Combining Language Independent Part-of-Speech Tagging Tools

György Orosz[1], László János Laki[1], Attila Novák[1], and Borbála Siklósi[2]

1 **MTA-PPKE Language Technology Research Group –**
Pázmány Péter Catholic University, Faculty of Information Technology
50/a Práter street, Budapest, Hungary
{oroszgy, laki.laszlo, novak.attila}@itk.ppke.hu

2 **Pázmány Péter Catholic University, Faculty of Information Technology**
50/a Práter street, Budapest, Hungary
siklosi.borbala@itk.ppke.hu

Abstract

Part-of-speech tagging is a fundamental task of natural language processing. For languages with a very rich agglutinating morphology, generic PoS tagging algorithms do not yield very high accuracy due to data sparseness issues. Though integrating a morphological analyzer can efficiently solve this problem, this is a resource-intensive solution. In this paper we show a method of combining language independent statistical solutions – including a statistical machine translation tool – of PoS-tagging to effectively boost tagging accuracy. Our experiments show that, using the same training set, our combination of language independent tools yield an accuracy that approaches that of a language dependent system with an integrated morphological analyzer.

1998 ACM Subject Classification I.2.7 Natural Language Processing

Keywords and phrases part-of-speech tagging, combination, agglutinative languages, machine learning, machine translation

Digital Object Identifier 10.4230/OASIcs.SLATE.2013.249

1 Introduction

Part-of-speech tagging is one of the basic and most studied tasks of computational linguistics. There are several freely available language independent solutions which are usually based on statistical methods. The robust and accurate operation of these tools is crucial, since they are usually one of the first components of any linguistic processing chain. Thus errors propagating from this level affect the result of systems performing more complex language processing tasks.

In our present work, we describe a method of combining two independent tools: the HMM-based PurePos [14] and the HuLaPos PoS tagger [12] based on the Moses decoder [11]. Deeper investigation of incorrectly classified words has reflected that the overlap between the errors made by each of these systems is very small. Inspired by this observation, we experimented with possibilities of combining the knowledge of these systems. We prove that using the combination of the two language independent systems yields a better result than using a simple majority voting of three tools by extending the investigation to a third system as well. Our results also show that for Hungarian, the tagging accuracy of the presented language independent method approaches that of the augmented version of PurePos that employs a language dependent morphological analyzer.

2nd Symposium on Languages, Applications and Technologies (SLATE'13).
Editors: José Paulo Leal, Ricardo Rocha, Alberto Simões; pp. 249–257
OpenAccess Series in Informatics
OASICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

2 Tools

In this article, we explore ways of combining the following tools:

1. **PurePos** is an open source hybrid system for full morphological disambiguation: it is capable of not just selecting the most probable tag for a token, but assigning a lemma as well. It is based on hidden Markov models, but it can use an integrated morphological analyzer module as well to tag unseen words and to assign lemmas. The tool is based on algorithms described by Brants [3] and Halácsy [9], but what distinguishes it from them is the complete integration of a morphological analyzer. Its extremely short training time is due to the usage of a simple smoothed trigram model while some tweaks in the implementation result in a high precision. It is implemented in Java, thus it can be easily extended and is portable. Integration of a morphology results in a further boost in its PoS tagging accuracy and also makes lemmatization possible.
2. **HuLaPos**: is a morphological annotation tool based on an SMT decoder(HuLaPos [12]). It is possible to create a near state-of-the-art system that we created with minimal preprocessing and – compared to corpus sizes needed for SMT – a relatively small training set. Another advantage of the SMT translation process applied for PoS tagging is that it is able to consider the context of a word in both directions. Its only weakness is that it is not capable of tagging out-of-vocabulary (OOV) words not represented in the training corpus. However with smoothing based on the distribution of rare words, the error rate of tagging of OOV words was decreased.
3. **OpenNLP**: In our experiments, we also used the **maximum entropy** and **perceptron learning** algorithms implemented in the OpenNLP toolkit[1] [2]. These are very popular annotation methods, since the feature sets used for training can be easily adapted for new tasks. However the main drawback of these algorithms is that their training time is extremely high compared to HMM based models. They were employed in our tests with their default feature set.

