Automatic parsing as an efficient pre-annotation tool for historical texts

Hanne Eckhoff
UiT The Arctic University of Norway
hanne.m.eckhoff@uit.no

Aleksandrs Berdičevskis
UiT The Arctic University of Norway
aleksandrs.berdicevskis@uit.no

Abstract

Historical treebanks tend to be manually annotated, which is not surprising, since state-of-the-art parsers are not accurate enough to ensure high-quality annotation for historical texts. We test whether automatic parsing can be an efficient pre-annotation tool for Old East Slavic texts. We use the TOROT treebank from the PROIEL treebank family. We convert the PROIEL format to the CONLL format and use MaltParser to create syntactic pre-annotation. Using the most conservative evaluation method, which takes into account PROIEL-specific features, MaltParser by itself yields 0.845 unlabelled attachment score, 0.779 labelled attachment score and 0.741 secondary dependency accuracy (note, though, that the test set comes from a relatively simple genre and contains rather short sentences). Experiments with human annotators show that preparsing, if limited to sentences where no changes to word or sentence boundaries are required, increases their annotation rate. For experienced annotators, the speed gain varies from 5.80% to 16.57%, for inexperienced annotators from 14.61% to 32.17% (using conservative estimates). There are no strong reliable differences in the annotation accuracy, which means that there is no reason to suspect that using preparsing might lower the final annotation quality.

1 Introduction

Parsing historical texts is a complicated venture. One challenge is high variation on all levels both across and within texts, in particular the absence of standardised spelling. Another is the small number of texts available in digital form, and the even smaller amount of annotated resources which could facilitate the development of new tools (Pettersson et al., 2012). Moreover, the overall amount of existing texts can be small too, which means that the gain achieved by developing highly specialised tools can be limited. In the meantime, historical linguists usually expect their corpora to have high-quality annotation, and tend to be less tolerant towards errors than computational linguists on average, which is probably reasonable, given the relatively small sizes of the corpora. With this in mind, it is not surprising that historical treebanks tend to be manually annotated (Piotrowski, 2012). One way to make use of automatic annotation would be to develop parsers that can handle historical texts (Schneider, 2012). Another would be use off-the-shelf tools for pre-annotation and then correct their output manually. In this paper, we test whether the latter approach is efficient for Old East Slavic (also known as Old Russian) texts. The idea to combine pre-annotation with subsequent manual correction is, of course, not at all new. It has been used, for instance, in the development of the TIGER treebank of German (Brants et al., 2002), the SynTagRus treebank of Russian (Apresjan et al., 2006) and ICEPAHC, the diachronic treebank of Icelandic (Rögnvaldsson et al., 2012). We are not, however, aware of any systematic evaluation of whether this routine is more efficient than a fully manual annotation with respect to historical texts.

In section 2, we describe the treebank we use for this purpose. In section 3, we outline the format conversions we have to perform and the technical details of our parsing experiments. In section 4, we describe a parsing experiment in an idealised setting, where the test set has manually corrected morphological annotation and lemmatisation. In section 5, we move to realistic experiments, where the parser

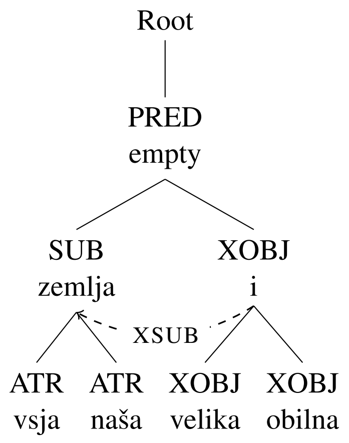

Figure 1: *vsja zemlja naša velika i obilna* 'all our land is great and bountiful' (Primary Chronicle 20.1–2, Codex Laurentianus)

has to deal with texts that have not been manually corrected. We present the results in section 6 and conclude in section 7.

