Constructing a Lossless and Extensible Part-Of-Speech Tagger

Jeffrey Regier
jregier1@swarthmore.edu

Abstract

This paper describes the design of a lossless and extensible part-of-speech tagger, with the intent of illuminating general principles underlying part-of-speech tagging. To this end, the described tagging program has been designed with form emphasized above performance and even above completeness. A key design premise is that a tagger is most naturally constructed through the implementation of largely independent methods of inferencing tags (e.g. analyzing suffixes, analyzing context). However, occasionally communication between these methods is necessary, as is communication of each method's output to a component that ultimately decides each word's part-of-speech. It is argued that a "stochastic-tag", implemented as a class and defined roughly as a set of pairings between each part-of-speech tag and its hypothesized likelihood, is an appropriate and perhaps optimal vehicle for communication. Finally, several techniques of combining the stochastic tags returned by methods are evaluated.

1 Introduction

Deducing the correct part-of-speech (POS) of the vast majority of words in a testing corpus requires little more than recalling the POS most frequently assigned to the same word type in a training corpus of reasonable size. More sophisticated POS taggers attempt to exceed this level of performance by correctly tagging 1) words types not found in the training corpus, and 2) words types with multiple parts-of-speech (POSes).

The arrangement of POSes is typically viewed as governed either by a set of simple rules with many exceptions or by distributions which deviate from randomness. These views correspond to two paradigms between which most tagging programs are divisible. Rule-based taggers generate a set of rules for choosing a tag, often including rules that are exceptions to other rules. Choice of the tag in the testing corpus is simply a matter of applying the rules invented during the training section of the corpus.

Stochastic taggers, on the other hand, have much similarity to Markov models. They too learn from the training corpus, but not merely by creating rules to be used during the testing phase. Their learning takes the form of improving the accuracy of rules' parameters. While there are many taggers that claim to be hybrids, it is perhaps more natural to view the set of rule-based taggers as the subset of stochastic-taggers with parameters that may only hold binary values.

2 Research Goals

This research is motivated by the search for a deeper understanding of the principles that underlie POS tagging. To this end, a POS tagger has been constructed from scratch with the goal of learning as much as possible from the aspects of the document which have been selected for exploration. Restated, a key design goal is to model language such that POS deduction can occur without any loss of information due to processing failure. While the categories "loss of information due to collection failure" and "loss of information due to processing failure" are perhaps not disjoint, an example may clarify this key distinction. Information available from a word's prefix, context, etc. can be lost if no attempt is made to process that aspect of a word. Information can also be lost when an aspect of a word is collected, but either for reasons of efficiency, ease, or oversight some of the available information is not fully incorporated into the ultimate prediction of that word's POS.

In order to minimize the loss of information due to processing, the program's form is emphasized in the design process above all else. Thus, some features which could have improved accuracy were omitted because they were not central to building a framework for lossless POS tagging. For one part of the program, for which programming the optimal method proved beyond the scope of this research, several suboptimal methods were suggested and clearly marked as suboptimal.

It has already been suggested that designing a tagger that minimizes loss of information due to processing failures is a appropriate approach to understanding POS tagging and language. Additionally, if the framework to which components are added is readily extensible, then it is likely the framework has captured some truth about language. Conversely, if the framework designed captures the complexities of language without using techniques that sacrifice a full understanding either due to obscure structure (e.g. neural networks) or full consideration (e.g. rule-based POS taggers), then it will be simple to building new considerations into the existing framework.

[perhaps use this paragraph instead of some of the previous sentences] The introduction of rule-based taggers is one way to counteract the incredible complexity of creating a stochastic-based tagger. The motivation behind this project is that another method of overcoming the complexity of creating a stochastic model of language is by placing an explicit focus on structure and simplicity, and admission of areas of the tagger which will require further work, rather than garbling ideas together to avoid acknowledging the need for future research and to get higher accuracy measures at the expense of a consistent methodology for addressing considerations.