We have seen in the case of the PurePos system that morphological knowledge is very useful, especially in the case of agglutinating languages (such as Hungarian), but a morphological analyzer is often not available and it is very time consuming to build one and it requires the involvement of expert linguists.

3 Motivation for Tagger Combination

We investigated the tagging performance (table 1) and errors (figure 1) of the above described four systems and found, that though the accuracy of PurePos is higher relative to the others, its errors overlap only slightly with those of the HuLaPos system. There are also cases where words mistagged by PurePos and HuLaPos are correctly tagged by one of the two modules of OpenNLP. However, the error sets of the maxent and perceptron learning algorithms are very similar to each other.

We performed our experiments on a modified version of the Hungarian Szeged Corpus [6], in which PoS annotation was automatically converted to morphosyntactic tags used by the Hungarian HuMor morphological analyzer [13, 15]. It was done for the purpose of comparing our results with an available state-of-the-art hybrid disambiguator. The taggers described above use the same rich tagset that is available in the training corpus and provided by the

[1] Henceforward PE denotes the preceptron learning method while ME denotes the maximum entropy learning method of the toolkit.

Table 1 Tagging precision of the baseline systems.

PoS tagger	Precision
PurePos	97.85%
HuLaPos	97.57%
OpenNLP perceptron	93.86%
OpenNLP maxent	93.03%

analyzer. In the case of the training data this amounts to more than a thousand different tags. 10% of the corpus was separated for testing and another 10% of the corpus is used for development and tuning purposes. Each set contains about 7100 sentences, while the rest, about 57000 sentences, were used for training of the systems.

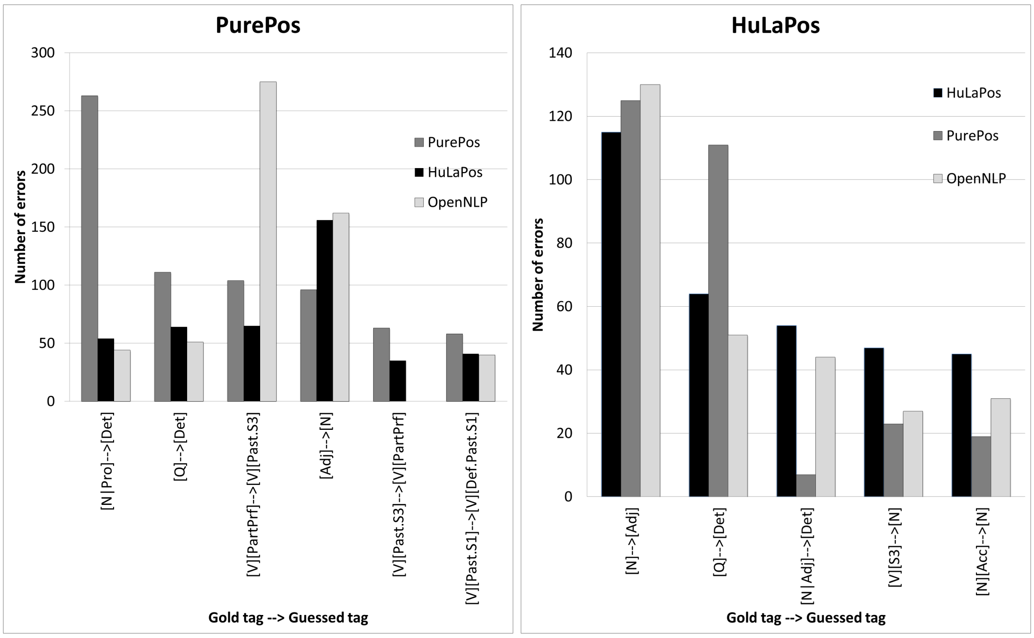

Figure 1 Comparing the most frequent errors of PurePos and HuLaPos.

