11-712: NLP Lab Report

Maya Tydykov

April 25, 2014

1 Basic Information about Russian

The Russian language is an Indo-European language spoken primarily in Russia and in other parts of the world by approximately 162 million people. It belongs to the Eastern branch of the Slavic language family (Lewis et al., 2013). Russian is a free word order language, although according to (Dryer and Haspelmath, 2013), it is primarily Subject-Verb-Object (SVO). Russian has three genders and six cases, which are marked via suffixes on words. It is written using the Cyrillic script, which was originally created for 9th-10th century Slavic language speakers in order to translate the Bible along with other church texts ("Cyrillic alphabet").

2 Past Work on the Syntax of Russian

A wide range of phenomena concerning the syntax of Russian has been studied in recent years. (Franks, 2005) presents an overview of several issues which have recently been a focused on in the course of these studies. These issues include that of the second dative and nearest conjunct agreement phenomena in Russian. Another subject which has recieved a great deal of attention is that of numerals in Slavic languages. One issue in the domain of Russian syntax that has been studied extensively and that is particularly relevant to the problem of parsing is word order. There is a large body of work concerning Russian's free word order, which is referred to as scrambling in literature on syntax. An overview of the most influential works on the subject of free word order in Russian (as well as other Slavic languages) over the last several decades can be found in (Franks, 2005), and some recent work on this topic includes (Bailyn, 2008).

Russian word order has had a significant impact on the prefered methods of parsing Russian. The fact that Russian has free word order, as well as Russian's rich morphology, make it difficult to parse using a constituency framework because of the increase in the number of rules that would result from such an attempt. Thus, instead of constituency parsing, dependency parsing has been the standard method used for parsing Russian (Skatov et al., 2013). In 2012, the NLP Evaluation forum RU-EVAL held a Russian syntactic parsing evaluation task in which seven dependency parsing systems were evaluated. The purpose of the conference was to get an overview of the current state of the art in parsing for Russian. The top two systems at the conference (ranked by F1-score) were Compreno and ETAP-3, both of which use primarily rule-based approaches (Gareyshina et al., 2012). The third-place parser in the competition, SyntAutom, also a rule-based parser, is an automata-based system (Antonova and Misyurev, 2012; Gareyshina et al., 2012). According to (Skatov et al., 2013), none of the rule-based parsers evaluated as part of the task are openly available to the public. Another parsing method, implemented in the DictaScope Syntax system, which is itself incoporated into a commerical product, was recently described in (Skatov et al., 2013). This method combines constituency and dependency parsing, attempting to eliminate disadvantages of each.

Few dependency parsers for Russian have been made openly available to the public. One such system is the Russian Link Grammar parser, based on the Link Grammar formalism introduced in (Sleator and Temperley, 1993). This formalism is similar to the dependency structure formalism in that it focuses on creating links between words rather than on grouping words into constituencies. However, the Link Grammar formalism differs in that the links are undirected (i.e., there is no head or child word), links can form cycles, and there is no root word. Another system is Russian Malt, a machine learning system that does not incorporate any rules. This system achieved a score in the RU-EVAL task which would have put it into third place, but did not formally participate in the competition (Sharoff and Nivre, 2011; Gareyshina et al., 2012).

3 Available Resources

Building a supervised parser requires annotated training data. While some well-developed resources such as annotated corpora exist for Russian, to the best of the author's knowledge, few of them are openly available. One unannotated corpus that is freely available to use is the MultiUN corpus, consisting of cleaned data in XML format, extracted from the United Nations Website (Eisele and Chen, 2010). This corpus could be handy in developing a parser because it has already been preprocessed. One problem with using this corpus, however, is that its domain is limited. In order to develop an annotated corpus for for more general use (i.e., without focusing on one domain), one can use Wikipedia, since many articles are available in Russian. Specifically, there are currently 1,085,000 articles in Russian on Wikipedia (Wik, 2014). Using Wikipedia to develop a parser allows the freedom to develop a parser across domains or to choose some subset of domains to work on.

