Combining Multiple Classifiers to Improve Part of Speech Tagging: A Case Study for Brazilian Portuguese

Rachel V. Xavier Aires[1], Sandra M. Aluísio[2], Denise C. S. Kuhn[1], Marcio L. B. Andreeta[2], Osvaldo N. Oliveira Jr.[3]

[1]Núcleo Interinstitucional de Lingüística Computacional (NILC), ICMC-USP
CP 668, 13560-970 São Carlos, SP, Brazil
{rachel, denise} @nilc.icmc.sc.usp.br
[2]Instituto de Ciências Matemáticas e de Computação, USP, CP 668, 13560-970 São Carlos, SP, Brazil
{sandra, andreeta}@icmc.sc.usp.br
[3]Instituto de Física de São Carlos, USP, CP 369, 13560-970 São Carlos, SP, Brazil
chu@if.sc.usp.br

Abstract. Four taggers have been trained on a 100,000-word corpus of Brazilian Portuguese, namely Unigram (Treetagger), N-gram (Treetagger), transformation-based (TBL) and Maximum-Entropy tagging (MXPOST). The latter displayed the best accuracy (88.73%), which is still much lower than the state-of-the-art accuracy for English. The low accuracy is attributed to the reduced size of the training corpus. Twelve methods of combination were used, four of which led to an improvement over the MXPOST accuracy. The best result (89.42%) was obtained with a majority-wins voting strategy.

1 Introduction

Classifier combination is a powerful tool for improving the performance of neural networks and machine learning algorithms [1], [2], which has recently been applied successfully in natural language processing in a number of tasks, including part of speech (POS) tagging, word sense disambiguation, parsing and speech recognition. In particular, for POS tagging[1] van Halteren et al. [3] and Brill and Wu [4] have demonstrated the improvement in lexical disambiguation accuracy for English, using several methods for combining the individual classification decisions (outputs) of taggers trained on the same corpus data. Even though POS tagging is a well-understood and widely researched task, for which several language model representations have been employed, e.g. decision trees, rules, case-base, maximum entropy model [5], [6], [7], [8], it is still open to improvements. For human taggers are consistent with one another at the 98% level [9] and the outstanding data-driven

[1] Choosing the proper morphosyntactic tag for a word in a particular context from a set of possible tags can be seen as a problem of classification.

methods applied to tagging seem to achieve comparable accuracy — 96-97% correctly tagged words[2].

Following the basic idea that an ensemble of classifiers tends to perform better than a single one, as already demonstrated for English, we trained four publicly available taggers — Unigram (TreeTagger), N-gram (TreeTagger), transformation-based (TBL) and Maximum-Entropy tagging (MXPOST) — on a mixed 105,000-word Brazilian Portuguese corpus to explore their combination. The latter tagger displayed the best accuracy (88.73%), which is still unsatisfactory not only from the point of view of results from state-of-the-art accuracy for English, but also from the practical viewpoint for 90% accuracy means that, on average, each sentence contains 3 errors, considering that common sentences in running texts have an average size of 30 words. Since POS tagging is a very basic task in most NLP understanding systems, starting with this error rate in each sentence could be a severe drawback, especially considering that the propagation of such errors could grow more than linearly. The low accuracy is attributed to the reduced training corpus; high number of both unknown words and words that appear with different tags; tagset with verb subcategorization; and use of a mixed corpus.

Twelve methods of combination were used, four of which led to an improvement over the MXPOST accuracy. The choice among the several methods for combining classifiers is important because one drawback in combining classifiers is the increase in processing time. The work described in this paper is part of a larger project which aims at building a POS tagger for Brazilian Portuguese in order to tag a 35 million word corpus of Brazilian texts (NILC Corpus)[3]. As a byproduct the results may help in the creation of morphosyntactic rules for the ReGra grammar checker for Brazilian Portuguese [10].