3 Program Structure

3.1 Parsing

This research's emphasis form is natural to accommodated by parsing the training corpus and the testing corpus into separate document objects containing sentence objects which in turn contain word objects.

Each document is ultimately a series of word tokens (including punctuation) but the sentence is nonetheless a relevant unit since it is the largest unit of inferencing. Because a human reader has no trouble figuring out words' POSes without looking beyond the current sentence, it is reasonable that the program not do so either. Of course, this is assuming that the reader is not using other sentences to learn about the language. The equivalent to a reader with knowledge of language is the program running on the testing phase. The exception to this rule is that words used near a sentence are likely to be used with the same POS in later sentences. This exception could be dealt with via horizontally accessed variables. [horizontal?]

Each Word object contains a type, which is the text made lowercase of the word (case is another field). Just as the sentence method handle context sensitive rule deduction and application, the word class handles context free inferencing. Context free inferencing includes operating a dictionary and set of stochastic rules, and will be discussed later.

3.2 Stochastic-Tag

In other taggers, each word is associated with one or at most a few tags. Ultimately, tagging is applying a series of methods to a given word, and combining the output of these methods. If the output of each method is merely a tag or set of tags, then combining is probably best performed by simply picking the POS returned by all methods. This severely limits the abilities of taggers, since most methods of inferencing are not totally confident about and equally confident in the tags in the set that they report. Naturally, there are ways to work around an unwillingness of methods to return both the absolute and relative likelihood of every POS. However, if abstraction barriers are to be respected (and doing so is key to dealing with the large amount of complexity likely inherent to dealing with language), then each method must describe its confidence in its each tag. The stochastic-tag provides is the vehicle for doing so.

The data portion of a stochastic-tag associates each POS with a number representing the likelihood that it is the tag. Accessor methods to return the most likely tag, the percent likelihood of each tag, and the confidence in the most likely tag. Additional modifier methods make automatic some types of adjustments of the likelihood of a POS. Finally, some more complex methods allow the averaging and summing of multiple stochastic-tags. Finally, a product method the confidence in every POS according to a vector taken as a parameter.

[should be proofread]In whole, the stochastic-tag provides a convenient format for relaying the information required of each component. The proportion of the sum of the numbers associated with each of the POSes made up of each number relates the relative likelihoods of each part. Greater deviation from 1 divided by the number of POSes indicates greater amounts of information provided by the component. Similarly, the total sum of the numbers provides confidence in the measure. The design of the components will show how the stochastic-tag's format for relaying information is convenient and ultimately enables inter-component comparison and combination.

4 Context Free Methods

Context free inferencing refers to a set of methods used to deduce POS that do not consider a word's context within a sentence. There are several methods for doing this.

4.1 Past Word Type Usage

A single Dictionary object is created to serve as the store for all the words encountered during the training phase. During the testing phase the dictionary is used to retrieve statistics about the uses of the word. Word that have not been encountered in the training section will not be found in the dictionary. Similarly, words that have been encountered only a few times may take on novel POSes in the testing corpus. Actually, there is a chance of a word type taking on any formerly unseen POS in the testing corpus if the word type has only been encountered a finite number of times in the training corpus. More generally, given a finite number of encounters with a word type in the training corpus the sample distribution of POSes will differ from the true population distribution for a given word type. There is never a case where dictionary lookup is certain to be correct, and thus more information is always potentially helpful to choosing a tag. Since the dictionary tends to be the premere source of information about a word token's POS, in the case of words not found in the dictionary other methods become especially influential. In isolation of other methods, the dictionary correctly tags approximately 85% of words when given a large training corpus. The addition of a few simple rules such as guessing noun when no instances of the word are found increases accuracy 92%. However, this type of modification has been commented out of the program's code because it is inconsistent with the stochastic approach underlying this research.

