Stomp, a POS-tagger with a different view

Jonas Sjöbergh
Department of Numerical Analysis and Computer Science
KTH, Nada, SE-100 44 Stockholm, Sweden
jsh@nada.kth.se

Abstract

In this paper a data-driven method for Part-of-Speech tagging not using any n-grams of tags is presented. The method matches the text to be tagged to as long continuous strings from the training data as possible and assigns each match the same tags as the matching part of the training data. There is also a back-off method for short matches and the treatment of unknown words differs from the most common ways of handling them. The method is evaluated on Swedish text. The method is slightly less accurate than available state of the art taggers. It is faster than some of these, and uses different information, which makes it a useful addition to an ensemble of taggers, especially for unknown words.

1 Introduction

Part-of-speech tagging is a very important step in most advanced language technology systems. It is a nontrivial problem due to ambiguous words and unknown words, i.e. words not in the training data. POS-tagging is harder for some languages than others. Typical accuracy for Swedish taggers is between 94% and 96% (Megyesi 01). Taggers may either be based on manually written rules for a specific language (Karlsson *et al.* 95), language independent, but trained on a tagged corpus (Brants 00; Ratnaparkhi 96; Schmid 94; Brill 92) or a combination of both (Carlberger & Kann 99).

An obvious way to improve tagging is to combine the results of several taggers, hoping to take advantage of the fact that different taggers are good at tagging different constructions. This is called using an ensemble of classifiers (in this case taggers). A good overview of why ensembles are good and different ways of combining classifiers is given in (Dietterich 97). The basic idea is that classifiers making uncorrelated errors can correct each other.

We have developed a new data-driven tagger, Stomp (**Sto**ckholm **M**atching **P**art-of-speech tagger), primarily for Swedish but relatively language independent, that does not use n-grams of tags at all. This was intended to make it different from other taggers and thus hopefully useful in an ensemble of taggers. Stomp tags text by matching sequences of words. It finds the longest match between a word in its current context and the training data and assigns the tag in the matching data to the word.

Stomp has slightly lower accuracy, 94.5%, than other available taggers. It is slower than the fastest available taggers, but faster than some (and very fast when training). Stomp has high accuracy on some types of words, which are not necessarily easy to tag for other types of taggers. It handles unknown words differently than most taggers. Some types of unknown words are handled well. Stomp can also find inconsistencies in the annotation of a tagged corpus.

2 Details of the algorithm

2.1 Known words

For a known word, Stomp proceeds as follows: Find all matches between this word in this context and the training corpus. Select the match that is "best". Which match is "best" is measured by computing the product of the lengths of the matching left context and the matching right context. To rank one-sided matches by length, a small constant is added to the lengths of the matching contexts before multiplying. The word is then assigned the same tag as the matching word in the best match.

If there are several equally good matches, the most common matching tag (in these matches) is chosen. If it is still a tie, the one first encountered is chosen.

Many known words always have the same tag in the training corpus, and will thus always be assigned this tag. These words are detected and assigned their tag without the computationally heavy matching procedure. Not all words that are unambiguous in the training data are actually unambiguous, accuracy on these words is 98% in the tests.

2.2 Unknown words

Stomp was developed for use primarily on Swedish, which is a compounding language. Over 60% of the unknown words are compounds. When an unknown word is found, Stomp first checks if it is a compound of words it already knows. In Swedish, the last part of a compound determines the word class, so only the last part is checked.

For an unknown n-letter word with characters $c_1c_2...c_n$ this is done by checking if $c_ic_{i+1}...c_n$ is a known word, for $2 \leq i \leq n-6$. If a substring matching a known word is found, the unknown word is replaced by the known (sub)word, using the longest if several alternatives exist. This is done when reading the text, so the replacement word is used for tagging the unknown word and as context for neighboring words.

Many Swedish compounds end with a word shorter than six letters (which is the limit Stomp uses), but allowing shorter words leads to other problems. Many common word suffixes for regular inflection forms are also known words in the lexicon. One example is “ande”, which is a common adjective suffix and also a noun.

