Methods for Amharic Part-of-Speech Tagging

Björn Gambäck[†‡] **Fredrik Olsson**[†] **Atelach Alemu Argaw**[⋆] **Lars Asker**[⋆]
[†]Userware Laboratory
Swedish Institute of Computer Science
Kista, Sweden
{gamback,fredriko}@sics.se
[‡]Dpt. of Computer & Information Science
Norwegian University of Science & Technology
Trondheim, Norway
gamback@idi.ntnu.no
[⋆]Dpt. of Computer & System Sciences
Stockholm University
Kista, Sweden
{atelach,asker}@dsv.su.se

Abstract

The paper describes a set of experiments involving the application of three state-of-the-art part-of-speech taggers to Ethiopian Amharic, using three different tagsets. The taggers showed worse performance than previously reported results for English, in particular having problems with unknown words. The best results were obtained using a Maximum Entropy approach, while HMM-based and SVM-based taggers got comparable results.

1 Introduction

Many languages, especially on the African continent, are under-resourced in that they have very few computational linguistic tools or corpora (such as lexica, taggers, parsers or tree-banks) available. Here, we will concentrate on the task of developing part-of-speech taggers for Amharic, the official working language of the government of the Federal Democratic Republic of Ethiopia: Ethiopia is divided into nine regions, each with its own nationality language; however, Amharic is the language for country-wide communication.

Amharic is spoken by about 30 million people as a first or second language, making it the second most spoken Semitic language in the world (after Arabic), probably the second largest language in Ethiopia (after Oromo), and possibly one of the five largest languages on the African continent. The actual size of the Amharic speaking population must be based on estimates: Hudson (1999) analysed the Ethiopian census from 1994 and indicated that more than 40% of the population then understood Amharic, while the current size of the Ethiopian population is about 80 million.[1]

[1]82.5 million according to CIA (2009); 76.9 according to Ethiopian parliament projections in December 2008 based on the preliminary reports from the census of May 2007.

In spite of the relatively large number of speakers, Amharic is still a language for which very few computational linguistic resources have been developed, and previous efforts to create language processing tools for Amharic—e.g., Alemayehu and Willett (2002) and Fissaha (2005)—have been severely hampered by the lack of large-scale linguistic resources for the language. In contrast, the work detailed in the present paper has been able to utilize the first publicly available medium-sized tagged Amharic corpus, described in Section 5.

However, first the Amharic language as such is introduced (in Section 2), and then the task of part-of-speech tagging and some previous work in the field is described (Section 3). Section 4 details the tagging strategies used in the experiments, the results of which can be found in Section 6 together with a short discussion. Finally, Section 7 sums up the paper and points to ways in which we believe that the results can be improved in the future.

2 Amharic

Written Amharic (and Tigrinya) uses a unique script originating from the Ge'ez alphabet (the liturgical language of the Ethiopian Orthodox Church). Written Ge'ez can be traced back to at least the 4th century A.D., with the first versions including consonants only, while the characters in later versions represent consonant-vowel (CV) pairs. In modern Ethiopic script each syllograph (syllable pattern) comes in seven different forms (called orders), reflecting the seven vowel sounds. The first order is the basic form; the others are derived from it by modifications indicating vowels. There are 33 basic forms, giving 7*33 syllographs, or *fidels* ('*fidel*', lit. 'alphabet' in Amharic, refers both to the characters and the entire script). Unlike Arabic and Hebrew, Amharic is written from left to right. There is no agreed upon spelling standard for compound words and the writing system uses several ways to denote compounds

Proceedings of the EACL 2009 Workshop on Language Technologies for African Languages – AfLaT 2009, pages 104–111,
Athens, Greece, 31 March 2009.

	form	pattern
root	*sbr*	CCC
perfect	*säbbär*	CVCCVC
imperfect	*säbr*	CVCC
gerund	*säbr*	CVCC
imperative	*sbär*	CCVC
causative	*assäbbär*	as-CVCCVC
passive	*täsäbbär*	täs-CVCCVC

Table 1: Some forms of the verb *sbr* ('break')

2.1 Amharic morphology

A significantly large part of the vocabulary consists of verbs, and like many other Semitic languages, Amharic has a rich verbal morphology based on triconsonantal roots with vowel variants describing modifications to, or supplementary detail and variants of the root form. For example, the root *sbr*, meaning 'to break' can have (among others!) the forms shown in Table 1. Subject, gender, number, etc., are also indicated as bound morphemes on the verb, as well as objects and possession markers, mood and tense, beneficative, malfactive, transitive, dative, negative, etc.