Preprocessing the corpus prior to annotation requires a tokenizer. One available tokenizer is the Lingua Ru OpenCorpora Tokenizer, which uses a probabilistic algorithm to do the tokenization (Surikov). Another tool that will be necessary is a part of speech tagger. One freely available part of speech tagger that can be used to tag Russian is the TreeTagger tool. This tool can be combined with parameter files from (Sharoff et al., 2008) in order to provide part of speech tags for Russian. TreeTagger uses a binary decision tree to estimate transition probabilities (Schmid, 1994). The parameter files used to train the TreeTagger for Russian were trained "on the disambiguated version of the Russian National Corpus" (Sharoff et al., 2008). There is a choice of two parameter files for part of speech tags - one using only basic part of speech tags, and another using more detailed part of speech tags. The detailed part of speech tags are composed of a general part of speech category as the first letter of the tag, and morphological information as the rest of the tag.

It is also useful to have a visualization tool to perform the annotation of the parse trees for training and testing data. One such tool is the DgAnnotator, which provides a simple interface and uses CoNLLX format for both input and output(Attardi).

The parsing itself can be done using an existing tool. One such tool is the TurboParser, a non-projective parser. This is ideal for languages such as Russian that have free word order. TurboParser incorporates third-order feature models and performs at speeds comparable to projective parsers (Martins et al., 2013). This parser is freely available and works with the standard CoNLLX format, making it easy to combine with other tools such as DgAnnotator.

Finally, in building a parser, it is helpful to include features beyond part of speech tags. Such features include lemmas and morphological information. The TreeTagger tool is capable of producing lemmas for the text it is tagging, and morphological features can be extracted from the detailed part of speech tags that are used with the TreeTagger tool (since, as mentioned above, the detailed part of speech tags are composed of a category followed by morphological information). Another source of lemmas and morphological information is the Pymorphy tool (Korobov), which is a rule-based analyzer that is based on the morphological dictionaries and work of (Sokriko and Toldova, 2006).

However, neither tool is an ideal source of these features. TreeTagger takes context into account when producing lemmas (because it looks at part of speech tags to do so), but is unable to produce lemmas for words it does not already know, and so results in many "unknown" type lemmas. Pymorphy has a predictor for unknown lemmas, but there is sometimes unresolved ambiguity when one attempts to retrieve a lemma or morphological information for a particular word, and there is no clear way of determining which of the multiple choices returned to use.

4 Survey of Phenomena in Russian

There has been much work on different phenomena in the Russian language. The "second dative" phenomenon in Russian concerns constructions with two special semipredicatives , where a semipredicative is "an adjective that makes an adjunct predication of some item in the sentence, auxiliary to the main subject-predicate relation(80 Greenberg and Franks, 1991). These two special semipredicatives are "odin", which means "alone", and "sam", which means "oneself". They differ from other semipredicatives in their declension and case marking and in that their case always agrees with an antecedent in the same clause that they appear in if the clause is simple. The antecedent with which they agree can be a subject or an object, whereas with normal semipredicatives, the antecedent they agree with must be a subject. Furthermore, when the semipredicatives do not agree with their antecedent, their case in these situations is dative, rather than instrumental, which is the non-agreeing case for other semipredicatives. The second dative appears in infinitive phrases where the subject of the infinitive is not in nominative case or it is in nominative case, but there is an overt complementizer between the infinitive phrase and the matrix clause (Greenberg and Franks, 1991). Several differing explanations have been proposed for the phenomena of the second dative; according to (Franks, 2005), one adequate explanation is provided by checking theory.

Another phenomenon is that of nearest conjunct agreement, in which the verb can agree with the conjunct nearest to it. This phenomenon is usually seen in sentences that start with a prepositional phrase and where the verb is not in accusative case and comes before the subject, which is conjoined. (Franks, 2005) mentions one possible explanation for this phenomenon could be "LF feature lowering".

