MorphoDiTa-based Tagger Addapted to the Polish Language Technology

Maciej Piasecki, Wiktor Walentynowicz

G4.19 Research Group, Computational Intelligence Department
Wrocław University of Technology, Wrocław, Poland
maciej.piasecki@pwr.edu.pl, wiktor.walentynowicz@pwr.edu.pl

Abstract

We present a new morpho-syntactic tagger for Polish called MorphoDiTa-pl, which is based on the adaptation of the MorphoDiTa tagger developed originally for the Czech language. Following its basis, MorphoDiTa-pl utilises a rich feature averaged perceptron neural network for morphological analysis and morpho-syntactic disambiguation of the Polish language and produces results in the National Corpus of Polish (NCP) tagset. MorphoDiTa-pl performance represents the state of the art for Polish. However, contrary to other taggers, it is a complete, self-contained tool and relatively easy to be installed. Morphological analyser, an integral part of the tagger, was trained on automatically combined data from the SGJP dictionary of Polish (a basis for the Morfeusz 2 analyser) and also data acquired from the 1M subcorpus of NCP. The paper describes the process of creating a model for this solution, experiments and their results.

1. Introduction

Morpho-syntactic tagging of the Polish language has been an active research topic since at least the 2003 year in which a very interesting tagger for partial disambiguation of Polish (Rudolf, 2003) based on linguistic hand-crafted rules was published and also the first statistical tagger was constructed (Dębowski, 2003). Over the years a couple of taggers have been developed on the basis of different methods and approaches. However, in spite of all these efforts morpho-syntactic tagging of Polish seems to be still not completely solved problem. Firstly, taggers for Polish achieve significantly lower accuracy than English ones, but a tagset for Polish is typically 10 times larger than for English. However, the significantly lower accuracy results in more than one error per sentence on average. Secondly, many of the existing Polish taggers are difficult to be installed and not enough efficient for processing large volumes of text. Thirdly, there is a permanent problem over the years with the increasing discrepancy between the Polish best morphological analyser – Morfeusz 2.0 (Woliński, 2014) and the only Polish training corpus for taggers – 1-million word sub-corpus of the National Corpus of Polish (Przepiórkowski et al., 2012). A robust tagger must provide some means to effectively combine these two significantly incompatible Language Technology (LT) elements.

As the construction of a robust tagger as a language tool, appropriate for large scale applications, from scratch is quite laborious, we were looking for ready-to-use solutions, especially among the languages similar to Polish. The Czech language is not only similar to Polish, especially on the morphological and syntactic levels that are crucial for tagging, but also is supported by well developed LT, including robust tools for PoS tagging. Thus, our goal was to adapt a state-of-the-art tagger for Czech, namely *MorphoDiTa* (Straková et al., 2014) to Polish using minimal effort, i.e. a large scale, practical exercise in porting LT to another language.

In the rest of the paper, we will revisit the problem of incompatibilities among the basic elements of the Polish LT, discuss the conversion of tagsets, present briefly MorphoDiTa and its wrapper-like adaptation to Polish. Finally, we will present experiments and results.

2. Related Works

The first taggers were already mentioned: partial tagger of Rudolf (Rudolf, 2003) and *Trigram Tagger* (Dębowski, 2003; Dębowski, 2004). The former was based on a very interesting set of linguistically sophisticated rules for elimination of tags not matching the context. The rules were initially expressed formally in a way enabling direct implementation. Unfortunately, due to hard to understand reasons, the rules were finally published in an informal and transformed shape (Rudolf, 2003), so their real versions have been practically lost. *Trigram Tagger* (Dębowski, 2004) was a relatively simple statistical tagger, probably the first full tagger for Polish, that was trained and tested on IPI PAN Corpus (IPIC) (Przepiórkowski, 2004) (i.e. the first morpho-syntatically disambiguated corpus of Polish). *Trigram Tagger* achieved a reasonable accuracy, and was used initially to process IPIC, but has never become a practical language tool.

(Piasecki and Gaweł, 2005) presented a pattern-based tagger (i.e. applied in a memory-based learning scheme) in which patterns were extracted from IPIC by a genetic algorithm. It presented lower accuracy than the first two.