2 The Tromsø Old Russian and OCS Treebank

The Tromsø Old Russian and OCS[1] Treebank (TOROT) is, to our knowledge, the only existing treebank of Old East Slavic and Middle Russian. It currently contains approximately 205,000 tokens equally divided between the two language stages.[2] The treebank belongs to a larger family of historical Indo-European treebanks originating in the project *Pragmatic Resources in Old Indo-European Languages* (PROIEL) at the University of Oslo, Norway, which built a parallel treebank consisting of the New Testament in its Greek original and translations into Latin, Gothic, Classical Armenian and Old Church Slavonic (OCS) (Haug et al., 2009). TOROT is an expansion of the OCS part of the PROIEL treebank.

All treebanks in the PROIEL family share an enriched dependency grammar scheme and a set of open-source online tools for annotation. The dependency scheme is inspired by Lexical-Functional Grammar, and is notable for allowing empty verb and conjunction nodes in ellipsis, gapping and asyndetic coordinations, and for the use of secondary dependencies to capture control and raising phenomena, shared arguments and predicate identity.[3] Figure 1 illustrates how empty tokens and secondary dependencies are used in a null copula construction.

The online annotation tool allows annotators to work online without any local installation of software. The tool guides the annotator through the annotation workflow: sentence division and tokenisation adjustment, lemmatisation and morphological annotation, and finally rule-guided manual dependency analysis. After the analysis, every sentence is reviewed, i.e. proofread by another project member.

The annotation tool allows import of texts with automatic pre-annotation. This is done as a matter of course for lemmatisation and morphological annotation in TOROT. In this paper we exploit the possibility to also import syntactic pre-annotation. It should be noted that the current annotation tool does not easily lend itself to merging sentences with syntactic pre-annotation. It is possible, but it requires a number of manual adjustments, which have been shown in pilot experiments to slow the annotators down to the extent that all speed gain is lost. In section 5, we therefore deal only with sentences that need no boundary adjustments.

[1]Old Church Slavonic

[2]TOROT can be browsed at `https://nestor.uit.no`, and versioned downloads may be obtained from `http://torottreebank.github.io/`. For a detailed description of the treebank, we refer the reader to Eckhoff and Berdičevskis (2015).

[3]See Haug et al. (2009). Full annotation guidelines and documentation are found at `http://folk.uio.no/daghaug/syntactic_guidelines.pdf`.

3 Parsing and format conversion

For the parsing experiments we used MaltParser (Nivre et al., 2007), version 1.9.0.[4] An earlier version of MaltParser was shown to be able to parse modern Russian with decent accuracy (82.3% labelled attachment score, Nivre et al. (2008)). For optimising the parser, we used MaltOptimizer (Ballesteros and Nivre, 2012), version 1.0.3.[5] For evaluating parsing results in the CONLL format (this section), we use MaltEval (Nilsson and Nivre, 2008), for evaluating full-fledged annotation in the PROIEL format we use a specialised script (see below).

In order to do the parsing, we convert the PROIEL format annotation to CONLL-X format (MaltParser 1.9.0 accepts the more convenient CONLL-U as input, but MaltOptimizer and MaltEval do not).[6] After parsing is done, the output of MaltParser is converted back to PROIEL. In converting PROIEL to CONLL, however, we lose information about empty nodes and secondary dependencies (see section 2). Technically, this information can be represented in the CONLL format (by adding empty tokens with dummy form, lemma and features and by using columns 9 and 10 for secondary dependencies), but MaltParser cannot reconstruct either empty tokens or secondary dependencies on its own, which means that we have to sacrifice them. Shared dependencies are simply removed, while empty tokens are weeded out from the trees according to a set of rules (designed to be simple and yet to facilitate subsequent restoration of empty tokens).