The phenomenon of numerals in Russian involves the fact that the nominals quantified by numerals greater than "five" are in genitive case. However, when in an oblique phrase, the numeral and nominals are in oblique case.

A phenomenon which is particularly relevant to the choice of depedency parsing framework is Russian's free word order. Although Russian is free word order, as mentioned above, some orders are prefered over others in neutral situations. Specifically, the neutral ordering of a sentence is generally SVO. Adjectives and demonstratives usually come before the noun they modify, though the order can be changed for various reasons (Bivon, 1971). Adpositions in Russian come before the noun phrase they modify (prepositions)(Dryer and Haspelmath, 2013). In generative literature, the free word order phenomenon is frequently referred to as "scrambling"(Franks, 2005). (Bailyn, 2008) questions the existence of scrambling as a way of accounting for free word order and instead proposes that a certain syntactic processes can be used to explain it. He argues that Russian is "underlyingly SVO", and that most alternative orders come about from a syntactic "Generalized Inversion" process and from Dislocation, rather than from a generalized scrambling process.

The phenomena in Russian of the second dative, nearest conjunct agreement, numerals and case, and free word order are only a handful of a many more various phenomena which characterize the Russian language. Several more of these are discussed in (Franks, 2005).

5 Initial Design

The initial round of development used the Turbo Parser (Martins et al., 2013) to train a supervsed model using annotated training data from Wikipedia. One of the Wikipedia dumps for Russian was used as the pool from which the training, development and testing corpus were extracted. A subset of the corpus was extracted at random, from which development and test sets were chosen such that they contained approximately 1000 tokens each. The rest of the subset was reserved for training data. The data was first manually preprocessed to remove titles and other text that were not part of the body of the Wikipedia article, such as filenames and links to other pages. All data has been tokenized using the Lingua Ru OpenCorpora Tokenizer tool, mentioned in section 3 above. Some automatic postprocessing was performed on the tokenized data in order to correct some common mistakes made by the tokenizer (stripping off punctuation marks from the end of tokens that are not abbreviations), and, finally, some manual postprocessing was performed to corrected for the cases where the token was indeed an abbreviation in order to keep the punctuation as part of the token. The data was then part-of-speech tagged using the TreeTagger tool, also mentioned in section 3 above. Since the author could not find an easy to use, freely available tool that can be used to split Russian sentences, the 'SENT' part of speech tag output by TreeTagger was used to split sentences. This resulted in mostly correct sentence splitting but did not always work when abbreviations ending in periods were in a sentence, thus resulting in some incorrectly split sentences. Next, the data was annotated using the DgAnnotator tool, also mentioned in section 3. Annotation of the data required many decisions about how to make attachments between words. In Russian, many annotation decisions can be made by asking semantic questions which can then be answered by looking at the morphological suffix on various words. However, this does not always work in a straightforward way. A discussion of some of the annotation decisions made follows.

In Russian it is grammatically correct to conjoin 2 independent clauses with a comma, a dash, a colon, or a semicolon rather than an "and", "but", or some other conjunction word. In this case, there will be multiple heads attached to the root in one sentence. In example (1), the two independent clauses are connected by a comma; each has a verb as the head of the clause and there is no word conjoining the two clauses. In this case, "usmatrivali", "observed", is the head of the first clause and is attached directly to the root, and "objavljali", "announced", is the head of the second clause, also attached directly to the root.

(1) В нем усматривали автопародию, его объявляли рупором идей Воннегута.
v nem usmatrivali avtoparodiju, evo objavljali ruporom idej Vonneguta.
'In him was they observed a parody, and he was announced as a loudspeaker of the ideas of Vonnegut.'

If the conjunction word is present, then the heads of the clauses being conjoined will be children of the conjunction word, which will itself be directly attached to the root. Thus, in (2), "i", meaning "and", is the head of the main verb in each clause, namely, of "i" in the first clause (since the first clause contains two main verbs - "obnaruzhil", "discovered", and "zahvatil", "captured"), and "bil", meaning "was", in the second clause.

