Tagset conversion with decision trees

Bartosz Zaborowski[1] and Adam Przepiórkowski[1,2]

[1] Institute of Computer Science, Polish Academy of Sciences
[2] University of Warsaw

bartosz.zaborowski@ipipan.waw.pl
adamp@ipipan.waw.pl

Abstract. This paper addresses the problem of converting part of speech – or, more generally, morphosyntactic – annotations within a single language. Conversion between tagsets is a difficult task and, typically, it is either expensive (when performed manually) or inaccurate (lossy automatic conversion or re-tagging with classical taggers). A statistical method of annotation conversion is proposed here which achieves high accuracy, provided the source annotation is of high quality. The paper also presents an evaluation of an implementation of the converter when applied to a pair of Polish tagsets.

Keywords: morphosyntactic annotation, part of speech tagsets, decision trees

1 Introduction

Most annotated corpora use various types of tags to encode additional information along words. Depending on the language and requirements they can be just parts-of-speech (POS-tags) or they can contain more complex morphosyntactic information. The sets of tags used in various corpora usually differ. They quite often differ even for corpora of the same language. Unfortunately, various Natural Language Processing tools tend to be tied to specific tagsets, or, at least, they require some effort and high quality resources to be able to switch to some other tagset. For these reasons sometimes there is a need to convert a corpus from one tagset to another. Usually, the conversion has to be automatic, since manual (re-)annotation is expensive. As in the general tagging problem, high quality results are expected, which makes the task of tagset conversion difficult.

There are two usual approaches to the automatic conversion of a corpus from one tagset to another. The first, very common, is to use a state-of-the-art tagger, train it on a large corpus tagged with the target tagset and then apply it to the source corpus. When the source corpus is tagged manually, the method is lossy in the sense that it does not make any use of these initial gold standard tags. The second common approach is to manually write a set of rules converting particular tags. Potentially it can be a very accurate method. However, this accuracy is very expensive. Additionally, the more the tagsets differ, the more difficult it is to write the rules.

The rule-based approach was deeply explored by Daniel Zeman (e.g. [13]), who proposed a rule-based conversion method involving an interset, a common tagset for different tagsets and languages. This paper explores the statistical approach.

2 Definition of the task

Let us precisely define the task. Given a corpus tagged simultaneously with two different tagsets, the task is to train statistical conversion methods such that, given a word – or a segment[3] – and its context tagged with one tagset, the converter finds the best fitting tag from the other tagset. In this article the first tagset (the one which a given text is tagged with) will be called the *source tagset*, and the other – the *target tagset*. Similarly, a *source tag* for a given word will be a tag taken from the source tagset and a *target tag* for this word will be a tag from the target corpus for this word.

3 Baseline approach

The simplest tagger, frequently used as a bootstrap or a baseline, is the unigram tagger. For a slightly different task from tagging, namely tagset conversion, we can modify the unigram tagger to make use of the information derived from source tags. Instead of computing frequencies of tags for each word from the training corpus, the algorithm computes frequencies of target tags for each source tag. On the basis of this information, the baseline algorithm assigns to a given word the most frequent target tag for the source tag of this word. If the source tag does not appear in the training corpus, the most frequent target tag in the whole training corpus is assigned.

4 Improvements

This section describes and discusses possible ways to improve the correctness of the algorithm. All ideas were tested on a conversion from the IPIPAN Corpus tagset ([6]) to the National Corpus of Polish (NKJP) tagset ([10]).[4] These tagsets are compared in [5]; see also Appendix B. Experiments were performed on parts of the Enhanced Corpus of Frequency Dictionary of Contemporary Polish ([3]), which is manually annotated with tags from both tagsets. Unless stated differently, experiments were performed on a small part of the corpus,

[3] The term *word* is understood here as a maximal sequence of letters, digits and some punctuation marks (e.g., hyphens), i.e., "from space to space"; on the other hand, *segment* is the bit of text that's assigned a morphosyntactic tag. Usually the two are the same, but Polish tagsets assume that some words consist of a number of segments (compare with English *don't* sometimes split for tagging to *do* and *n't*).

