PurePos 2.0: a hybrid tool for morphological disambiguation

György Orosz and **Attila Novák**
Pázmány Péter Catholic University, Faculty of Information Technology
MTA-PPKE Natural Language Processing Group
50/a Práter street, Budapest, Hungary
{oroszgy, novak.attila}@itk.ppke.hu

Abstract

We present PurePos, an open-source HMM-based automatic morphological annotation tool. PurePos can perform tagging and lemmatization at the same time, it is very fast to train, with the possibility of easy integration of symbolic rule-based components into the annotation process that can be used to boost the accuracy of the tool. The hybrid approach implemented in PurePos is especially beneficial in the case of rich morphology, highly detailed annotation schemes and if a small amount of training data is available. Evaluation of the tool was on a Hungarian corpus revealed that its hybrid components significantly improve overall annotation accuracy.

1 Introduction

Part-of-speech tagging is one of the basic and commonly studied tasks of natural language processing. High accuracy of morphosyntactic annotation is crucial since tagging is usually part of a language processing pipeline, thus tagging errors propagate. Several PoS tagging tools have been created and made available during the years, however, PoS tagging is just the subtask of morphological annotation: in addition to the morphosyntactic tag, the lemma needs to be identified for each token. For morphologically not very rich languages, like English, a cascade of a tagger and a stemmer may yield an acceptable performance, but in the case of morphologically rich languages, incorporating morphological knowledge in the form of a morphological analyzer (MA) into the tagging process seems to be necessary not only to obtain high tagging accuracy but also to provide correct lemmata.

Sequence tagging tasks are often solved using statistical modelling techniques, since having a huge amount of annotated data, a decent method can learn important regularities, and applying this knowledge can yield highly accurate results. Smoothing techniques are commonly used in statistical natural language processing applications to alleviate problems caused by data sparseness. However, this prevents purely statistical models from being able to exclude events from the model that are known to be impossible to occur. Rule-based tools can find their niche here: one can either use rules to filter out agrammatical sequences, or ones that do not occur in a given domain. Hybrid methods combining statistical and rule-based approaches are getting more and more popular, since these are often able to yield a level of performance not attainable by either the statistical or the rule-based component alone.

In this paper, we describe the improvements that we made to an open source tool, PurePos, which, combining statistical models with symbolic and rule-based components, can generate highly accurate morphological annotation. Our paper is structured as follows. First, we motivate the model with annotation scenarios where a hybrid approach can be expected to perform significantly better than a purely statistical solution. Then the components of the tool are introduced. We describe the disambiguation process implemented in the tool, focusing on methods that enable us utilize the knowledge of the built-in MA and algorithms that we use to lemmatize words unknown to the MA. Finally, we evaluate our tool in a scenario where the annotation task involves a language with a very rich agglutinating and compounding morphology, an annotation scheme with very detailed distinctions and a rather modest amount of training data.

Proceedings of Recent Advances in Natural Language Processing, pages 539–545,
Hissar, Bulgaria, 7-13 September 2013.

2 The need of a hybrid annotation model

2.1 Agglutinating languages

If we compare an agglutinating language like Finnish with English in terms of the coverage of vocabulary by a corpus of a given size, we find that although there is a much higher number of different word forms in the Finnish corpus, these still cover a much smaller percentage of possible word forms of the lemmata in the corpus than in the case of English. Creutz et al. (2007) have compared the number of different word forms encountered in a corpus as a function of corpus size for English and agglutinating languages like Finnish, Estonian or Turkish. While a 10-million-word portion of their English newspaper corpus has less than 100,000 different word forms, a corpus of the same size for Finnish contains well over 800,000. On the other hand, however, while an open class English word has no more than 4–6 different forms, it has several hundred or thousand different productively suffixed forms in any of the agglutinating languages discussed in that paper. Moreover, there are much more different possible morphosyntactic tags in the case of agglutinating languages (corresponding to the different possible inflected forms) than in English (several thousand vs. a few dozen). Thus data sparseness is threefold:

- an overwhelming majority of possible word forms of lemmata occurring in the corpus is totally absent,
- word forms that do occur in the corpus have much less occurrences, and
- there are also much less examples of tag sequences, what is more, many tags may not occur in the corpus at all.

The identification of the correct lemma is not trivial either, especially in the cases of guessed lemmata. One such case from Hungarian is briefly discussed in (Orosz and Novák, 2012).

