Data-Driven Part-of-Speech Tagging of Kiswahili

Guy De Pauw[1], Gilles-Maurice de Schryver[2,3], and Peter W. Wagacha[4]

[1] CNTS - Language Technology Group, University of Antwerp, Belgium
guy.depauw@ua.ac.be
[2] African Languages and Cultures, Ghent University, Belgium
gillesmaurice.deschryver@ugent.be
[3] Xhosa Department, University of the Western Cape, South Africa
[4] School of Computing and Informatics, University of Nairobi, Kenya
waiganjo@uonbi.ac.ke

Abstract. In this paper we present experiments with data-driven part-of-speech taggers trained and evaluated on the annotated Helsinki Corpus of Swahili. Using four of the current state-of-the-art data-driven taggers, TnT, MBT, SVMTool and MXPOST, we observe the latter as being the most accurate tagger for the Kiswahili dataset.We further improve on the performance of the individual taggers by combining them into a committee of taggers. We observe that the more naive combination methods, like the novel plural voting approach, outperform more elaborate schemes like cascaded classifiers and weighted voting. This paper is the first publication to present experiments on data-driven part-of-speech tagging for Kiswahili and Bantu languages in general.

1 Introduction

It is well-known that Part-of-Speech (POS) taggers are crucial components in the development of any serious application in the fields of Computational Linguistics (CL), Natural Language Processing (NLP) or Human Language Technology (HLT). While great strides have been made for (major) Indo-European languages such as English, Dutch and German, work on the Bantu languages is scarcely out of the egg. The Bantu languages - of which there are roughly five to six hundred - are basically agglutinating in nature, are characterized by a nominal class system and concordial agreement, and are spoken from an imaginary line north of the Democratic Republic of the Congo all the way down to the southern tip of the African continent.

A particularly active region with regard to work on POS taggers for the Bantu languages is South(ern) Africa, but so far the projects have unfortunately not gone much beyond the development of (proposed) tagsets and, in some cases, prototype modules for morphological analysis. In this regard, the EAGLES tagset was adjusted for Setswana [1], a different tagset and suggestions to venture into Transformation-Based Tagging were presented for isiXhosa [2], yet another tagset and a combination of rule-based symbolic tagging and statistical tagging were offered as a corpus-processing tool for Sesotho sa Leboa [3,4], and a prototype finite-state morphological analyzer was developed for isiZulu [5,4].

For Kiswahili — a Bantu language spoken by up to fifty million people in East Africa (which makes it one of the most widely spoken African languages) — the situation is markedly different. Close to two decades of work at the University of Helsinki resulted in a

Petr Sojka, Ivan Kopeček and Karel Pala (Eds.): TSD 2006, LNAI 4188, pp. 197–204, 2006.

relatively large corpus, the Helsinki Corpus of Swahili (HCS) [6], which has been thoroughly analyzed and carefully annotated using a two-level finite-state formalism, with morphological disambiguation carried out using a Constraint Grammar Parser [7]. The POS tag information in HCS allows one to use supervised learning techniques to build data-driven POS taggers and to perform a quantitative comparative evaluation of the available techniques. The latter is exactly the purpose of this paper.

2 An Annotated Corpus of Kiswahili: HCS

Lexical ambiguity in Kiswahili is limited, making POS tagging relatively straightforward, but still far from trivial, as illustrated in the following example:

(1) *paka*	*alianguka*	*ndani*	*ya*	*maji*
`cat`	`fell`	`inside`	`of`	`water`
noun	**verb**	**adverb**	adjective	**noun**
verb		noun	**preposition**	

To tackle this disambiguation problem, we investigate the applicability of existing data-driven POS taggers. These methods have in common that they require a large amount of annotated data to induce the word class disambiguation task. For the experiments we used the POS tag annotated part of the aforementioned HCS as our training material.