[4] These tagsets are also described at `http://www.korpus.pl/en/cheatsheet/` and `http://nkjp.pl/poliqarp/help/en.html`.

consisting of about 30,000 segments from scientific texts. It will be called the *development* subcorpus. The experiments generally consisted in performing a 10-fold cross-validation on this subcorpus.

Most of the concepts described in this section can be adapted to various types of tagsets, not only positional (cf. section 4.3).The only requirement is that there exists a deterministic method to extract from tags different kinds of information represented in the source and target tagsets.

4.1 Choice of classifier

The baseline algorithm described in the previous section can be seen as a 1R classification algorithm ([2]) in the case when there is only a single attribute (the source tag) and a class attribute (the target tag) and thus the selection of the best attribute is trivial. The 1R algorithm gives quite good results for typical data. However, it is commonly known that there exist a number of more complex classification algorithms which achieve better results using more than one attribute. As a starting point for comparison between various classifying algorithms we decided to use a small set of attributes which intuitively may contain additional information, and thus, improve the performance. The selected attributes are: a positionally encoded source tag of the word immediately preceding the given word, a positionally encoded source tag of the word and a positionally encoded source tag of the word immediately following the given word. The class attribute remains the same (a plain target tag). In the case of the current tagsets we get a class and a set of $3 * 13$ attributes, most of which are $NULL$ (lack of value of grammatical categories of words which do not have those categories and do not inflect for them). The positional encoding and the context are discussed in more detail in the following two sections.

A number of experiments were performed using different classifiers from the WEKA ([11]) data-mining library. Due to performance reasons, a rough classifier selection was performed on a smaller subcorpus of approx. 5K segments (a part of the 30K development corpus).

The actual number of classifying algorithms tested amounted to 45 (all the available and applicable classifiers from WEKA, except for meta-classifiers), with over 400 different configurations. The best performing and simultaneously relatively fast four classifiers were: DecisionTable, PART, J48 and J48graft. They achieved from 89.0% to 90.2% of correctness in a reasonable computing time of at most a few minutes. After the second, more fine-grained comparison on the whole 30K development corpus, the J48 algorithm was chosen. Actually, since the J48 is a java implementation of the C4.5 ([7]) algorithm, and due to performance reasons, in later experiments we used an improved version of the original native implementation: the C5.0.

4.2 Context

The use of context is one of the most obvious improvements in tagging. It is also one of the most intuitive improvements with regard to the task of tagset

conversion. However, it is never known in advance how large the context should be for the best results on a particular language and tagset. Hence, we performed a number of experiments using various context sizes and independently changing the left context size and the right context size starting from zero (no context at all) up to 3 preceding/following segments. Like in the previous subsection, the only information available for the classifier were positionally encoded source tags for each word of the context. For the 30K development subcorpus results varied slightly from 91.9% to 92.4%. Not all context configurations improved the performance in comparison to the empty context which scored somewhere in the middle (92.1%). Larger right contexts tended to give worse results, as they probably introduced too much information noise. The best configuration found was: the right context of one segment and the left context consisting of three segments.

It is worth noting that some of the ideas described in the following sections interfere with the attributes used in these experiments. The final best configuration with all those changes is described in Section 4.7.

4.3 Positional tags

Both tagsets assumed here are positional. Since both contain a large number of tags, it is likely that not enough instances of each tag will be found to successfully train any statistical methods. Even in the relatively large corpus (>300K tokens) described in the evaluation section, not all of over one thousand possible source tags appear even once. It gets even more complicated when there is a need to extract rules from the context – there sometimes are thousands or even millions of bi-grams or trigrams to cover every possible combination of tags representing a simple relation between words. To overcome this problem, source tags are split into multiple attributes: one attribute for the grammatical class and a separate attribute for each of the grammatical categories represented in the tagset.