In general, coordination is represented in PROIEL using a version of Prague-style coordination (according to the classification in Popel et al. (2013)): All conjuncts are attached under the coordinating conjunction (empty conjunction if the coordination is asyndetic), the conjunction and the conjuncts have the same incoming relation, as demonstrated in figure 1. When an empty conjunction is removed during conversion to CONLL, the first conjunct is promoted into its place and gets a special temporary relation “coord” (absent from the PROIEL inventory). During back-conversion to PROIEL, all nodes which have the incoming relation “coord” are demoted, and empty conjunctions are restored, their original relations restored from other conjuncts. Note that the removal of empty conjunctions leads to asyndetic coordination being represented in CONLL using what Popel et al. (2013) call Stanford style (the first conjunct is the head and the remaining conjuncts and conjunctions are attached under it), the solution that is also used in Universal Dependencies.[7] It means that syndetic and asyndetic coordination are represented differently in CONLL, which can arguably worsen the MaltParser’s performance. It could potentially be amended by converting the syndetic coordination to the same style, but at the cost of, first, additional work, and second, potential loss of precision during back-conversion (in some cases it is not possible to restore the original structure and labels).

For empty verbs, the conversion rules depend on whether the verb has a predicative complement (“xobj”, see figure 1). If it does, the complement is promoted to the verb’s place, and all other dependents of the verb are attached under it. If it does not, all the dependents are placed directly under the node that was the head of the empty verb. In all cases, all the dependents preserve their incoming relation labels, i.e. we are losing information about the relation label that the empty verb had. When converting back to PROIEL, a number of simple heuristics are used to reinsert empty verbs, to make sure that the nodes placed directly under root conform to PROIEL requirements and to restore secondary dependencies for control and raising (“xsub”, see figure 1). No attempt is made to restore other types of secondary dependencies, though that is technically possible (Berdičevskis and Eckhoff, 2015).

Obviously, some accuracy is lost during conversion and back-conversion processes. One particularly important source of errors are structures where an empty token has another empty token as a head. Before we move on to the parsing experiments, we want to evaluate how large the inevitable loss is. Since the parsed output and the gold standard can potentially differ in the number of tokens (due to the presence/absence of empty tokens), this makes applying standard attachment scores impossible. Various evaluation methods that could be applied in such cases have been developed (Tsarfaty et al., 2011).

[4]http://www.maltparser.org/

[5]http://nil.fdi.ucm.es/maltoptimizer/install.html

[6]All data and scripts are found at http://dx.doi.org/10.18710/WIZHJN

[7]http://universaldependencies.org/

However, we use a simple script that was created specifically for PROIEL format and can handle both token misalignment and secondary dependencies.[8] Notably, this affects the LAS and UAS (labelled and unlabelled attachment score), since a missed or superfluous empty token counts as an attachment error.

The results of comparing the original manual annotation of the test set used in section 4 and the outcome of its conversion to CONLL and back to PROIEL are shown in table 1.

UAS	LAS	Empty token error	Secondary dependency accuracy
0.967	0.920	0.008	0.837

Table 1: Back-conversion accuracy, test set

The back-conversion is not perfect, but given that some loss of information is inevitable due to the richness of PROIEL format, we judge the final accuracy as acceptable. We use the same evaluation method for both parsing experiments, see sections 4 and 6. Note that the lossy conversion process contributes to making the PROIEL evaluator's UAS and LAS lower than those of MaltEval in section 4.

4 Parsing in an ideal world

Using the conversion process described in section 3, we first evaluate how well MaltParser can cope with Old East Slavic texts and PROIEL annotation in principle. We run the experiment in an idealised setting: For both training and test sets we use texts that have already been manually annotated and reviewed. That means that sentence division, tokenisation, lemmatisation and morphological annotation are as perfect as they can be, something which is never true for texts that have not been manually processed (see section 5). Obviously, the training sets also have high-quality syntactic annotation.