2.2 Resource-poor languages

A great proportion of resource-poor languages (that lack annotated corpora) is morphologically complex. To create an annotated corpus for these languages, an iterative workflow seems to be a feasible approach as it is proposed in (Orosz and Novák, 2012). First, a very small subset of the corpus is disambiguated manually, and the tagger is trained on this subset. Then another subset of the corpus is tagged automatically and corrected manually, yielding a new, bigger training corpus and this process is repeated. The higher the accuracy of the automatic annotation tool is, the less time human annotators need to spend manually correcting the results, and the less annotation errors are likely to remain in the resulting annotation.

2.3 Domain adaptation

Statistical models trained on a specific corpus, or even on balanced corpora, usually perform worse on texts from a different domain. The incorporation of symbolic morphological knowledge in the form of a high-coverage MA in the tagging procedure can successfully reduce the effect of domain differences. Miller et al. (2006) have shown that the incorporation of domain-specific lexical resources significantly improves performance. Such resources, however, can only increase accuracy in a consistent manner in the case of a morphologically rich language if the resource also covers suffixed forms of the domain-specific lexical items. Furthermore texts from a specific domain often have domain-specific syntactic and lexical patterns that can be made use of to gain accuracy.

Even in the case of ample training data, the tool may fail to generate correct annotation if the model implemented in it is not capable to capture some relevant generalization, e.g. a second-order HMM model may not capture long-distance agreement constraints, which results in random noise. In such a case, and for each of the use cases described above, applying additional linguistic constraints can improve accuracy. PurePos was made capable of incorporation of linguistic constraints and lexical knowledge both at its input and its output. It is capable of reading partially disambiguated input, where not only possible tags but their lexical probabilities can also be specified in the input for each individual token. In addition, it is capable of generating a k-best list of annotations with scores assigned to each annotated output, which can be used by either further parsing tools or machine learning systems.

3 Disambiguation model

The morphological annotation model performs lemma identification after determining the most

probable morphosyntactic tag for each word. In this section, we describe the tagging and lemmatization models implemented in PurePos.

3.1 PoS tagging model

Our aim was to build a system that is not only highly accurate but has a short training time as well. Fast turnaround time is e.g. needed in the iterative corpus creation scenario described above. In order to achieve high accuracy and fast training time, PurePos uses methods introduced in TnT (Brants, 2000) and HunPos (Halácsy et al., 2007). The tagging model is a linearly interpolated n-gram-based contextual model[1], and it uses unigram or bigram lexical models.

$$P(t_k|t_{k-1,k-n+1}) = \sum_{i=1}^{n} \lambda_i \hat{P}(t_k|t_{k-1,k-i+1}) \quad (1)$$

In (1), $\hat{P}$'s are maximum-likelihood conditional probability estimates of different left context sizes, while the interpolation parameters (λ_i) are calculated in a context-independent way using deleted estimation. This algorithm iteratively increases the score of a model weight if that is the most confident one for a trigram found in the training data. PurePos, like HunPos and TnT, maintains a separate lexical model for special tokens, and employs a guessing algorithm for determining the tags for previously unseen words. This guesser estimates PoS tag probabilities for unknown words based on the suffix distribution of rare words. For decoding, HunPos offers a slightly sped-up version of the Viterbi algorithm, which, while it gains on speed, loses a little accuracy. Besides keeping the Viterbi decoder, beam search was added to PurePos, which can be selected as an alternative decoding algorithm. When using beam search, the updated version of PurePos is capable of providing k-best output, outputting for each candidate annotated sequence its score, which is used for ranking candidates:

$$\begin{aligned} &Score(w_{1,m}, t_{1,m}) \\ &= log \prod_{i=1}^{m} P(w_k|t_k) P(t_k|t_{k-1}, ..., t_{k-n+1}) \end{aligned} \quad (2)$$

[1]The software is able to incorporate higher-order models as well, but in practice, a smoothed trigram model is generally used.

Employing morphological knowledge

In addition to statistical modelling, the tagger can incorporate knowledge provided by a morphological analyzer. In a previous version of PurePos, this could only be done through integration of a symbolic component using a Java API. The updated version is capable to read pre-analyzed text from the input, which means that any morphological analyzer can be used. If possible analyses are specified in the input for a token, tagging options as well as lemmas are restricted to the ones in the input for that token.

While the usage of morphological information might seem at first sight to be simple, there are several corner cases that need to be handled. First of all, a problem arises when the model is requested to assign a probability mass (either lexical or contextual) to an unseen tag. This occurs when an unseen tag is input to the system either as user input or by the integrated analyzer: in the default implementation, there is no way to calculate a lexical probability for this event. The same problem arises when a new morphosyntactic tag is included as a candidate analysis for a word that was seen in the training data but was never observed with that tag. These annotations were ignored by the original algorithm implemented in HunPos thus yielding obviously erroneous tagging.