Surprisingly, this idea is only partially profitable. The positional encoding of tags for words in the context is clearly more profitable in terms of the correctness. However switching back to the full-tag for the currently tagged word (preserving positional encoding of the context) produces better results. Especially, when the context set to empty, positional encoding of the currently tagged word causes the converter to perform worse. It turns out that separating tags deprives the classifier of useful knowledge about relations between grammatical categories. Furthermore, we observed the same bad influence on performance when the target tags were split and each grammatical category was classified separately. This negative impact of positional encoding is more visible on the small 30K development subcorpus, but can be also observed on much larger data.

4.4 Retrieving information from the orthographic form and the lemma

The most useful information available to the converter is contained in the orthographic form of a word. Unfortunately, the statistical method does not allow to

use this information directly due to data sparsity. However, some parts of the orthographic information can still be extracted. For inflectional languages such as Polish a lot of the morphological information can be obtained by analyzing only a small prefix and suffix of the orthographic form. The same goes for the lemma of words (since the source corpus is annotated manually, it can be assumed that it is also lemmatized). Another useful type of information which can be easily extracted is whether the word starts with a capital letter or not. The first idea was to extract such prefixes, suffixes and the-first-letter-cases from the word and from each of the words in the context. After some tests we narrowed down the extraction of prefixes and suffixes to one word only (without the context). The gain from including prefixes and suffixes of words from the context was not clear (in some cases it decreased correctness). It seems that for corpora of sizes similar to the development corpus (and even for larger data) prefixes/suffixes introduce too much information noise and hardly ever point at useful information. Another finding was that there is no single best prefix size or suffix size for all words. The best results can be obtained by using a few classification attributes with extracted prefixes/suffixes of different length and leave the choice of length in particular cases to the classifier.

During the tests we found out that the best configuration for the conversion task is to include the-first-letter-case for a given word and each word from the context and to extract a single-letter prefix and one-, two- and three-letter suffixes of the orthographic form of the word. Including prefixes and suffixes of the lemma didn't seem to improve results of the experiments.

4.5 Error driven learning of additional attributes

In most languages there are words which are grammatical exceptions, hard to handle by general rules. As mentioned in the previous subsection, the orthographic form or the lemma cannot be used directly to distinguish such words and treat them separately because of limited resources. However, since only a small percentage of all words behave like this, we may treat differently only such words. This leads us to the question of how to find such words. The answer is simple: the conversion algorithm can be used to find a list of words whose tags are classified incorrectly. It is done by conducting a cross-validation of the converter on training data. Of course, the converter in such a case uses only those improvements which are described in the previous sections. The set of the most frequently appearing words from this list is a good approximation of the relevant set.

Note that the set constructed this way may also contain words whose tags are frequently incorrect because of their frequent adjacency to a grammatically exceptional word, even if this exceptional word is easy to tag (e.g., because it has just one interpretation in the lexicon). In order to force the classifier to take such situations into consideration, the classifier should memorize also the context for each of the words from the set. Then, for each position in the context, it should prepare a similar set of the most frequent words at this position.

Finally, after some experiments, the classifier was enlarged with a pair of attributes for each of the words in the context each signifying "the orthographic form / the lemma is the context of a special-treatment word X" or "neither orthographic form nor the lemma is an expected context of a special word". Of course, also a pair of attributes marking whether the given word is a "special treatment word" was added. Additionally, it appeared that it is also worth to store single-letter prefixes and suffixes of the lemmata of "special treatment" words in separate attributes. The optimal solution seem to be to use only the most frequent 1/3rd of the list of problematic words for preparing the attributes.

4.6 Using information from morphological analyzer

The algorithm we describe is guessing tags for all given words. Although by design it cannot produce illegal tags (according to the tagset), there are rare cases when the classifier selects a tag not possible for a given segment. If a comprehensive morphological analyzer using the target tagset is available, it can help the classifier to overcome this problem. In order to correct such cases, a frequency table of target tags is prepared on the training data. Then, when such a case occurs, the classifier result is replaced with a tag from the set of tags proposed by analyzer which has the highest frequency.