We use the 20160616 release of TOROT. As a test set, we take 20% of the sentences (randomly selected) from the Primary Chronicle, Laurentian Codex, one of the most important Old East Slavic manuscripts. The test set consists of 1453 sentences and 11105 tokens. We experiment with three training sets: LAV80, which consists of the remaining 80% of the Laurentian Codex (5816 sentences, 44174 tokens), KIEV, which comprises LAV80 and all other texts written at the same period (i.e. before 1400, during the so-called Kievan era; 8561 sentences, 76374 tokens), and ORV,[9] which comprises KIEV and all other Old East Slavic texts in the corpus (17661 sentences, 165104 tokens). We optimise the parser to each of the training sets. Following MaltOptimizer's recommendations, we use cross-validation for LAV80 and KIEV, but not ORV, since the ORV sample is large enough to make it unneccesary.

It can be expected that ORV will perform better than KIEV, while KIEV will perform better than LAV80 due to the differences in size. On the other hand, it is not impossible that genre will also matter. We expect LAV80 to be most similar to the test set, and ORV, which is most diverse as regards both genre and time of writing, to be least similar. The results show that size is the more important parameter, although the differences in performance are small. We estimate accuracy using both MaltEval on the CONLL format and the PROIEL evaluator on converted PROIEL xml, as shown in table 2. Note that the PROIEL evaluator yields considerably lower scores, both because it counts empty token errors as attachment errors, and because of the loss of accuracy in the conversion process (see section 3).

When preparsing texts for annotators to work on (see section 5), we use ORV as the training set.

5 Parsing in the real world

We are, however, interested in using parsing for practical purposes, i.e. for syntactic preprocessing to speed up the annotation process. Under these conditions, we are faced with a number of challenges.

[8]The script (found at `http://dx.doi.org/10.18710/WIZHJN`) compares gold and the compared analysis sentence by sentence, aligning the tokens of each. Since no retokenisations were allowed, all token number discrepancies were due to missed or superfluous empty tokens. In the PROIEL xml, empty tokens are always placed at the end, and are thus easy to align. If an empty token in either text had no match, it was aligned with nil. Such nil alignments are reported as "empty token errors". Note that since empty token errors also count as attachment errors, in this respect our UAS and LAS measures are stricter and more accurate than the ones obtained from MaltEval. "Secondary dependency accuracy" is the share of correctly attached secondary dependencies that are also correctly labelled and have the correct target, divided by the number of secondary dependencies in gold. LAS/UAS are calculated for primary dependencies only, not for secondary ones.

[9]orv is the ISO 639-2 code for Old Russian

Parse	Malt UAS	Malt LAS	UAS	LAS	Empty token error	Secondary dependency accuracy
LAV80	0.823	0.766	0.791	0.719	0.018	0.601
KIEV	0.831	0.775	0.797	0.727	0.018	0.622
ORV	0.839	0.786	0.804	0.734	0.018	0.627

Table 2: Parsing accuracy, test set

In order to preserve as much linguistic information as possible, we use manuscript-near text transcriptions with original mediaeval punctuation. Since we have a relatively large base of lemmatised and annotated forms, we are able to use a statistical tagger to provide morphological pre-annotation, as well as part-of-speech tags and lemma guesses. Old East Slavic displays particularly complex and linguistically interesting orthographic variation due to South Slavic influence, and it is not desirable to simply use a normalised text. The orthographic variation is, naturally, an impediment to statistical tagging. We solve the problem by normalising both the training data and the new text behind the scenes during the pre-annotation process, while the tokens stored in the treebank remain unnormalised (for a detailed description, see Berdičevskis et al. (2016)). However, the morphological and lemma/part-of-speech information we can provide is not good enough to use directly as linguistic data, and errors in the morphological preprocessing will necessarily cause problems in the parsing process as well. A single-feature morphological error, such as mistaking an accusative for a nominative, will typically produce label errors. A part-of-speech error, such as a verb misanalysed as a noun, can easily throw off the dependency structure of the whole sentence.