Amharic nouns (and adjectives) can be inflected for gender, number, definiteness, and case, although gender is usually neutral. The definite article attaches to the end of a noun, as do conjunctions, while prepositions are mostly prefixed.

2.2 Processing Amharic morphology

The first effort on Amharic morphological processing was a rule-based system for verbs (and nouns derived from verbs) which used root patterns and affixes to determine lexical and inflectional categories (Bayou, 2000), while Bayu (2002) used an unsupervised learning approach based on probabilistic models to extract stems, prefixes, and suffixes for building a morphological dictionary. The system was able to successfully analyse 87% of a small testdata set of 500 words.

The first larger-scale morphological analyser for Amharic verbs used XFST, the Xerox Finite State Tools (Fissaha and Haller, 2003). This was later extended to include all word categories (Amsalu and Gibbon, 2005). Testing with 1620 words text from an Amharic bible, 88–94% recall and 54–94% precision (depending on the word-class) were reported. The lowest precision (54%) was obtained for verbs; Amsalu and Demeke (2006) thus describe ways to extend the finite-state system to handle 6400 simple verbal stems generated from 1300 root forms.

Alemayehu and Willett (2002) report on a stemmer for Information Retrieval for Amharic, and testing on a 1221 random word sample stated "Manual assessment of the resulting stems showed that 95.5 percent of them were linguistically meaningful," but gave no evaluation of the correctness of the segmentations. Argaw and Asker (2007) created a rule-based stemmer for a similar task, and using 65 rules and machine readable dictionaries obtained 60.0% accuracy on fictional text (testing on 300 unique words) and 76.9% on news articles (on 1503 words, of which 1000 unique).[2]

3 Part-of-Speech Tagging

Part-of-speech (POS) tagging is normally treated as a classification task with the goal to assign lexical categories (word classes) to the words in a text. Most work on tagging has concentrated on English and on using supervised methods, in the sense that the taggers have been trained on an available, tagged corpus. Both rule-based and statistical / machine-learning based approaches have been thoroughly investigated. The Brill Tagger (Brill, 1995) was fundamental in using a combined rule- and learning-based strategy to achieve 96.6% accuracy on tagging the Penn Treebank version of the Wall Street Journal corpus. That is, to a level which is just about what humans normally achieve when hand-tagging a corpus, in terms of interannotator agreement—even though Voutilainen (1999) has shown that humans can get close to the 100% agreement mark if the annotators are allowed to discuss the problematic cases.

Later taggers have managed to improve Brill's figures a little bit, to just above 97% on the Wall Street Journal corpus using Hidden Markov Models, HMM and Conditional Random Fields, CRF; e.g., Collins (2002) and Toutanova et al. (2003). However, most recent work has concentrated on applying tagging strategies to other languages than English, on combining taggers, and/or on using unsupervised methods. In this section we will look at these issues in more detail, in particular with the relation to languages similar to Amharic.

3.1 Tagging Semitic languages

Diab et al. (2004) used a Support Vector Machine, SVM-based tagger, trained on the Arabic Penn

[2]Other knowledge sources for processing Amharic include, e.g., Gasser's verb stem finder (available from `nlp.amharic.org`) and wordlists as those collected by Gebremichael (`www.cs.ru.nl/~biniam/geez`).

Treebank 1 to tokenize, POS tag, and annotate Arabic base phrases. With an accuracy of 95.5% over a set of 24 tags, the data-driven tagger performed on par with state-of-the-art results for English when trained on similar-sized data (168k tokens). Bar-Haim et al. (2008) developed a lexicon-based HMM tagger for Hebrew. They report 89.6% accuracy using 21 tags and training on 36k tokens of news text. Mansour (2008) ported this tagger into Arabic by replacing the morphological analyzer, achieving an accuracy of 96.3% over 26 tags on a 89k token corpus. His approach modifies the analyses of sentences receiving a low probability by adding synthetically constructed analyses proposed by a model using character information.