TaKIPI (Piasecki, 2007), whose development started in late 2005, became the first publicly available and more widely used morpho-syntactic tagger for Polish. It was used for the final processing of IPIC and expressed accuracy measured on tokens close to 93%. A small percentage of tags are left non-disambiguated. TaKIPI works on the basis of a combination of a small number of rules (≈30) and a large number of decision trees automatically built for different *classes of ambiguity* arranged into the subsequent layers of disambiguation[1], i.e. a paradigm later named tiered-tagging. The decision trees also include simple rules (constraints) providing complex information to the machine learning process. TaKIPI rules are expressed in the JOSKIPI language a (Piasecki, 2006) that was also used in several applications in the extraction of the linguistic knowledge from text. TaKIPI is written in C++, still in use, and its main limitation is tight connection to the IPIC

[1] On each layer selected tag parts are disambiguated: first grammatical class, next number and gender, finally case.

tagset and format (very unfortunately differing in small but significant details from NCP). TaKIPI uses an older version of Morfeusz (Woliński, 2014).

WCRFT (Radziszewski, 2013; Radziszewski, 2012) is a statistical tagger based on the Conditional Random Fields (CRF) machine learning algorithm. It was trained on the 1 million sub-corpus of NCP (1M-NCP), like all taggers discussed from this point on. WCRFT inherited from TaKIPI division of tagging into layers, calling overtly this scheme "tiered tagging", but used only simple features to describe the disambiguation contexts. For WCRFT a complex of several tools was build including: Corpus2 (Radziszewski and Śniatowski, 2012) – for reading corpora and *Maca* (Radziszewski and Śniatowski, 2011; Radziszewski and Śniatowski, 2011) – an expandable morphological analyser being a wrapper on Morfeusz (Woliński, 2014). Corpus2 and Maca can be set up for any tagset. Due to the complex structure, and the use of several libraries plus Morfeusz, WCRFT requires some efforts to be installed.

As WCRFT was not enough fast for processing very large volumes of texts, it was re-implemented in C++ and published as *WCRFT2* tagger (Radziszewski and Warzocha, 2014). It uses also a simplified model trained with a smaller number of features. WCRFT2 is the fastest tagger for Polish ever built, is available as a Web Service from CLARIN-PL[2], but expresses slightly lower accuracy than WCFT. Both WCRFT and WCRFT2 were trained with Morfeusz SJAT, so they express lower quality when used with Morfeusz 2.

Pantera[3] (Acedański, 2010) is a tagger developed for processing NCP. Pantera is based on a combination of an interesting adaptation of Brill's transformation learning to Polish and tiered tagging. It expresses accuracy close to WCRFT, uses Morfeusz, also for tokenisation, as an external module and is not the simplest system to install.

The CRF algorithm was also used for the construction of *Concraft* tagger (Waszczuk, 2012). CRF has been adapted to the problem and tuned in a sophisticated way. Concraft was written in less popular Haskell programming language, and uses Morfeusz for morphological analysis and tokenisation. Its accuracy is slightly better than WCRFT.

As the number of taggers was growing at least twice attempts were made to build an ensemble of taggers, i.e. combining several taggers run in parallel for final decision, e.g. (Śniatowski and Piasecki, 2011), (Kobyliński, 2014). In both cases the achieved results were better than any of the individual taggers combined in the ensemble. However, combining several taggers into a new one only increased problems with efficiency, installation and licenses.

All of these solutions require additional modules or external libraries, e.g. for prior additional tokenisation, in order to achieve compatibility between morphological analysis, the tagset in which they work, and the training corpus.

3. Morphological Analysis and Corpus

As it was earlier mentioned Morfeusz 2 and 1M-NCP express quit many disturbing discrepancies:

1. Morfeusz 2 uses a slightly different tagset than 1M-NCP,
2. beyond the tagset differences, the same tokens can be assigned more tags in Morfeusz 2.0, and *vice versa*,
3. disambiguated tags in 1M-NCP are not generated by Morfeusz 2 for the same words,
4. some differences between Morfeusz and 1M-NCP tags are caused by errors, but which ones?

As manual correction of 1M-NCP was beyond the scope of the work presented here, we decided to automatically merge the morphological information from 1M-NCP and Morfeusz 2.0 in a way preserving disambiguation in 1M-NCP, and next to train MorphoDiTa morphological analyser. In this way, a tagger will be able to use a very rich dictionary of Morfeusz 2.0, but for the cost of inevitable, but hopefully limited, errors in the training data and performance.