2 Methods for Combining Classifier Outputs Used in POS Tagging

Ideally, the method of combination should take advantage of the strengths of individual classifiers, avoid their weaknesses, and improve classification accuracy [11]. However, the difficulty here is to combine the outputs in such a way that failures of some of the classifiers do not affect the correct output from the others. If one classifier provides the correct answer, then the whole system should also give the correct answer. This is the ideal situation, but there is no method that can warrant that to happen. A widely used method of combination of classifiers is the cooperative (or parallel) combination, in which each classifier has its own input and output, which are independent from the others. This method is applied after all classifiers have provided their output. Other possibilities of combination are cascaded and hierarchical [1]. van Halteren et al. and Brill and Wu have chosen the parallel type of combination. In what follows we present several techniques implemented by them to combine the outputs of classifiers for the tagging task. Table 1 shows the three major ways to combine the

[2] It is important to note that when accuracy is measured on truly ambiguous words the numbers are lower.

[3] http://www.nilc.icmc.sc.usp.br/tools.html#CORPUS

outputs and the techniques implemented by van Halteren et al. **(H)** and by Brill and Wu **(B)**.

Table 1. Methods for combining outputs

1) Random Decision		
2) Linear Combinations	Simple Majority-wins Voting	Majority1**(H)** Majority2**(B)**
	Weighted Voting	Tot Precision **(H)** Tag Precision **(H)** Precision-Recall **(H)**
	Pairwise Voting **(H)**	
3) Non-Linear Combinations	Stacking[4]	Tags **(H)** Tags+Word **(H)** Tags+Context **(H)** Pick Tag **(B)** Pick Tagger **(B)**

In a **Random Decision** system, after all classifiers provide their output, the output of one of them is chosen randomly, and serves as reference for assessing the efficiency of the other methods that include some criteria for selecting a tag. In **Majority1,** each tagger is allowed to vote for the tag of its choice and the tag with the highest number of votes is selected. In case of ties, a random selection from among the winning tags is made. For **Majority2**, the POS tag that appeared as the choice of the largest number of classifiers is picked as the answer. In ties, the tagger's output with the highest overall accuracy is chosen. We have also implemented another simple voting strategy, in which majority is only considered when 51% of the classifiers choose the same tag. If there is a tie, the choice is the tagger's output with the highest overall accuracy. This system is referred to as **Majority3**.

In **Tot Precision**, each tagger votes its accuracy (overall precision) on the suggested tag. The votes for the same class are added, and the final result is the class with more votes. In case of ties, a random selection from among the winning tags is made. **Tag Precision** follows the same idea of Tot Precision, but each tagger votes its precision (class precision) on the suggested tag. **Precision-Recall** is similar, but each tagger votes its precision (class precision) on the suggested tag plus 1 minus the recall of the opponents for its tag, i. e., precision_{tag} + (1 - $\text{recall-opponent1}_{tag}$) + ... + (1 - $\text{recall-opponentN}_{tag}$). **Pairwise Voting** allows a tag suggested by a minority (or even none) of the taggers to still have a chance to win using information from the taggers in pairs. It considers situations where one tagger suggests T_1 and another T_2, and estimates the probability that in this situation the tag should actually be T_x. For combination, every tagger pair is taken in turn and allowed to vote (with the probabilities for the apropriate tag) for each possible tag, not just the ones suggested by the taggers. For example, if we have the cases listed below, the choice will be T_{x2}.

[4] Wolpert [17] illustrated stacking using nearest neighbor predictors to form nonlinear combinations, but they can also be used in linear combinations.

T_1	T_2	T_x	Probabilities
T_1	T_2	T_{x1}	(2/12)%
T_1	T_2	T_{x2}	(5/12)%
T_1	T_2	T_{x3}	(3/12)%
T1	T2	T_{x4}	(2/12)%