A first prototype POS tagger for Amharic used a stochastic HMM to model contextual dependencies (Getachew, 2001), but was trained and tested on only one page of text. Getachew suggested a tagset for Amharic consisting of 25 tags. More recently, CRFs have been applied to segment and tag Amharic words (Fissaha, 2005), giving an accuracy of 84% for word segmentation, using character, morphological and lexical features. The best result for POS-tagging was 74.8%, when adding a dictionary and bigrams to lexical and morphological features, and 70.0% without dictionary and bigrams. The data used in the experiments was also quite small and consisted of 5 annotated news articles (1000 words). The tagset was a reduced version (10 tags) of the one used by Getachew (2001), and will be further discussed in Section 5.2.

3.2 Unsupervised tagging

The desire to use unsupervised machine learning approaches to tagging essentially originates from the wish to exploit the vast amounts of unlabelled data available when constructing taggers. The area is particularly vivid when it comes to the treatment of languages for which there exist few, if any, computational resources, and for the case of adapting an existing tagger to a new language domain.

Banko and Moore (2004) compared unsupervised HMM and transformation-based taggers trained on the same portions of the Penn Treebank, and showed that the quality of the lexicon used for training had a high impact on the tagging results. Duh and Kirchhoff (2005) presented a minimally-supervised approach to tagging for dialectal Arabic (Colloquial Egyptian), based on a morphological analyzer for Modern Standard Arabic and unlabeled texts in a number of dialects. Using a trigram HMM tagger, they first produced a baseline system and then gradually improved on that in an unsupervised manner by adding features so as to facilitate the analysis of unknown words, and by constraining and refining the lexicon.

Unsupervised learning is often casted as the problem of finding (hidden) structure in unlabeled data. Goldwater and Griffiths (2007) noted that most recent approaches to this problem aim to identify the set of attributes that maximizes some target function (Maximum Likelihood Estimation), and then to select the values of these attributes based on the representation of the model. They proposed a different approach, based on Bayesian principles, which tries to directly maximize the probability of the attributes based on observation in the data. This Bayesian approach outperformed Maximum Likelihood Estimation when training a trigram HMM tagger for English. Toutanova and Johnson (2007) report state-of-the-art results by extending the work on Bayesian modelling for unsupervised learning of taggers both in the way that prior knowledge can be incorporated into the model, and in the way that possible tags for a given word is explicitly modeled.

3.3 Combining taggers

A possible way to improve on POS tagging results is to combine the output of several different taggers into a committee, forming joint decisions regarding the labeling of the input. Roughly, there are three obvious ways of combining multiple predicted tags for a word: random decision, voting, and stacking (Dietterich, 1997), with the first way suited only for forming a baseline. *Voting* can be divided into two subclasses: unweighted votes, and weighted votes. The weights of the votes, if any, are usually calculated based on the classifiers' performance on some initial dataset. *Stacking*, finally, is a way of combining the decisions made by individual taggers in which the predicted tags for a given word are used as input to a subsequent tagger which outputs a final label for the word.

Committee-based approaches to POS tagging have been in focus the last decade: Brill and Wu (1998) combined four different taggers for English using unweighted voting and by exploring contextual cues (essentially a variant of stacking). Aires et al. (2000) experimented with 12 different ways of combining the output from taggers for Brazilian

Portuguese, and concluded that some, but not all, combinations yielded better accuracy than the best individual tagger. Shacham and Wintner (2007) contrasted what they refer to as being a naïve way of combining taggers with a more elaborate, hierarchical one for Hebrew. In the end, the elaborated method yielded results inferior to the naïve approach. De Pauw et al. (2006) came to similar conclusions when using five different ways of combining four data-driven taggers for Swahili. The taggers were based on HMM, Memory-based learning, SVM, and Maximum Entropy, with the latter proving most accurate. Only in three of five cases did a combination of classifiers perform better than the Maximum Entropy-based tagger, and simpler combination methods mostly outperformed more elaborate ones.

Spoustová et al. (2007) report on work on combining a hand-written rule-based tagger with three statistically induced taggers for Czech. As an effect of Czech being highly inflectional, the tagsets are large: 1000–2000 unique tags. Thus the approach to combining taggers first aims at reducing the number of plausible tags for a word by using the rule-based tagger to discard impossible tags. Precision is then increased by invoking one or all of the data-driven taggers. Three different ways of combining the taggers were explored: serial combination, involving one of the statistical taggers; so called SUBPOS pre-processing, involving two instances of statistical taggers (possibly the same tagger); and, parallel combination, in which an arbitrary number of statistical taggers is used. The combined tagger yielded the best results for Czech POS tagging reported to date, and as a side-effect also the best accuracy for English: 97.43%.[3]

4 The Taggers

This section describes the three taggers used in the experiments (which are reported on in Section 6).

