POS tagger combinations on Hungarian text

András Kuba, László Felföldi, András Kocsor
Research Group on Artificial Intelligence,
University of Szeged,
Aradi vrt. 1, H-6720, Hungary
{akuba, lfelfold, kocsor}@inf.u-szeged.hu

Abstract

In this paper we will briefly survey the key results achieved so far in Hungarian POS tagging and show how classifier combination techniques can aid the POS taggers. Methods are evaluated on a manually annotated corpus containing 1.2 million words. POS tagger tests were performed on single-domain, multiple domain and cross-domain test settings, and, to improve the accuracy of the taggers, various combination rules were implemented. The results indicate that combination schemas (like the Boosting algorithm) are promising tools which can significantly degrade the classification errors, and produce a more effective tagger application.

1 Introduction an related works

Part-of-speech (POS) tagging is perhaps one of the most basic tasks in natural language processing. In this paper we will review the current state-of-the-art in Hungarian POS tagging, and investigate the possibilities of improving the results of the taggers by applying classifier combination techniques.

We used a Transformation Based Learner (TBL) as the base tagger for combination experiments, and the learner algorithm determines the set of applicable combination schemes. From this set we chose two algorithms called *Bagging* and *Adaboost.M1*.

In the next subsection, the most important published results of the last few years in Hungarian POS tagging are summarized. The TBL tagger is described in detail in Section 2, then the corpora and the data sets we used for our investigations are presented in Section **??**. The classifier combination and details about the implementation of this technique are described in Section 3. After the results of the boosted tagger are presented in Section 4. Lastly, some conclusions about the effectiveness and efficiency of our boosting approach are made in the final section.

1.1 POS Tagging of Hungarian Texts

Standard POS tagging methods were applied to Hungarian as soon as the first annotated corpora appeared that were big enough to serve as a training database for various methods. The TELRI corpus (Dimitrova et al., 1998) was the first corpus that was used for testing different POS tagging methods. This corpus contains approximately 80,000 words. Later, as the Hungarian National Corpus (Váradi, 2002) and the Manually Annotated Hungarian Corpus (the Szeged Corpus) (Alexin et al., 2003) became available, an opportunity was provided to test the results on bigger corpora (153M and 1.2M words, respectively).

In recent years several authors have published many useful POS tagging results in Hungarian. It is generally believed that, ow-

ing to the fairly free word order and the agglutinative property of the Hungarian language, there are more special problems associated with Hungarian than those of the Indo-European languages. However, the latest results are comparable to results achieved in English and other well-studied languages. Fruitful approaches for Hungarian POS tagging are Hidden Markov Models, Transformation Based Learning and rule-based learning methods.

One of the most common POS tagging approaches is to build a tagger based on Hidden Markov Models (HMM). Tufis (Tufis et al., 2000) reported good results with the Trigrams and Tags (TnT) tagger (Brants, 2000). A slightly better version of TnT was employed by Oravecz (Oravecz and Dienes, 2002), and it achieved excellent results. In their paper, Oravecz and Dienes (Oravecz and Dienes, 2002) argue that regardless of the rich morphology and relatively free word order, the POS tagging of Hungarian with HMM methods is possible and effective once one is able to handle the data sparsity problem. They used a modified version of TnT that was supported by an external morphological analyzer. In this way the trigram tagger was able to make better guesses about the unseen words and therefore to get better results. An example of the results achieved by this trigram tagger is presented in the first row of Table 1.

Another approach besides the statistical methods is the rule-based learning one. A valuable feature of the rule-based methods is that the rules these methods work with are usually more intelligible to humans than the parameters of statistical methods. For Hungarian, a few such approaches are available in the literature.

In a comprehensive investigation, Horváth et al. (Horváth et al., 1999) applied five different machine learning methods to Hungarian POS tagging. They tested C4.5, PHM, RIBL, Progol and AGLEARN (Alexin et al., 1999) methods on the TELRI corpus. The results of C4.5 and the best tagger found in this investigation (RIBL) are presented in the second and third rows of Table 1.