The impact of this modification certainly depends on the quality and the size of the dictionary used by the morphological analyzer. In our case the Morfeusz SGJP analyzer ([12, 9]) is used. It covers approximately 98.5% of the 30K development subcorpus and on this data only a slight improvement (about 1% reduction of the number of errors) was observed.

4.7 Fine-tuning of parameters

All the improvements proposed above interfere with each other. Although all of them give a correctness gain even when used together, the parameters found to be optimal for individual optimizations do not have to be optimal when all optimizations are applied at the same time. Hence, we performed another set of experiments to fine-tune various parameters. As opposed to the previous tests, here the evaluation was carried out on different sizes of corpora: small (5K tokens), medium (30K tokens) and large corpora (120K tokens). Like the development subcorpus, all of them are parts of the manually annotated Enhanced Corpus of Frequency Dictionary of Contemporary Polish. The small and medium corpora consisted of scientific texts, the large one contained also some news and fragments of plays/dramas.

For each corpus size, we performed a number of tests for slightly changed parameter values and observed for which parameters the best values change between corpora. As supposed, slight trends appeared for the context size and length of optimal suffixes, and more surprisingly, for the percentage of the most frequent problematic words used to calculate "special-treatment" attributes. All of them rose together with the corpus size. Finally, the best configuration of classifier attributes found was:

- left and right context sizes: 1 word
- different context for the "special treatment" attributes: 2 words left, 1 word right context; used for memorizing whole words (orthographic form or lemma)
- suffixes for the orthographic form: one-, two- and three-letter
- no prefixes for the orthographic form at all
- "special treatment": single-letter prefix and suffix for the lemma of a word, two-letter suffix of the orthographic form of the processed word.
- the most frequent half of the list of the problematic words used for preparing "special treatment" attributes

5 Influence of various improvements on performance

A set of experiments was performed to evaluate how the improvements described above interfere with each other. The experiments were performed on the development subcorpus (30k tokens) using 10-fold cross-validation. Table 1 shows min/max/average correctness computed on results of experiments with particular improvements enabled or disabled. It should give a rough indication of how useful each concept is. The detailed raw results of each of the experiments are shown in the appendix A. All the improvements and their parameters were in the final state, as described in Section 4.7.

improvement	**enabled?**	**correctness (%)**			**time and**
		avg	**max**	**min**	**memory used**
information from orth form	yes	94.88	95.18	94.43	61min, 206 MB
	no	93.48	95.17	90.89	21min, 138 MB
context usage	yes	94.26	95.18	91.11	52min, 180 MB
	no	94.11	95.12	90.89	30min, 163 MB
special treatment words	yes	95.04	95.18	94.72	45min, 216 MB
	no	93.32	94.93	90.89	37min, 128 MB
information from morphoanalyzer	yes	94.45	95.18	92.62	41min, 172 MB
	no	93.91	95.16	90.89	41min, 172 MB
positional encoding of source tags	context + word	94.13	95.11	90.89	39min, 151 MB
	context only	94.23	95.18	91.04	38min, 146 MB
	no	94.18	95.16	91.04	46min, 219 MB

Table 1. How useful each improvement is? Resources usage applies to the whole cross-validation (time is CPU time).

The results in Table 1 show that the biggest amount of valuable information for Polish comes from *special treatment* words (exceptions or so). The orthographic form is also very usable, but consumes significantly more resources. Surprisingly, neither the context nor the positional encoding is very important.

6 Evaluation

The final evaluation was performed on a large part of the Enhanced Corpus of Frequency Dictionary of Contemporary Polish (ECFDCP; [1, 4]) – on all available texts annotated manually both with the IPIPAN Corpus tagset and the National Corpus of Polish tagset which were aligned at the level of segmentation. It consisted of about 377,000 segments from scientific texts, news, essays and plays/dramas. During the evaluation, the target corpus was reanalyzed morphologically by means of Morfeusz analyzer (1.52% of segments were unknown to the analyzer). 56.1% of the tokens were ambiguous or had no interpretation, and the average number of ambiguous tags per token was 4.13 (including those for which there was no tag proposed by the analyzer). Due to similar tagsets, conversion of 90.64% of tags was trivial (renaming of values for grammatical class and corresponding categories) and approximately this level of correctness can be easily achieved by general, rule-based tagset converters such as DZ Interset ([13]). Unfortunately, there is no so called *driver* for the NKJP tagset for DZ Interset tool, therefore there is no possibility to directly compare this approach with our approach.