Words not recognized as compounds are tagged with the help of the hapax words (words occurring only once in the training data). These make up approximately half the words in the training lexicon, or 4,5% of the words in the training texts. If the last four letters of the unknown word is the same as the suffix of a hapax words, then it is treated as if it occurred on all the positions where hapax words with this suffix occur. If there is no hapax with this suffix in the training data the word is treated as if it occurred on all positions where any hapax word occurs.

47% of all unknown words are recognized as compounds and 88% of these are tagged correctly. For other unknown words the accuracy is 68%, which gives a total accuracy of 77% on unknown words. State of the art taggers achieve 80% to 90% accuracy on unknown words on the same dataset.

2.3 Back-off for short matches

Tagging accuracy on words with short matches is low, and short matches are common. Stomp behaves much like a unigram tagger on words with short matches. One way of increasing tagging accuracy on these words is to use the tags in their contexts.

Stomp has a back-off method using these tags. After doing the tagging as described earlier, Stomp rechecks all words with short matches, this time matching first on words as before, and when no more words match, continuing the matching on tags (tags in the training data and the tags assigned earlier). This changes about 3% of the tags, increasing tagging accuracy from 93.8% to 94.5%, but takes a lot of time, more time than the original tagging step.

Of the tags that are changed, 8% are one error changed to another error, 33% are a correct tag changed (to an incorrect one) and 59% are an incorrect tag changed to the correct tag.

Stomp rechecks words starting on the last word and working backwards. Checking the words in any other order would also work. Different orders can give different results, since the matching tag context changes for some words when a word is retagged. In practice the order makes very little difference. Likewise, running the back-off again after the first back-off has finished changes very few tags, about 0.1%.

2.4 Length of matches

In the tests performed, the mean length of matches, including the word itself, is 2.8 words when tagging. When using the back-off method, matches for the treated words are increased from 2.3 words to 3.6 words. These matches and the long matches for words where no back-off was used, have a mean length of 4.0 words. This was for a balanced training corpus of 1.1 million words and a test text of about 60 000 words from the same domain. Unambiguous words were not included in these numbers, since no matching is done for them.

3 Performance

3.1 Tagging accuracy

The Stockholm-Umeå Corpus (SUC) (Ejerhed *et al.* 92), a manually corrected tagged corpus of Swedish, was used for training and testing. The tag set in SUC was slightly modified, resulting in a tag set of 150 tags.

Training and testing was performed by splitting SUC into two parts: a test set consisting of about 58 000 words, and a training set consisting of the rest of the corpus, about 1.1 million words. This results in approximately 5% of the words in the test data being unknown words. To increase reliability of results, the part of SUC used as test data was chosen in 10 different ways (all 10 test sets were disjoint) and the training and testing repeated once for every choice. The test data was chosen to be as balanced as possible.

Testing was done by stripping the tags from the test data and letting the tagger tag the text. An assigned tag was then deemed correct if it was the same as the original tag.

Stomp tags 94.5% of all words correctly (93.8% without back-off). It tags 77% of unknown words correctly, which means 22% of all errors were unknown words. A baseline unigram tagger, choosing the most

Type of match	Stomp	No back-off	fnTBL	Mxpost	TnT	Words
All words	94.5	93.8	95.6	95.5	95.9	100.0
Known words	95.5	94.9	96.5	96.1	96.3	94.6
Unknown words	77.4	75.8	79.8	85.1	88.5	5.4
Compound unknown words	88.4	87.8	82.2	85.5	91.2	2.6
Non-compound unknown	67.6	64.9	77.7	84.9	86.0	2.9
Word only	80.5	75.5	86.4	88.5	90.0	6.2
Short edge	92.0	90.9	93.9	93.9	94.1	35.4
Long edge	96.6	96.1	97.0	96.4	96.5	0.9
1+1 word	93.9	93.9	94.9	95.0	94.3	9.2
Short good	95.4	95.4	95.9	96.1	95.2	5.4
Long good	97.8	97.8	97.1	97.0	96.4	1.6
Unambiguous word	98.7	98.7	98.3	97.9	98.8	41.3