Performing a deep error analysis, we collected the most frequent error types that together represent 30% of all errors of each tool. From figure 1. one can conclude, that there are some error types, that are specific to PurePos. These are the following: mistagging of demonstrative pronouns as definite articles (az[N|Pro] 'that' vs. az[Det] 'the') and the numeral egy[Q] 'one' as the indefinite article egy[Det] 'a'). There is a significant difference between the performance of tagging past participles ([V][PartPrf]): these are mistagged as simple past verb forms ([V][Past.S3]) by PurePos more often than by HuLaPos.

On the other hand, the SMT system very often assigns wrong tags to word forms not seen in the training set while it performs better for words that were seen. Besides, some typical errors of HuLaPos are the following: mistagging adjectives ([Adj]) as nouns ([N]) and assigning a nominative noun tag ([N]) to verbs ([V][S3]) and accusative nouns ([N][ACC]).

While there are some cases where the perceptron learning method could guess the right label while both HuLaPos and PurePos missed it (such as the verbal tag [V][Past.S3]),

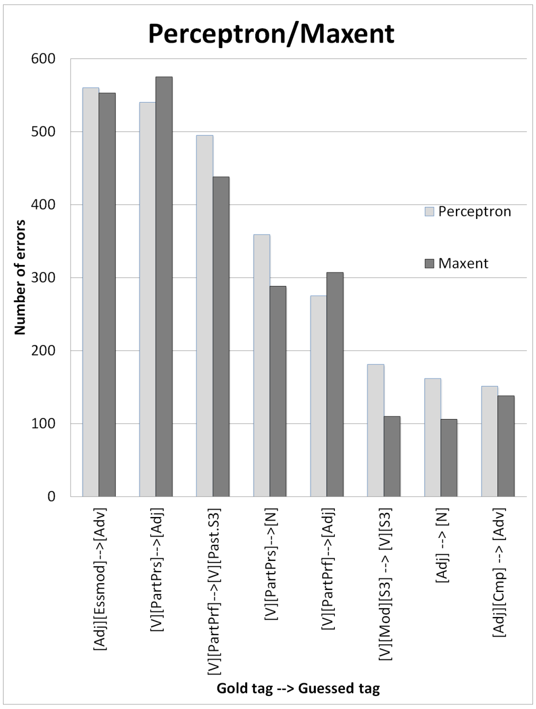

Figure 2 Comparing the ME and PE methods error rate.

its overall tagging precision is significantly lower. Moreover, comparing the ME and PE methods (see figure 2) we can conclude that their errors are mostly overlapping. This is due to the fact that both their feature sets and the decoding algorithm are the same, only the training algorithm differs.

Table 2 The maximal knowledge of tagger combinations.

Name	Precision
Max2: PurePos + HuLaPos	98.84%
Max3: PurePos + HuLaPos + PE	99.21%
Max4: PurePos + HuLaPos + PE + ME	99.27%

Hypothetical maximum Relying on the above error analysis we can calculate what performance a hypothetical combination algorithm could reach that always succeeds in selecting the best of all tags proposed by the taggers to be combined. This oracle tagger was simulated by presuming that one could always decide which tagger to trust. Thus the combination of two (Max2), three (Max4) or four (Max4) tools may perform significantly better than each of the individual ones, if one manages to combine them right. The ideal combined tagger that could aggregate the knowledge of the PurePos, HuLaPos and the PE systems, could lower the error rate of the best one with almost 66% percent. One that combines the best two: PurePos and HuLaPos only, could also achieve an almost of 46% error rate reduction. While mining the knowledge of all four systems could in theory result in the best tagger, since the two methods derived from OpenNLP highly correlate, we skip the poorly performing maxent tagger in this investigation.

In the rest of this paper, we focus only on the combination of the two or three best performing systems.

