Part-Of-Speech Ambiguity and Unknown Word Guessing with Different Machine Learning Techniques in Comparison

Daniel Nagel
University of Tübingen
daniel.nagel@student.uni-tuebingen.de

Abstract

Different approaches of assigning a specific part-of-speech (POS) tag to each corresponding word of a natural language sentence have been tried and developed over the years. It is a fact, that the accuracy of part-of-speech tagging for highly ambiguous words and so called 'unknown' words, which are words, probably not learned directly from a corpus, is substantially lower than that for known words. Such ambiguous and unknown words account for a non-negligible portion of the tagging errors we get when trying to tag a text or sentence. In this paper we present a comparison of some of the different approaches of part-of-speech tagging for English and evaluate their performance on two small sets of sentences. We compared several statistical taggers and machine learning machanisms like different n-gram taggers, the 'Brill Tagger', a classifier-based tagger, a perceptron tagger and the predictions from a Support Vector Machine as a tagger, to tag our test sets. To improve our scores, we also tried out to influence our scores by different sizes of training corpora as well as different tagsets to see how the different taggers improve with a change of one of those factors when they encounter the ambiguous or unknown words. We found out that even some small changes in the setup can improve the tagging evaluation scores. In the end, we found out that the best performance was done by the Perceptron tagger which scored an accuracy of 89% and a F_1-score of 90% for the 'ambiguous words' and 92% and 93% for the 'unknown words, respectively'.

1 Introduction

Manning (1999) mentions in his famous book that the ultimate goal of research on natural language processing is to parse and understand language. For this reason, some research in nlp tasks has focused on the structure that is inherent in language without requiring complete understanding. One of those task is part-of-speech tagging. Parts-of-speech are useful because they carry a large amount of information about the word they are attatched to and the surrounding neighbors. By tagging, we try to find out which syntactic category is the most likely one for a particular word in a sentence or n-gram. The parts of speech themselves can be divided into two broad supercategories: closed-class types and open-class types. The closed classes are the ones that are relatively fixed and do not underly a change in membership. One example here are prepositions, because new prepositions are rarely created. Such closed-class words are usually function words which are mostly quite short, occur relatively frequently, and often have structuring uses in grammar. To the other category, the open-class words, belong for example nouns and verbs. They are called open class because nouns and verbs can be loanwords and words that are created during the normal process of language development (Jurafsky and Martin, 2017). Part-of-speech tagging is further the process of converting a already given sentence, which is a string or a list of words, into a list of tuples. Each tuple is represented by the form '(word, tag)'. The 'tag' itself is simply a part-of-speech tag and states, as mentioned above, of which specific category the 'word' is. But the task of assigning the most likely syntactic category with one of the possible tagging methods also carries some problems because the corresponding POS tag is not always easy to find for a computer. For a machine, finding the correct POS tag is a disambiguation task, which tells us that some words we want to tag can be ambiguous: they have more than one possible part-of-speech and our goal is to resolve these ambiguities by choosing the best tag for the given context. Unfortunately not only part-of-speech ambiguity seems to be a big problem in natural language processing. Part-of-speech tag-

gers also have some problems with words that they have not encountered while learning because they can not cover all possible words due to its restricted size and so more unknown words leads to more guessing of the corresponding tags, thereby resulting in more errors. In this paper, we are going to focus on the phenomena mentioned above and have a look at a couple of different approaches to see how different approaches to part-of-speech tagging handle the problem of ambiguity and unknown words.

2 Methodology

In order to approach the problem of part-of-speech ambiguity and unknown words in part-of-speech tagging we had to analyse the behaviour of different taggers in different setups. As libraries for programming we chose NLTK, the Natural Lanugage Toolkit, which is a suite of libraries and programs for symbolic and statistical natural language processing and sklearn, a machine learning library, both written for the Python programming language.

2.1 Ambiguous and unknown words

As a first step we chose 20 sentences that contain words with a highly ambiguous part-of-speech tag depending on the context in which the word is embedded. Each sentence has additionally up to three sentences next to it, to show the reader another view and another reading of the particular part of speech in a different environment [see Appendix A]. For the unknown words we searched mainly the Oxford English Dictionary[1] for their regularly updating 'new words' lists and extracted 15 samples of words that might be unknown to a part-of-speech tagger into a sentence [see Appendix A]. After choosing the sentences we had to set a 'gold standard' for evaluation. For evaluation purposes, we annotated both sets of sentences by hand.