(2) Однако подводную лодку обнаружил и захватил английский эсминец «Ворон», и
odnako podvodnuju lodku obnaruzhil i zahvatil anglijskij esminez "Voron", i
Джонсон был взят на борт эсминца.
Dzhonson bil vzjat na bort esminza
'However, the English destroyer "Raven" discovered and captured the submarine, and Johnson was taken on board the destroyer. '

For compound nouns where one of the nouns is not proper and the other one is, the one that is not the proper noun will be used as the head of the proper noun, since one can argue that the proper noun is modifying the non-proper noun (making it more specific). An example of this is (3), where "choreographer" is the head of "Peter", and "Peter" is the head of "Sandrin".

(3) хореограф Пьер Сандрини
horeograf Pier Sandrin
‘’choreographer Pier Sandrin’

Another consideration is that many sentences in Russian are grammatical without verbs. In these sentences, the next most important word in the sentence will be selected as the head. Thus, in (4), this will be “beach”.

(4) Пляж чистый и обустроенный .
pljazh chistij i obustrojennij .

’beach clean and equipped .’

Another common, arbitrary case is that of an auxiliary verb followed by an infinitive. In this case, the auxiliary will be the head of the infinitive and the subject of the clause, since the sentence will be grammatical if the infiinitive verb is taken out, but will not be grammatical if the auxiliary verb is taken out. The infinitive will be the head of the objects, direct objects, and obliques in the clause. In example (5), “mozhno”, which translates as “can”, is the head, and “vospolzovatsja”, “use”, is its child.

(5) воспользоваться которыми можно за отдельную плату
vospolzovatsja kotorimi mozhno za otdelnuju platu

’which one can use for a separate fee’

For compound acronyms, the first one will be made the head and all subsequent parts will attach in a chain. For example, in (6), "ДВО" is the head of "РАН".

(6) Дальневосточное отделение Российской академии наук (ДВО РАН)
dal’nevostochnoje otdelenije rossijskoj akademiji nauk (dvo ran)

’Far East Department of the Russian Academy of Science (FED RAS).’

In many situations, the case of a word helps determine whether it should be the head or the child of another word. For example, in (7), 7500 is the head of "chelovek" ("people"), because "chelovek" is in the genitive case, answering the question "7500 who/what?". Removing the 7500 (along with its modifier, "okolo", meaning "about"), changes the meaning of the sentence to just one person doing work, whereas removing "chelovek" just underspecifies 7500 of exactly who or what is doing the work.

(7) трудится около 7500 человек
truditsja okolo 7500 chelovek

about 7500 people are working’

In contrast to (7), in (8), "godu", meaning "year", is the head of "1919", because "godu" is in prepositional case, answering the question "In what?" asked by the preposition "v", meaning "in", and "1919" is an adjective in prepositional form describing "godu".

(8) В 1919 году
v 1919 godu

'In the year 1919'

When there are two noun clauses with a dash in between, the dash often signifies an "is-a" relationship, where the noun clause after the dash is often clarifying or explaining the first part. In such situations, the head noun of the noun clause before the dash will be made the head of the main noun of the noun clause after the dash. This is shown in (9), where "uliza", meaning "street", is the head of "Promenad".

(9) От отелей пляж отделяет пешеходная улица — « Променад » .
Ot otelej pljazh otdeljajet peshehodnaja uliza - "Promenad" .

'The hotels are separated from the beach by a pedestrian walkway called "Promenad".'

In prepositional phrases, the preposition is used as the head of the phrase. This is shown in (10), where "v", meaning "in", is the head of the phrase "v balet", meaning "in the ballet". "v" is the child of the verb "vkluchil", meaning "included".

(10) включил танец в балет
vkluchil tanez v balet

'included the dance in the ballet.'

Finally, punctuation will simply be attached to the root.