Sentence division is a larger problem. Texts are imported into TOROT with preliminary sentence division based on the original punctuation. Old East Slavic texts generally use syntactically motivated punctuation, but the punctuation usually indicates smaller syntactic units than the sentence. In the chronicle texts we are using in this paper, punctuation often separates subordinate clauses and participial constructions from the main clause, often quite neatly isolating a verb with its arguments. The conjuncts in a coordination are often also separated by punctuation, which is less convenient. A typical example is seen in (1), where the punctuation separates an adverbial participial construction from the main clause, and an adverbial PP from the participial construction. Within the main clause we see that the two coordinated complements of the preposition *s* 'with' are also split by punctuation.

(1) i poide s družinoju svoeju. i perejaslavci. vzem mltvu vъ stěm mixailě. u jepspa jeufimьja.
'and he went with his retinue. and the Pereyaslavians. having received prayer in St. Michael. from Bishop Jevfimij.' (The Suzdal' Chronicle, 6654)

Since we do not wish to use texts with editorial punctuation, and in many cases do not even have access to such punctuation, the final sentence division must necessarily be manual.[10] However, if the sentence division is done manually before syntactic preprocessing, we would be doing double work and would be likely to lose any speed gain.[11] Our syntactic preprocessing should therefore be performed before manual sentence boundary adjustment. This presents another problem (see section 2): if the annotators change boundaries of pre-parsed sentences, the effort required to save the existing trees will cost more than annotating sentences from scratch, as demonstrated in pilot experiments.

For the purposes of our experiment, we therefore selected a straightforwardly narrative passage from the Suzdal' Chronicle (year entries 6654, 6655, 6656, ms. Codex Laurentianus). We imported the text with preliminary sentence division according to the original punctuation, and manually selected only the sentences that did not need sentence boundary adjustment. Note that this limitation is a realistic one for our setting: The current version of the annotation tool will delete any preparsing if the annotator uses the in-built token and sentence boundary adjustment tools. The selected subset thus reflects the set of

[10]Automatic sentence division is not straightforward, since adverbial elements freely occur both before and after the main verb, making it difficult to automatically assess whether they belong to a preceding or subsequent verb.

[11]Note, however, that the developers of the diachronic Icelandic treebank ICEPAHC opted for exactly this solution: manual detection of clause boundaries before processing (Rögnvaldsson et al., 2012, 1980).

sentences that would retain preparsing in a real annotation setting.

We divided the passage into two approximately equal portions, hereafter referred to as Batch 1 and Batch 2 (see Table 4). Recall that it is impractical to change boundaries of pre-parsed sentences and attempt to save the trees. Due to the workflow organisation in TOROT, it is equally impractical to check the sentence division manually first and pre-parse later. A realistic solution is to pre-parse texts as they are, but spend no effort on preserving the trees if sentence boundaries have to be changed. However, in order to evaluate a potential speed gain, we run our experiment in a slightly artificial setting, i.e. limit the test set to the sentences that do not need boundary adjustment (as checked manually by the authors).

It should be noted that this experimental design systematically excludes longer and more complex sentences, which are known to be more difficult for the parser. The difference in sentence length between selected and non-selected sentences is given in table 3.

Batch	**Sentence status**	**Mean**	**Max**	**Min**	**Median**
batch 1	selected	4.9	10	2	5
batch 1	non-selected, unadjusted	4.9	10	2	5
batch 1	non-selected, adjusted	4.6	10	2	4
batch 2	selected	5.8	10	3	5
batch 2	non-selected, unadjusted	4.9	10	2	5
batch 2	non-selected, adjusted	10	21	2	9.5

Table 3: Sentence length in selected and non-selected sentences

Batch	**Entries**	**Selected sentences**	**Selected tokens**	**Lemma/POS**	**Morphology**
batch 1	6654, 6656	57	327	0.873	0.873
batch 2	6655	60	345	0.871	0.871

Table 4: Overview of selected sentences, accuracy of lemmatisation and morphological preprocessing.