However, if a tag pair (T_1-T_2) has never been observed in the Tuning Corpus, we use information on individual taggers (the probability of each tag T_x given the tagger suggested T_i). van Halteren et al. (1998) and by Brill and Wu (1998) used the **Stacking** approach where the outputs of a number of classifiers and some additional information become the features for a next learner (a Memory-based learner) producing 5 variants (or versions): Tags, Tags+Word, Tags+Context by van Halteren et al., Pick Tag, and Pick Tagger by Brill and Wu. In **Tags,** each case consists of the tags suggested by the component taggers and the correct tag, whereas the word in focus is also included in **Tags+Word**. During tagging, the similarity measure used is the frequency of a tag. In **Tags+Context**, each case consists of the information by Tags+Word plus the tags suggested by all taggers for the previous and the next position. In **Pick Tag** and **Pick Tagger** each case consists of all information of Tags+Context plus the previous and next word. The first uses this context to specify which tag to output where the latter uses it to specify which tagger to trust.

3 Tagging Systems Trained for Brazilian Portuguese Data

The experiments described in this paper are based on three systems with publicly available implementations: TreeTagger[5], Brill's Transformation-based Learning (TBL) system[6], and MXPOST[7]. Each of them uses different representations for the language model — decision trees, rules, and a maximum entropy model, respectively, and different features for POS tagging. TreeTagger also allows the generation of Unigram and Bigram taggers[8] which were trained for Brazilian Portuguese data. The three systems are described below.

3.1 TreeTagger

The probabilistic methods used for tagging are based on first and second orders Markov Models. Owing to the large number of parameters (especially in the case of trigrams), these methods are inadequate for estimating low probabilities when a limited set of training data is employed. TreeTagger, described in detail in [5],

[5] Executable code for the TreeTagger can be downloaded from http://www.ims.uni-stuttgart.de/projekte/corplex/DecisionTreeTagger.html.
[6] Brill's system is available at http://www.cs.jhu.edu/~brill.
[7] Ratnaparkhi's implementation is available at ftp://ftp.cis.upenn.edu/pub/adwait/jmx/MXPOST.html.
[8] As the Bigram tagger trained has accuracy very similar to the Trigram we are not considering it in the combination experiment.

obviates this difficulty by using a decision tree to obtain reliable estimates of the transition probabilities. This decision tree automatically determines the appropriate size of the context to generate reliable probabilities. TreeTagger also allows contexts like “tag-1 = ADJ and tag-2 <> ADJ and tag-2 <> DET”, besides allowing bigrams, trigrams and unigrams. The lexicon employed by TreeTagger contains the *a priori* probability of each word, and is split into three parts: *fullform lexicon*, with fully-written words, *suffix lexicon* for morphological analysis and *default entry* through which a given class may be chosen as default. During the search for a word in the lexicon, the *fullform lexicon* is consulted first. If the word is found, a probability is returned. In case the word is not found, the *suffix lexicon* is activated. If this also fails, then the *default entry* is returned. When Penn-Treebank corpus [12] was used for a comparison between TreeTagger and a traditional tagger based on trigrams (which do not use decision trees), TreeTagger performed better with an accuracy of 96.36% against 96.06% for the traditional tagger.

3.2 Brill’s TBL

The tagging starts with an initial corpus annotation, for example the most likely tag for a word if it is known or a guess based upon morphological knowledge if it is unknown. Then, a sequence of previously learned rules are applied[9] changing the tags based upon the contexts in which they appear. These rules are applied in sequence, obeying the order they appear in the transformation list. An example of a contextual rule is “Change a tag from Noun to Verb if the previous tag is a Modal” [4]. The contexts used for changing a tag are the words and tags within a window of three words. This hybrid approach to tagging (involves both a statistical and a symbolic component) is described in detail in [6].

3.3 MXPOST

This system is described in [8] where the context features used for POS tagging is the focus word and two words before and after, and the two previous tags. Ratnaparkhi trains a Maximum Entropy model that combines the advantages of statistically based models using Markov models or decision trees, and of Brill’s TBL.

3.4 NILC Tagset and Tagged Corpus

Several versions of the tagset, hereafter referred to as the NILC tagset, were used to cope with distinct phenomena as further data were treated. The present tagset contains 37 morphosyntactic tags (excluding tags formed by operators used in contractions and clitic pronouns)[10] and 16 of punctuation. The number of tags is important insofar as it

---

[9] In our case, 163 transformations for tagging unknown words and 334 contextual rules were learned.