Merging process was done in the following way:

1. Word forms from the SGJP dictionary (Saloni et al., 2015) were processed by the Morfeusz 2.0 tool in order to obtain tags in the NCP format.
2. Word forms not present in 1M-NCP were immediately included into the morphological training data (MTD).
3. If for a word form in 1M-NCP some tags from SGJP were missing, we added them.
4. If a word form from 1M-NCP had some tags additional in comparison to Morfeusz 2.0, we simply removed them, unless a given tag was marked as a disambiguating one in any occurrence of the word form.

As a word form can have different sets of tags assigned across its occurrences in 1M-NCP, we merged its descriptions before the procedure was started. Next, 1M-NCP was also converted to match the compiled MTD.

The treatment of agglutinates appeared to be quite serious problem. MorphoDiTa performs its own tokenisation based on the assumption that character sequences between white characters are not split. However, in the NCP tagset even several word forms can be concatenated in one continuous sequence of characters. This happens in the case of combinations of verbs and agglutinates or participles. For instance, *chciałabym* is interpreted as consisting of *chciała* (finite verb) + *by* (participle – subjunctive meaning) + *m* (agglutinate – a form of the *być* 'to be' verb).

We found two solutions to this problem. Firstly, we can expand the NCP tagset by introducing additional grammatical classes and/or attributes in order to describe such concatenated sequences as single tokens. In this case all the work is next done by MorphoDiTa, and after tagging is completed, the assigned tags and tokenisation must be converted back to the NCP standard. As a result a kind of a wrapper on MorphoDiTa-based tagger needs only to be built. Secondly, we can use Morfeusz 2.0 as pre-tokeniser and train and apply MorphoDiTa on data prepared in this way.

[2] http://ws.clarin-pl.eu/tager.shtml

[3] `http://zil.ipipan.waw.pl/PANTERA`

4. Polish to Czech Tagset Conversion

In order to increase the tagger's performance we tried to closely map NCP tagset onto the Prague Dependency Treebank (PDT) tagset (Hajič and Hajičová, 1997) format. Both are positional, and the NCP tagset follows the Czech one to some extent that simplified the task. Moreover, the most important was to express the information from NCP tags in PDT format, so we had freedom in adding classes and new attribute values to PDT. MorphoDiTa adapts easily to the content of the tagset. We only tried to preserve the meaning of different positions in PDT tags, in order to stay close to the definitions of training vectors developed for MorphoDiTa.

To match the first position in PDT tagset, i.e. PoS, NCP grammatical classes had to be merged. For the second position – grammatical class – we could directly map 23 out 36 NCP grammatical classes to their similar Czech counterparts. For the rest of 11, we tried to select Czech classes of the same PoS. We added also three new classes to represent concatenated tokens.

Concerning grammatical categories, 6 (gender, number, case, person, grade and negation) could be directly mapped to PDT tag position, but in a different order inside the tag. NCP number and gender have smaller number of values than in the PDT tagset, and the Polish animal masculine gender was mapped just to the Czech masculine, due to the lack of other option. For the NCP categories missing in PDT we extended PDT tags with additional positions corresponding to the original NCP order.

In order to cope with the problem of representing word-internal tokenisation, e.g. *pseudo-participle* + *agglutinate*, discussed in Sec. 3., we added three additional grammatical classes to the tagset to signal 'compound' word forms. This allowed us to inform the tagger that after the completion of the process these forms should be broken down into smaller ones and the values of attributes should be transferred between them. Due to the fact that the forms of the grammatical class `winien` (verbs similar to English *should*) have all the necessary attributes to split it in a later process, it was not necessary to add PoS class in this case. Therefore, the tagset used internally in MorphoDiTa-pl tagger for Polish language has 39 grammatical classes.

5. MorphoDiTa-based for Polish

MorphoDiTa: Morphological Dictionary and Tagger (Straková et al., 2014) is an open-source tool for morphological analysis and morpho-syntactic disambiguation (tagging). Its construction is based on several modules, and allows for easy transfer and use without dependency on application of some additional software for specific subtasks. MorphoDiTa dissambiguates text in two steps:

1. all possible pairs of lemma and POS tag are identified for each token,

2. the most likely pair is selected on the basis of the context described by a set of defined features.