4.1 Hidden Markov Models: TnT

TnT, "Trigrams'n'Tags" (Brants, 2000) is a very fast and easy-to-use HMM-based tagger which painlessly can be trained on different languages and tagsets, given a tagged corpus.[4] A Markov-based tagger aims to find a tag sequence which maximizes $P(word_n|tag_n) * P(tag_n|tag_{1...n-1})$, where the first factor is the emit (or lexical) probability, the likelihood of a word given certain tag, and the second factor is the state transition (or contextual) probability, the likelihood of a tag given a sequence of preceding tags. TnT uses the Viterbi algorithm for finding the optimal tag sequence. Smoothing is implemented by linear interpolation, the respective weights are determined by deleted interpolation. Unknown words are handled by a suffix trie and successive abstraction.

Applying TnT to the Wall Street Journal corpus, Brants (2000) reports 96.7% overall accuracy, with 97.0% on known and 85.5% on unknown words (with 2.9% of the words being unknown).

4.2 Support Vector Machines: SVMTool

Support Vector Machines (SVM) is a linear learning system which builds two class classifiers. It is a supervised learning method whereby the input data are represented as vectors in a high-dimensional space and SVM finds a hyperplane (a decision boundary) separating the input space into two by maximizing the margin between positive and negative data points.

SVMTool is an open source tagger based on SVMs.[5] Comparing the accuracy of SVMTool with TnT on the Wall Street Journal corpus, Giménez and Màrquez (2004) report a better performance by SVMTool: 96.9%, with 97.2% on known words and 83.5% on unknown.

4.3 Maximum Entropy: MALLET

Maximum Entropy is a linear classification method. In its basic incarnation, linear classification combines, by addition, the pre-determined weights used for representing the importance of each feature to a given class. Training a Maximum Entropy classifier involves fitting the weights of each feature value for a particular class to the available training data. A good fit of the weights to the data is obtained by selecting weights to maximize the log-likelihood of the learned classification model. Using an Maximum Entropy approach to POS tagging, Ratnaparkhi (1996) reports a tagging accuracy of 96.6% on the Wall Street Journal.

The software of choice for the experiments reported here is MALLET (McCallum, 2002), a freely available Java implementation of a range of machine learning methods, such as Naïve Bayes, decision trees, CRF, and Maximum Entropy.[6]

[3] As reported on `ufal.mff.cuni.cz/compost/en`

[4] `www.coli.uni-saarland.de/~thorsten/tnt`

[5] `www.lsi.upc.edu/~nlp/SVMTool`

[6] `mallet.cs.umass.edu`

5 The Dataset

The experiments of this paper utilize the first medium-sized corpus for Amharic (available at `http://nlp.amharic.org`). The corpus consists of all 1065 news texts (210,000 words) from the Ethiopian year 1994 (parts of the Gregorian years 2001–2002) from the Walta Information Center, a private news service based in Addis Ababa. It has been morphologically analysed and manually part-of-speech tagged by staff at ELRC, the Ethiopian Languages Research Center at Addis Ababa University (Demeke and Getachew, 2006).

The corpus is available both in *fidel* and transcribed into a romanized version known as SERA, System for Ethiopic Representation in ASCII (Yacob, 1997). We worked with the transliterated form (202,671 words), to be compatible with the machine learning tools used in the experiments.

5.1 "Cleaning" the corpus

Unfortunately, the corpus available on the net contains quite a few errors and tagging inconsistencies: nine persons participated in the manual tagging, writing the tags with pen on hard copies, which were given to typists for insertion into the electronic version of the corpus—a procedure obviously introducing several possible error sources.

Before running the experiments the corpus had to be "cleaned": many non-tagged items have been tagged (the human taggers have, e.g., often tagged the headlines of the news texts as one item, end-of-sentence punctuation), while some double tags have been removed. Reflecting the segmentation of the original Amharic text, all whitespaces were removed, merging multiword units with a single tag into one-word units. Items like '`"`' and '`/`' have been treated consistently as punctuation, and consistent tagging has been added to word-initial and word-final hyphens. Also, some direct tagging errors and misspellings have been corrected.

