Combining Part-of-Speech Tagger and Inflectional Lexicon for Croatian

Željko Agić*, Marko Tadić, Zdravko Dovedan***

*Department of Information Sciences
**Department of Linguistics
Faculty of Humanities and Social Sciences
University of Zagreb
Ivana Lučića 3, HR-10000 Zagreb
{zeljko.agic, marko.tadic, zdravko.dovedan}@ffzg.hr

Abstract

This paper investigates several methods of combining output of a second order hidden Markov model part-of-speech/morphosyntactic tagger and a high-coverage inflectional lexicon for Croatian. Our primary motivation was to improve overall tagging accuracy of Croatian texts by using our newly-developed tagger. We also wanted to compare its tagging results – both standalone and utilizing the morphological lexicon – to the ones previously described in (Agić and Tadić, 2006), provided by the TnT statistical tagger which we used as a reference point having in mind that both implement the same tagging procedure. At the beginning we explain the basic idea behind the experiment, its motivation and importance from the perspective of processing the Croatian language. We also describe all the tools and language resources used in the experiment, including their operating paradigms and input and output format details that were of importance. With the basics presented, we describe in theory all the possible methods of combining these resources and tools with respect to their paradigm, input and production capabilities and then put these ideas to test using the F-measure evaluation framework. Results are then discussed in detail and conclusions and future work plans are presented.

1. Introduction

After obtaining satisfactory results of the preliminary experiment with applying a second order hidden Markov model part-of-speech/morphosyntactic tagging paradigm by using TnT tagger on Croatian texts – detailed description of the experiment given in (Agić and Tadić, 2006) and TnT tagger described in (Brants, 2000) – we decided to give it a try to reach a higher level of accuracy based on these results. (Please note that abbreviation HMM is used instead hidden Markov model and PoS/MSD tagging instead part-of-speech/morphosyntactic tagging further in the text).

In the section about our future work plans in (Agić and Tadić, 2006), we provided two main directions for further enhancements:

(i) producing new, larger and more comprehensive language resources, i.e. larger, more precisely annotated and systematically compiled corpora of Croatian texts, maybe with special emphasis on genre diversity and

(ii) developing our very own PoS/MSD tagger based on HMMs – being that TnT is available to public only as a black-box module – and then altering it by adding morphological cues about Croatian language or other rule-based modules.

We considered both courses of action as being equally important; HMM PoS/MSD trigram taggers make very few mistakes when trained on large and diverse corpora encompassing most of morphosyntactic descriptions for a language and, on the other hand, these taggers rarely seem to achieve 97-98% accuracy on PoS/MSD (excluding the tiered tagging approach by (Tufiş, 1999.) and (Tufiş, Dragomirescu, 2004)) without the help of rule-based modules, morphological cues or other enhancements which in fact turn statistical tagging systems into hybrid ones. Therefore, we have reasonably chosen to undertake both courses of action in order to create a robust version of Croatian PoS/MSD tagger that would be able to provide us automatically with new and well-annotated Croatian language resources.

However, knowing that manual production of MSD-tagged corpora takes time and human resources, we put an emphasis on developing and fine-tuning the trigram tagger in this experiment. Here we describe what is probably the most straightforward of fine-tuning options – combining the tagger and the Croatian morphological lexicon (HML), described in (Tadić and Fulgosi, 2003) and implemented in form of Croatian lemmatization server, described in (Tadić, 2006) and available online at hml.ffzg.hr.

Section 2 of the paper describes all the tools, language resources, annotation standards, input and output formats used in the experiment, while section 3 deals in theory with various methods of pairing trigram tagger and the before-mentioned morphological lexicon. Section 4 defines the evaluation framework that would finally provide us with results. Discussion and conclusions along with future plans are given in sections 5 and 6.

2. Tools, resources and standards

In this section, we give detailed insight on tools and resources used in the experiment, along with other facts of interest – basic characteristics of available annotated corpora and input-output file format standard used.