MorphoDiTa provides also an already built-in statistical guesser, which predicts both tags and lemmas for tokens not covered by the MTD. In most cases, the use of the guesser improves the results produced by the tagger.

MorphoDiTa is based on a rich feature averaged perceptron neural network and is trained on the basis of feature vectors describing contexts of ambiguous tokens in texts. A simple language of the feature representation was introduced. It enables simple operations like reading the word form, lemma or an attribute value on a position specified by offset, but also more complex operations like searching for the first noun to the left. The used feature representation (Hajič et al., 2009) has one very strong limitation – feature values must be atomic, so it is not possible to read attributes of tokens that has not been yet disambiguated, i.e. practically all to the right of the current position, e.g. it is not possible to obtain a set of possible cases from a noun following the current position, as it is the case in TaKIPI. The only exceptions are word forms, as they are non-ambiguous. As a result, several types of morphological agreement are not 'visible' to the tagger due to this limitation, e.g. adjectives occurring before nouns that is very frequent in Polish.

A definition of the feature vector must be provided in a form: one feature specification per line. The definition is next applied to every ambiguous token in the training data and a list of value vectors plus the token tags as the decisions is used for training the network. During the experiments we used two main sets of features:

1. a full set of features proposed for the Czech language in MorphoDiTa,

2. the set extended with features testing the value of the attribute *number* of the preceding tokens.

We used the vector for the Czech language because it has been developed as a result of numerous experiments on the Czech corpora, while Polish and Czech languages are similar. However, quite surprisingly this vector does not include any information about the *number* attribute.

In addition, we used three different versions of the above generic vectors, templates suggested for languages with rich inflection in MorphoDiTa documentation[4] that differ in the length of the context window: *generic2*, *generic3* and *generic4*: *generic2* has a window wide for one form in each direction, *generic3* for two forms, and *generic4* for three. The version of our tagger delivered to the evaluation in the PolEval contest is based on the *generic3* architecture.

6. Experiments and Results

In order to test the models we used a collection made available as part of the PolEval 2017 competition for Task 1(C). We compared the following models:

1. *PolEval* in Table 1 – a model that does not require an external tokeniser,

2. *Generic2-Generic4 with guesser* – models based on the Czech generic vector, (see Sec. 5.) and context windows of different length,

[4] `http://ufal.mff.cuni.cz/morphodita/users-manual#czech_morphological_system`

Model	Tags	Lemmas	Both
PolEval (baseline)	89.67%	95.78%	89.28%
PolEval with guesser	91.33%	97.10%	90.42%
Generic2 with guesser	90.71%	97.69%	90.19%
Generic3 with guesser	91.74%	97.81%	91.25%
Generic4 with guesser	91.75%	97.91%	91.25%
Generic3 extended vector with guesser	91.62%	97.83%	91.13%

Table 1: Results from the test set

Model	Time [s]
PolEval-baseline	27.512
PolEval with guesser	29.125
Generic2 with guesser	7.894
Generic3 with guesser	34.160
Generic4 with guesser	82.841
Generic3 extended w. guess	46.387

Table 2: Execution times

Task	Accuracy
Tag	95.75%
Lemma	97.80%
Tag + Lemma	95.03%
Only PoS tag	99.18%

Table 3: Czech state-of-art models performance

3. *Generic3 extended vector* – a model based on the extended generic vector (the number attribute added).

All models (with the exception of *PolEval-baseline*) use a statistical guesser, as well as Morfeusz 2.0 as an external tokeniser compatible with the NCP tagset.

The data set used in the learning process has been divided into a learning and validation sets – both composed of disjoint parts of the 1M-NCP. In Tab. 1 we present results of the taggers' performance on the test set. The values indicate, respectively: accuracy of tag disambiguation, accuracy of lemma disambiguation, percentage of correctly marked both tag & lemma pairs for one token. As the efficiency of a tagger is important for many practical applications, execution times in seconds are shown in Tab. 2.

Analysing these results, we can notice that models based on the *generic3* and *generic4* vectors are the best. The *generic3* model, being slightly weaker, is more than twice as fast as *generic4*. For tasks that do not require high precision, but high speed, the *generic2* vector model is best suited. The model that does not require an external tokeniser is doing about 1 percent worse than those with such a tokeniser. It is also faster than them. The model in which the set of features was extended achieved a worse result than the others.