Time expressions and numbers have not been consistently tagged at all, but those had to be left as they were. Finally, many words have been transcribed into SERA in several versions, with only the cases differing. However, this is also difficult to account for (and in the experiments below we used the case sensitive version of SERA), since the SERA notation in general lets upper and lower cases of the English alphabet represent different symbols in *fidel* (the Amharic script).

5.2 Tagsets

For the experiments, three different tagsets were used. Firstly, the full, original 30-tag set developed at the Ethiopian Languages Research Center and described by Demeke and Getachew (2006). This version of the corpus will be referred to as 'ELRC'. It contains $200,863$ words and differs from the published corpus in way of the corrections described in the previous section.

Secondly, the corpus was mapped to 11 basic tags. This set consists of ten word classes: Noun, Pronoun, Verb, Adjective, Preposition, Conjunction, Adverb, Numeral, Interjection, and Punctuation, plus one tag for problematic words (unclear: `<UNC>`). The main differences between the two tagsets pertain to the treatment of prepositions and conjunctions: in 'ELRC' there are specific classes for, e.g., pronouns attached with preposition, conjunction, and both preposition and conjunction (similar classes occur for nouns, verbs, adjectives, and numerals). In addition, numerals are divided into cardinals and ordinals, verbal nouns are separated from other nouns, while auxiliaries and relative verbs are distinguished from other verbs. The full tagset is made up of thirty subclasses of the basic classes, based on type of word only: the tags contain no information on grammatical categories (such as number, gender, tense, and aspect).

Thirdly, for comparison reasons, the full tagset was mapped to the 10 tags used by Fissaha (2005). These classes include one for Residual (`R`) which was assumed to be equivalent to `<UNC>`. In addition, `<CONJ>` and `<PREP>` were mapped to Adposition (`AP`), and both `<N>` and `<PRON>` to `N`. The other mappings were straight-forward, except that the 'BASIC' tagset groups all verbs together, while Fissaha kept Auxiliary (`AUX`) as its own class. This tagset will be referred to as 'SISAY'.

5.3 Folds

For evaluation of the taggers, the corpus was split into 10 folds. These folds were created by chopping the corpus into 100 pieces, each of about 2000 words in sequence, while making sure that each piece contained full sentences (rather than cutting off the text in the middle of a sentence), and then merging sets of 10 pieces into a fold. Thus the folds represent even splits over the corpus, to avoid tagging inconsistencies, but the sequences are still large enough to potentially make knowledge sources such as n-grams useful.

Fold	TOTAL	KNOWN	UNKNOWN
fold00	20,027	17,720	2,307
fold01	20,123	17,750	2,373
fold02	20,054	17,645	2,409
fold03	20,169	17,805	2,364
fold04	20,051	17,524	2,527
fold05	20,058	17,882	2,176
fold06	20,111	17,707	2,404
fold07	20,112	17,746	2,366
fold08	20,015	17,765	2,250
fold09	20,143	17,727	2,416
Average	20,086	17,727	2,359
Percent	—	88.26	11.74

Table 2: Statistics for the 10 folds

Table 2 shows the data for each of the folds, in terms of total number of tokens, as well as split into known and unknown tokens, where the term UNKNOWN refers to tokens that are not in any of *the other* nine folds. The figures at the bottom of the table show the average numbers of known and unknown words, over all folds. Notably, the average number of unknown words is about four times higher than in the Wall Street Journal corpus (which, however, is about six times larger).

6 Results

The results obtained by applying the three different tagging strategies to the three tagsets are shown in Table 3, in terms of average accuracies after 10-fold cross validation, over all the tokens (with standard deviation),[7] as well as accuracy divided between the known and unknown words. Additionally, SVMTool and MALLET include support for automatically running 10-fold cross validation on their own folds. Figures for those runs are also given. The last line of the table shows the baselines for the tagsets, given as the number of tokens tagged as regular nouns divided by the total number of words after correction.

6.1 TnT

As the bold face figures indicate, TnT achieves the best scores of all three taggers, on all three tagsets, on *known* words. However, it has problems with the unknown words—and since these are so frequent in the corpus, TnT overall performs worse than the other taggers. The problems with the unknown words increase as the number of possible tags increase, and thus TnT does badly on the original tagging scheme ('ELRC'), where it only gets

[7]The standard deviation is given by $\sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \overline{x})^2}$ where $\overline{x}$ is the arithmetic mean ($\frac{1}{n}\sum_{i=1}^{n} x_i$).