After some general data clean-up and disposal of duplicate sections, we had a corpus of 3,656,821 words (169,702 sentences) available. To obtain a reasonable spread in language usage, we randomized the sentences in the corpus, so that the tagger would not be biased towards a particular type of text during training. Given the expansive size of the corpus, full 10-fold cross validation experiments were not feasible. We therefore randomly divided the corpus into a 80% training set (2,927,846 words), a 10% validation set (362,866 words) on which the optimal parameters of the algorithms could be established, and finally a 10% blind test set (366,109 words) for evaluation on unseen text.

3 Data-Driven Taggers

The last 15 years have witnessed corpus-based methods making tremendous headway in providing accurate and robust POS taggers. Many of these tools have since been made publicly available, so that they can relatively easily be applied to new languages when annotated corpora become available. In this section, we briefly introduce the taggers used for the experiments.

TnT (Trigrams'n'Tags): Hidden Markov Modeling One of the most common approaches to data-driven POS tagging is using Hidden Markov Models (HMMs). A very sophisticated HMM tagger is the Trigrams'n'Tags (TnT) tagger[1] [8]. It improves on previous HMM approaches through the use of well established smoothing methods and its more sophisticated processing of unknown words, capitalized words and sentence boundaries.

[1] TnT is available from `http://www.coli.uni-saarland.de/~thorsten/tnt/`

MXPOST: Maximum Entropy Modeling Maximum entropy modeling has consistently been achieving top performance on a variety of NLP tasks. The maximum entropy tagger, MXPOST[2] [9], is typically able to beat most other POS taggers in a direct comparison [10]. Like most other taggers, it uses lexical information about the word to be tagged, contextual features (preceding, following tags) and morphological features (prefix, suffix letters).

MBT: Memory-Based Learning With its emphasis on symbolic processing and its inherent robustness to exceptions, Memory-Based Learning (MBL) is particularly well suited for NLP classification tasks. The Memory-Based Tagger (MBT)[3] [11] induces two taggers from the training data: one for known words and one for unknown words, the former using contextual clues, while the latter also uses orthographical features.

SVMTool: Support Vector Machines Support Vector Machines (SVMs) have been successfully applied to a wide range of classification tasks [12], but only recently has an SVM-based POS tagging tool become available: SVMTool[4] [13], which functions as a set of pre- and postprocessing scripts for SVM-Light [14]. SVMTool has been shown to outperform TnT on English data [13], but has so far not been extensively compared to other methods and on other datasets.

4 Experiments: Individual Tagger Performance

In this section, we outline the performance of the individual data-driven taggers trained and evaluated on the Kiswahili dataset. The training, validation and test sets outlined in Section 2 were kept constant for all of the experiments, allowing for a systematic and direct comparison between the tagging methods.

In a first phase, algorithmic parameters and information source are optimized on the basis of the validation set. The taggers obtained from this training and optimization phase are subsequently used to tag the held-out test set. The accuracy of the respective taggers is calculated by comparing the output of the taggers to the gold-standard annotation provided by HCS.

The average per-word lexical ambiguity in the Kiswahili dataset is quite favorable, with only an average of 1.3 possible tags per word. This figure indicates that (on the basis of the HCS tagset) there is not a lot of lexical ambiguity in Kiswahili. Roughly 3% of the words (about 12,000 words) in the validation set, as well as the test set, are unknown, meaning that they do not occur in the training set.

The limited lexical ambiguity is further illustrated by the high score achieved by the baseline method: a simple statistical unigram method, which assigns to each of the known words in the test set the tag it has been most often associated with in the training set. For unknown words, it assigns the tag most frequently associated with unknown words in the validation set (PROPNAME). This baseline method already achieves more than 97% accuracy on known words (Table 1), but does not handle unknown words very well with a score of only 18.59%.