An important reason for which we examined solutions that do not use an external tokeniser (i.e. Morfeusz 2.0) is the fact that we aim at providing a tool that can be easily installed and does not introduce troublesome external software dependencies. MorphoDiTa is a tool that only requires the installation of a standard C++ library and downloading an executable file from the repository with the model. For example, in the *Ubuntu 16.04* system, it is enough just to execute the command: *apt install libc6*, then to download the executable file with the model and one can start using MorphoDiTa. In addition, the model can be used in the original MorphoDiTa package. However, if one wants to use an external tokeniser, such as *Toki* (Radziszewski and Śniatowski, 2011), one needs to have his own compilation and to install multiple libraries in his operating system.

Only after, MTD has been created and taggers based on it have been evaluated on the test set, we could identify errors in MTD that have decreased the obtained results. First of all, errors in 1M-NCP are more frequent than one could expect, some are as significant as the use of a wrong lemma, e.g., prepositions *za* '≈behind,beyond' for *na* '≈on,for', not mentioning errors in tags or lower inter-annotator agreement on disambiguation of different types of ambiguity. For instance, there are errors in the value of the vocalicity for agglutinates occurring together with the word forms of the grammatical class `winny` (mostly words similar to *should*). However, what is the exact source of this errors requires deeper investigation. Unfortunately, due to the large size of 1M-NCP its manually correction was beyond the scope of the work presented here.

7. Conclusions

Despite significant development of tools for morphological analysis and morphological disambiguation for the Polish language, there is still a lack of robust taggers that would allow independent work from start to finish and do not require any other software or libraries. We have presented MorphoDiTa-pl, which is a morpho-syntactic tagger for the Polish language. It achieves the state-of-the-art performance, and is a technologically matured solution. MorphoDiTa-pl is available on open licence for download[5], but also as a high-performance Web Service[6]. In order to improve further the accuracy of the tagger, we need to correct errors in 1M-NCP, as well to merge the annotations with Morfeusz 2.0 in a semi-automated way. The work done by us was also an interesting exercise in adaptation of the LT for Czech language to Polish. A significant difference between the results reported for MorphoDiTa for the Czech language and ours for Polish needs closer investigation. As the MorphoDiTa algorithm expresses its natural limitations (e.g. the lack of the right context, too simple features, the lack of the tiered tagging model, relatively simple Machine Learning algorithm), it is worth to

[5] `git clone http://nlp.pwr.wroc.pl/morphodita-pl-poleval.git`

[6] `http://ws.clarin-pl.eu/morphoDiTa.shtml`

look for new solutions for tagging Polish on the basis of the experience collected.

Acknowledgment Works funded by the Polish Ministry of Science and Higher Education within CLARIN-PL Research Infrastructure.

8. References

Acedański, Szymon, 2010. A morphosyntactic Brill tagger for inflectional languages. In *Advances in Natural Language Processing*. Springer, pages 3–14.

Calzolari, Nicoletta, Khalid Choukri, Thierry Declerck, Hrafn Loftsson, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis (eds.), 2014. *Proceedings of the Ninth International Conference on Language Resources and Evaluation, LREC 2014*. Reykjavík, Iceland: ELRA.

Dębowski, Łukasz, 2003. A reconfigurable stochastic tagger for languages with complex tag structure. In *Proceedings of the Workshop on Morphological Processing of Slavic Languages. 10th Conf. of the European Chapter of Association for Computational Linguistics*. Budapest.

Dębowski, Łukasz, 2004. Trigram morphosyntactic tagger for Polish. In M. A. Kłopotek, S. T. Wierzchoń, and K. Trojanowski (eds.), *Intelligent Information Processing and Web Mining, Proceedings*. Zakopane, Poland: Springer.

Hajič, Jan and Eva Hajičová, 1997. Syntactic tagging in the Prague Dependency Treebank. In R. Marcinkeviciene and N. Volz (eds.), *Proceedings of the Second European Seminar "Language Applications for a Multilingual Europe"*. Kaunas, Lithuania: TELRI.

Hajič, Jan, Jan Raab, Miroslav Spousta, et al., 2009. Semi-supervised training for the averaged perceptron POS tagger. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics.

Kobyliński, Łukasz, 2014. PoliTa: A multitagger for Polish. In (Calzolari et al., 2014), pages 2949–2954.