[10] The NILC tagset is available at http://www.nilc.icmc.sc.usp.br/tools.html#TAGGER.

affects the performance of the learning algorithm of some taggers available in the WWW, for example, the Brill's TBL.

The manually tagged data used for this research consist of 104.966 tagged words selected from the corrected part of the NILC corpus (http://cgi.portugues.mct.pt/acesso/). There are texts from 3 different genres: newspapers (56.653 words), textbooks (16.259 words), and literature (32.054 words). This corpus were used by Bick in his hand-crafted tagger [13], and the results helped us to observe some problems with our corpus, which were then manually corrected by linguists working at NILC. We divided the corpus into 3 parts: the first part consists of 80% of the data (84.511 words), called **Training Corpus**, and is used to train each chosen tagger. The second part, **Tuning**, consists of 10% of the data (10.433 words), and is used to develop the combination methods; and the third part, **Testing**, consists of the remaining 10% (10.022 words) and is used to evaluate performance of the several combination methods. A 80/10/10 ratio for training/tuning/testing was used in order to have independent data in the training, tuning and testing phases.

Table 2 shows the accuracy of the taggers (in the Tuning corpus), which is far from the level required for POS tagging. The reason for the low accuracy is the size of the training corpus used when compared to those used to train English data (from 1 to 2 million words) with the same taggers. In order to probe how important the increase in the corpus could be, we tested the MXPOST tagger with the BNC corpus, for English, using the Claws 7 tagset, and observed an improvement in accuracy from 84.24% to 95.33% when the training corpus was increased from 100,000 to 1,000,000 words. Other factors that reflect the importance of the size of the training corpus are: for the experiments with English in [3], the testing and tuning corpora had approximately 2.5% of unknown words (new tokens), compared to ca. 15% in our experiments for Portuguese. Also, the number of known words that appear with different tags from the training corpus was 0.2% for English and here it was ca. 0.3%. This lower performance in individual taggers is reflected in the combinations, as discussed in Section 5. Another worth mentioning point is that in the NILC Tagset the verb transitivity was also considered, which is not common in other tagging procedures. When the transitivity is not taken into account, the overall accuracy of the taggers increase, reaching 91,11% for MXPOST, compared to 88.57% in Table 2.

Table 2. Accuracy of individual taggers

	Accuracy in Tuning (%)	Accuracy in Testing (%)
TreeTagger as Unigram	79.13%	77.16%
Brill's TBL	86.63%	86.42%
TreeTagger as Trigram	87.40%	87.12%
MXPOST	88.57%	88.73%

The best accuracy is that of MXPOST, corroborating the results presented by [3] and [4]. The reason why MXPOST performs better may lie in the advantages offered by the Maximum Entropy model. It can model subtle dependencies among variables in a way that is not possible with traditional predictive modeling techniques. Moreover, it makes no unwarranted assumptions about the data, unlike decision trees and neural networks, and this makes the Maximum Entropy model to perform better on unseen data. A surprise here is that the trigram tagger outperforms the results from the latter authors, probably due to the decision tree trigram used here instead of the traditional trigram tagger used by van Halteren et al. and Brill and Wu. In addition, this performance appears to depend on the size of the training corpus, which makes Treetagger to outperform Brill's TBL in a smaller corpus.

The accuracy of MXPOST obtained here is above that obtained for contemporary Portuguese using a statistical tagger (84.5%) [14], and close to the value quoted for a neural network tagger (88.7%) [15]. Such comparisons should be analyzed with caution because parameters vary widely in the experiments by distinct groups, especially the tagset and the sizes (and their ratio) of the training and testing corpora. In particular, the works [14] and [15] employed a smaller tagset than the NILC tagset and did not consider verb transitivity.