Tagger	Accuracy
TnT + Morph. Ana.	98.11%
C4.5	97.60%
Best method (RIBL)	98.03%
RGLearn	97.32%
TBL	91.94%
TBL + Morph. Ana.	96.52%
Best combination	96.95%

Table 1: Results achieved by various Hungarian POS taggers.

Hócza (Hócza et al., 2003) used a different rule generalization method called RGLearn. Row 4 shows the test results of that tagger in Table 1. Transformation Based Learning is a rule-based method that we will discuss in depth in Section 2. Megyesi (Megyesi, 1999) and Kuba et al. (Kuba et al., 2004) produced results with TBL taggers that are given in Table 1, in rows 5 and 6, respectively. Kuba et al. (Kuba et al., 2004) performed experiments with combinations of various tagger methods. The combinations outperformed their component taggers in almost every case. However, in the different test sets, different combinations proved the best, so no conclusion could be drawn about the best combination. The combined tagger that performed best on the largest test set is shown in row 7 of Table 1.

2 The TBL tagger

Transformation Based Learning (TBL) was introduced by Brill (Brill, 1995) for the task of POS tagging. Brill's implementation consists of two processing steps. In the first step, a lexical tagger calculates the POS tags based on lexical information only (word forms). The result of the lexical tagger is used as a *first guess* in the second run where both the word forms and the actual POS tags are applied by the contextual tagger. Both lexical and contextual taggers make use of the TBL concept.

During training, TBL performs a greedy search in a rule space in order to find the rules that best improve the correctness of the cur-

rent tagging. The rule space contains rules that change the POS tag of some words according to their environments. From these rules, an ordered list is created. In the tagging phase, the rules on the rule list are applied one after another in the same order as the rule list. After the last rule is applied, the current tag sequence is returned as a result.

For the Hungarian language, Megyesi applied this technique initially with moderate success. (Megyesi, 1998) The weak part of her first implementation was the lexical module of the tagger, as described in (Megyesi, 1999). With the use of extended lexical templates, TBL produced a much better performance but still lagged behind the statistical taggers.

We chose a different approach that is similar to (Kuba et al., 2004). The *first guess* of the TBL tagger is the result of the baseline tagger. For the second run, the contextual tagger implementation we used is based on the fnTBL learner module. (Ngai and Florian, 2001) We used the standard parameter settings included in the fnTBL package.

2.1 Baseline Tagger

The baseline tagger relies on an external morphological analyzer[1] to get the list of possible POS tags. If the word occurs in the training data, the word gets its most frequent POS tag in the training. If the word does not appear in the training, but representatives of its ambiguity class (words with the same possible POS tags) are present, then the most frequent tag of all these words will be selected. Otherwise, the word gets the first tag from the list of possible POS tags.

Some results produced by the baseline tagger and the improvements achieved by the TBL tagger are given in Table 2.

Test Domain	Baseline	TBL
full corpus	94.94%	96.52%
business news	97.56%	98.26%
cross-domain	79.51%	95.79%

Table 2: Accuracy of the baseline tagger and the TBL tagger.

3 Classifier Combinations

The goal of designing pattern recognition systems is to achieve the best possible classification performance for the specified task. This objective traditionally led to the development of different classification schemes for recognition problems the user would like solved. Experiments shows that although one of the designs should yield the best performance, the sets of patterns misclassified by the different classifiers do not necessarily overlap. These observations motivated the relatively recent interest in combining classifiers. The main idea behind it is not to rely on the decision of a single classifier. Rather, all of the inducers or their subsets are employed for decision-making by combining their individual opinions to produce a final decision.

3.1 Bagging

The *Bagging* (Bootstrap aggregating) algorithm (Breiman, 1996) applies majority voting (*Sum Rule*) to aggregate the classifiers generated by different bootstrap samples. A bootstrap sample is generated by uniformly sampling m instances from the training set with replacement. T bootstrap samples $B_1, B_2, ..., B_T$ are generated and a classifier C_i is built from each bootstrap sample B_i.