4 Tagger Combinations

The task of combining several classification systems is traditionally composed of two subtasks. First, one has to select the appropriate features that may be used, then one must select an appropriate combining algorithm. (In data mining, this procedure is commonly referred to as stacking learners.) Although part-of-speech tagging is a classification task [16], where the tagger assigns the most probable tag to each token in the sentence, it is not done in a token by token manner, rather sentence by sentence. Consequently, the individual tagging events are not independent. Most of the statistical tagging algorithms heavily rely on this fact: finding the most probable tag sequence for the sentence instead of individually disambiguating the morphological class of each token [14, 3, 9, 16]. Considering this, one could create a sentence- or a token-based combination system. A sentence-based solution would select the proper tagger for each sentence, while a token-based one does the same for each token. The former is a feasible method of combining MT (machine translation) systems, but for taggers, we opted for token-based combination.

4.1 Related Works

Creating and applying part-of-speech taggers has a long history starting from rule-based systems to applying machine learning methods. One of the first attempts of combining such methods was done by Brill and Wu [4]. They propose an instance based learning system for tagging a token that employs contextual clues such as the surrounding words and their suggested tags. Hajič et al. use a series of tagger with a rule-based approach [8] to combine disambiguators in order to improve overall tagging accuracy. A comprehensive study was presented by Halteren et al. [10] in which a detailed overview about previous combination attempts is given mainly using machine learning techniques. They also present several combination methods and systematically compare and evaluate them. For the optimal usage of the training corpus cross-validation is used to train the second-level classifier. All of these works conclude that the "combination of several different learning systems enables to raise the performance ceiling".

4.2 Voting

As a baseline system, we implemented a standard token–based unweighted voting scheme. As we saw in section 4, the errors of the three best-performing systems differ significantly, but the error distributions of the two methods in OpenNLP seem to be correlated. That is why we only took the three of the best performing systems for this stacking, namely: PurePos, HuLaPos and PE. The tags are calculated as follows:

1. tag the sentence with all systems,
2. for each token choose the tag that has the most votes ,
3. if there is no such, take the one proposed by PurePos.

Applying this scheme to the development set, it (Comb3) increases the overall accuracy to 98.18%, that is a 15.35% error reduction rate. Applying the same simple voting scheme to all of the four available taggers (Comb4) results in inferior performance compared to Comb3. The reason for this is that typical errors of the ME and PE systems co-occur and, and they can together win the vote with a wrong suggestion.

Compared to PurePos, this growth is significant (see table 3), but this combination yields just 24.26% of the hypothetical maximum error reduction rate.

Table 3 Comparing the accuracy of the simple voting scheme.

Tagger name	Precision
Comb3	98.18%
Comb4	98.10%
PurePos	97.85%
Max4	99.21%

4.3 Stacking

In our further combination experiments, we took into account the possibilities of stacking only the two best performing systems. As previously, we used the token based combination approach. A commonly used method in the area of stacking is to use a metalearner that is built upon various algorithms that solve the same task using significantly different methods. It learns which classifier to trust in various contexts, thus discovers the best way to combine their output. The models learnt by the actual systems applied on the task to be solved are usually called level-0 models, while the one that is learnt by the combiner is called level-1 model. This examination implies the following questions:

1. Using a fixed size corpus, what is the best way to use all the knowledge in the data?
2. What sort of combining algorithms perform the best?
3. Given a combiner, what is the most informative feature combination?

In data mining tasks, it is usual to use level-0 attributes for the level-1 classifier [17], but in our case that is hard to apply, since each tagger we use has a different kind of feature sets. We applied features that are commonly used in taggers, and added some more, that are specific for our task:

- the word to be tagged and words that precede or follow it
- guessed tags from PurePos and HuLaPos for the actual word, the previous word, the next word, the second previous word, the second word on the right,
- at most ten long suffixes of the word
- whether the word contains a hyphen,
- whether the word contains a dot,
- whether the word starts with an uppercase letter.