4 Tagger Complementarity for Brazilian Portuguese Data

We shall use the same measure used by Brill and Wu to analyze how different the errors of the taggers are. The complementary rate of taggers X and Y is defined as:

$$\text{Comp}(X,Y) = (1 - (\text{\# of common errors}/\text{\# of errors in X only}))*100 \quad (\mathbf{1})$$

and measures the percentage of time when tagger X is wrong that tagger Y is correct. As shown in Table 3, the complementary rates are high. For example, when MXPOST is wrong, the TreeTagger as Trigram is right 31.2% of the time, and when the TreeTagger as Trigram is wrong, MXPOST is right 37.59% of the time.

Table 3. Complementary rate between taggers

Y X	**Unigram**	**Trigram**	**Brill's TBL**	**MXPOST**
Unigram	_	52.68%	55.81%	64.49%
Trigram	21.61%	_	28.69%	37.59%
Brill's TBL	31.03%	32.83%	_	38.35%
MXPOST	35.15%	31.20%	27.85%	_

For combination to work well, i.e. to give better tagging accuracy, the chosen taggers must differ in the errors they make, because otherwise the combination would be vain. Table 4 shows that in 94.35% of words in the Tuning corpus (10.433 words), at least one of the four taggers selects the correct tag. When three taggers (without Unigram) are employed, in 93.15% of the words there is at least one correct tagging (third column). These results are encouraging, but they represent at the same time an ideal situation, which probably will not be reached.

Table 4. Tagger agreement in tuning

	All taggers	Taggers without unigram
All Taggers Correct	72.15%	80.85%
Majority Correct	14.14 %	7.76%
Correct Present Without Majority	4.32 %	1.41%
Minority Correct	3.74%	3.14%
All Taggers Wrong	5.65%	6.84%

In other words, if we had an oracle that was able to always pick up the correct tag from the different outputs, we could make improvements on the results of the best accurate tagger, MXPOST. Table 5 shows that the error rates decrease as we add further taggers to the combination pool. For example, when the oracle works with the four trained taggers the error rate is 5.655%, i.e. a 50.5% error rate reduction obtained over MXPOST error rate. This means that we can expect additive improvements, even though we do not know when saturation will be reached.

Table 5. Combination with optimal selection of tags

	MXPOST	+ Trigram	+ Brill's TBL	+ Unigram
% of the time all are wrong	11.42%	7.86%	6.84%	5.655%
% oracle improvement		31.20%	40.13%	50.50%

5 Combination Methods and Results for Brazilian Portuguese Data

The results shown in Table 5 refer to the maximum possible improvement in relation to the MXPOST tagger. In practice, combinations may not be so efficient, as shown in Table 5 for several combinations of the 4 taggers. Four combination methods, Majority3, TotPrecision, Tag Precision, and Tag Pair do outperform MXPOST (88.73%), whereas the other combinations lead to less accurate results (Table 6).

Table 6. Accuracy of combination methods

Random Decision	84.49%
Majority1	87.33%
Majority2	87.72%
Majority3	89.42%
Tot Precision	89.28%
Tag Precision	89.01%
Precision-Recall	88.58%
Tag Pair	88.89%
Tags	88.11%
Tags+word	88.12%
Pick Tag	81.27%
Pick Tagger	87.46%

These results differ from those obtained for English, where all combination of classifiers led to an improvement in the performance of the best individual tagger. In our experiments, because the training corpus was relatively small, a large number of unknown words had to be tagged and therefore random choices in cases of ties were actually worse than the best classifier. Improvements were only obtained in cases where smart criteria were used to pick the best choice, for which the best result was the Majority3 voting. It is possible that as the size of the training corpus is increased, other combinations may also outperform MXPOST.