2.2 Tagset

The first parameter we want to investigate is the impact of the chosen tagset. Therefore we chose two different sized and detailed tagsets, the 'Penn Treebank' and the 'Brown' tagset to annotate our sentences.

[1] https://public.oed.com/updates/new-words-list-march-2018/

2.3 Corpora

Next we wanted to examine the impact of different corpora on the behaviour of our taggers. For this case we focussed not only on learning from a single corpus but also on learning from different corpora with their different annotation styles and sizes. NLTK provides a couple of different corpora that can easily be accessed. For our task we picked the Penn Treebank corpus, the Brown corpus, the CoNLL 2000 tagged chunking corpus, and the MASC tagged corpus. The Treebank corpus contains approximately 40,000 words of the Wall Street Journal. The Brown corpus is by far the biggest with 1,150,000 words. The CoNLL 2000 corpus which contains around 270,000 words and the MASC tagged corpus, which consists of approximately 500,000 words, taken from the Open American National Corpus. It is important to mention, that all the annotated corpora use the Penn Treebank tagset except for the Brown corpus, which is the only one in our collection that sticks to the Brown annotations.

2.4 Taggers

For this paper we picked only a small collection of possible taggers. We started with the the unigram tagger, which tags, as the name already suggests, every token separately and without context information. The unigram tagger is a rather simple statistical algorithm. For each token the most likely part-of-speech tag is assigned. To avoid the assignment of 'None', we implemented a default tagger as a backoff tagger. The next step involves higher n-gram taggers. We wrote a program that lets a bigram tagger tag the same sentences trained on the exact same corpora as the unigram tagger. Since we have bigrams now, it only makes sense to implement a backoff tagger here for the default bigram tagger to reach an acceptable accuracy, this time we chose a unigram- and a default tagger. To come to an end with the simple n-gram taggers, the last tagger to compare the outcome with was a trigram tagger which is based on second order Markov models. Again, we used a backoff chain, containing a bigram-, a unigram- and a default tagger to improve our socres. The next tagger in our line of experiments was the TnT Tagger, with a silghtly different algorithm. The TnT tagger uses second order Markov models for part-of-speech tagging. "The states of the model represent tags, outputs represent the words. Transition probabili-

ties depend on the states, thus pairs of tags. Output probabilities only depend on the most recent category" (Brants, 2000). Another, more complex tagger that we used next was the Brill tagger. "The tagger works by automatically recognizing and remedying its weaknesses, thereby incrementally improving its performance" (Brill, 1992). As an initial tagger for the Brill tagger chain, we went with a simple trigram tagger with a bigram-, unigram- and default-tagger backoff chain. The classifier-based POS tagger class is a subclass of classifier-based tagger, which is a sequential tagger that uses a classifier to choose the tag for each token in a sentence. It implements a feature detector that combines many of the techniques of the other taggers mentioned here into a single feature set. The feature detector finds multiple length suffixes, does regular expression matching, and looks at the uni- to trigram history to produce a complete set of features for each word. The produced feature sets are used to train the internal classifier and also for classifying words into part-of-speech tags (Perkins, 2014). The second to last tagger, the perceptron tagger, uses, as the name suggests, a neural network to learn the part-of-speech tags. The perceptron is an algorithm for learning a binary classifier. It is also the standard NLTK tagger and is already trained on the Penn Treebank corpus. To train the tagger on additional corpora, we had to implement our own perceptron tagger with the desired training corpora. The last tagger we used was a Support Vector Machine (SVM). Support Vector Machines are machine-earning algorithms for binary classification. They try to find the hyperplane that maximises the margin which is the distance between the hyperplane and the nearest points. In case the classes are not linearly separable, the SVM mapps the feature vectors into a higher dimensional space via a kernel function, where they can be separated easier. For multiclass classification, commonly used methods are One vs. Rest and One vs. One. Since we use the LinearSVC from the sklearn library, we also use the default One vs. Rest approach.