Piasecki, Maciej, 2006. Hand-written and automatic rules for Polish tagger. In Petr Sojka, Ivan Kopeček, and Karel Pala (eds.), *Text, Speech and Dialogue, 9th International Conference, Brno, Czech Republic, Proceedings*, volume 4188 of *LNCS*. Springer.

Piasecki, Maciej, 2007. Polish tagger TaKIPI: Rule based construction and optimisation. *Task Quarterly*, 11(1–2):151–167.

Piasecki, Maciej and Bartłomiej Gaweł, 2005. A rule-based tagger for Polish based on Genetic Algorithm. In Mieczysław A. Kłopotek, Sławomir T. Wierzchoń, and Krzysztof Trojanowski (eds.), *Intelligent Information Processing and Web Mining, Proceedings, Gdańsk, Poland*, Advances in Soft Computing. Springer.

Przepiórkowski, Adam, 2004. *The IPI PAN Corpus: Preliminary version*. Institute of Computer Science, Polish Academy of Sciences.

Przepiórkowski, Adam, Mirosław Bańko, Rafał L. Górski, and Barbara Lewandowska-Tomaszczyk (eds.), 2012. *Narodowy Korpus Języka Polskiego [in Polish]*. Wydawnictwo Naukowe PWN.

Radziszewski, Adam, 2012. WCRFT. CLARIN-PL digital repository, `http://hdl.handle.net/11321/35`.

Radziszewski, Adam, 2013. A tiered CRF tagger for Polish. In R. Bembenik, Ł. Skonieczny, H. Rybiński, M. Kryszkiewicz, and M. Niezgódka (eds.), *Intelligent Tools for Building a Scientific Information Platform: Advanced Architectures and Solutions*. Springer Verlag.

Radziszewski, Adam and Tomasz Śniatowski, 2011. MACA. CLARIN-PL digital repository, `http://hdl.handle.net/11321/20`.

Radziszewski, Adam and Tomasz Śniatowski, 2011. Maca — a configurable tool to integrate Polish morphological data. In *Proceedings of the Second International Workshop on Free/Open-Source Rule-Based Machine Translation*.

Radziszewski, Adam and Tomasz Śniatowski, 2012. Corpus2. CLARIN-PL digital repository, `http://hdl.handle.net/11321/10`.

Radziszewski, Adam and Radosław Warzocha, 2014. WCRFT2. CLARIN-PL digital repository, `http://hdl.handle.net/11321/36`.

Radziszewski, Adam and Tomasz Śniatowski, 2011. Maca – a configurable tool to integrate Polish morphological data. In *Proceedings of FreeRBMT11*.

Rudolf, Michał, 2003. *Metody automatycznej analizy korpusu tekstów polskich: pozyskiwanie, wzbogacanie i przetwarzanie informacji lingwistycznych.*. Ph.D. thesis, Uniwersytet Warszawski.

Saloni, Zygmunt, Marcin Wolińskia, Robert Wołosz, Włodzimierz Gruszczyński, and Danuta Skowrońska, 2015. *Słownik gramatyczny języka polskiego. [Grammatical dictionary of Polish]*. SGJP, 3rd edition.

Śniatowski, Tomasz and Maciej Piasecki, 2011. Combining polish morphosyntactic taggers. In P. Bouvry, M. Kłopotek, F. Leprévost, M. Marciniak, A. Mykowiecka, and H. Rybiński (eds.), *Security and Intelligent Information Systems - International Joint Conferences, SIIS 2011, Warsaw, Poland, June 13-14, 2011, Revised Selected Papers*, volume 7053 of *LNCS*. Springer, pages 359–369.

Straková, Jana, Milan Straka, and Jan Hajič, 2014. Open-Source Tools for Morphology, Lemmatization, POS Tagging and Named Entity Recognition. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*. Baltimore, Maryland: Association for Computational Linguistics.

Waszczuk, Jakub, 2012. Harnessing the CRF complexity with domain-specific constraints. The case of morphosyntactic tagging of a highly inflected language. In *Proceedings of the 24th International Conference on Computational Linguistics (COLING 2012)*. Mumbai, India.

Woliński, Marcin, 2014. Morfeusz reloaded. In (Calzolari et al., 2014), pages 1106–1111.