In the case of nominal attributes[2], an additional `<none>` value is needed to which nominal attributes of words never seen in the level-1 training data are mapped. We use the same solution for the output tag feature, because in the case of an agglutinating language like Hungarian, all elements of the morphological tag set (over 1000 different tags in our case) cannot be expected to be present in the training corpus. While the predicted level-1 class label generally could in theory be either the correct tag or the name of the preferred system, we chose the latter approach, since, due to the huge number of possible PoS tags, the training data for a system using the former approach would be extremely sparse.

David Wolpert, the inventor of stacking, proposed to use a “relatively global, smooth” level-1 learner, thus we investigated the following classifiers, which in addition to being simple, were shown to be able to handle nominal attributes: Naïve Bayes [5] , IB1 and IBk [1]. Since our features are clearly not independent, Naïve Bayes is not expected to perform very well, thus we used it as another baseline. The instance based methods fit to our model rather

[2] In data mining terminology nominal attributes are ones, that have fixed set of possible values.

well since, in the case of nominal attributes (and that is indeed what we have), the distance function is the square root of the number of attributes that are the same, thus providing a simple but efficient classification rule[3].

Training with Cross-Validation

For the best utilization of the corpus, we applied training with cross-validation. We split the training set into 5 equal sized parts and trained level-0 taggers (PurePos, HuLaPos) five times using 4/5 of the corpus, and the rest was annotated by both taggers in each round. The union of these annotated parts was used for training the level-1 classifier. Thus the full training data was available for level-1 training, yet separating the two phases of the training process. In addition, this workflow made it possible that level-0 taggers also be trained on the full training data in the end.

Table 4 Tagging precision of the combined tagger methods.

Combination	Precision
IBk, k=1	98.32%
IB1	98.30%
Naïve Bayes	98.26%

Evaluating the combination methods on the development set (table 4), we can state that the best results were obtained unsing instance based learning (IB), more specifically the algorithm called IBk in the WEKA framework[4] [7], with the k parameter set to 1. The aggregation of the two best performing classifiers results in a significantly better accuracy than the simple voting scheme of 3 or 4 systems, with the IB learning algorithms beating all the others as expected.

5 Evaluation

Table 5 Evaluating the tagger combinations on the test set.

Name	Precision
PurePos	97.89%
Morphologically augmented PurePos	**98.57%**
Simple voting of 3	98.19%
Combination with Naïve Bayes	98.28%
Combination with IB1	98.36%
Combination with IBk, k=1	**98.39%**
Maximal knowledge of Purepos and HuLaPos	98.86%

Choosing the best performing IBk combination, we compared (5. table) its performance on the held out test set with that of the baseline systems and the hypothetical best combination. While the simple voting scheme yielded only 30.93% and Naïve Bayes 40.21%, IBk reached

[3] Other machine learning algorithms – such as C4.5 – were also considered to be involved, but unfortunately they were not able to handle the large amount of data and the huge number of discrete features that were provided during the experiments.

[4] Weka is a collection of machine learning algorithms for data mining tasks.

51.55% of the hypothetical maximum improvement. The best performing IBk system produces only 14.18% more errors than the morphologically augmented PurePos system, which includes a language specific symbolic component. Our PoS combination results are in accordance with the observations made by Halteren et al. [10] that stacking performance is significantly better than voting. Because there is only a little overlap between the errors made by the taggers, we could achieve a significant error rate reduction by combining only two taggers.

We are not aware of any previous work exploiting the strengths of an SMT system, thus on of the achievements of this work is on discovering the possible supplementary usage of HuLaPos in such a task. Beside of this the feature set proposed by Brill [4] was also extended to be able to perform well with agglutinative languages such as Hungarian.

6 Conclusion

In this paper, we presented a combination of two PoS taggers that is able to decrease the error rate of the better tagger by 23.70%. One of this tools was a machine translation system, that were discovered to greatly complement the HMM one. While the combination uses a machine learning algorithm, we presented a way of using the whole training data for training the level-1 and level-0 models at the same time. The performance of the combined system approximates that of a morphologically augmented one, which heavily relies on language dependent linguistic knowledge. The presented method could be used as a language independent high precision PoS tagging tool. Since our results are promising we are planning to extend the investigation of our PoS combining technique to full morphological disambiguation[5] and other languages as well.