Table 1: Tagging accuracies (in % correctly tagged words) for different types of matches. "Edge" means the matching word was the first or last word of the match, where long means at least 4 matching words, and short means 2 or 3. "1+1" means there was one word on each side in the match. "Good" means there was matching context on both sides, where short means 2 or 3 words on one side and 1 word on the other, long is all other two-sided matches. Unambiguous words means words with only one tag in the training data. Unknown words are included in the matching measurements, since Stomp treats these as regular words (though not as the unknown word itself, see Section 2.2) when matching.

common tag for known words and the most common open word class tag for unknown words, achieves 87.3% accuracy (25.4% on unknown words) on the same data.

In Table 1 accuracy information for different types of matches is presented. There is also accuracy measurements for three state of the art taggers: a Brill-tagger, fnTBL (Ngai & Florian 01); a maximum entropy tagger, Mxpost (Ratnaparkhi 96) and an HMM-tagger, TnT (Brants 00).

Stomp performs well on long matches, especially on long matches with matching context on both sides. It also performs well on unknown words it believes are compounds. On these types of matches it outperforms several of the state of the art taggers. These types of matches are not very common, though, and Stomp performs poorly on short matches, which are common. Using the scores of the matches Stomp uses to choose the best match, it is easy to separate the different types of matches. It is thus easy to use Stomp for only some types of words, and let another tagger tag the rest.

Stomp makes good use of larger training sets, since the information it uses (series of words) is so sparse. Not having any larger annotated resources available, three million words of newspaper clips from the web were automatically tagged with a voting ensemble of taggers. When this was added to the training data of Stomp, the accuracy increased to 95.1%, despite the new data not being 100% correct (probably around 96.5% correct). About half the increase was on unknown words which were no longer unknown since they occurred in the new training texts, which Stomp generally makes a lot of errors on. The other half was on known words, though, so Stomp seems to use large training sets well.

Training size in words	Words per second
4 000 000	2 200
2 000 000	3 400
1 000 000	4 800
100 000	25 000
10 000	36 000

Table 2: Tagging speed, measured on tagging only (including back-off, ignoring time for reading corpus, printing output etc.). Measured on a SunBlade 100.

3.2 Tagging speed

Since tagging is done by matching a text to the corpus, tagging time increases with both corpus size and text size. Most taggers have a separate training step, and then only depend on the text size. Stomp has zero training time (no training is done), while tagging time is quite high. This is to be expected, since Stomp uses a form of instance based learning, which generally makes the classification of new data computationally heavy.

Tagging a text of 58 000 words with training data consisting of 1.1 million words takes 30 seconds on a SunBlade 100. Of this, 15 seconds are spent on reading the corpus, 3.5 seconds on tagging and 9 seconds on back-off for short matches. This amounts to 2 000 words per second all in all, and 4 000 words per second excluding the time for reading training data.

Other taggers vary in speed, of the taggers in Section 3.1, TnT is very fast (8 seconds on the same task), while Mxpost and fnTBL are slower than Stomp (a few minutes).

4 Applications

4.1 Finding errors in a tagged corpus

Stomp's very high accuracy on long matches can be useful, for instance when correcting a manually tagged corpus. If a long word sequence is found several times in a corpus and the annotation differs for words in the middle of the sequence, there is likely to be an error or inconsistency in the annotation.

This was tested on the SUC corpus (Ejerhed *et al.* 92). Stomp was used to find all matches with at least two matching words on each side where the tagging differed in different parts of the corpus. This gave about 2 000 matches. Some of these were then manually checked by a linguist. In most cases any of the two tags could have been used in both matches, so the tagging was inconsistent (one of the tags should have been used for all occurrences or the ambiguity should have been kept), and in some cases one of the annotations was wrong (and in some cases there was a genuine difference between the two matches).