	ELRC	BASIC	SISAY
TnT	85.56	92.55	92.60
STD DEV	0.42	0.31	0.32
KNOWN	**90.00**	**93.95**	**93.99**
UNKNOWN	52.13	82.06	82.20
SVM	**88.30**	**92.77**	**92.80**
STD DEV	0.41	0.31	0.37
KNOWN	89.58	93.37	93.34
UNKNOWN	**78.68**	**88.23**	**88.74**
Own folds	88.69	92.97	92.99
STD DEV	0.33	0.17	0.26
MaxEnt	87.87	92.56	92.60
STD DEV	0.49	0.38	0.43
KNOWN	89.44	93.26	93.27
UNKNOWN	76.05	87.29	87.61
Own folds	**90.83**	**94.64**	**94.52**
STD DEV	1.37	1.11	0.69
BASELINE	35.50	58.26	59.61

Table 3: Tagging results

a bit over 50% on the unknown words (and 85.6% overall). For the two reduced tagsets TnT does better: overall performance goes up to a bit over 92%, with 82% on unknown words.

Table 3 shows the results on the default configuration of TnT, i.e., using 3-grams and interpolated smoothing. Changing these settings give no substantial improvement overall: what is gained at one end (e.g., on unknown words or a particular tagset) is lost at the other end (on known words or other tagsets). However, per default TnT uses a suffix trie of length 10 to handle unknown words. Extending the suffix to 20 (the maximum value in TnT) gave a slight performance increase on 'ELCR' (0.13% on unknown words, 0.01% overall), while having no effect on the smaller tagsets.

6.2 SVM

The SVM-tagger outperforms TnT on unknown words, but is a bit worse on known words. Overall, SVM is slightly better than TnT on the two smaller tagsets and clearly better on the large tagset, and somewhat better than MaxEnt on all three tagsets.

These results are based on SVMTool's default parameters: a one-pass, left-to-right, greedy tagging scheme with a window size of 5. Previous experiments with parameter tuning and multiple pass tagging have indicated that there is room for performance improvements by $\approx 2\%$.

6.3 Maximum Entropy

The MaxEnt tagger gets results comparable to the other taggers on the predefined folds. Its overall

Word$_n$; Tag of *Word*$_n$
Prefixes of *Word*$_n$, length 1-5 characters
Postfixes of *Word*$_n$, length 1-5 characters
Is *Word*$_n$ capitalized?
Is *Word*$_n$ all digits?
Does *Word*$_n$ contain digits?
Does *Word*$_n$ contain a hyphen?
Word$_{n-1}$; Tag of *Word*$_{n-1}$
Word$_{n-2}$; Tag of *Word*$_{n-2}$
Word$_{n+1}$
Word$_{n+2}$

Table 4: Features used in the MaxEnt tagger

performance is equivalent to TnT's on the smaller tagsets, but significantly better on 'ELRC'.

As can be seen in Table 3, the MaxEnt tagger clearly outperforms the other taggers on all tagsets, when MALLET is allowed to create its own folds: all tagsets achieved classification accuracies higher than 90%, with the two smaller tagsets over 94.5%. The dramatic increase in the tagger's performance on these folds is surprising, but a clear indication of one of the problems with n-fold cross validation: even though the results represent averages after n runs, the choice of the original folds to suit a particular tagging strategy is of utmost importance for the final result.

Table 4 shows the 22 features used to represent an instance (*Word*$_n$) in the Maximum Entropy tagger. The features are calculated per token within sentences: the starting token of a sentence is not affected by the characteristics of the tokens ending the previous sentence, nor the other way around. Thus not all features are calculated for all tokens.

6.4 Discussion

In terms of accuracy, the MaxEnt tagger is by far the best of the three taggers, and on all three tagsets, when allowed to select its own folds. Still, as Table 3 shows, the variation of the results for each individual fold was then substantially larger.

It should also be noted that TnT is by far the fastest of the three taggers, in all respects: in terms of time to set up and learn to use the tagger, in terms of tagging speed, and in particular in terms of training time. Training TnT is a matter of seconds, but a matter of hours for MALLET/MaxEnt and SVMTool. On the practical side, it is worth adding that TnT is robust, well-documented, and easy to use, while MALLET and SVMTool are substantially more demanding in terms of user effort and also appear to be more sensitive to the quality and format of the input data.