The other corpora appearing in Table 2 are parts of the ECFDCP described in Section 4.7. The evaluation was made by performing a 10-fold cross-validation.

algorithm	**corpus**	**correctness**			**resources used**
		all	**ambiguous**	**nontrivial**	
here	full 377k	96.12%	93.08%	58.54%	20h, 10.5GB of RAM
baseline	full 377k	92.42%	86.49%	19.00%	5min, 600MB of RAM
here	large 120k	95.41%	91.82%	50.95%	9h, 1.9GB of RAM
baseline	large 120k	92.85%	87.25%	23.59%	2min, 350MB of RAM
here	small 5k	94.08%	89.45%	36.74%	6min, 80MB of RAM
baseline	small 5k	92.83%	87.22%	23.38%	1min, 30MB of RAM

Table 2. Results of evaluation. Resources usage applies to the whole crossvalidation (time is CPU time).

As can be seen, the conversion approach gives quite high correctness even with a simple baseline algorithm. The baseline even for the small corpus reaches the correctness level reported for state-of-the-art taggers trained on much larger corpora of hundreds thousands tokens (e.g. PANTERA: 92.95%, WMBT: 93.00%, evaluated on 1-million corpus by [8][5]). The presented "here" approach achieves significantly higher correctness than state-of-the-art taggers. Furthermore, even when trained on the 30K development subcorpus and tested on the

[5] Those numbers are not strictly comparable with the results of our converter, since the mentioned taggers made use of a morphological analysis from the gold standard corpus during the evaluation.

rest of the full corpus (that is, 347K segments), this method gives 94.95% of correctness. It is still better than the correctness achieved by the abovementioned taggers trained on 1M corpus.

All the above "here" results were obtained using parameters tuned for the large corpus. The evaluation was performed on a computer with a 3.1GHz AMD FX processor running a 64-bit Linux and a Ruby interpreter (version 1.9.3). The baseline, as well as parts of the improved converter (data pre-/postprocessing), were implemented in the Ruby language; the C5.0 classifier used was an original native implementation. The speed and resource usage optimization was not a concern and it possibly could be up to 2 times better.

A list of most common errors is presented in Table 3. It clearly indicates that an inconsistency of the manual annotation or a different understanding of the same tags is one of the main causes of errors. Another information hard to guess by the converter are optional grammatical categories (see appendix B for details).

% of all errors	source tag	target tag (gold standard)	selected tag
2.14	conj	qub	conj
1.91	adv:pos	qub	adv:pos
1.73	qub	conj	qub
1.54	qub	qub	conj
0.99	qub	adv	qub
0.88	adv:pos	adv:pos	adv
0.82	conj	conj	qub
0.63	adj:pl:gen:m3:pos	adj:pl:gen:n:pos	adj:pl:gen:m3:pos
0.61	ger:sg:gen:n:perf:aff	subst:sg:gen:n	ger:sg:gen:n:perf:aff
0.60	subst:sg:nom:f	subst:sg:nom:m1	subst:sg:nom:f
0.58	qub	qub	adv
0.55	qub	qub	adv:pos
0.55	subst:sg:nom:n	subst:sg:acc:n	subst:sg:nom:n
0.53	qub	adv	subst:sg:nom:m3
0.51	subst:sg:gen:f	subst:pl:gen:f	subst:sg:gen:f
0.50	qub	qub	subst:sg:nom:m3
0.44	num:pl:nom:m3:rec	num:pl:nom:m3:congr	num:pl:nom:m3:rec
0.44	adj:pl:nom:m3:pos	adj:pl:nom:n:pos	adj:pl:nom:m3:pos
0.42	conj	adv	adv:pos
0.42	ppron3:sg:nom:f:pri	ppron12:sg:nom:f:pri	ppron12:sg:nom:m1:pri

Table 3. A list of most common errors from the evaluation on the full corpus, covering 17% of the total number of errors.