[2] MXPOST is available from `ftp://ftp.cis.upenn.edu/pub/adwait/jmx/jmx.tar.gz`

[3] MBT is available from `http://ilk.uvt.nl/software.html`

[4] SVMTool is available from `http://www.lsi.upc.es/~nlp/SVMTool/`

Table 1. Accuracy scores on blind test set (366K words) and approximate CPU times for the individual taggers

Tagger	Accuracy Scores			CPU Time	
	Known Words	Unknown Words	Total	Train	Tag
Baseline	97.01%	18.59%	94.50%	4s	1s
TnT	98.00%	91.66%	97.79%	9s	4s
MBT (default)	98.39%	90.59%	98.14%	3m	20s
MBT (optimal)	98.46%	91.61%	98.25%	6m	8m
SVMTool	98.48%	91.30%	98.24%	±80h	15s
MXPOST	98.61%	93.32%	98.44%	±5h	90s

Table 1 indicates that the **TnT** tagger by far exhibits the most efficient processing times of all data-driven taggers[5]. Despite an exhaustive optimization phase on the validation set (which still revealed the default settings to perform the best), the performance of the TnT tagger trails in direct comparison to the other taggers. It nevertheless establishes a significant increase compared to the baseline tagger, particularly with respect to unknown words.

The default **MBT** uses a context of two disambiguated tags to the left of the word to be tagged and one ambiguous tag to the right. Table 1 shows that the default MBT performs quite well for known words, but is lacking for unknown words. We subsequently performed extensive optimization experiments during which we established the ideal information source and optimal algorithmic parameters. For known and unknown words, this equaled to three tags before and after the word to be tagged. For unknown words, we also took into account five prefix and suffix letters and information on capitalization, hyphenation and numerical characters within the word. While this optimization had a significantly positive effect on the accuracy of the tagger, particularly on the processing of unknown words, it has a detrimental effect on CPU time during classification.

Typical for SVM-based methods, **SVMTool** has a laborious training phase, but very attractive efficiency properties during classification. The training phase of the SVMTool tagger is rather problematic with a processing time of several days, which rendered optimization experiments unfeasible. Table 1 therefore presents the accuracy scores on the test set using the default radial basis kernel. As expected however, tagging time is very favorable and SVMTool's performance is easily able to match that of the optimized MBT. Its lower performance on processing unknown words means it achieves a barely significantly lower score than MBT, but we are confident that further optimization experiments can at least level the field.

In direct comparison with other data-driven taggers, the **MXPOST** tagger further establishes its state-of-the-art status. The default settings of MXPOST were confirmed as performing the best during optimization experiments on the validation set, except for the number of iterations (we used 500 iterations during training instead of the default 200). Table 1 illustrates that MXPOST is able to achieve the highest accuracy, with a particularly impressive accuracy score for unknown words. Compared to the baseline tagger, MXPOST achieves an error reduction rate of 72% (54% on known words, 92% on unknown words).

[5] Approximate CPU time was measured on a dual 64bit AMD Opteron 2.44GHz system with 6GB RAM.

Data analysis showed that most taggers are able to resolve ambiguity well. The MXPOST tagger for instance has an accuracy of more than 94% on ambiguous words. Interestingly however, both TnT and MBT beat MXPOST when it comes to unambiguous words, which means MXPOST makes slightly more mistakes on words that should not be considered ambiguous. MXPOST seems to avoid overfitting the training data, by a more loose definition of lexical ambiguity, while the other 3 taggers tend to choose the single tag associated with the word in the lexicon.

The most common mistake made by all taggers is the tagging of a preposition (PREP) as an agentative particle (AG-PART) and vice versa. This accounts for almost 20% of all tagging errors. Other common mistakes include the tagging of a noun as a verb and the tagging of a proper name as a noun.

5 Experiments: System Combination

While some studies [15] suggest that classifier bias can be minimized given an exhaustive search through algorithmic parameters and information source, in practice most data-driven taggers exhibit quite different tagging behavior given the same data set. In the system combination experiments, we try to exploit these differences by combining the output of the taggers to create a type of tagging committee that agrees on a tag for a word. Data analysis indeed shows that only 97.23% of the time do the taggers all predict the same tag. 96.75% of the time do the taggers agree on a tag which matches the correct tag. Furthermore, only 0.5% of the time, do the taggers all agree on the same erroneous tag.

These figures indicate that there is enough disagreement between the individual taggers to obtain a considerable increase using system combination. This type of system combination is again performed in two processing steps: first we use the four data-driven taggers to tag the validation set and test set. We then create a new dataset with 6 columns: the word, the four tagger predictions and the gold-standard tag. The upper bound performance of any given combination method can be found on the last line of Table 2. If we were to have an oracle which, given the four possible predicted tags, always chooses the correct one, we could obtain a tagging accuracy of 99.44%.