4.2 Past POS Usage

A crude method of guessing POSes is to always guess the POS that most frequently occurs in the training corpus without even considering the specific word type. However, appropriately filling a stochastic-tag requires including information other than that the most likely POS has 100% chance of correctness while the other POSes have a 0% chance of correctness. Rather, accurate information concerning the likelihood of each POS must be included to avoid unfairly undermining the estimates of other methods. It is thus more appropriate to fill each field with the percent of tags in the training document with that POS. By being truthful about its certainty in each POS, this component can hope to have its information accurately weighted upon combination with other POSes.

Applied in isolation, this method will guess that every word is a noun, with accuracy of approximately 21%. An example will make clear the value of avoiding the rule-based method of guessing noun when a word type is not found in the dictionary, even without consideration of methods other than "Past Word Type Usage" and "Past POS Frequency". Obviously, when "Past Word Type Usage" reports that it has seen the word type 0 times, it is more reasonable to follow the suggestions from "Past POS Usage". Less obviously, when "Past Word Type Usage" reports that it has encountered a certain word token a small but nonzero number of times, then there still is some information about smoothing that could potentially be obtained from "Past POS Usage".

4.3 Suffix Identification

Some words have suffixes which provide useful hints as to their POS. For example, words ending in "ly" are likely to be adverbs while words ending in "ing" are frequently verbs. There are also many endings that provide less certain mandates, but nonetheless slightly increase the likelihood of a word taking on a certain POS or decrease the likelihood of a word having another POS. More generally, shared word endings influence the distribution of probabilities.

Words can share suffixes of different lengths with multiple classes of words. How are multiple shared suffixes to be reconciled? With a stochastic-tag, of course. The stochastic-tag features an accumulate function, used to sum the contents of multiple stochastic-tags. If a word has one suffix that occurs many times in the testing corpus but provides no clear mandate, then it should be like adding unbiased white noise to the stochastic-tag. The number associated with each POS in the stochastic-tag returned by this component is the number of words with a matching suffix in the train corpus. If a suffix of length 3 is matched then a suffix of length 2 will also be matched, bringing up questions of double counting. This is a recurring theme, which tragically is beyond the scope of this project. Nonetheless, even in isolation this method performs much better than randomness, choosing the correct tag for approximately 55% of words when recall is set to 35%. Ultimately, there will be no need to manually set the level of recall, since doing so is inconsistent with this research's objectives.

4.4 Affix Transformation

Some words, when stripped of a prefix or a suffix, may match a word type in the dictionary. However, just because a word shares a root word with a word in the dictionary does not mean that it has the same POS. In fact, languages frequently rely on affixes to change a word's POS. While recall will be lower for this method than for "suffix identification", it will be more accurate for the words it finds, as is illustrated by an example. While "ly" may be highly correlated with adverb, if the root word is not itself a verb then "ly" is likely be a red herring (or at least to be over confident).

In practice this method is useful only when recall is set at less than 12%.

5 Context Sensitive Methods

There is also information available about a word's POS based on its location and surroundings within a sentence.

5.1 Preceding Word's Type

Certain words tend to immediately precede words of certain POSes. For example, "the" tends to precede a noun. By identifying these words in the testing corpus a stochastic-tag can be returned indicating all information relevant to each word.

5.2 Surrounding Words' POSes

Words with certain POSes tend to be located in the same position relative to other words with certain POSes. For example, articles tends to precede nouns, either directly or perhaps with an adjective in between the article and the noun. By identifying these words in the testing corpus a stochastic-tag can be returned of the suspected POS distribution of the current word. However, there is one major difficulty with this method that was not relevant to the "Preceding Word's Type" component. The type of the preceding token is certain, so applying the data collected from the training corpus is not difficult. During the training phase it is not permissible to use the truth about the previous words POSes to deduce the current word's POS. An estimate must be substituted for the truth, and for simplicity this estimate comes from the previous word's dictionary-based tag. Again, unlike the truth, the dictionary-based tag is a stochastic-tag, and so it does not clearly specify one POS. Using the "product" function of stochastic tag it is possible to use this tag as a vector for scaling distribution for each POS, and subsequently accumulating the resulting stochastic-tags.