2.1. Lemmatization

At the first stage of the experiment, we had available the Croatian morphological lexicon in two forms – one was the generator of Croatian inflectional word-forms described in (Tadić, 1994) and another was the Croatian lemmatization server (Tadić, 2006). As it can be verified at hml.ffzg.hr, the server takes as input a UTF-8 encoded verticalized file. File verticalization is required because the server reads each file line as a single token which is used as a query in lemma and MSD lookup. Output is provided in form of a text file and an equivalent HTML browser output.

Therefore, a text document could be obtained from the server containing all (lemma, token, MSD) triples and a computer program or a programming library could be created and used in our experiment as a black-box module. In order to do so, we obtained the TMT library, described in (Šilić et al., 2007) that had implemented a very fast and efficient dictionary module based on finite state automata, storing triples of wordforms, lemmas and tags into an incrementally constructed data structure. The TMT dictionary module has therefore provided us with the needed C++ object-oriented interface that we could use to get e.g. all lemmas and MSDs for a token, all MSDs for a (token, lemma) pair etc. A working lemmatization interface was now at our disposal and it could be used both as an input-output black-box and as a rule-based module to be integrated with the second-order HMM tagging paradigm at runtime.

2.2. PoS/MSD-tagging

The Croatian statistical PoS/MSD-tagger (CPT from this point on) was developed and made available as an early beta-version for purposes of validation in this experiment. Although many statistical taggers are now available in the community for scientific purposes – for example, the TnT tagger (Brants, 2000) and the HunPos tagger (Halácsy et al., 2007), a completely open-source reimplementation of TnT – and could be utilized in our research scheme, we chose to produce our own tagger so that we could alter its modus operandi by request and also be able to integrate it at will within other, larger natural language processing systems that are currently under development. CPT is written in standard C++ although some help from the HunPos development team and additional interpretation of the HunPos OCaml source itself was necessary.

At this moment, the tagger implements only a second order hidden Markov model tagging paradigm (trigram tagger), utilizing a modified version of the Viterbi algorithm (Thede and Harper, 1999), linear interpolation, successive abstraction and deleted interpolation as smoothing and default unknown word handling paradigms which are de facto standard methods, found in both TnT and HunPos. CPT presumes token emission upon reached state and is trained as a visible Markov model (VMM), i.e. on pre-tagged corpora, from which it acquires transition and emission probability matrices, as described in e.g. (Manning and Schütze, 1999).

Input and output formats of CPT are once again virtually identical to ones of TnT and HunPos; the training procedure takes a verticalized, sentence delimited corpus and creates the language model – i.e. the probability matrices – and the tagging procedure itself takes as input a verticalized, sentence delimited, non-tagged text and before mentioned language model matrices, providing an output formatted as is required of the training input: a verticalized text containing a token and MSD per line.

Since our tagger is still in beta-version, these procedures do not offer any possibility of setting the parameters to the user although the implementation of these options is taken into account. The further planned work on CPT beta is discussed in section 6 together with other possible research directions.

2.3. Annotated corpora

The Croatia Weekly 100 Kw newspaper corpus (CW100 corpus further in the text) consists of articles extracted from seven issues of the Croatia Weekly newspaper, which has been published from 1998 to 2000 by the Croatian Institute for Information and Culture. This 100 Kw corpus is a part of Croatian side of the Croatian-English parallel corpus described in detail in (Tadić, 2000). The CW100 corpus was manually tagged using the MULTEXT-East version 3 morphosyntactic specifications detailed in (Erjavec, 2004) and encoded using XCES standard (Ide et al., 2000). The corpus consists of 118529 tokens, 103161 of them being actual wordforms in 4626 sentences, tagged by 896 different MSD tags. Nouns make for a majority of corpus wordforms (30.45%), followed by verbs (14.53%) and adjectives (12.06%), which is in fact a predictable distribution for a newspaper corpus.

PoS	% corpus	Diff. MSD
Noun	30.45%	119
Verb	14.53%	62
Adjective	12.06%	284
Adposition	9.55%	9
Conjunction	6.98%	3
Pronoun	6.16%	312
Other	20.27%	107

Table 2.1 PoS distribution on the corpus

Details are provided in Table 2.1. Please note that the PoS category Other includes acronyms, punctuation, numerals etc. A more detailed insight on the CW100 corpus facts and preprocessing can be found in (Agić and Tadić, 2006).