Many decisions extend the above examples in more complicated ways, using these guidelines as a kind of algorithm for more complex sentences with many different clauses to find the head of each clause. Additionally, other situations that do not fall into common cases require decisions to be made on a case-to-case basis during the course of annotation.

6 System Analysis on Corpus A

7194 tokens, including punctuation, were annotated during the first round of development. There were 1059 tokens, including punctuation, in test corpus A. The results are summarized in Figure 1.

Features	Basic Model	Standard Model
DPOS	46.66%	46.41%
DPOS, CPOS	49.81%	48.55 %
CPOS	64.44%	63.68 %

Figure 1: Results of evaluation of parser on Corpus A.

7 Lessons Learned and Revised Design

The initial development of the parser showed how varied the results can be with a minimal amount of training data. This can be seen in the large difference in unlabeled attachment accurarices when using data labeled with only coarse vs coarse and detailed part of speech tags vs only detailed part of speech tags. The detailed part of speech tags may not be very good features because they incorporate detailed morphological information and are thus very numerous. This would result in a high degree of sparisty and, thus, overall decreased performance of the parser. This is also likely why the coarse part of speech tags do significantly better than the detailed part of speech tags.

In the next round of development, it may help to include additional information, such as morphological features and lemmas. One way to extract morphological features is to use the detailed part of speech tags and to split them into two parts, one part for the word category (a very coarse part of speech tag) and the other part for the morphological information that it carries. Thus, a tag such as "Ncmsnn", which can be broken down as N=Noun, Type=common, Gender=masculine, Number=singular, Case=nominative, and Animate=no, can be split into a coarse part of speech tag ("N") and the morphological information ("csmsnn"). Another source of morphological information is the morphological analyzer Pymorphy, mentioned in Section 3. One issue with this tool, however, is that it can only consider one word form at a time, and thus sometimes results in ambiguity. As a first pass, this can be dealt with, albeit coarsely, by selecting the first morphological feature that is returned, if there are multiple features returned.

Lemmas can be produced using the TreeTagger tool during the part of speech tagging process. Although the lemmas produced by this tool incorporate the word context, they can be problematic because of unknown word forms. The Pymorphy analyzer can also be used to retrieve the lemma for a word and does not result in unknowns because there is a predictor which accounts for them. However, this tool results in ambiguity for some word forms. Additionally, it only looks at one word at a time, ignoring context. Thus, while it is likely that TreeTagger could result in more accurate lemmas than Pymorphy, the overall usefulness of TreeTagger lemmas to the parser would likely be lowered because of the unkown word forms. One way to mitigate this problem could be to substitute Pymorphy lemmas (by choosing, once again, the first lemma that is returned, in the case that more than one are returned) whenever TreeTagger comes across an unknown word form. In the next round of development several experiments will be run combining the aforementioned features from these two tools to see which combinations perform the best on Corpus A and Corpus B.

8 System Analysis on Corpus B

Once the morphological analysis and lemmatization from the TreeTagger and Pymorphy tools was incorporated into the parsing framework, several experiments were run on the data that was annotated for part 6, where each experiment retained different combinations of features to use as input for the TurboParser. The parameters and their possible values were the following:

- Detailed part of speech tags: included or not
- Coarse part of speech tags: From coarse part of speech tagset, from splitting the detailed part of speech tags as described in section 7, or not included
- Morphological features: From splitting detailed part of speech tags, as described in section 7, from the Pymorphy tool, or not included
- Lemmas: From TreeTagger, from Pymorphy, or not included

  - If lemmas from TreeTagger included, whether or not Pymorphy lemmas were substituted for unknowns

- TurboParser mode: basic or standard

These combinations resulted in a total of 144 experiments on Corpus A and 144 experiments on Corpus B. There were 1135 tokens, including punctuation, in Corpus B. Some of the more interesting results of these experiments are shown in Figure 2, below. A discussion of the results follows.