Simple settings described above make it impossible to estimate probabilities for unknown tags, thus they get zero probability (and negative infinity as a score), which affects the whole tagging sequence making it unreliable.

It is also important to note that in case of tagging morphologically rich languages, the cardinality of the tag set usually exceeds one thousand, which results in data sparseness. This is especially problematic when the amount of the training data is low. Adaptation to new domains or tasks may also lead to the expansion of the tag set, which is difficult to handle with other existing tools.

We employ the following method to deal with problematic cases: if a token has only one (unseen) candidate analysis, that one is selected, and the lexical probability of the word-tag pair is assumed to be 1, while the contextual probabilities of forthcoming tags are taken from a lower level (unigram) model. When multiple candidates exist and at least one label is missing from the training data, PurePos is able to estimate lexical and contextual probabilities through mapping it to a pre-

viously seen morphosyntactic tag. For this, the user must setup a configuration file in which morphosyntactic label mapping rules can be formulated using regular expressions.

3.2 Lemmatization model

The updated implementation of PurePos contains a lemma identification process that selects the lemma candidate that has the maximal probability according to following conditional model:

$$\arg\max_{l} P(l|t,w) \tag{3}$$

I.e. the most probable lemma given the token and part-of-speech tag is selected. In practice, this probability is estimated in two ways. First, assuming that the lemmata are independent from words and tags, their probability can be estimated with unigram maximum likelihood estimates $\hat{P}(l)$, which are derived from relative frequencies. In addition, reformulating the core of (3), we get

$$P(l|t,w) = \frac{P(l,t|w)}{P(t|w)} \tag{4}$$

As the task is to select an optimal lemma for a fixed word and label pair, $P(t|w)$ is constant and can be ignored. The rest is approximated by using smoothed suffix models as described in (Brants, 2000). In order to efficiently store *(lemma, tag)* pairs, they are represented as suffix transformations that are to be performed to get the lemma from the word form in case of the given tag. This model is not only used for calculating probabilities but also employed for generating the lemma candidates. To utilize the strengths of both models, we use log-linear interpolation:

$$P(l|w,t) = P(l)^{\lambda_1} P(l,t|w)^{\lambda_2} \tag{5}$$

The idea of estimating the $\lambda_{1,2}$ parameters is similar to that used for the interpolation of PoS n-gram models (see section 3.1), but instead using positive weights, negative penalty scores are added to the parameter for the model performing poorly for a given *(word, PoS tag, lemma)* triplet (see algorithm 1).

Having the $\lambda_{1,2}$ parameters calculated, lemmatization is performed after morphosyntactic disambiguation. If there are full morphological analyses provided by the MA, then the lemmata provided by the analyses are taken as candidates, otherwise the lemma-guesser provides them. Finally, PurePos selects the candidate that satisfies (3).

3.3 Hybrid components

In addition to the exhaustive use of the morphological knowledge described above, PurePos provides facilities for users to incorporate extra lexical or grammatical knowledge through the input to the tagger. One can provide pre-analyzed input that not only contains full morphological analysis of tokens but contains lexical distribution data, which can be used to locally override lexical distributions in the model used by the tagger coming from the training corpus. This facility can be used e.g. to provide domain-specific lexical distribution information if the distribution of analyses for a given lexical item are markedly different in the given domain from that in the training corpus. The same facility can be used to filter out candidates agrammatical in the given context, e.g. capturing long-distance agreement constraints that the trigram tagging model cannot handle.

Using the built-in k-best search algorithm and the variable beam size, it is possible to generate output that is apt for post-processing. Advanced machine learning techniques and further parsing algorithms can also benefit from the k-best output format, since the disambiguation scores for sentences are also output.

4 Evaluation

In this section, we present a tagging task that we used as a test case to evaluate the methods described above. In a project aiming at the creation of an annotated corpus of Middle Hungarian texts (Novák et al., 2013), [2] an adapted version of Hungarian HuMor (Prószéky and Novák, 2005; Prószéky, 1994) morphological analyzer was used. This tool was originally made to annotate contemporary Hungarian, but the grammar and lexicon were modified so that the tool can handle morphological constructions that existed in Middle Hungarian but have since disappeared from the language. In the experiments described here, we used a manually checked disambiguated portion of this corpus. The data was annotated using a rich variant of the HuMor tagset, the cardinality of which is over a thousand.