3 Analysis

3.1 Ambiguous words

In Table 1 we can see the performance, more precisely the accuracy and the precision, recall and the F_1-score of the different taggers on our 'ambiguous words'. Table 2 shows the performance

Tagger	Corpus	Acc.	P.	R.	F1
Unigram	Treebank	78%	80%	78%	78%
Unigram	Brown	80%	79%	80%	79%
Unigram	CoNLL	80%	81%	81%	80%
Unigram	MASC	83%	83%	83%	83%
Bigram	Treebank	80%	82%	80%	80%
Bigram	Brown	84%	83%	84%	83%
Bigram	CoNLL	84%	84%	84%	83%
Bigram	MASC	85%	85%	85%	84%
Trigram	Treebank	80%	82%	80%	80%
Trigram	Brown	84%	82%	84%	83%
Trigram	CoNLL	83%	84%	83%	83%
Trigram	MASC	85%	85%	85%	85%
TnT	Treebank	78%	85%	78%	80%
TnT	Brown	87%	86%	87%	86%
TnT	CoNLL	85%	89%	85%	86%
TnT	MASC	87%	89%	87%	88%
Brill	Treebank	83%	83%	83%	82%
Brill	Brown	83%	82%	83%	82%
Brill	CoNLL	84%	85%	84%	84%
Brill	MASC	85%	85%	85%	85%
Class.	Treebank	82%	86%	82%	83%
Class.	Brown	83%	85%	83%	83%
Class.	CoNLL	83%	86%	83%	84%
Class.	MASC	88%	90%	88%	89%
Percep.	**Treebank**	**89%**	**91%**	**89%**	**90%**
Percep.	Brown	83%	87%	83%	84%
Percep.	CoNLL	86%	88%	86%	86%
Percep.	MASC	82%	85%	82%	82%
SVM	Treebank	84%	86%	85%	85%
SVM	Brown	81%	84%	81%	82%
SVM	CoNLL	86%	87%	86%	86%
SVM	MASC	82%	83%	82%	82%

Table 1: Accuracy, Precision, Recall and F_1-score for 'ambiguous words'

on the 'unknown words'. For the 'ambiguous words' sentences, when we take a look at the accuracy, the taggers performed from 78% to 89%. None of them went below an accuracy of 78%. One can notice, that the worse results in accuracy and F_1 were tagged by taggers trained on the Treebank Corpus, which has only 40,000 words, significantly fewer words than the other corpora we used. The other corpora, ranging from 270,000 to 1,150,000 words, performed similarly to each other in most of the cases and better than the same taggers trained on the Treebank Corpus. The bad F_1-scores were as well achieved when trained on the treebank corpus. We discard the 82% of the Perceptron and the SVM on the MASC corpus, because they did not work with the full training set so some reason we couldn't figure out. One can notice, that training on a very big corpus like the Brown corpus, which is of course more time consuming, has no significant effect on the accuracy, no matter which tagger is used. The impact on the F_1-score, is more significant. Here we achieved good scores with the MASC corpus. A larger tag set as used by the brown corpus is not helpful either. Astonishing was, that the best performance was the combination of perceptron tagger and treebank corpus, which was unexpected, because the treebank corpus performed mostly worse when compared to the other corpora when used with other taggers. This might be the reason, why this particular combination is the ntlk standard POS-tagger.

3.2 Unknown words

When taking a look at Table 2, we see, that with unknown words, the outcome is not as consistent as with the ambiguous sentences. In terms of accuracy, the worse results are split between the Treebank and the Brown corpus. The accuracy varies across all the tagger. The worse F_1-scores are dominated by tagger trained on the Brown corpus. When using the more sophisticated tagger, the TnT tagger and below, we see an improvement in F_1-score. The best accuracy is shared by the combination of Perceptron tagger, trained on the Treebank, and the Perceptron tagger, trained on CoNLL2000. These combinations also result in the leading F_1-score for the unknown words, except for the the recall, which scored 99% when we used the combination of TnT tagger, trained on the MASC corpus. To mention an exceptions