Table 2. Results of system combination experiments

Method	Known Words	Unknown Words	Total
MXPOST	98.61%	93.32%	98.44%
Majority Voting	98.53%	93.12%	98.36%
Weighted Voting	98.59%	93.68%	98.42%
Plural Voting	98.72%	92.72%	98.56%
MXPOST+LLU	98.79%	93.32%	98.61%
Cascaded Classifier	98.63%	93.37%	98.46%
Oracle	99.52%	96.85%	99.44%

The first combination method we consider, simple majority voting, chooses for each word the tag that is most often predicted by the taggers. Ties are resolved randomly. This

combination method improves on all of the individual taggers, except MXPOST. Apparently, many of the correct tags suggested by MXPOST are outvoted by the other taggers. To counter this effect, we implemented two more refined voting methods: weighted voting and plural voting. Interestingly, weighted voting in which the weight of each classifier's vote is equal to its observed accuracy on the validation set, again fails to yield a performance increase.

We also experimented with a more naive voting method, plural voting, in which we attribute MXPOST four votes, MBT and SVMTool three votes and TnT two votes. These values were manually chosen on the basis of their performance on the validation set. Plural voting achieves a higher accuracy on the test set than any of the individual taggers. To our knowledge, plural voting has not yet been attempted as a system combination technique. It is therefore interesting to observe that this very naive combination method outperforms the more sophisticated weighted voting method.

We previously observed that MXPOST makes more mistakes on unambiguous words than the other taggers, but is better at handling ambiguous words. Since ambiguity information is available before tagging, we are able to propose a combined system, where MXPOST tags ambiguous and unknown words and a simple lexicon lookup approach handles unambiguous words. This almost trivial combination method yields a substantial performance increase with an overall tagging accuracy of 98.61% (MXPOST+LLU in Table 2).

A last combination method takes the output of the taggers and transforms them into instances that can be used as training material for a machine learning algorithm, with the gold-standard tag as the class to be predicted. The tagged validation set was used to create a training set for a memory-based classifier which classified the instance base generated from the test set. The output tags were then considered as the final tag proposed by the tagger committee. Table 2 shows that the cascaded classifier is indeed able to improve on any of the individual taggers with an overall accuracy of 98.46%. Interestingly however, this combination method underperforms compared to the more naive combination methods.

The best system combination method (MXPOST+LLU) achieves an error reduction rate of more than 11% compared to the best individual tagger. While this increase in accuracy is not as dramatic compared to those observed for other languages and datasets [10], it nevertheless establishes further proof that system combination is able to overcome the individual taggers' bias to a significant extent. Moreover, given the upper-bound accuracy obtained by the oracle, there is still ample room for improvement for other system combination methods, especially for the disambiguation of unknown words.

6 Future Work and Conclusion

In this paper we presented experiments with data-driven part-of-speech taggers trained and evaluated on the annotated Helsinki Corpus of Swahili. We selected four of the current state-of-the-art data-driven taggers, TnT, MBT, SVMTool and MXPOST, and observed the latter as being the most accurate tagger for this dataset. In another set of experiments, we further improved on the performance of the individual taggers by combining them into a committee of taggers. Surprisingly, we observed the more naive combination methods, like the novel plural voting approach, outperform more elaborate schemes like cascaded classifiers and weighted voting.

This paper presents the first direct comparison of data-driven taggers on this particular data set. We are confident that significant increases in tagging accuracy can still be

obtained through various stages of algorithmic optimization and more refined system combination methods. The results of SVMTool in particular can undoubtedly be improved through the selection of a more appropriate kernel and a thorough validation phase. Furthermore, the inclusion of other data-driven tagging methods such as CRF++, WPDV [10] or Transformation-Based Tagging [16] might also improve the performance of the system combination methods.