References

1 David W Aha, Dennis Kibler, and Marc K Albert. Instance-based learning algorithms. *Machine Learning*, 6(1):37–66, 1991.

2 Jason Baldridge, Thomas Morton, and Gann Bierner. The OpenNLP maximum entropy package, 2002.

3 Thorsten Brants. TnT - A Statistical Part-of-Speech Tagger. In *Proceedings of the sixth conference on Applied natural language processing*, number i, pages 224–231. Universitsität des Saarlandes, Computational Linguistics, Association for Computational Linguistics, 2000.

4 Eric Brill and Jun Wu. Classifier combination for improved lexical disambiguation. In *Proceedings of the 17th international conference on Computational linguistics-Volume 1*, pages 191–195. Association for Computational Linguistics, 1998.

5 William B. Cavnar and John M. Trenkle. N-Gram-Based Text Categorization. *Ann Arbor MI*, 48113(2):161–175, 1994.

6 Dóra Csendes, János Csirik, and Tibor Gyimóthy. The Szeged Corpus: A POS tagged and syntactically annotated Hungarian natural language corpus. In *Proceedings of the 5th International Workshop on Linguistically Interpreted Corpora LINC 2004 at The 20th International Conference on Computational Linguistics COLING 2004*, pages 19–23, 2004.

7 Eibe Frank, Mark Hall, Geoffrey Holmes, Richard Kirkby, Bernhard Pfahringer, Ian H Witten, and Len Trigg. Weka – a machine learning workbench for data mining. *Data Mining and Knowledge Discovery Handbook*, pages 1269–1277, 2010.

[5] There is also a need of high precision disambiguation between lemmas especially in the case of agglutinating languages.

8 Jan Hajič, Pavel Krbec, Pavel Květoň, Karel Oliva, and Vladimír Petkevič. Serial combination of rules and statistics: A case study in czech tagging. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, pages 268–275. Association for Computational Linguistics, 2001.

9 Péter Halácsy, András Kornai, and Csaba Oravecz. HunPos: an open source trigram tagger. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, pages 209–212, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

10 Hans Van Halteren, Jakub Zavrel, and Walter Daelemans. Improving Accuracy in Word Class Tagging through the Combination of Machine Learning Systems. *Computational Linguistics*, 27(2):199–229, 2001.

11 Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. Moses: Open source toolkit for statistical machine translation. In Annie Zaenen and Antal Van Den Bosch, editors, *Computational Linguistics*, volume 45 of *ACL '07*, pages 177–180. Association for Computational Linguistics, Association for Computational Linguistics, 2007.

12 László János Laki. Investigating the Possibilities of Using SMT for Text Annotation. In *SLATE 2012 - Symposium on Languages, Applications and Technologies*, pages 267–283, Braga, Portugal, 2012. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik.

13 Attila Novák. Milyen a jó humor? In *Magyar Számítógépes Nyelvészeti Konferencia 2003*, pages 138–145., Szeged, 2003.

14 György Orosz and Attila Novák. PurePos – an open source morphological disambiguator. In Bernadette Sharp and Michael Zock, editors, *Proceedings of the 9th International Workshop on Natural Language Processing and Cognitive Science*, pages 53–63, Wroclaw, 2012.

15 Gábor Prószéky and Attila Novák. Computational Morphologies for Small Uralic Languages. In *Inquiries into Words, Constraints and Contexts.*, pages 150–157, Stanford, California, 2005.

16 Adwait Ratnaparkhi. A maximum entropy model for part-of-speech tagging. In *Proceedings of the conference on empirical methods in natural language processing*, volume 1, pages 133–142, 1996.

17 Ian H. Witten, Eibe Frank, and Mark A. Hall. *Data Mining: Practical Machine Learning Tools and Techniques*. 3rd edition, 2011.