The text batches were imported into the annotation web tool in four versions. In all four versions, the selected sentences were provided with automatic lemmatisation and part-of-speech tags with an accuracy of approx. 87%, as seen in table 4.[12] In two of the versions, the selected sentences were also preparsed as described in section 3. The annotators were presented with the full text entries, but used the presence of pre-annotation as an indication as to whether a sentence was to be annotated or not. They were also presented with a list of the ids of the selected sentences.

We selected four annotators for the experiment. Two of them, Volodimer and Olga,[13] had several years of experience with PROIEL-style annotation, and had annotated and reviewed around 200,000 tokens each. The other two, Rogned' and Lybed', were relatively inexperienced. Both had only annotated for a time span of a few months, and both had annotated a little over 2500 tokens each. Lybed' had been inactive for several months, Rogned' for several years. Thus, Volodimer and Olga had a perfect grasp of all technical aspects of the annotation tool, and very detailed knowledge of the annotation scheme. Lybed' and Rogned', on the other hand, had a good working knowledge of the annotation tool, but a much less detailed grasp of the annotation scheme. We expected Lybed' and Rogned' to benefit considerably more from the syntactic preprocessing.

The experienced annotators Volodimer and Olga received the following instructions:

- not to look at each other's annotation solutions
- to keep an accurate record of the time spent
- to annotate at their usual pace, but not spend time on major consistency checks

[12]Although the number of morphology and lemmatisation errors happen to be the same in both batches, they do not always occur in the same tokens – there are tokens with correct morphology but wrong lemmatisation and vice versa in both batches.

[13]We refer to the annotators by nicknames taken from the Primary Chronicle.

- to annotate *only* sentences with preprocessing and to use the sentence id list to go faster
- not to split, merge or retokenise, to accept the authors' sentence division and tokenisation judgements, even if they disagreed

Volodimer did the preprocessed text before the non-preprocessed one, while Olga did the non-preprocessed text before the preprocessed one. Rogned' and Lybed' were given the same instructions, but, unlike Volodimer and Olga, were both instructed to do the non-preprocessed text before the preprocessed one, thus deviating from the Latin square design. We made this change because both Rogned' and Lybed' were relatively inexperienced and had been away from annotation for a good while. They might therefore learn a lot from the first batch, which could improve their performance on the second batch. For the consequences of this design choice and handling the potential bias created by it, see section 6.

After the annotators had finished their annotation, their analyses were downloaded and saved, and the analyses in the production version of the text[14] were reviewed by the first author. This corrected version was also downloaded and saved, and serves as the gold standard in all of the comparisons in this paper.

6 Results: speed and quality

In the Suzdal' experiment, we see that all four annotators gained speed from working with preparsed sentences (table 6). The most inexperienced annotator, Rogned', had the greatest speed gain. To control for the potential bias caused by the fact that Rogned' and Lybed' were asked to do the the unparsed portion first, we also calculated separate annotation speeds for the first and second halves of each batch.[15].

	Unparsed 1	Unparsed 2	Parsed 1	Parsed 2
Rogned'	1.96	3.73	5.93	4.43
Lybed'	1.78	2.67	2.63	3.73

Table 5: Annotation speed gain, inexperienced annotators, tokens per minute

As seen in table 5, both did indeed increase their speed considerably from the first to the second half of the unparsed batch. We see that the benefits of preparsing are not easy to separate from the benefits of gained experience. However, it should be noted that Rogned', the least experienced annotator, does not gain speed between the first and second half of the preparsed batch, which suggests that most of her speed gain is due to the preparsing. In table 6 we report the gain from the second half of the unparsed batch to the whole preparsed batch as "conservative gain".