7 Conclusions

In this paper we showed that the task of tagset conversion – in the sense of re-tagging a corpus – can be significantly improved by using information taken from the previous manual annotation. We presented various types of information extractable from manually annotated corpus that can be used in the conversion process. A comparison of results of using each type of information revealed that the most valuable information is related to orthographic forms of words, but the source tags from the context are also useful for the algorithm. It is clear that these two types of information are partially redundant. However, using them together can further improve the correctness achieved by the converter.

We presented results which demonstrate an advantage of the current approach over using classical taggers for the task of tagset conversion. As opposed to classical taggers, this approach performs well even when there is only very little training data available for the target tagset.

References

1. Bień, J.S., Woliński, M.: Wzbogacony korpus *Słownika frekwencyjnego polszczyzny współczesnej*. In: Linde-Usiekniewicz, J. (ed.) Prace lingwistyczne dedykowane prof. Jadwidze Sambor, pp. 6–10. Uniwersytet Warszawski, Wydział Polonistyki (2003)
2. Holte, R.C.: Very simple classification rules perform well on most commonly used datasets. Machine Learning 11, 63–91 (1993)
3. Kurcz, I., Lewicki, A., Sambor, J., Szafran, K., Woronczak, J.: Słownik frekwencyjny polszczyzny współczesnej. Wydawnictwo Instytutu Języka Polskiego PAN, Cracow (1990)
4. Ogrodniczuk, M.: Nowa edycja wzbogaconego korpusu słownika frekwencyjnego. In: Gajda, S. (ed.) Językoznawstwo w Polsce. Stan i perspektywy, pp. 181–190. Komitet Językoznawstwa, Polska Akademia Nauk and Instytut Filologii Polskiej, Uniwersytet Opolski, Opole (2003), `http://www.mimuw.edu.pl/~jsbien/MO/JwP03/`
5. Przepiórkowski, A.: A comparison of two morphosyntactic tagsets of Polish. In: Koseska-Toszewa, V., Dimitrova, L., Roszko, R. (eds.) Representing Semantics in Digital Lexicography: Proceedings of MONDILEX Fourth Open Workshop. pp. 138–144. Warsaw (2009)
6. Przepiórkowski, A., Woliński, M.: The unbearable lightness of tagging: A case study in morphosyntactic tagging of Polish. In: Proceedings of the 4th International Workshop on Linguistically Interpreted Corpora (LINC-03), EACL 2003. pp. 109–116 (2003)
7. Quinlan, J.R.: C4.5 Programs for Machine Learning. Morgan Kaufmann, Los Alios, CA (1993)
8. Radziszewski, A., Acedański, S.: Taggers gonna tag: an argument against evaluating disambiguation capacities of morphosyntactic taggers. In: Text, Speech and Dialogue: 14th International Conference, TSD 2012, Brno, Czech Republic. Lecture Notes in Artificial Intelligence, Springer-Verlag (2012)
9. Saloni, Z., Gruszczyński, W., Woliński, M., Wołosz, R.: Słownik gramatyczny języka polskiego. Wiedza Powszechna, Warsaw (2007)

10. Szałkiewicz, Ł., Przepiórkowski, A.: Anotacja morfoskładniowa NKJP. In: Przepiórkowski, A., Bańko, M., Górski, R.L., Lewandowska-Tomaszczyk, B. (eds.) Narodowy Korpus Języka Polskiego. Wydawnictwo Naukowe PWN, Warsaw (2012)
11. Witten, I.H., Frank, E.: Data Mining: Practical machine learning tools and techniques. Morgan Kaufmann, San Francisco, 2nd edn. (2005), `http://www.cs.waikato.ac.nz/ml/weka/`
12. Woliński, M.: Morfeusz — a practical tool for the morphological analysis of Polish. In: Kłopotek, M.A., Wierzchoń, S.T., Trojanowski, K. (eds.) Intelligent Information Processing and Web Mining, pp. 503–512. Advances in Soft Computing, Springer-Verlag, Berlin (2006)
13. Zeman, D.: Reusable tagset conversion using tagset drivers. In: Proceedings of the Sixth International Conference on Language Resources and Evaluation, LREC 2008. ELRA, Marrakech (2008)