7 Conclusions and Future Work

The paper has described experiments with applying three state-of-the-art part-of-speech taggers to Amharic, using three different tagsets. All taggers showed worse performance than previously reported results for English. The best accuracy was obtained using a Maximum Entropy approach when allowed to create its own folds: 90.1% on a 30 tag tagset, and 94.6 resp. 94.5% on two reduced sets (11 resp. 10 tags), outperforming an HMM-based (TnT) and an SVM-based (SVMTool) tagger. On predefined folds all taggers got comparable results (92.5-92.8% on the reduced sets and 4-7% lower on the full tagset). The SVM-tagger performs slightly better than the others overall, since it has the best performance on unknown words, which are four times as frequent in the 200K words Amharic corpus used than in the (six times larger) English Wall Street Journal corpus. TnT gave the best results for known words, but had the worst performance on unknown words.

In order to improve tagging accuracy, we will investigate including explicit morphological processing to treat unknown words, and combining taggers. Judging from previous efforts on combining taggers (Section 3.3), it is far from certain that the combination of taggers actually ends up producing better results than the best individual tagger. A pre-requisite for successful combination is that the taggers are sufficiently dissimilar; they must draw on different characteristics of the training data and make different types of mistakes.

The taggers described in this paper use no other knowledge source than a tagged training corpus. In addition to incorporating (partial) morphological processing, performance could be increased by including knowledge sources such as machine readable dictionaries or lists of Amharic stem forms (Section 2.2). Conversely, semi-supervised or unsupervised learning for tagging clearly are interesting alternatives to manually annotate and construct corpora for training taggers. Since there are few computational resources available for Amharic, approaches as those briefly outlined in Section 3.2 deserve to be explored.

Acknowledgements

The work was partially funded by Sida, the Swedish International Development Cooperation Agency through SPIDER (the Swedish Programme for ICT in Developing Regions).

Thanks to Dr. Girma Demeke, Mesfin Getachew, and the ELRC staff for their efforts on tagging the corpus, and to Thorsten Brants for providing us with the TnT tagger.

References

Rachel V. Xavier Aires, Sandra M. Aluísio, Denise C. S. Kuhn, Marcio L. B. Andreeta, and Osvaldo N. Oliveira Jr. 2000. Combining classifiers to improve part of speech tagging: A case study for Brazilian Portuguese. In *15th Brazilian Symposium on AI*, pp. 227–236, Atibaia, Brazil.

Nega Alemayehu and Peter Willett. 2002. Stemming of Amharic words for information retrieval. *Literary and Linguistic Computing*, 17:1–17.

Saba Amsalu and Dafydd Gibbon. 2005. Finite state morphology of Amharic. In *5th Recent Advances in Natural Language Processing*, pp. 47–51, Borovets, Bulgaria.

Saba Amsalu and Girma A. Demeke. 2006. Non-concatinative finite-state morphotactics of Amharic simple verbs. *ELRC Working Papers*, 2:304-325.

Atelach Alemu Argaw and Lars Asker. 2007. An Amharic stemmer: Reducing words to their citation forms. *Computational Approaches to Semitic Languages*, pp. 104–110, Prague, Czech Rep.

Michele Banko and Robert C. Moore. 2004. Part of speech tagging in context. In *20th Int. Conf. on Computational Linguistics*, pp. 556–561, Geneva, Switzerland.

Roy Bar-Haim, Khalil Simaan, and Yoad Winter. 2008. Part-of-speech tagging of modern Hebrew text. *Natural Language Engineering*, 14:223–251.

Abiyot Bayou. 2000. Design and development of word parser for Amharic language. MSc Thesis, Addis Ababa University, Ethiopia.

Tesfaye Bayu. 2002. Automatic morphological analyser: An experiment using unsupervised and autosegmental approach. MSc Thesis, Addis Ababa University, Ethiopia.

Thorsten Brants. 2000. TnT — a statistical part-of-speech tagger. In *6th Conf. Applied Natural Language Processing*, pp. 224–231, Seattle, Wash.