3. Combining tagger and lemmatizer

Four different methods were considered while planning this experiment. They all shared the same preconditions for input and output file processing, as described in the previous section. We now describe in theory these methods of pairing our trigram tagger and morphological lexicon.

3.1. Tagger as a disambiguation module

The first idea is based on very high text coverage displayed by HML (more than 96.5% for newspaper texts) and the derived lemmatization interface: the text, consisting of one token per line to be tagged, could serve as lemmatizer input, the module providing as output all known MSDs given a wordform in each output line. The tagger would then be used only in context of its trained knowledge of tag sequence probabilities. A program module should therefore be derived from basic tagger function set – a module using its tag transition probabilities matrix to find the optimal tag sequence in the search space, now narrowed by using lemmatizer output instead of a generally poor lexical base acquired at training.

3.2. Lemmatizer as an unknown word handler

A second-order HMM tagger is largely dependent on its matrix of transition and emission probabilities, both of which are in our case obtained from previously annotated corpus by a training procedure. As mentioned before, both our tagger and TnT use visible Markov model training procedures. It is well-known that a large gap occurs when comparing PoS/MSD tagging accuracies on tokens known and unknown to the tagger in terms of the training procedure. If the training procedure encounters wordforms and discovers their respective tag distributions at training, error rates for tagging these words decrease substantially compared to tagging words that were not encountered at training. Improving trigram tagger accuracy therefore often means implementing an advanced method of guessing distributions of tags for unknown wordforms based on transition probabilities and other statistical methods, e.g. deleted interpolation, suffix tries and successive abstraction. Namely, TnT tagger implements all the methods listed above. However, TnT can never link a wordform to an unknown tag, i.e. a tag that was not previously acquired by the training procedure. We based our second method of pairing HML and CPT on that fact alone: it should be investigated whether HML – as a large base of wordforms and associated lemmas and MSDs – could serve as unknown word handling module for the tagger at runtime.

In more detail, the idea builds on (Halacsy et al., 2006) and (Halacsy et al., 2007) and is basically a simple logical extension of the unknown word handling paradigm using suffix tries and successive abstraction (Samuelsson, 1993). Trigram tagger such as TnT uses algorithms to disambiguate between tags in tag lists provided by emission probability matrix for a known wordform. Upon encountering an unseen wordform, such a list cannot be found in the matrix and must be constructed from another distribution, e.g. based on wordform suffixes and implemented in the suffix trie. Successive abstraction contributes by iteratively choosing a more general distribution, i.e. distribution for shorter suffixes until a distribution of tags is finally assigned. This results in large and consequently low-quality distributions for unknown wordforms, resulting in lower tagging accuracy. Taking high coverage of HML into consideration, idea was to choose from the suffix trie distribution only those MSDs on which both HML and trie intersect, falling back to suffix tries and successive abstraction alone when both lemmatizer and tagger fail to recognize the wordform. By this proposition, we utilize (wordform, tag) probabilities as given by the suffix trie and yet choose only meaningful (wordform, tag) pairs, i.e. pairs confirmed by applying the morphological lexicon.

3.3. Lemmatizer as a preprocessing module

In this method, we train a trigram tagger using the VMM training method and obtain matrices of transition and emission probabilities. The latter one, emission probability matrix, links each of the tokens found in the training corpus to its associated tags and counts, as is shown in Figure 3.1. The figure provides an insight on similarities and differences of storing language specific knowledge of tagger and lemmatizer.