	Unparsed	Preparsed	Gain	Conservative gain
Olga	5.19	6.05	16.57%	–
Volodimer	3.45	3.63	5.80%	–
Rogned'	2.51	4.93	96.41%	32.17%
Lybed'	2.21	3.06	38.46%	14.61%

Table 6: Annotation speed, tokens per minute

The quality of parsing is reported in tables 7 and 8. We see that the quality of the raw parse is actually better than the "ideal-world" parse in section 4 (0.804 UAS, 0.734 LAS, with the same training set), even though we used imperfect automatic morphology and part-of-speech assignment/lemmatisation. This is probably due to our selection of text and sentences: The choice of a straightforward narrative passage means that the syntax is considerably simpler than it would be in e.g. a religious passage. The fact that we narrowed our selection down to only full sentences naturally delimited by the mediaeval punctuation effectively excluded all long sentences, which are more difficult for the parser.

[14]The non-preparsed version of Batch 1 and Batch 2 as analysed by Olga and Volodimer.

[15]Batch 1 part 1: year entry 6654 (174 tokens), Batch 1, part 2: year entry 6656 (153 tokens), Batch 2, part 1: the first 190 tokens of year entry 6655, Batch 2, part 2: the last 155 tokens of year entry 6655.

Status	Annotator	UAS	LAS	Empty token error	Secondary dependency accuracy
Raw parse		0.843	0.783	0.121	0.714
No preparsing	Olga	0.954	0.945	0.030	0.952
No preparsing	Rogned’	0.948	0.792	0	0.666
Preparsed	Volodimer	0.988	0.970	0	0.952
Preparsed	Lybed’	0.945	0.878	0	0.762

Table 7: Parsing accuracy, Batch 1

Status	Annotator	UAS	LAS	Empty token error	Secondary dependency accuracy
Raw parse		0.848	0.776	0.574	0.769
No preparsing	Volodimer	0.991	0.983	0	0.962
No preparsing	Lybed’	0.925	0.884	0	0.808
Preparsed	Olga	0.980	0.954	0	0.962
Preparsed	Rogned’	0.945	0.827	0	0.692

Table 8: Parsing accuracy, Batch 2

As expected, there is a great performance gap between the experienced and inexperienced annotators. All human annotators have better UAS than the parser, but the inexperienced annotators do not have much better LAS than the parser, and they sometimes perform worse when it comes to secondary dependencies. For Volodimer and Olga, there is no discernable quality gain or drop under the preparsed condition. For the inexperienced annotators, there are some differences: Lybed’ has a better UAS under the preparsed condition, and Rogned’ has a better LAS under the preparsed condition (but note again that, unlike Volodimer and Olga, they have become noticeably more experienced after having dealt with the unparsed condition).

7 Conclusions

While it is not possible, given state of the art, to create high-quality historical corpora using automatic parsing only, it is not unlikely that parsing can be an efficient pre-annotation tool. We examined whether this is the case using Old East Slavic texts. Using the most conservative evaluation method, which takes into account PROIEL-specific features, such as empty tokens, MaltParser yields 0.804 UAS, 0.734 LAS, 0.627 secondary dependency accuracy on an “ideal” test set and 0.845 UAS, 0.779 LAS, 0.741 secondary dependency accuracy on a “real” test set. Note that while the real test set does not boast perfect morphological annotation and lemmatisation, it comes from a relatively simple genre (narrative) and was limited to contain few long sentences, which probably explains the high performance.

Experiments with human annotators show that preparsing, if limited to sentences where no changes to word or sentence boundaries are required, increases their annotation rate. For experienced annotators, the speed gain varies from 5.80% to 16.57%, for inexperienced annotators from 14.61% to 32.17% (using conservative estimates). Since the current version of the annotation tool only allows us to retain preparsing in sentences that meet the test set conditions, this is a gain which would not be lost if we were to introduce preparsing as a routine procedure. There are no strong reliable differences in the annotation accuracy, which means that there is no reason to suspect that using preparsing might lower the annotation quality (if any effect can be expected, it is that of higher consistency).

From that, we can conclude that even given that historical texts are difficult to parse and that the current annotation interface of the TOROT is not well-suited for syntactic pre-annotation, parsing can still be used as an efficient pre-annotation tool.