6 Conclusions and Future Directions

The results presented here indicate clearly that the size of the training corpus must be increased considerably before the accuracy for POS tagging for Portuguese reaches the levels today obtained for English. The results were encouraging, though, because if the trend observed for English upon increasing the size of the corpus applies to Portuguese, one can expect to reach 98% with a one million word corpus. Another consequence of a small training corpus was the number of unknown words, which affected the output of combinations. Unlike the case of English, only some combinations led to improvements on the accuracy of MXPOST, the most accurate tagger. Therefore, we are currently increasing the size of our corpus, for which we are applying the most accurate combination of classifiers, followed by human revision that will be supported by a software tool. The latter will not only deal automatically with the division of the corpus into training, tuning and testing, among the three genres of text, but also help in identifying the tags and contexts posing most problems in classification. We shall also implement other taggers, including a symbolic one developed at NILC, and test the bagging algorithm [16] with our data mainly because it is an effective variance reduction method that is useful when data is not abundant, besides being very successful when applied to unstable classifiers as neural nets and decision trees. The new results will be used to investigate whether the complementary rates increase in comparison with the taggers already investigated, which is to be followed by implementation of other methods of combination from Table 1.

Acknowledgments

The financial support from Finep (#88-98-05910002-10) and CNPq (Brazil) is acknowledged.

References

1. Bick, E.: Automatic Parsing of Portuguese. In Proc. of II Encontro para Processamento Computacional do Português Escrito e Falado. Curitiba. (1996) 91-100
2. Brill, E.: Transformation-based error-driven learning of natural language: A case study in part of speech tagging. Computational Linguistics, v. 21, n.4. (1995) 543-565
3. Brill, E. and Wu, J.: Classifier Combination for Improved Lexical Disambiguation. ACL 98. (1998).
4. Charniak, E.: Statistical techniques for natural language parsing. AI Magazine. (1997). Also available in http://www.cs.brown.edu/people/ec/home.html#biography
5. Daelemans, W., Zavrel, J., Berch, P. and Gillis, S.: MBT: A Memory-Based Part of Speech Tagger-Generator. In Proceedings of the Fourth Workshop on Very Large Corpora / Ejerhed E. (eds.), Copenhagen. (1996) 14-27
6. Dietterich, T.: Machine-Learning Research: Four Current Directions. AI Magazine. Winter 1997. (1997) 97-136
7. Ho, T. K., Hull, J. J. and Srihari, S. N.: Decision Combination in Multiple Classifier Systems. IEEE Transations on pattern analysis and machine intelligence. Vol.16, no.1. January 1994. (1994)
8. Marcus, M., Santorini, B and Marcinkiewicz, M.: Building a large annotated corpus of English: the Penn Treebank. Computational Linguistics, Vol. 19, 1993. (1993)
9. Marques, N. C. and Lopes, G. P. : A Neural Network Approach to Part-of-Speech Tagging. In Proc. of II Encontro para Processamento Computacional do Português Escrito e Falado. Curitiba. (1996) 1-9
10. Martins, R.T., Hasegawa, R., Nunes, M.G.V., Montilha, G. and Oliveira, Jr. O.N.: Linguistic issues in the development of ReGra: a Grammar Checker for Brazilian Portuguese.Natural Language Engineering, Volume 4 (Part 4 December 1998), Cambridge University Press. (1998) 287-307
11. Ratnaparkhi, A.: A Maximum Entropy Part-of-Speech Tagger. Proceedings of the First Empirical Methods in Natural Language Processing Conference. Philadelphia, Pa. (1996)
12. Sharkey, A.J.C.: Combining Artificial Neural Nets: Ensemble and Modular Multi-Net Systems. London: Springer. (1999)
13. Schmid, H.: Probabilistic part-of-speech tagging using decision trees. In Proc. of International Conference on New Methods in Language Processing. (1994) 44-49
14. van Halteren, H., Zavrel, J. and Daelemans, W.: Improving Data Driven Wordclass Tagging by System Combination. COLING-ACL 1998. (1998) 491-497
15. Villavicencio, A. et al.: Part-of-Speech Tagging for Portuguese Texts. In Anais do Simpósio Brasileiro de Inteligência Artificial (SBIA´95) (1995)
16. Breiman, L.: Bagging Predictors. Machine Learning, 24, (1996) 123-140
17. Wolpert, D.: Stacked Generalization. Neural Networks, Vol. 5, (1992) 241-259