Bagging algorithm

Require: Training Set $S = \{\mathbf{x}_1, \ldots, \mathbf{x}_m\}$
Ensure: Combined classifier $\hat{C}$
  for $i = 1 \ldots T$ **do**
    S' = bootstrap sample from S
    Train classifer C_i on S'
  end for
  $\hat{C}(\mathbf{x}) = \underset{j}{\operatorname{argmax}} \sum_i \mathbf{I}[C_i(\mathbf{x}) = \omega_j]$

For a given bootstrap sample, an instance in the training set will have a probability $1 - (1 - 1/m)^m$ of being selected at least once from the m instances that are randomly picked from the training set. For large m,

[1] We used the Humor (Prószéky, 1995) analyzer developed by MorphoLogic Ltd.

this is about 1-1/e = 63.2%. This perturbation causes different classifiers to be built if the inducer is unstable (e.g. ANNs, decision trees) and the performance may improve if the induced classifiers are uncorrelated. However, Bagging can slightly degrade the performance of stable algorithms (e.g. kNN) since effectively smaller training sets are used for training.

3.2 Boosting

Boosting (Freund and Schapire, 1996) was introduced by Shapire as a method for improving the performance of a weak learning algorithm. AdaBoost changes the weights of the training instances provided as input for each inducer based on classifiers that were previously built. The final decision is made using a weighted majority voting schema for each classifier, whose weights depend on the performance of the training set used to build it.

Adaboost algorithm

Require: Training Set $S = \{\mathbf{x}_1, \ldots, \mathbf{x}_m\}$
Ensure: Combined classifier $\hat{C}$
$\mathbf{d}_j^{(1)} = 1/N$ for all $j = 1 \ldots m$
for $t = 1 \ldots T$ **do**
Train classifier C_t with respect to the distribution $\mathbf{d}^{(t)}$.
Calculate the weighted training error ϵ_t of C_t: $\epsilon_t = \sum_{j=1}^{m} \mathbf{d}_j^{(t)} \mathbf{I}[C_t(\mathbf{x}_j) \neq \omega_j]$
if $\epsilon_t > 1/2$ **then** Exit
Set α_t: $\alpha_t = \frac{1}{2} log \frac{1-\epsilon_t}{\epsilon_t}$
$\mathbf{d}_j^{(t+1)} = \mathbf{d}_j^{(t+1)} \frac{1}{Z_t} e^{-\alpha_t I[C_t(\mathbf{x}) \neq \omega_j]}$,
where Z_t is the normalization constant, such that: $\sum_{j=1}^{m} \mathbf{d}_j^{(t+1)} = 1$
end for
$\hat{C}(\mathbf{x}) = \underset{j}{\text{argmax}} \sum_t \alpha_t \mathbf{I}[C_t(\mathbf{x}) = \omega_j]$

The boosting algorithm requires a weak learning algorithm whose error is bounded by a constant strictly less than 1/2. In the case of multi-class classifications this condition might be difficult to guarantee, and various techniques may need to be applied to get round this restriction.

There is an important issue that relates to the construction of weak learners. At step t the weak learner is constructed based on the weighting $\mathbf{d}^t$. Basically, there are two approaches for taking this weighting into account. In the first approach we assume that the learning algorithm can operate with reweighted examples. For instance, when the learner minimizes a cost function, one can construct a revised cost function which assigns weights to each of the examples. However, not all learners can be easily adapted to such an inclusion of the weights. The other approach is based on resampling the data with replacement. This approach is more general as it is applicable to all kinds of learners.

4 Boosted results for TBL

TBL belongs to the group of learners that generates abstract information, i.e. only the class label of the source instance. Although it is possible to transcribe the output format to confidence type, this limitation degrades the range of the applicable combination schemes. *Min*, *Max* and *Prod Rules* cannot produce a competitive classifier ensemble, while *Sum Rule* and *Borda Count* are equivalent to majority voting. From the set of available boosting algorithms we may only apply those methods that do not require the modification of the learner.

For the experiments we chose *Bagging* and a modified *Adaboost.M1* algorithm as boosting. Since the learner is incapable of handling instance weights, individual training datasets were generated by bootstrapping (i.e. resamping with replacement). The original Adaboost.M1 algorithm requires that the weighted error should be below 50%. In this case the modified algorithm reinitializes the instance weights and goes on with the processing.