References

Juri Apresjan, Igor Boguslavsky, Boris Iomdin, Leonid Iomdin, Andrei Sannikov, and Victor Sizov. 2006. A syntactically and semantically tagged corpus of Russian: State of the art and prospects. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation, LREC 2006, Genoa, Italy, May 22–28, 2006*, pages 1378–1381.

Miguel Ballesteros and Joakim Nivre. 2012. MaltOptimizer: A system for MaltParser optimization. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC)*, Istanbul.

Aleksandrs Berdičevskis and Hanne Eckhoff. 2015. Automatic identification of shared arguments in verbal coordinations. In *Computational Linguistics and Intellectual Technologies: Proceedings of the International Conference Dialogue 2015*, pages 33–43, Moscow.

Aleksandrs Berdičevskis, Hanne Eckhoff, and Tatiana Gavrilova. 2016. The beginning of a beautiful friendship: Rule-based and statistical analysis of Middle Russian. In *Computational Linguistics and Intellectual Technologies: Proceedings of the International Conference Dialogue 2016*, pages 99–111, Moscow.

Sabine Brants, Stefanie Dipper, Silvia Hansen, Wolfgang Lezius, and George Smith. 2002. The TIGER treebank. In *Proceedings of the Workshop on Treebanks and Linguistic Theories*, Sozopol.

Hanne Martine Eckhoff and Aleksandrs Berdičevskis. 2015. Linguistics vs. digital editions: The Tromsø Old Russian and OCS Treebank. *Scripta & e-Scripta*, 14–15:9–25.

Dag Trygve Truslew Haug, Marius Jøhndal, Hanne Martine Eckhoff, Eirik Welo, Mari Johanne Bordal Hertzenberg, and Angelika Müth. 2009. Computational and linguistic issues in designing a syntactically annotated parallel corpus of Indo-European languages. *Traitement Automatique des Langues*, 50.

Jens Nilsson and Joakim Nivre. 2008. MaltEval: an evaluation and visualization tool for dependency parsing. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Marrakech. http://www.lrec-conf.org/proceedings/lrec2008/.

Joakim Nivre, Johan Hall, Jens Nilsson, Atanas Chanev, Gülsen Eryigit, Sandra Kübler, Svetoslav Marinov, and Erwin Marsi. 2007. MaltParser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(2):95–135.

Joakim Nivre, Igor Boguslavsky, and Leonid L. Iomdin. 2008. Parsing the SynTagRus treebank of Russian. In Donia Scott and Hans Uszkoreit, editors, *Proceedings of the 22nd International Conference on Computational Linguistics (COLING)*, pages 641–648.

Eva Pettersson, Beáta Megyesi, and Joakim Nivre. 2012. Parsing the past – identification of verb constructions in historical text. Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities, European Association for Computational Linguistics, Avignon.

Michael Piotrowski. 2012. *Natural Language Processing for Historical Texts*. Morgan & Claypool, San Rafael.

Martin Popel, David Marecek, Jan Stepánek, Daniel Zeman, and Zdenek Zabokrtský. 2013. Coordination structures in dependency treebanks. In *ACL (1)*, pages 517–527. The Association for Computer Linguistics.

Eiríkur Rögnvaldsson, Anton Karl Ingason, Einar Freyr Sigurðsson, and Joel Wallenberg. 2012. The Icelandic Parsed Historical Corpus (IcePaHC). In *Proceedings of the Eighth International Conference on Language Resources and Evaluation, LREC 2012, Istanbul, Turkey, May 23–25, 2012*, pages 1977–1984.

Gerold Schneider. 2012. Adapting a parser to historical English. *Studies in Variation, Contacts and Change in English*, 10.

Reut Tsarfaty, Joakim Nivre, and Evelina Andersson. 2011. Evaluating dependency parsing: Robust and heuristics-free cross-annotation evaluation. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, EMNLP 2011, 27-31 July 2011*, pages 385–396.