Exper-iment	Features					Unlabeled Attachment Score	
	Lemma	No Unknown	Morphology	Coarse POS	Model Type	Corpus A	Corpus B
1	pymorphy			tagset	basic	**65.45 %**	76.76 %
2	pymorphy		pymorphy	split	basic	64.44 %	**77.86 %**
3	treetagger	False	pymorphy	tagset	standard	**65.32 %**	76.32 %
4	treetagger	False		split	standard	62.42 %	**78.41 %**
5	treetagger	False			basic	48.05 %	52.09 %
6	treetagger	True			basic	46.03 %	51.87 %
7	treetagger	False			standard	47.54 %	54.07 %
8	treetagger	True			standard	44.77 %	53.63 %

Figure 2: Evaluation of parser on corpora A and B using different combinations of features extracted using different tools.

The results in Figure 2 show that the combination of features that performs best varies when evaluated on corpus A and on corpus B. The results are better on both corpora when detailed part of speech tags are omitted. The results using detailed part of speech tags are not shown in figure, but using detailed part of speech tags always resulted in lower performance than using coarse part of speech tags. Using both detailed and any kind of coarse part of speech tags also always resulted in lower performance than using just coarse part of speech tags.

The best performing combination of features on Corpus A, using the basic model, is the one where lemmas are used from Pymorphy, coarse part of speech tags are used from the tagset provided with TreeTagger, and morphological features are omitted. This can be seen in experiment no. 1.

The best combination of features on Corpus B, using the basic model, however, uses both morphological features and lemmas from Pymorphy and coarse part of speech tags obtained from splitting the detailed part of speech tags from the TreeTagger tagset. This can be seen in experiment no. 2.

Using the standard model for TurboParser, the best performing combination of features on Corpus A, shown in experiment no. 3, is the one where lemmas are extracted from treetagger and unknowns are not replaced, morphological features are extracted from Pymorphy, and coarse part of speech tags from the tagset are used.

The best performing combination on Corpus B is shown in experiment no. 4, where only lemmas are extracted from Pymorphy, and coarse part of speech tags are derived from splitting the detailed part of speech tagset from TreeTagger.

Finally, experiments 5-8 attempt to isolate the effects of using lemmas from TreeTagger and replacing unknowns with lemmas from Pymorphy in order to determine whether or not replacing unknowns improves parser performance. The results are suggest that parser performance is not improved by this process, since the scores decrease whenever unknowns are replaced.

These results suggest that using TreeTagger lemmas and replacing unknowns is never preferred to either using TreeTagger lemmas and not replacing unknowns, or using Pymorphy lemmas. It is also interesting to note how varied the optimal combination of features is, depending on the case. Of course, the variability could be due to the small amount of training data. It is likely that a different combination of features would turn out to be the best-performing one with the addition of more training data. Furthermore, the nature of the two test sets is quite different. Corpus A contains a relatively long article with mathematical information and many equations and is thus not quite as representative of most of the training data as are the articles in Corpus B, and it may not be so surprising that a different combination of features performs better on it.

9 Final Revisions

During the final round of development, additional data was annotated, resulting in a total of 10,376 annotated tokens, including punctuation. The same experiments were run on this larger set of data as in Section 8. Results showing the best-performing combination of features on each test corpus using the basic and standard model are shown in Figure 3.

10 Future Work

The work described in this report has resulted in a small annotated training corpus and some insight into the best combination of features beyond part of speech tags that can improve performance when the parser is trained on this corpus. However, there is certainly room for improvement. One oversight in the addition of the morphological features was that they were not split by attribute-value and were instead inserted as a whole into each relevant slot for each token. In the future, separating them by attribute-value would likely decrease sparsity of morphological information and improve the results. Additionally, the problems with the preprocessing described in section 5 resulted in some badly split sentences which were annotated to the best of the author's ability but may have hurt the performance of the parser. A different method of preprocessing might lead to more proper sentences which would be easier to annotate and may improve the performance of the parser. One could also try eliminating all badly-split sentences in the training corpus to see if that improves or worsens performance. Futhermore, annotation errors could result in inconsistencies and noise in the annotation of the training data. This could have a particularly large effect on the parser, given the small amount of training data. The largest problem, overall, is thus the small amount of training data. Therefore, in addition to simply annotating more data for training, a possibility for future development aimed at reducing parser confusion due to such errors is to try an active learning method. Such a method, based on parser confidence, would involve searching for other Wikipedia articles that are most likely to help reduce confusion and increase the unlabeled attachment score. Additional directions for future work are to explore other sources of deriving lemmas from text and other sources of morphological information.