The Boosting algorithm is based on weighting the independent instances based on the classification error of the previously trained

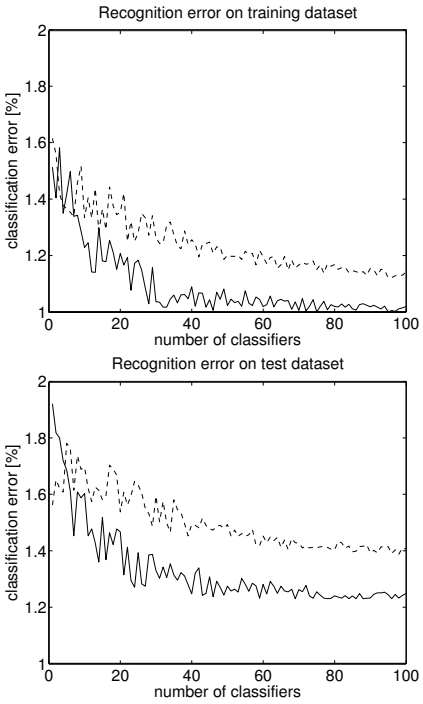

Figure 1: Classification error of *Bagging* and *Boosting* algorithm on the training and training datatsets (dashed: Bagging, solid: Boosting).

learners. TBL operates on words, but words are not treated as independent instances. Their context, the position in the sentence, affects the building of the classifier. Thus instead of the straightforward selection of words, the boosting method handles the sentences as instance samples. The classification error of the instances are calculated as the arithmetic mean of the classification errors of the words in the corresponding sentence. Despite this, the combined final error is expressed as the relative number of the misclassified words.

The training and testing datasets were chosen from the business news domain of the Szeged Corpus. The train database contained about 128,000 annotated words in 5700 sentences. The remaining part of the domain, the test database, has 13,300 words in 600 sentences.

The results of training and testing error rates are shown in Fig.1. The classification error of the stand-alone TBL algorithm on the test dataset was 1.74%. Boosting is capable of decreasing it to below 1.31%, which means a 24.7% relative error reduction. As the graphs show, boosting achieves this during the first 20 iterations, so further processing steps cannot make much difference to the classification accuracy. It can also be seen that the training error does not converge to a zero-error level. This behavior is due to the fact that the learner cannot maintain the 50% weighted error limit condition. Bagging achieved only a moderate gain in accuracy, its relative error reduction rate being 18%.

5 Conclusions

In this paper we investigated the possibility of improving the classification performance of POS taggers by applying classifier combination schemas. For the experiments we chose TBL as a tagger and *Bagging* and *Boosting* as combination schemas. The results indicates that Bagging and Boosting can reduce the classification error of the TBL tagger by 18% and 24.7%, respectively.

It is clear that further improvements could be made by tailoring the tagger algorithm to the requirements of more sophisticated boosting methods like Adaboost.M2 and other derivatives optimized for multi-class recognition. Another promising idea for more effective combinations is that of applying confidence-type generative (ANN, kNN, SVM) or discriminative (HMM) learners (Duda et al., 2001; Bishop, 1995; Vapnik, 1998). These kinds of classifiers provide more information about the recognition results and can improve the cooperation between the combiner and the classifiers. In the future we plan to investigate these alternatives for constructing better tagger combinations.

References

Zoltán Alexin, Szilvia Zvada, and Tibor Gyimóthy. 1999. Application of AGLEARN on Hungarian Part-of-Speech tagging. In D. Parigot and M. Mernik, editors, *Second Workshop on Attribute Grammars and their Applications, WAGA'99*, pages 133–152, Amsterdam, The Netherlands. INRIA rocquencourt.

Zoltán Alexin, János Csirik, Tibor Gyimóthy, Károly Bibok, Csaba Hatvani, Gábor Prószéki, and László Tihanyi. 2003. Manually annotated Hungarian corpus. In *Proceedings of the Research Note Sessions of the 10th Conference of the European Chapter of the Association for Computational Linguistics, EACL'03*, pages 53–56, Budapest, Hungary.