Eric Brill and Jun Wu. 1998. Classifier combination for improved lexical disambiguation. In *17th Int. Conf. on Computational Linguistics*, pp. 191–195, Montreal, Canada.

Eric Brill. 1995. Transformation-based error-driven learning and Natural Language Processing: A case study in part of speech tagging. *Computational Linguistics*, 21:543–565.

CIA. 2009. *The World Factbook — Ethiopia*. The Central Intelligence Agency, Washington, DC. [Updated 22/01/09.]

Michael Collins. 2002. Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In *Empirical Methods in Natural Language Processing*, pp. 1–8, Philadelphia, Penn.

Girma A. Demeke and Mesfin Getachew. 2006. Manual annotation of Amharic news items with part-of-speech tags and its challenges. *ELRC Working Papers*, 2:1–17.

Mona Diab, Kadri Hacioglu, and Daniel Jurafsky. 2004. Automatic tagging of Arabic text: From raw text to base phrase chunks. In *HLT Conf. North American ACL*, pp. 149–152, Boston, Mass.

Thomas G. Dietterich. 1997. Machine-learning research: Four current directions. *AI magazine*, 18:97–136.

Kevin Duh and Katrin Kirchhoff. 2005. POS tagging of dialectal Arabic: A minimally supervised approach. *Computational Approaches to Semitic Languages*, pp. 55–62, Ann Arbor, Mich.

Sisay Fissaha and Johann Haller. 2003. Amharic verb lexicon in the context of machine translation. In *10th Traitement Automatique des Langues Naturelles*, vol. 2, pp. 183–192, Batz-sur-Mer, France.

Sisay Fissaha. 2005. Part of speech tagging for Amharic using conditional random fields. *Computational Approaches to Semitic Languages*, pp. 47–54, Ann Arbor, Mich.

Mesfin Getachew. 2001. Automatic part of speech tagging for Amharic: An experiment using stochastic hidden Markov model (HMM) approach. MSc Thesis, Addis Ababa University, Ethiopia.

Jesús Giménez and Lluís Màrquez. 2004. SVMTool: A general POS tagger generator based on support vector machines. In *4th Int. Conf. Language Resources and Evaluation*, pp. 168–176, Lisbon, Portugal.

Sharon Goldwater and Thomas L. Griffiths. 2007. A fully Bayesian approach to unsupervised part-of-speech tagging. In *45th ACL*, pp. 744–751, Prague, Czech Rep.

Grover Hudson. 1999. Linguistic analysis of the 1994 Ethiopian census. *Northeast African Studies*, 6:89–107.

Saib Mansour. 2008. Combining character and morpheme based models for part-of-speech tagging of Semitic languages. MSc Thesis, Technion, Haifa, Israel.

Andrew Kachites McCallum. 2002. MALLET: A machine learning for language toolkit. Webpage.

Guy De Pauw, Gilles-Maurice de Schryver, and Peter W. Wagacha. 2006. Data-driven part-of-speech tagging of Kiswahili. In *9th Int. Conf. Text, Speech and Dialogue*, pp. 197–204, Brno, Czech Rep.

Adwait Ratnaparkhi. 1996. A maximum entropy model for part-of-speech tagging. In *Empirical Methods in Natural Language Processing*, pp. 133–142, Philadelphia, Penn.

Danny Shacham and Shuly Wintner. 2007. Morphological disambiguation of Hebrew: A case study in classifier combination. In *Empirical Methods in Natural Language Processing*, pp. 439–447, Prague, Czech Rep.

Drahomirá Spoustová, Jan Hajič, Jan Votrubec, Pavel Krbec, and Pavel Květoň. 2007. The best of two worlds: Cooperation of statistical and rule-based taggers for Czech. *Balto-Slavonic Natural Language Processing*, pp. 67–74. Prague, Czech Rep.

Kristina Toutanova and Mark Johnson. 2007. A Bayesian LDA-based model for semi-supervised part-of-speech tagging. In *21st Int. Conf. Advances in Neural Information Processing Systems*, pp. 1521–1528, Vancover, B.C.

Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *HLT Conf. North American ACL*, pp. 173–180, Edmonton, Alberta.

Atro Voutilainen. 1999. An experiment on the upper bound of interjudge agreement: The case of tagging. In *9th European ACL*, pp. 204–208, Bergen, Norway.

Daniel Yacob. 1997. The System for Ethiopic Representation in ASCII — 1997 standard. Webpage.