is an accomplishment for achieving progress in all three.