When evaluating taggers, they will often make "errors" on words with inconsistent annotation, since several suggestions are correct, but only one will be considered correct in the evaluation, and the tagger has no way of guessing which tag was used in this part of the test data. They also degrade the quality of the training data by introducing differences in the annotation where there is no real difference. This is also true for words where the annotation is wrong.

See (Källgren 96) for a thorough discussion of evaluation of automatic taggers, tagging errors and ambiguous words in SUC.

4.2 Usefulness in an ensemble

The intention when creating Stomp was to create a tagger which uses different information than most other taggers. Mxpost uses information similar to the information Stomp uses, Mxpost looks at (among other things) the preceding and following word, and the tags in the context of a word to tag. For long matches Stomp uses different information than Mxpost, though, and they also treat unknown words differently.

Tagger	Accuracy (%)
TnT	95.9
fnTBL	95.6
Mxpost	95.5
TreeTagger	95.1
Stomp	94.5
TnT+Mxpost+TreeTagger	96.2
Mxpost+TreeTagger+Stomp	96.3
TnT+TreeTagger+Stomp	96.1
TnT+Mxpost+Stomp	96.4
Mxpost+TreeTagger+fnTBL	96.3
TnT+TreeTagger+fnTBL	96.1
TnT+Mxpost+fnTBL	96.5
All five	96.5

Table 3: Tagging accuracy of ensemble taggers when trained on one million words of Swedish. More than one tagger means simple voting was used, with ties broken by the most accurate tagger. The ensemble with all five taggers is actually more accurate, by almost 0.1% (significant using McNemar's test at the 5% level (Everitt 77)), than the best trio, but the difference is too small to show up in this table.

To test whether Stomp is actually useful in an ensemble of taggers a small test was performed. An ensemble was created by using several publicly available taggers, TnT (Brants 00), Mxpost (Ratnaparkhi 96) and TreeTagger (Schmid 94). These were trained and tested in the same way as Stomp, see Section 2.2. The taggers then voted on which tag to choose, with ties being resolved by using the tag suggested by the most accurate single tagger in the ensemble. The accuracies of the ensembles is presented in Table 3.

When exchanging one of the taggers for Stomp and then using the new ensemble, the ensemble accuracy increased except when removing Mxpost (TreeTagger and TnT are quite similar so they do not complement each other very well).

Then the same was done with fnTBL (Ngai & Florian 01), which also differs a lot from the taggers in the ensemble, and is also very accurate alone (unlike Stomp). fnTBL increased the accuracy of the ensembles about as much as Stomp, once by a little more, twice by a little less, but the difference was small.

In all cases except when removing Mxpost, the ensembles with Stomp had greater accuracy on unknown words than the original ensemble and the ensemble with fnTBL instead of Stomp. This indicates that it is mainly the handling of unknown words in Stomp that is useful for the ensembles. Also, if the minimum length allowed for compounds is lowered in Stomp, the accuracy of an ensemble with Stomp increases slightly, while the accuracy of Stomp decreases noticeably. The known word accuracy is also increased in ensembles with Stomp, so it contributes useful information there too, but not as much as for unknown words.

Finally, all five taggers were combined. This gave the highest accuracy of all, although not much higher than the best trio. All ensembles were more accurate (significant using McNemar's test at the 5% level (Everitt 77)) than the best tagger (TnT) alone.

One could also use Stomp's scoring of matches to determine which tagger to trust. One simple example would be to use the tag chosen by Stomp for words with long two-sided matches, the tag chosen by Mxpost for other two-sided matches and the tag chosen by TnT for all other words.

Stomp has also been used in a more thorough evaluation of ensemble methods (Sjöbergh 03).

5 Conclusions

Stomp is not the fastest tagger, and it is not the most accurate tagger either. It is faster than several other taggers, though, and it uses different information than most. Stomp can be successfully combined with other taggers, by for instance using it as one tagger in an ensemble.