It was obvious that lemmatizer and lexicons acquired by training share properties and therefore it was possible to create a lemmatizer-derived module for error detection and correction on the acquired lexicon used internally by the tagger. From another perspective, lemmatizer and acquired lexicon could also be merged into a single resource by a well-defined merging procedure.

```
%% ...
ime 26 Ncnsa 24 Ncnsn 2
imena 8 Ncnpa 1 Ncnpg 1 Ncnpn 3 Ncnsg 3
imenima 2 Ncnpd 1 Ncnpi 1
imenom 3 Ncnsi 3
imenovan 2 Vmps-smp 2
imenovana 1 Vmps-sfp 1
imenovanja 3 Ncnpg 2 Ncnsg 1
imenovanje 1 Ncnsv 1
imenovanjem 1 Ncnsi 1
imenovanju 4 Ncnsl 4
%% ...
```

```
%% ...
ime ime Ncnsa ime Ncnsn ime Ncnsv
imenima ime Ncnpd ime Ncnpi ime Ncnpl
imenom ime Ncnsi
%% ...
```

Figure 3.1 Emission probability matrix file and lemmatizer output comparison

3.4. Lemmatizer as a postprocessing module

Similar to using lemmatizer's language knowledge before tagging, it could also be used after tagging. Output of the tagger could then be examined in the following manner:

1. Input is provided both to tagger and lemmatizer, each of them giving an output.
2. The two outputs are then compared, leading to several possibilities and corresponding actions:
   a. Both tagger and lemmatizer give an answer. Lemmatizer gives an unambiguous answer identical to the one provided by the tagger. No action is required.
   b. Both tagger and lemmatizer give an answer. Lemmatizer gives an unambiguous answer and it is different from the one provided by the tagger. Action is required and we choose to believe the lemmatizer as a manually assembled and therefore preferred source of language specific knowledge.
   c. Both tagger and lemmatizer give an answer. Lemmatizer gives an ambiguous answer, i.e. a sequence of tags. One of the tags in the sequence is identical to taggers answer. We keep the tagger’s answer, being now confirmed by the lemmatizer.
   d. Both tagger and lemmatizer give an answer. Lemmatizer gives an ambiguous answer and none of the tags in the sequence matches the one provided by the tagger. A module should be written that takes into account the sequence provided by the lemmatizer and does re-tagging in a limited window of tokens in order to provide the correct answer. Basically, we define a window sized 3 tokens/tags and centered on the ambiguous token, lookup the most frequent of various trigram combinations available for the window (these are given by lemmatizer!) in transition probability matrix of the tagger and assign this trigram to the

window, disambiguating the output. By this we bypass tagger knowledge and once again choose to prefer lemmatizer output.
   e. Tagger provides an answer, but token is unknown to the lemmatizer. We keep the tagger's answer, this being the only possible course of action.
   f. Tagger does not provide an answer and lemmatizer does. If its answer is unambiguous, we assign it to the token. If it is ambiguous, we apply the procedure described in option 2d.
3. Final output produced by the merge is then investigated by the evaluation framework.

It should by all means be noted that each of the presented paradigms had to undergo a theoretical debate and possibly – if considered to be a reasonable course of action – a full sequence of tests described in section 4 in order to be accepted or rejected for introducing overall improvement of tagging accuracy or creating additional noise, respectively. Details are given in the following sections.

4. Evaluation method

As a testing paradigm, we chose the F-measure framework for evaluation on specific PoS and general accuracy for overall tagging performance. Firstly, we provide a comparison of CPT and TnT: overall PoS vs. MSD accuracy and also F-measures on nouns, pronouns and adjectives, proven to be the most difficult categories in (Agić and Tadić, 2006). We then discuss the proposed tagger-lexicon combinations and provide the measures – overall accuracy and F1-scores for those methods judged as suitable and meaningful at the time of conducting the investigation.

Each test consists of two parts: the worst-case scenario and the default scenario. Worst-case is a standard tagging accuracy measure scenario created by taking 90% of the CW100 corpus sentences for training and leaving the other 10% for testing; therefore, in a way, this scenario guarantees the highest number of unknown words to be found at runtime. The default scenario chooses 90% of sentences from the CW100 pool for training and then 10% for testing from the same pool, making it possible for sentences to overlap in these sets. The default scenario is by definition not a standard measure scenario and was introduced in order to respect the nature of random occurrences in languages, leaving a possibility (highly improbable) of tagger encountering identical sentences at training and at runtime.