Exper-iment	Features					Unlabeled Attachment Score	
	Lemma	No Unknown	Morphology	Coarse POS	Model Type	Corpus A	Corpus B
1	treetagger	False	pymorphy	tagset	Basic	**68.10 %**	79.74 %
2			pymorphy	tagset	Basic	**67.21 %**	80.07 %
3	treetagger	True	pymorphy	tagset	Standard	**66.58 %**	**80.40 %**
4	treetagger	False			Basic	49.68 %	55.29 %
5	treetagger	True			Basic	48.17 %	54.19 %
6	treetagger	False			Standard	49.81 %	56.83 %
7	treetagger	True			Standard	45.90 %	56.17 %

Figure 3: Evaluation of parser on corpora A and B using different combinations of features extracted using different tools on more training data.

With more training data, the combinations with the highest unlabeled attachment scores, when using lemmas, are always the ones using lemmas from the TreeTagger tool. Furthermore, all morphological features came from Pymorphy, and all coarse part of speech tags came from the tagset. It is also interesting to note that the highest scoring combination of features on both corpora for the standard model is the same, and that in this combaintion, TreeTagger unknowns are replaced. However, experiments 4-7 support the results of the analogous experiments in section 8 that replacing unknowns when the TreeTagger tool is used does not improve the results, even if more training data is used. Although it is promising to see that with the addition of more data, there seems to emerge a pattern of which tools provide the best features and in which combinations, there is still not enough annotated training data to be able to draw a concrete conclusion that these combinations are meaningful and indeed the best ones.

References

Cyrillic alphabet. Encyclopaedia Britannica Online Academic Edition. URL `http://www.britannica.com/EBchecked/topic/148713/Cyrillic-alphabet`.

Russian Wikipedia. Web, January 2014. URL `http://en.wikipedia.org/wiki/Russian_Wikipedia`.

Wiktionary. Web, January 2014. URL `http://en.wikipedia.org/wiki/Wiktionary`.

A. A. Antonova and A. V. Misyurev. Russian dependency parser syntautom at the dialogue-2012 parser evaluation task. In *Computational Linguistics and Intellectual Technologies*, 2012.

Giuseppe Attardi. URL `http://medialab.di.unipi.it/Project/QA/Parser/DgAnnotator/`.

John Frederick Bailyn. *Word Order and Scrambling*. Blackwell Publishing Ltd, 2008.

Valentina Balkova, Andrey Sukhonogov, and Sergey Yablonsky. Russian WordNet. pages 31–38. URL `http://www.fi.muni.cz/gwc2004/proc/127.pdf`.

R. Bivon. *Element Order*, volume 7 of *Studies in the Modern Russian Language*. Cambridge University Press, Cambridge, 1971.

Matthew S. Dryer and Martin Haspelmath, editors. *WALS Online*. Max Planck Institute for Evolutionary Anthropology, Leipzig, 2013. URL `http://wals.info/languoid/lect/wals_code_rus`.

Andreas Eisele and Yu Chen. Multiun: A multilingual corpus from united nation documents. In Daniel Tapias, Mike Rosner, Stelios Piperidis, Jan Odjik, Joseph Mariani, Bente Maegaard, Khalid Choukri, and Nicoletta Calzolari (Conference Chair), editors, *Proceedings of the Seventh conference on International Language Resources and Evaluation*, pages 2868–2872. European Language Resources Association (ELRA), 5 2010.