The handling of unknown words, especially those believed to be compounds, seems promising even though it is quite naive. A more advanced method based on these principles might be a good addition even to some state of the art taggers. This was tested by letting Stomp change unknown words believed to be compounds and then running Mxpost on the resulting text, which resulted in an unknown word accuracy of 86.6% compared to 85.6% when using Mxpost on the original text (tested on 57 000 words).

Stomp can also be used to find inconsistencies and errors in the annotation of a tagged corpus.

6 Future work

Since Stomp uses very sparse information, using Stomp with more training data to see how high the accuracy will get would be interesting. Evaluating Stomp on other languages reasonably similar to Swedish, other types of tagging, such as shallow parsing, could also be done.

Improving tagging of words Stomp finds hard could probably be done, for instance by using statistics of common tag n-grams. This would make Stomp behave more like other taggers, which might not be desirable even if the tagging accuracy is increased.

7 Acknowledgments

This work has been funded by The Swedish Agency for Innovation Systems (VINNOVA).

I would like to thank my supervisor professor Viggo Kann. I would also like to thank Stockholm University for letting me use the Stockholm-Umeå Corpus.

References

(Brants 00) Thorsten Brants. TnT – a statistical part-of-speech tagger. In *Proceedings of the 6th Applied NLP Conference, ANLP-2000*, Seattle, USA, 2000.

(Brill 92) Eric Brill. A simple rule-based part-of-speech tagger. In *Proceedings of ANLP-92, 3rd Conference on Applied Natural Language Processing*, Trento, IT, 1992.

(Carlberger & Kann 99) Johan Carlberger and Viggo Kann. Implementing an efficient part-of-speech tagger. *Software – Practice and Experience*, 29(9):815–832, 1999.

(Dietterich 97) Thomas Dietterich. Machine learning research: Four current directions. *AI Magazine*, 18(4):97–136, 1997.

(Ejerhed *et al.* 92) Eva Ejerhed, Gunnel Källgren, Ola Wennstedt, and Magnus Åström. The linguistic annotation system of the Stockholm-Umeå Corpus project. Technical report, Department of General Linguistics, University of Umeå (DGL-UUM-R-33), Umeå, Sweden, 1992.

(Everitt 77) Brian Everitt. *The Analysis of Contingency Tables*. Chapman and Hall, 1977.

(Källgren 96) Gunnel Källgren. Linguistic indeterminacy as a source of errors in tagging. In *Proceedings of COLING-96*, Copenhagen, Denmark, 1996.

(Karlsson *et al.* 95) Fred Karlsson, Atro Voutilainen, Juha Heikkila, and Atro Anttila. *Constraint Grammar, A Language-independent System for Parsing Unrestricted Text*. Mouton de Gruyter, 1995.

(Megyesi 01) Beáta Megyesi. Comparing data-driven learning algorithms for POS tagging of Swedish. In *Proceedings of NAACL-2001*, Carnegie Mellon University, Pittsburgh, USA, 2001.

(Ngai & Florian 01) Grace Ngai and Radu Florian. Transformation-based learning in the fast lane. In *Proceedings of NAACL-2001*, Carnegie Mellon University, Pittsburgh, USA, 2001.

(Ratnaparkhi 96) Adwait Ratnaparkhi. A maximum entropy part-of-speech tagger. In *Proceedings of the Empirical Methods in Natural Language Processing Conference*, University of Pennsylvania, Philadelphia, USA, 1996.

(Schmid 94) Helmut Schmid. Probabilistic part-of-speech tagging using decision trees. In *Proceedings of the International Conference on New Methods in Language Processing*, Manchester, UK, 1994.

(Sjöbergh 03) Jonas Sjöbergh. Combining pos-taggers for improved accuracy on Swedish text. In *Proceedings of NoDaLiDa 2003*, Reykjavik, Iceland, 2003.