Note that we do not include testing scenarios debating on training set size as a variable: in this test, we consider improving overall tagging accuracy and not investigating HMM tagging paradigm specifics as in (Agić and Tadić, 2006), being that conclusions on this specific topic were already provided by that test environment.

5. Results

The first set of results we present is from the set of tests evaluating overall tagging accuracy of CPT on full MULTEXT East v3 MSD and on PoS information (the first letter of the MSD tag, not comparable to English PoS of e.g. English Penn Treebank) only. Acquired results are displayed in Table 5.1.

		TnT		CPT	
		MSD	PoS	MSD	PoS
Worst case	Overall	86.05%	96.53%	86.05%	**96.84%**
	Known	89.05%	98.29%	**89.26%**	**98.42%**
	Unknown	**66.04%**	86.02%	65.95%	**87.29%**
	Corp. unk.	**13.07%**	14.40%	**13.77%**	14.11%
Default case	Overall	**97.54%**	98.51%	97.51%	**99.31%**
	Known	98.04%	98.74%	**98.05%**	**99.43%**
	Unknown	62.21%	83.11%	**63.75%**	**88.39%**
	Corp. unk.	**1.42%**	1.51%	1.59%	**1.13%**

Table 5.1 Overall tagging accuracy on MSD and PoS

It could be stated from this table that results on TnT and CPT are virtually identical and the differences exist merely because testing environment – mainly the number of unknown words – was variable. It is however quite apparent that CPT outperformed TnT on part-of-speech, especially regarding unknown tokens, but this should be taken with caution as well, being that CPT dealt with fewer unknowns in that specific test.

Second testing case considers combining CPT and the inflectional lexicon. Before presenting the results and in order to interpret them correctly, it should be stated that only two of the four initially proposed merging methods were chosen to proceed to the practical testing session: method (3.2) using the lemmatizer as an unknown world handler (3.4) using the lemmatizer as a postprocessing module to resolve potential errors produced by the tagger. We rejected applying (3.1) tagger as a disambiguation module for lemmatizer output because it would be costly to develop yet another tagger-derived procedure to handle transition probabilities only and because this procedure would, in fact, do nothing different than a common HMM-based tagger does with its own acquired lexicon: disambiguates its ambiguous entries upon encountering them in the text and applying the transition matrix and handling procedures on unknown words.

The idea of lemmatizer as preprocessing module (3.3) was also rejected, mainly because we were unable to define precisely how to merge its database to the one acquired by tagger at training procedure. Being that tagger assigns each entry with a number of its occurrences overall and number of occurrences under various MSDs and, in order to apply the lemmatizer, we would have to assign these numbers so the tagger could understand the new entries – if we assign all to 1, it does not contribute and is redundant and if we assign any other number, we are in fact altering the tagging procedure outcome in such a manner that is not in any way bound by the language model, i.e. the training corpus. Therefore, we proceed with considering proposed cases (3.2) and (3.4) only.

We have also omitted PoS results from this testing case because TnT and CPT are both able to achieve an accuracy over 95% without additional modules so we were focused in improving MSD accuracy, keeping in mind that most errors do not occur on PoS but on sub-PoS levels resolvable by the lexicon. Details are provided by Table 5.2.

The first apparent conclusion is that method (3.4) that cleans up the errors on tagger output has failed and that it has failed on unknown words – where we could have expected it (or hoped for it) to perform better. The reason is, on the other hand, quite obvious: the tagger applies a tag to an unknown word using transition probabilities and smoothing procedures that are proven to operate quite satisfactory in TnT, HunPos and now CPT; when the postprocessing lemmatizer-based module encounters a word tagged as unknown – this word is rarely unambiguous on HML – therefore, a resolution module using transition probabilities had to be applied quite frequently and this module clearly and expectedly does not outperform default procedures (suffix tries, successive abstraction, deleted interpolation).