Tagger	**Corpus**	**Acc.**	**P.**	**R.**	**F1**
Unigram	Treebank	83%	90%	83%	82%
Unigram	Brown	83%	84%	83%	81%
Unigram	CoNLL	84%	90%	84%	82%
Unigram	MASC	83%	89%	83%	82%
Bigram	Treebank	84%	92%	84%	85%
Bigram	Brown	82%	84%	82%	80%
Bigram	CoNLL	86%	92%	86%	85%
Bigram	MASC	86%	93%	86%	86%
Trigram	Treebank	84%	92%	84%	85%
Trigram	Brown	83%	85%	83%	81%
Trigram	CoNLL	86%	92%	86%	85%
Trigram	MASC	86%	93%	86%	86%
TnT	Treebank	79%	98%	79%	86%
TnT	Brown	81%	92%	81%	84%
TnT	CoNLL	81%	98%	81%	86%
TnT	**MASC**	83%	**99%**	83%	88%
Brill	Treebank	85%	89%	85%	84%
Brill	Brown	85%	85%	85%	83%
Brill	CoNLL	86%	89%	86%	84%
Brill	MASC	88%	93%	88%	88%
Class.	Treebank	84%	90%	84%	86%
Class.	Brown	80%	88%	80%	82%
Class.	CoNLL	88%	93%	88%	90%
Class.	MASC	90%	93%	90%	91%
Percep.	**Treebank**	**92%**	95%	**92%**	**93%**
Percep.	Brown	86%	88%	86%	86%
Percep.	**CoNLL**	**92%**	94%	**92%**	**93%**
Percep.	MASC	84%	91%	84%	86%
SVM	Treebank	90%	94%	90%	92%
SVM	Brown	84%	88%	85%	85%
SVM	CoNLL	89%	94%	89%	91%
SVM	MASC	86%	91%	87%	88%

Table 2: Accuracy, Precision, Recall and F_1-score for 'unknown words'

found in the unknown words section, the classifier-based tagger trained on the Brown corpus had an unexpected low accuracy (80%), whereas the same tagger trained on the MASC corpus reached over 90% accuracy. This is consistent with the statement above, where we state that a bigger corpus and a larger tag-set neccessarily do not have a positive impact on the tagging task.

4 Discussion

We found out that there are a few screws to turn which can help us to get along better with ambiguous and unknown words, two major problems while part of speech tagging. We all agree with the fact that it is impossible to teach a part-of-speech tagger, be it unsupervised or supervised, be it by applying rules or statistics, to correctly guess the right part-of-speech tag for every word. But as we showed, there are some parameters which could at least increase the chance of a proper solution during the process of part-of-speech tagging. When we take a look at the corpora used in our experiment setups, we see an positive impact of the corpus size on the accuracy, precision, recall, and F1 score for most of our taggers. The scores vary depending on how big the specific corpus is but mostly improve when taking a bigger corpus with more words covered to learn. More input increases the chance that the tagger finds the word while learning and makes it more likely to get the correct tag. When we take a look at the performance of the Penn Treebank corpus with its 40,000 words and compare it to the next bigger one, the MASC corpus with 270,000 words, we can see an enormous improvement in the evaluation scores as mentioned in the last section. Unfortunately it is not as easy as that; this pattern depends on the tagger we used, since we got worse outcomes with bigger corpora and the same tagger, and sometimes better outcomes with smaller corpora but more sophisticated tagging algorithms. The best example here is the Perceptron Tagger which is pretrained on the Treebank corpus. The next steps, from 270,000 to 500,000 or over 1,000,000 words, like the Brown corpus provides, did not show the same huge impact on the evaluation scores but with some taggers we could still see an improvement. Of course, we did not take a close look at each word or combinations of words in isolation - that would go beyond the work of this paper, but we think that the overall scores show the results we were searching for. Our findings show us furthermore that the quality of the corpus, not necessarily the number of words a corpus comes with is crucial for a good score. An explanation why the performance of the Brown corpus was sometimes way worse than expected leads us to the second parameter: The tagset. The complexity of the tagset could also play a role in tagging ambiguous or unknown words; the Brown tagset seems to be too specific for some tokens and sometimes confusing when compared to a 'simpler' tagset like Penn Treebank tagset. To make this statement definitively, we have to provide further evidence in a bigger test. Another result of our experiments is that a corpus can be as good as can be and cover a lot of possible phrases but you need a good tagger to learn the word/tag combinations, which leads us to the third possible parameter, the tagger. In the sections above we tested a lot of different tagger and their behaviour on different input. Here we found out that the more sophisticated tagger beat the simpler algorithms most of the time. Given the same test- and the same tag-set, we see an improvement for example when taking a higher order Markov model. Unfortunately this is not always true when we switch the corpus; here some corpora achieved better results with lower order Markov models, at least at some words or tags. The more sophisticated the tagger became, the better the average results, of course, with the corresponding setup consisting of tagging algorithm and corpus with the corresponding tagset. We achieved pretty good results with most of the taggers, especially at the classifier-based one, the SVM tagger and the perceptron tagger. Overall we can conclude that there are ways to get better scores for tagging sentences that contain ambiguous or unknown words but we showed that the solution is not as simple as we first thought. Tagging is still not at its end and there is much work to do to improve when trying to resolve ambiguity or to tag unknown words.