C. M. Bishop. 1995. *Neural Networks for Pattern Recognition*. Oxford University Press.

Thorsten Brants. 2000. TnT – a statistical part-of-speech tagger. In *Proceedings of the Sixth Applied Natural Language Processing, ANLP-2000*, Seattle, WA.

L. Breiman. 1996. Bagging predictors. *Machine Learning*, 24, no. 2:123–140.

Eric Brill. 1995. Transformation-based error-driven learning and natural language processing: A case study in part-of-speech tagging. *Computational Linguistics*, 21(4):543–565.

Ludmila Dimitrova, Tomaž Erjavec, Nancy Ide, Heiki Jaan Kaalep, Vladimir Petkevic, and Dan Tufis. 1998. Multext-east: Parallel and comparable corpora and lexicons for six Central and Eastern European languages. In Christian Boitet and Pete Whitelock, editors, *Proceedings of the Thirty-Sixth Annual Meeting of the Association for Computational Linguistics and Seventeenth International Conference on Computational Linguistics*, pages 315–319, San Francisco, California. Morgan Kaufmann Publishers.

R. O. Duda, P. E. Hart, and D. G. Stork. 2001. *Pattern Classification*. John Wiley and Son, New York.

Yoav Freund and Robert E. Schapire. 1996. Experiments with a new boosting algorithm. In Morgan Kaufmann, editor, *Proc International Conference on Machine Learning*, pages 148–156, San Francisco.

András Hócza, Zoltán Alexin, Dóra Csendes, János Csirik, and Tibor Gyimóthy. 2003. Application of ILP methods in different natural language processing phases for information extraction from Hungarian texts. In *Proceedings of the Kalmár Workshop on Logic and Computer Science*, pages 107–116, Szeged, Hungary.

Tamás Horváth, Zoltán Alexin, Tibor Gyimóthy, and Stefan Wrobel. 1999. Application of different learning methods to Hungarian Part-of-Speech tagging. In S. Džeroski and P. Flach, editors, *Proceedings of ILP99*, volume 1634 of *LNAI*, pages 128–139. Springer Verlag.

András Kuba, András Hócza, and János Csirik. 2004. POS tagging of Hungarian with combined statistical and rule-based methods. In Petr Sojka, Ivan Kopecek, and Karel Pala, editors, *Text, Speech and Dialogue, Proceedings of the 7th International Conference, TSD 2004*, pages 113–121, Brno, Czech Republic.

Beáta Megyesi. 1998. Brill's rule-based POS tagger for Hungarian. Master's thesis, Department of Linguistics, Stockholm University, Sweden.

Beáta Megyesi. 1999. Improving Brill's POS tagger for an agglutinative language. In *Proceedings of the Joint Sigdat Conference on Empirical Methods in Natural Language Processing and Very Large Corpora, EMNLP/VLC '99*, pages 275–284.

Grace Ngai and Radu Florian. 2001. Transformation-based learning in the fast lane. In *Proceedings of North American ACL 2001*, pages 40–47, June.

Csaba Oravecz and Péter Dienes. 2002. Efficient stochastic Part-of-Speech tagging for Hungarian. In *Proceedings of the Third International Conference on Language Resources and Evaluation, LREC2002*, pages 710–717, Las Palmas.

Gábor Prószéky. 1995. Humor: a morphological system for corpus analysis. In H. Retting, editor, *Language Resources for Language Technology, Proceedings of the First European TELRI Seminar*, pages 149–158, Tihany, Hungary.

Dan Tufis, Péter Dienes, Csaba Oravecz, and Tamás Váradi. 2000. Principled hidden tagset design for tiered tagging of Hungarian.

V. N. Vapnik. 1998. *Statistical Learning Theory*. John Wiley and Son.

Tamás Váradi. 2002. The Hungarian National Corpus. In *Proceedings of the Third International Conference on Language Resources and Evaluation, LREC2002*, pages 385–396, Las Palmas de Gran Canaria.