		TnT	CPT & 3.2	CPT & 3.4
Worst case	Overall	**86.05%**	85.58%	83.94%
	Known	**89.05%**	88.84%	88.18%
	Unknown	**66.04%**	65.13%	57.38%
	Corp. unk.	13.07%	13.77%	13.77%
Default case	Overall	97.54%	**97.97%**	97.88%
	Known	98.04%	**98.53%**	98.51%
	Unknown	62.21%	**63.49%**	59.40%
	Corp. unk.	01.42%	01.59%	01.59%

Table 5.2 Tagging accuracy with (3.2) unknown word handler and (3.4) postprocessing

Based on other, positive part of Table 5.2, we could end the section by stating that CPT, when combined with the HML-based lemmatizer in such a manner that the lemmatizer provides morphological cues to the tagger upon encountering unknown words, outperforms TnT by a narrow margin on the default MSD test case. However, a more sincere and exact statement – taking in regard all section 5 tables – would be that both TnT and CPT share the same functional dependency regarding the number of unknown words they encounter in the tagging procedure. That is, CPT outperforms TnT when less unknown tokens occur for him at runtime and vice versa, the lemmatizer contributing for around 1.3% improvement on unknown words. We could therefore argue that our beta-version of CPT tagger performs as well as TnT tagger – and that we have succeeded in implementing a state-of-the-art solution for tagging large-scale corpora of Croatian – given the test environment we had at hands, its drawbacks noted and hereby included.

In Table 5.3 we present results of evaluation broken down by three most difficult PoS categories: adjectives, nouns and pronouns. Data and analysis is given for PoS information only, as mentioned before.

		Adjective	Noun	Pronoun
TnT	Worst case	64.61%	82.10%	76.62%
	Default case	94.73%	96.89%	**97.11%**
CPT	Worst case	65.31%	80.85%	74.62%
	Default case	**95.86%**	**97.40%**	95.88%
CPT & 3.2	Worst case	**66.15%**	**81.71%**	**75.07%**
	Default case	95.06%	96.79%	95.82%

Table 5.3 Tagging accuracy with adjectives, nouns and pronouns

It can be clearly noticed that suggested combination mode (3.2) outperforms both TnT and CPT in the worst case scenario on all parts of speech since it has the support of HML when handling unknown words, that do occur frequently in this scenario. In the default case scenario, results are as expected, more even and inconclusive – default CPT actually outperforms lemmatizer combination because a unknown tokens were found in small numbers in the test sets, much too small for the lemmatizer to contribute significantly to overall tagging accuracy.

6. Conclusions and future work

In this contribution we have presented a beta-version of statistical PoS/MSD tagger for Croatian and proposed combining it with a large scale inflectional lexicon of Croatian, thus deriving a hybrid system for high-precision tagging of Croatian corpora. We have presented several possible types of combinations, tested and evaluated them using the F-measure evaluation framework. CPT provided results virtually identical to TnT – they differ only in hundredths of percentage in both directions in different evaluating conditions. This way we have shown that CPT functions at the level of state-of-the-art regarding HMM-based trigram tagging.

Our future directions for improvement of this system could and probably will fall into several different research pathways.

The first of them could be analyzing tagging accuracy on morphological (sub-part-of-speech) features in more detail and fine-tuning the tagger accordingly.

Various parameterization options could also be provided at tagger runtime. Such options could include parameters for unigram, bigram and trigram preference or implementing token emissions depending on previously encountered sequences (multiword unit dependencies).

Fine-tuned rule-based modules for Croatian language specifics could also be considered and applied before or after the statistical procedure. Another option would be the integration of lemmatizer into tagger as they have been programmed as separate modules.

The next direction would be to build a full lemmatizer which, unlike solution presented in this paper, gives a fully disambiguated output relying on the results of the tagger. Selection of proper lemmas from sets of possible ones would be done on the basis of tagger output, once again fine-tuning levels of confidence between tagger and lemmatizer similar to section 3 of the paper.

It should also be noted that (Agić and Tadić, 2008) takes into account an entirely different approach, putting an emphasis on corpora development. Namely, all the methods presented in previous sections are made exclusively for handling unknown word occurrences and they all required lots of time and human effort to be implemented. On the other hand, manual corpora development – although also requiring time and effort – is by definition less demanding and at the same time quite reasonable course of action: larger, better and more diverse corpora are always a necessity – a necessity that implicitly resolves unknown word issues as well. Courses of action could therefore be argued; we decided to take both throughout our future work in order to additionally improve tagging accuracy.