References

Thorsten Brants. 2000. Tnt: A statistical part-of-speech tagger. In *Proceedings of the Sixth Conference on Applied Natural Language Processing*, ANLC '00, pages 224–231, Stroudsburg, PA, USA. Association for Computational Linguistics.

Eric Brill. 1992. A simple rule-based part of speech tagger. In *Proceedings of the Third Conference on*

Applied Natural Language Processing, ANLC '92, pages 152–155, Stroudsburg, PA, USA. Association for Computational Linguistics.

Daniel Jurafsky and James H. Martin. 2017. *Speech and Language Processing (3Rd Edition, draft)*. Upper Saddle River, NJ, USA.

Christopher D. Manning and Hinrich Schuetze. 1999. *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, MA, USA.

Jacob Perkins. 2014. *Python 3 text processing with NLTK 3 cookbook*. Packt Publ., Birmingham.

A Appendix

Sentences containing ambiguous words

"Time **flies** like an arrow."
"Time and **flies** are like an arrow."
"We **saw** her duck with a saw."
"We **saw** her duck and it was small."
"The complex **houses** married and single soldiers and their families."
"The complex contains several **houses**."
"The **old** man the boat."
"The **old** man sits in the boat."
"The **old** train left the station."
"The **old** train the young men."
"The trash **can** be smelly."
"The trash **can** is smelly."
"The building **blocks** the sun."
"They are **building** blocks."
"There are **building** blocks."
"**That** that exists in that area makes me nervous."
"I know **that** this exists in the area of London."
"The statue **stands** in the park."
"The statue **stands** in the park are rusty."
"The cotton clothing is made of **grows** in Alabama."
"The cotton that **grows** in Alabama is good."
"The **back** door."
"The thing on my **back**."
"We win the voters **back**."
"I promised to **back** the bill."
"During its **centennial** year, The Wall Street Journal will report events of the past century that stand as milestones of American business history."
"We have a **centennial** this year."
"I have a **round** table."
"Yesterday we bought a **round** of cheese."
"I think you should **round** out your interests."
"I have to work the year **round**."
"She remembered everything in **minute** detail."
"She remembered that we meet in a **minute**."
"She grabs her poles and **skis** down the mountain."
"In winter she always **skis** down the mountain."
"The insurance company receives a lot of calls and **claims** that you submit your forms too late."
"He had his **claims** back in 1756."
"When the detective asks, the **honest** reply."
"When the detective asks, the **honest** people reply."
"I wonder if Will will sign his **will**."
"I have my own **will**."
"Rose **rose** to put rose roes on her **rose**."
"Rose woke up to water all the red **roses**.
"The flowers are **rose**."
"Let the captain **ship** to the ship!"
"The captain owns a **ship**."

Sentences containing 'unknown' words

"I like to wear my **mankini**."
"Yesterday I bought a lot of **bling**."
"This jewellery is very **droolworthy**."
"I forgot which state is a **Purple State**."
"I **androgynously** matched you wrong."
"This society is **degendered**."
"You **envenomate** your father."
"This is an **eventitive** event."
"This is an **eventitive**."
"**Everwhen** I go to the city this happens."
"I will **jackhammer** this building."
"I **jackhammer** this building."
"These are **unjournalistic** methods."
"The **unknot** arises in the mathematical theory of knots."
"The virus can then spread to new individuals through **skin-to-skin** contact."