Steven Franks. Slavic languages. In *Handbook of Comparative Syntax*, 2005.

Anastasia Gareyshina, Maxim Ionov, Olga Lyashevskaya, Dmitry Privoznov, Elena Sokolova, and Svetlana Toldova. RU-EVAL-2012: Evaluating dependency parsers for Russian. In *Proceedings of COLING 2012: Posters*, pages 349–360, Mumbai, India, December 2012. The COLING 2012 Organizing Committee. URL `http://www.aclweb.org/anthology/C12-2035`.

Gerald R. Greenberg and Steven Franks. A parametric approach to dative subjects and the second dative in slavic. *The Slavic and East European Journal*, 35(1):pp. 71–97, 1991. ISSN 00376752. URL `http://www.jstor.org/stable/309034`.

Mikhail Korobov. Morphological analyzer. URL `https://github.com/kmike/pymorphy`. Morphological analyzer (POS tagger / inflection engine) for Russian and English languages using converted AOT.

M. Paul Lewis, Gary F. Simons, and Charles D. Fennig. Ethnologue: Languages of the world, seventeenth edition. Web, 2013. URL `http://www.ethnologue.com/statistics/size`.

André F. T. Martins, Miguel Almeida, and Noah A. Smith. Turning on the turbo: Fast third-order non-projective turbo parsers. In *ACL (2)*, pages 617–622. The Association for Computer Linguistics, 2013. ISBN 978-1-937284-51-0. URL `http://dblp.uni-trier.de/db/conf/acl/acl2013-2.html#MartinsAS13`.

Helmut Schmid. Probabilistic part-of-speech tagging using decision trees, 1994.

Serge Sharoff and Joakim Nivre. The proper place of men and machines in language technology processing russian without any linguistic knowledge. In *Computational Linguistics and Intellectual Technologies*, 2011.

Serge Sharoff, Mikhail Kopotev, Tomaž Erjavec, Anna Feldman, and Dagmar Divjak. Designing and evaluating a russian tagset. In *In LREC'08*, 2008.

Dan Skatov, Sergey Liverko, Vladimir Okatiev, and Dmitry Strebkov. Parsing russian: a hybrid approach. In *Proceedings of the 4th Biennial International Workshop on Balto-Slavic Natural Language Processing*, pages 34–42, Sofia, Bulgaria, August 2013. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/W13-2406`.

Daniel D. Sleator and Davy Temperley. Parsing English with a link grammar. In *Proc. Third International Workshop on Parsing Technologies*, pages 277–292, 1993. URL `http://www.cs.cmu.edu/afs/cs.cmu.edu/project/link/pub/www/papers/ps/LG-IWPT93.ps`.

Aleksey Sokriko and Svetlana Toldova. Aot, 2006. URL `www.aot.ru`.

Alexey Surikov. Lingua-ru-opencorpora-tokenizer-0.06. Web. URL `http://search.cpan.org/~ksuri/Lingua-RU-OpenCorpora-Tokenizer-0.03/lib/Lingua/RU/OpenCorpora/Tokenizer.pm`.

S. Ju. Toldova, E. G. Sokolova, I. Astaf'eva, A. Gareyshina, A. Koroleva, D. Privoznov, E. Sidorova, L. Tupikina, and O. N. Lyashevskaya. Nlp evaluation 2011-2012: Russian syntactic parsers. In *Computational Linguistics and Intellectual Technologies*, volume 2, 2012.

Torsten Zesch, Christof Müller, and Iryna Gurevych. Extracting lexical semantic knowledge from wikipedia and wiktionary. In *Proceedings of the Conference on Language Resources and Evaluation (LREC), electronic proceedings*. Ubiquitious Knowledge Processing, Universität Darmstadt, Mai 2008. URL `http://www.ukp.tu-darmstadt.de/fileadmin/user_upload/Group_UKP/publikationen/2008/lrec08_camera_ready.pdf`.