7. Acknowledgements

This work has been supported by the Ministry of Science, Education and Sports, Republic of Croatia, under the grants No. 130-1300646-1776, 130-1300646-0645 and 036-1300646-1986.

8. References

Agić, Ž., Tadić, M. (2006). Evaluating Morphosyntactic Tagging of Croatian Texts. In Proceedings of the Fifth International Conference on Language Resources and Evaluation. ELRA, Genoa – Paris 2006.

Agić, Ž., Tadić, M. (2008). Investigating Language Independence in HMM PoS/MSD-Tagging. In Proceedings of the 30th International Conference on Information Technology Interfaces. Cavtat, Croatia, 2008, pp. 657-662.

Brants, T. (2000). TnT – A Statistical Part-of-Speech Tagger. In Proceedings of the Sixth Conference on Applied Natural Language Processing. Seattle, Washington 2000.

Erjavec, T. (2004). Multext-East Version 3: Multilingual Morphosyntactic Specifications, Lexicons and Corpora. In Proceedings of the Fourth International Conference on Language Resources and Evaluation. ELRA, Lisbon-Paris 2004, pp. 1535-1538.

Halácsy, P., Kornai, A., Oravecz, C. (2007). HunPos - an open source trigram tagger. In Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions. Association for Computational Linguistics, Prague, Czech Republic, pp. 209-212.

Halácsy, P., Kornai, A., Oravecz, C., Trón, V., Varga, D. (2006). Using a morphological analyzer in high precision POS tagging of Hungarian. In Proceedings of 5th Conference on Language Resources and Evaluation (LREC). ELRA, pp. 2245-2248.

Ide, N., Bonhomme, P., Romary, L., (2000). An XML-based Encoding Standard for Linguistic Corpora. In Proceedings of the Second International Conference on Language Resources and Evaluation, pp. 825-830. (see also at http://www.xces.org).

Manning, C., Schütze, H. (1999). Foundations of Statistical Natural Language Processing, The MIT Press, 1999.

Samuelsson, C. (1993). Morphological tagging based entirely on Bayesian inference. 9th Nordic Conference on Computational Linguistics NODALIDA-93. Stockholm University, Stockholm, Sweden.

Šilić, A., Šarić, F., Dalbelo Bašić, B., Šnajder, J. (2007). TMT: Object-Oriented Text Classification Library. Proceedings of the 29th International Conference on Information Technology Interfaces. SRCE, Zagreb, 2007. pp. 559-566.

Tadić, M. (1994). Računalna obrada morfologije hrvatskoga književnog jezika. Doctoral thesis. Faculty of Humanities and Social Sciences, University of Zagreb, 1994.

Tadić, M. (2000). Building the Croatian-English Parallel Corpus. In Proceedings of the Second International Conference on Language Resources and Evaluation. ELRA, Paris-Athens 2000, pp. 523-530.

Tadić, M., Fulgosi, S. (2003). Building the Croatian Morphological Lexicon. In Proceedings of the EACL2003 Workshop on Morphological Processing of Slavic Languages. Budapest 2003, ACL, pp. 41-46.

Tadić, M. (2006). Croatian Lemmatization Server. Formal Approaches to south Slavic and Balkan Languages. Bulgarian Academy of Sciences, Sofia, 2006. pp. 140-146.

Thede, S., Harper, M. (1999). A second-order Hidden Markov Model for part-of-speech tagging. In Proceedings of the 37th annual meeting of the Association for Computational Linguistics, pp. 175-182.

Tufiş, D. (1999). Tiered Tagging and Combined Classifiers. In F. Jelinek, E. Nöth (Eds.) Text, Speech and Dialogue, Lecture Notes in Artificial Intelligence 1692, Springer, 1999, pp. 28-33.

Tufiş, D., Dragomirescu, L. (2004). Tiered Tagging Revisited. In Proceedings of the 4th LREC Conference. Lisbon, Portugal, pp. 39-42.