6 Techniques for Combining Methods' Opinions

For each word, each method must not only suggest the most likely POS, but must suggest the likelihood of each element of the set of POSes. This approach largely isolates the methods, such that additional code is needed to combine the methods' outputs. Since the return type of each of the methods is a stochastic-tag, the most natural method of combining the information they contain is to create one "meta-stochastic-tag", and only then pick the POS deemed most likely by the "meta-stochastic-tag". Incidentally, leaving a "meta-stochastic-tag" associated with each word instead of just its most likely POS allows for more accurate context sensitive inferencing of neighboring words.

6.1 First Technique: Use Case Statements

A crude technique of combining the stochastic-tags returned by each component can be implemented using a series of case statements. Let the internal stability of a stochastic-tag be the maximal deviation of the estimated probability of any POS from a value denoting no information (i.e. $\frac{1}{|POSes|}$). Then, if the internal stability of the stochastic-tag returned by the "Past Word Type Usage" method is greater than 0 (i.e. the word type was encountered in the training corpus), let the meta-stochastic-tag be the "Past Word Type Usage" tag. Otherwise, use the suffix identification method's return value as the meta-stochastic-tag if the internal stability claimed by the suffix is over 50%. Precede in this fashion through the remainder of the methods, assigning the tag returned by "Past POS Usage" as a last resort.

Clearly this method is quite imperfect. The most glaring problem is that so little use is made of the information provided by the stochastic-tags returned by each methods. While with this technique each method's return value could potentially influence the final selection of a word's tag, the way this is achieved is far from optimal.

6.2 Second Technique: Pick the Most Confident Opinion

A superior technique lets the meta-stochastic-tag equal the method's output that makes the strongest claim to correctness. Without combining tags, even in theory no better choice can be made than choosing the method's output with the greatest internal stability (previously defined).

While conceptually this technique is an improvement over the first technique, it is still lacking. Recall that the motivation for introducing stochastic-tags was not to directly pick the method claiming to be the most certain. Rather, it was to allow the unbiased combination of information returned by various methods. These first two techniques are similarly suboptimal in that they both discard all of the returned tags except for one. Assuming that the components are accurately reporting the likelihood that they are correct, this does not preclude their being valuable information in the tags returned by the other components. Even given credible evidence, one is still aided by the collection of less credible pieces of evidence. Perhaps multiple pieces of less credible evidence all point mainly towards the same POS, thus overwhelming the most credible evidence.

6.3 Third Technique: Sum POS Histories

For this technique combination is performed by summing the number associated with each POS. Recall that this number as a proportion of the sum of the numbers associate with all of the POSes is the likelihood that this POS is correct, at least from the perspective of a particular method. However, this number is not a likelihood in isolation of the other POSes. This number is generated by seemingly similar processes across methods; when an instance is encountered a stochastic-tag's noteOccurrence method is called, which increases the count of the appropriate tag by 1. However, the tags returned by "Past POS Usage", for instance, will be called many times yet contains little information not accounted for by the dictionary-lookup method. However, with so many calls made to noteOccurrence, the stochastic-tag returned by "Past POS Usage" method will overwhelm other methods, effectively always choosing one of the least accurate tags. It was eventually realized that this number was not necessarily comparable among tags returned by different methods. This technique is discussed because its failings highlight a key difficulty in combining stochastic output from methods that generated their output in isolation.