A Detailed evaluation results

			no orth form			orth form		
			no pos	pos word + ctx	pos ctx only	no pos	pos word + ctx	pos ctx only
special –	morph –	ctx –	91.04	90.89	91.04	94.43	94.45	94.43
		ctx +	91.11	91.25	91.40	94.56	94.50	94.56
	morph +	ctx –	92.79	92.62	92.79	94.74	94.74	94.74
		ctx +	92.85	93.02	93.16	94.85	94.84	94.93
special +	morph –	ctx –	94.91	94.72	94.91	95.12	95.03	95.12
		ctx +	94.99	94.99	95.03	95.16	95.10	95.13
	morph +	ctx –	95.00	94.77	95.00	95.12	95.01	95.12
		ctx +	95.06	95.08	95.17	95.14	95.11	95.18

Table 4. Results for the experiments with enabling different combinations of improvements. Obtained on the 30K development corpus, values in percents. Background color reflects the results (darker – worse). Improvements names: ctx – context, pos – positional encoding, special – special treatment words, morph – morphological analyzer, orth form – information extracted from orthographic form. See Section 5 for more details.

B Comparison of the IPIPAN and NKJP tagsets

The tagsets used in this article were designed for two large corpora of Polish: the IPIPAN Corpus of Polish and the National Corpus of Polish (NKJP). Both are positional, with quite a large number of possible morphosyntactic tags. The sets of grammatical categories and grammatical classes are similar between the tagsets. Hence, most of tags from the one tagset have a corresponding tag in the other tagset. In Tables 5 and 6 we present a comparison of grammatical categories and grammatical classes between tagsets with differences **highlighted**.

Some statistics: the number of all theoretically possible tags in the IPIPAN tagset is 4389, 1357 of them were proposed in the lexical analysis, the number of tags actually used in the annotation is 913. The number of all theoretically possible tags in the NKJP tagset is 4187, proposed in the lexical analysis: 1697, 792 of them were actually used in the corpus annotation.

grammatical category	values for IPIPAN tagset	values for NKJP tagset
number	sg, pl	sg, pl
case	nom, gen, dat, acc, inst, loc, voc	nom, gen, dat, acc, inst, loc, voc
gender	m1, m2, m3, f, n	m1, m2, m3, f, n
person	pri, sec, ter	pri, sec, ter
degree	pos, **comp**, sup	pos, **com**, sup
aspect	imperf, perf	imperf, perf
negation	aff, neg	aff, neg
accommodability	congr, rec	congr, rec
accentability	akc, nakc	akc, nakc
post-prepositionality	npraep, praep	npraep, praep
agglutination	agl, nagl	agl, nagl
vocalicity	nwok, wok	nwok, wok
fullstoppedness	**(not present)**	**pun, npun**

Table 5. Grammatical categories and their values in the two tagsets.

grammatical class	morphosyntactic characteristics	
	categories for IPIPAN tagset	categories for NKJP tagset
num	number, case, gender, [**accommodability**]	number, case, gender, **accommodability**
numcol	number, case, gender, [**accommodability**]	number, case, gender, **accommodability**
adjc	**(class not present)**	**(no categories)**
adv	**degree**	[**degree**]
xxs	**number, case, gender**	**(class not present)**
comp	**(class not present)**	**(no categories)**
brev	**(class not present)**	**fullstoppedness**
burk	**(class not present)**	**(no categories)**
interj	**(class not present)**	**(no categories)**

Table 6. Grammatical classes with their morphosyntactic characteristics in the two tagsets. Only differing classes are shown here; 28 classes have been omitted (see references cited in text for full lists). Categories listed in [ ] are optional, their values may be present or not. The rest is required.