Future work will include learning curve experiments to determine how much data is minimally needed to obtain optimal performance. Thorough data analysis is further needed to investigate the way the taggers handle morphological issues in Kiswahili. Affixation is an important indicator of word class in Kiswahili and all of the data-driven taggers used in the experiments only cover this aspect indirectly on the level of the grapheme. Perhaps a more rigid morphologically inspired approach to part-of-speech tagging, where morphological analysis functions as a preprocessing step, might provide a significant performance increase. Despite the limitations of the taggers presented in this paper, we nevertheless hope that the results presented herein can function as a first benchmark for future research on data-driven part-of-speech tagging of Kiswahili, and Bantu languages in general.

References

1. van Rooy, B., Pretorius, R.: A word-class tagset for Setswana. Southern African Linguistics and Applied Language Studies **21(4)** (2003) 203–222.
2. Allwood, J., Grönqvist, L., Hendrikse, A.P.: Developing a tagset and tagger for the African languages of South Africa with special reference to Xhosa. Southern African Linguistics and Applied Language Studies **21(4)** (2003) 223–237.
3. Prinsloo, D.J., Heid, U.: Creating word class tagged corpora for Northern Sotho by linguistically informed bootstrapping. In: Proceedings of the Conference on Lesser Used Languages & Computer Linguistics (LULCL 2005), Bozen/Bolzano, Italy (2005 (to be published)).
4. Taljard, E., Bosch, S.E.: A comparison of approaches towards word class tagging: disjunctively vs conjunctively written Bantu languages. In: Proceedings of the Conference on Lesser Used Languages & Computer Linguistics (LULCL 2005), Bozen/Bolzano, Italy (2005 (to be published)).
5. Pretorius, L., Bosch, S.E.: Computational aids for Zulu natural language processing. Southern African Linguistics and Applied Language Studies **21(4)** (2003) 267–282.
6. Hurskainen, A.: HCS 2004 – Helsinki Corpus of Swahili. Compilers: Institute for Asian and African Studies (University of Helsinki) and CSC (2004).
7. Hurskainen, A.: Disambiguation of morphological analysis in Bantu languages. In: Proceedings of the Sixteenth International Conference on Computational Linguistics (COLING-96), Copenhagen, Denmark (1996) 568–573.
8. Brants, T.: TnT – a statistical part-of-speech tagger. In: Proceedings of the Sixth Conference on Applied Natural Language Processing (ANLP 2000), Seattle, WA, USA (2000) 224–231.
9. Ratnaparkhi, A.: A maximum entropy model for part-of-speech tagging. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, Somerset, NJ, USA (1996) 133–142.
10. van Halteren, H., Zavrel, J., Daelemans, W.: Improving accuracy in word class tagging through combination of machine learning systems. Computational Linguistics **27(2)** (2001) 199–230.
11. Daelemans, W., Zavrel, J., van den Bosch, A., van der Sloot, K.: MBT: Memory Based Tagger, version 2.0, Reference Guide. ILK Research Group Technical Report Series 03-13, Tilburg (2003).
12. Wagacha, P., Manderick, B., Getao, K.: Benchmarking Support Vector Machines using StatLog Methodology. In: Proceedings of Benelearn 2004, Machine Learning Conference of Belgium and the Netherlands, Brussels, Belgium (2004) 185–190.

13. Giménez, J., Màrquez, L.: SVMTool: A general POS tagger generator based on Support Vector Machines. In: Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC 2004), Lisbon, Portugal (2004) 43–46.
14. Joachims, T.: Making Large-scale SVM Learning Practical. In Schölkopf, B., Burges, C., Smola, A., eds.: Advances in Kernel Methods – Support Vector Learning. MIT Press, Boston, MA, USA (1999) 41–56.
15. De Pauw, G., Daelemans, W.: The role of algorithm bias vs information source in learning algorithms for morphosyntactic disambiguation. In: Proceedings of the Fourth Conference on Computational Natural Language Learning (CoNLL 2000), Lisbon, Portugal (2000) 19–24.
16. Brill, E.: A simple rule-based part-of-speech tagger. In: Proceedings of the Third Conference on Applied Natural Language Processing (ANLP '92), Trento, Italy (1992) 152–155.