6.4 Fourth Technique: Sum or Multiply POS Likelihoods

The simplest attempt at remedying the very fundamental problem raised by the third technique is to sum or multiply the likelihood of each POS instead of combining the number of calls to noteOccurrence. While dealing with percentages clearly improves on the unworkable model suggested by the third technique, in a way it is a step backwards. Certainly by using the percent rather than the number of calls to noteOccurrence the use of some of the most important features of the stochastic-tag are preserved: Summing two stochastic-tags that represent the same distribution will result in the same distribution of percentages. Multiplying a stochastic-tag that is 100% confident in one POS by another tag typically also gives returns a tag that is 100% in the appropriate POS. Both of these results are desirable. However, switching the tags that were summed in the previous example with the tags the were multiplied in the previous example provides counterexamples to the correctness of both methods.

Moreover, by dealing with likelihood instead of the absolute number, the ability to cope with small sample size is limited. Given the large number of English word types and the even larger number of bigrams, small sample size will be an issue even with the largest of corpora. Moreover, even if sample size was finite but not small, combining the stochastic-tags without loss of information would still require knowledge of the sample size.

6.5 Fifth Technique: Use Information External to the Methods

Before learning from a sentence of the training corpus, the tagger attempted to tag the sentence based on the data collected from the sentences that preceded it. It then preceded to judge its accuracy at tagging that sentence. Using a procedure which simulated the operations of a Kalman filter, a prediction of the current accuracy of each method was maintained throughout training. (A Kalman filter is a procedure which adjusts for the Gaussian distribution of noise.) By the time the tagger had finished learning it knew the optimal estimate of how each method would perform throughout the testing phase of the document, during which time it would be unable to improve itself since it did not have access to the truth.

Using a Kalman filter-like procedure generates the optimal estimate of how the tagger will perform on the training corpus. Many other methods would have resulted in worse performance. Method's accuracy from earlier is not a good indication of its current level of performance, since the method become more accurate as it trained. Also, a method's performance on the previous sentence would have been a poor way to gauge accuracy, since the previous sentence may have been unusually difficult or unusually simple to tag. Waiting until the tagger has processed the whole training corpus to gauge the abilities of the tagger also would not have been a good alternative; measures obtained from testing on the document on which training has occurred are largely meaningless. Finally, reserving part of the training corpus for testing prior to the actual testing is wasteful and suboptimal.

Based on the judgment of each method's accuracy, those methods consistently performing closer to their predictions were given appropriately more weight, using the stochastic-tag's accumulateByMerit function. Note that accuracy for a stochastic-tag is rarely a simple correct or incorrect, but rather the degree to which it is correct. A stochastic-tag's accuracy is the percent with which it votes for the correct tag. While it was heartening that accuracy could be so aptly defined, second guessing the internal measures of correctness entailed overwhelming complexity; further attempts along this trajectory were abandoned.

6.6 Suggestions for Future Research

Further development of the stochastic-tag is necessary if the output of methods is to be combined losslessly. Specifically, each method must accurately report more than the likelihood of each POS. Rather, each method must also adjust its output to account for the size of its sample. Modifying the "Past Word Type Usage" method such that it will adjust for sample size could be done perhaps by using the t-distribution to generate confidence intervals surrounding each estimate of POS likelihood. Generating confidence intervals around the unbiased estimates of several other methods would also be possible. Modifying operations on stochastic-tags to accommodate confidence intervals is well beyond the scope of this paper.

The covariance between methods also needs to be controlled. For example, when a word token has occurred often in the training corpus, "Past POS Usage" offers nothing that is not accounted for "Past Word Type Usage". Similarly, "Suffix Identification" is largely redundant when "Affix Transformation" is highly relevant.

Controlling for covariance is a standard method used when performing regression analyses, so perhaps those methods could be incorporated.

References

Brill, Eric. 1994. "A Report of Recent Transformation-Based Error Driven Learning." *http://www.cs.jhu.edu/brill/dissertation.html*

Carlberger, J. & Kann, V. 1999. "Implementing an Efficient Part-Of-Speech Tagger." *Software Practice and Experience, 29(9), 815-832.*

Charniak, Eugene. Statistical Language Learning. The MIT Press, Cambridge, MA (1996)