In order to simulate this annotation task, we split the corpus into three parts (Table 1). The tagger was trained on the biggest part, hybridization and adaptation methods were developed on a

[2] Historical corpus of informal language use [OTKA 81189]

Algorithm 1 Calculating parameters of the linear interpolated lemmatization model

```
for all (word, tag, lemma) do
    candidates ← generateLemmaCandidates(word, tag)
    maxUnigramProb ← getMaxProb(candidates, word, tag, unigramModel)
    maxSuffixProb ← getMaxProb(candidates, word, tag, suffixModel)
    actUnigramProb ← getProb(word, tag, lemma, unigramModel)
    actSuffixProb ← getProb(word, tag, lemma, suffixModel)
    unigramProbDistance ← maxUnigramProb − actUnigramProb
    suffixProbDistance ← maxSuffixProb − actSuffixProb
    if unigramProbDistance > suffixProbDistance then
        λ2 ← λ2 + unigramProbDistance − suffixProbDistance
    else
        λ1 ← λ1 + suffixProbDistance − unigramProbDistance
    end if
    normalize(λ1, λ2)
end for
```

Table 1: Characteristics of the used corpus

	Training	Dev.	Test
Documents	140	20	30
Clauses	12355	2731	2484
Tokens	59926	12656	11763

separate development subcorpus, while final evaluation was done on a test set. We used accuracy as a metric, with unambiguous punctuation tokens *not* taken into account (in contrast to how taggers are evaluated in general). The results were evaluated in a threefold way: PoS tagging accuracy and full morphological disambiguation accuracy were calculated for tokens, and the latter was also calculated to obtain a clause-level accuracy.

As baselines, we used the enhanced trigram-based algorithm derived from HunPos and implemented in PurePos (PP), while its combination with the HuMor analyzer (PPM) was also evaluated. As a lemmatization baseline, we used the unigram-based (UL) and the suffix-based model (SL) described in section 3.2. Performance of these systems is shown in Table 2. As the accuracy values indicate, suffix-based probability estimation could performed better when used together with a morphological analyzer, while when using no dedicated morphological component, the overall disambiguation accuracies applying either of the baseline lemmatization models were close to each other.

Basic lemmatization strategies can be improved through the model combination method described in Section 3.2. Results obtained by the combined approach are shown in Table 3. The presented algorithm yields an overall 3.2% relative error rate reduction compared to the best baseline (PPM+SL). The improvement is even more significant for in the case when a dedicated morphological analyzer is not used: the relative error rate reduction is 28.42% in this case (compared to PP+SL).

Table 2: Baseline disambiguation accuracies on the development set

	Tagging	Full	Clauses
PP+UL	93.20%	88.99%	55.58%
PP+SL	93.20%	89.01%	51.78%
PPM+UL	97.77%	97.22%	84.85%
PPM+SL	97.77%	97.50%	85.98%

To demonstrate the strengths of the hybrid PurePos, we present three models to enhance the performance of the tool. To that end, we utilized a development set to analyze common error types and to test hypotheses.

Table 3: Full disambiguation accuracies with the proposed lemmatization model measured on the development set

	Tokens	Clauses
Using a MA	97.58%	86.48%
Without a MA	92.14%	65.40%

Mapping tags

In contrast to other Hungarian annotation projects, the tag set used for annotating the historical cor-

pus distinguishes verb forms that have a verbal prefix from those that do not, because this is a distinction important for researchers interested in syntax.[3] This practically doubles the number of verb tags,[4] which results in data sparseness problems for the tagger. In case of a never encountered tag including a verbal prefix marking, mapping it to one without verbal prefix is a sensible solution since the distribution of prefixed and non-prefixed verbs largely overlap. Applying only this verbal mapping (TM), we could increase the clause level annotation accuracy to 86.53% that is 97.59% precision at token level.

Preprocessing

Another possible improvement is to employ rules that filter the input (FI). Exploiting the development set again, a preprocessing script was set up that employs five simple rules. Three of them catches frequent phrases such as *az a* 'that' in which *az* must be a pronoun. Another typical source of errors is the erroneous tagging or lemmatization of proper names that coincide with frequent common nouns or adjectives and the confusion of past participles as finite past verb forms. Implementing just a few rules for fixing these, we achieved 97.84% token accuracy and 86.77% clause accuracy on the development set.

k-best output

The k-best output of the tagger can either be used as a representation to apply upstream grammatical filters to or as candidates for alternative input to higher levels of processing. Five-best output for our test corpus has yielded an upper limit for attainable clause accuracy of 94.32%. While it is not directly comparable with the ones above, this feature could be successfully used also in self-training or in tagger combination schemes.

Applying the given hybridization steps to the test set, we can validate the performance improvements (see results in Table 4 [5]). Using 5-best output from the tagger, 92.30% of clauses have the golden annotation among the top 5 output.

Table 4: Disambiguation accuracies of the hybrid tool on the test set

	Tagging	Full	Clauses
Best baseline	96.72%	96.40%	80.52%
PurePos	96.72%	96.48%	80.95%
+TM	96.75%	96.51%	81.17%
+FI	96.83%	96.60%	81.55%
+FI +TM	96.87%	96.63%	81.77%

5 Conclusion

In this paper, we presented PurePos, an open-source full morphological annotation tool[6], which is based on simple and fast but effective models. The tagger is able to accommodate linguistic knowledge by using partially disambiguated input, including linguistic models that handle long-distance agreement constraints not covered by the core trigram HMM model. Its internal tag mapping interface can be used to handle problems caused by sparse tag data. Its data-driven lemmatization models are able to lemmatize words unseen in the training data and unknown to the morphological analyzer.

One can benefit from the usage of PurePos in cases of rich morphology, highly detailed annotation schemes or if a small amount of training data is available only. The possible application of linguistic knowledge makes it a feasible tool for rapid domain adaptation tasks as well.

Acknowledgement

Research reported in this paper was partially supported by the research project grants OTKA 81189, TÁMOP – 4.2.1.B – 11/2/KMR-2011-0002 and TÁMOP – 4.2.2/B – 10/1–2010–0014.

References

Thorsten Brants. 2000. TnT - A Statistical Part-of-Speech Tagger. In *Proceedings of the sixth conference on Applied Natural Language Processing*, pages 224–231.

Eric Brill. 1992. A simple rule-based part of speech tagger. In *Proceedings of the workshop on Speech and Natural Language*, pages 112–116.

[3] Hungarian verbal prefixes or particles behave similarly to separable verbal prefixes in most Germanic languages: they usually form a single orthographic word with the verb they modify, however, they are separated in certain syntactic constructions.

[4] 320 different verb tags occur in the corpus excluding verb prefix vs. no verb prefix distinction. This is just a fraction of the theoretically possible tags.

[5] Results in Tables 2 and 3 were obtained on the development set.

[6] `http://nlpg.itk.ppke.hu/software/purepos`

Mathias Creutz, Teemu Hirsimäki, Mikko Kurimo, Antti Puurula, Janne Pylkkönen, Vesa Siivola, Matti Varjokallio, Ebru Arisoy, Murat Saraçlar, and Andreas Stolcke. 2007. Morph-based speech recognition and modeling of out-of-vocabulary words across languages. *ACM Transactions on Speech and Language Processing (TSLP)*, 5(1):3.

Jan Hajič, Pavel Krbec, Květoň, Karel Oliva, and Vladimír Petkevič. 2001. 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.

Péter Halácsy, András Kornai, and Csaba Oravecz. 2007. HunPos: an open source trigram tagger. In *Proceedings of the 45th Annual Meeting of the ACL*, pages 209–212, Prague, Czech Republic.

Mans Hulden and Jerid Francom. 2012. Boosting statistical tagger accuracy with simple rule-based grammars. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey.

John E Miller, Michael Bloodgood, Manabu Torii, and K Vijay-Shanker. 2006. Rapid adaptation of pos tagging for domain specific uses. In *Proceedings of the HLT-NAACL BioNLP Workshop on Linking Natural Language and Biology*, pages 118–119.

Attila Novák, György Orosz, and Nóra Wenszky. 2013. Morphological annotation of old and middle hungarian corpora. In *Proceedings of the 7th Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities*, pages 43–48, Sofia, Bulgaria.

Csaba Oravecz and Péter Dienes. 2002. Efficient Stochastic Part-of-Speech Tagging for Hungarian. In *Third International Conference on Language Resources and Evaluation*, pages 710–717, Las Palmas, Spain.

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

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

Gábor Prószéky. 1994. Industrial applications of unification morphology. In *Proceedings of the fourth conference on Applied natural language processing -*, page 213, Morristown, NJ, USA.

Adwait Ratnaparkhi. 1996. A maximum entropy model for part-of-speech tagging. In *Proceedings of the conference on Empirical Methods in Natural Language Processing*, pages 133–142, Somerset, New Jersey.

Kristina Toutanova, Dan Klein, Christopher Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 173–180, Edmonton, Canada.