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 (**?**). Russian is a free word order language, although according to (**?**), 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. (**?**) 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 (**?**), and some recent work on this topic includes (**?**).

Russian word order has had a significant impact on the prefered methods of parsing. 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 (**?**). 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 (**?**). The third-place parser in the competition, SyntAutom, also a rule-based parser, is an automata-based system (**??**). According to (**?**), 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 (**?**). 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

(**?**). 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 (**??**).

3 Available Resources

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 (**?**). This corpus could be handy because it has already been preprocessed. One problem with using this corpus, however, is that it is limited to a specific domain. In lieu of using a corpus that has already been prepared for use in NLP tasks, one can use Wikipedia to develop an annotated corpus, since many articles are available in Russian. Specifically, there are currently 1,085,000 articles in Russian on Wikipedia (**?**). Using Wikipedia would solve the aforementioned problem of having a limited domain.

A lexicon is another important resource in building a parser. The Russian version of WordNet is a one such lexicon that may be useful(**?**), although it seems that project development has not progressed over the last several years and that it may not have been completed, as no recent information seems to be available about its current status. Another potential resource that can be used as a kind of lexicon is Wiktionary. Wiktionary is a free online, collaborative dictionary available in multiple languages, including Russian (**?**). The Java-based Wiktionary Library (JWKTL), described in (**?**), is a freely available Java API that provides access to Wiktionary in multiple languages and will help in processing the large amount of information available on Wiktionary.

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 (**?**). Another tool that will be necessary is a part of speech tagger. The only freely available part of speech tagger, as far as the author is aware, is the TreeTagger tool, which uses a binary decision tree to estimate transition probabilities (**?**). The parameter files used to train the TreeTagger for Russian were trained "on the disambiguated version of the Russian National Corpus" (**?**). There is a choice of two parameter files - one using only basic part of speech tags, and another using more detailed part of speech tags. It is also helpful to have a visualization tool to perform the annotation of the parse trees. One such tool is the DgAnnotator, which provides a simple interface; it uses CoNLLX format for both input and output(**?**).

4 Survey of Phenomena in Russian

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 **?**). 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 (**?**). Several differing explanations have been proposed for the phenomena of the second dative; according to (**?**), 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. (**?**) 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 using a depedency framework to parse Russian 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 (**?**). Adpositions in Russian come before the noun phrase they modify (prepositions)(**?**). In generative literature, the free word order phenomenon is frequently referred to as "scrambling"(**?**). (**?**) 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 (**?**).

5 Initial Design

The initial round of development used the Turbo Parser(**?**) 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 and development 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 (such as stripping off various punctuation marks from the end of a token that was not an abbreviation), and, finally, some manual postprocessing that corrected for the cases where the token was indeen an abbreviation. The data was then part-of-speech tagged using the TreeTagger tool, also mentioned in section 3 above. Since there is no known 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 with abbreviations, 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 systematic decisions about how to make attachments between words. A discussion of some of the 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"; in this case, there will be multiple heads attached to the root in one sentence. 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:

(1) хореограф Пьер Сандрини
horeograf Pier Sandrin
''choreographer Pier Sandrin'

Another consideration is that many sentences in Russian are grammatical without verbs. For example:

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

'beach clean and equipped .'

In these sentences, the next most important word in the sentence will be selected as the head. In the above example, this will be "beach".

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 the example below, "mozhno", which translates as "can", is the head, and "vospolzovatsja", "use", is its child.

(3) Большая часть пляжа занята шезлонгами под зонтами , воспользоваться которыми можно за отдельную плату (8 левов за шезлонг на день , и еще 8 , если вы захотите воспользоваться стоящим рядом зонтиком , на большинстве отрезков пляжа) .
bolshaja chast pljazha zanjata shezlongami pod zontami , vospolzovatsja kotorimi mozhno za otdelnuju platu (8 levov za shezlong na den , i eshe 8 , jesli vi zahotite vospolzovatsja stojashim rjadom zontikom , na bolshinstve otrezkov pljazha).

'A big part of the beach is taken up by beach chairs under umbrellas, which one can use for a separate fee (8 levs for a beach chair for a day, and another 8 if you want to use the adjacent umbrella, in most parts of the beach.'

In the case of an adjective appearing in front of a compound noun, the first noun in the compound will be selected as the head of the adjective.

For compound acronyms, if it is unclear what they stand for, make the first one the head and all subsequent ones attach in a chain. In the case of a cardinal that enumerates a noun, make the cardinal the head and the noun its child. In the case of a date and the Russian word for "year", make the word for "year" the head, since it is slightly better grammatically if the year is removed than if the word for "year" is removed.

In prepositional phrases, use the preposition as the head of the phrase.

Of course, many other decisions must be made on a case-to-case basis during the course of annotation.

6 System Analysis on Corpus A

7057 tokens were annotated during the first round of development. There were 1059 tokens in test corpus A. Training the Turbo parser using the standard mode resulted in 47.29 % unlabeled attachment accuracy using both coarse and detailed part of speech tags. Using only coarse part of speech tags resulted in a 63.81 % unlabeled attachment accuracy. Training the Turbo parser using basic mode resulted in 48.05% using both coarse and detailed part of speech tags. Training the Turbo parser using basic mode resulted in 64.82% using only coarse part of speech tags.

Features	Basic Model	Standard Model
DPOS	46.53%	46.03%
DPOS, CPOS	49.68%	48.55 %
CPOS	64.69%	63.56 %

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.

Morphological features can be produced in several different ways. One possibility 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 possible source of morphological information is the morphological analyzer Pymorphy (**?**), which is based on the morphological dictionaries and work of (**?**). One issue with this tool, however, is that it can only consider one word form at a time, and thus sometimes results in ambiguity.

Lemmas can be produced using the TreeTagger tool. The lemmas produced by this tool can be problematic, however, because of unknown word forms. The Pymorphy analyzer can also be used to retrieve the lemma for a word, albeit with ambiguity for some word forms. Thus, while it is likely that TreeTagger will 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 would be to substitute Pymorphy lemmas (by choosing the first lemma that is returned, in the case that more than one are returned) whenever TreeTagger comes across an unknown word form.

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 possible parameters were for detailed part of speech tags (whether or not they were included), coarse part of speech tags (whether or not they were included and whether they came from the coarse part of speech tagset provided for TreeTagger or from splitting the detailed part of speech tags as described in section 7, above), morphological features (whether or not they were included and whether htey came from splitting the detailed part of speech tags, as described in section 7, above, or from the Pymorphy tool), lemmas (whether or not they were included and whether they came from TreeTagger or from Pymorphy), and if the lemmas came from TreeTagger, whether or not Pymorphy lemmas were substituted for unknowns. These experiments were run on TurboParser in basic mode and in standard mode and were evaluated on both corpus A and corpus B. The results of these experiments can be seen in the figure below, where M refers to the morphological features, L refers to the lemmas, DPOS refers to the detailed part of speech tags, and CPOS refers to coarse part of speech tags. The experiments were run on TurboParser in basic mode and in standard mode. Several of the results of these experiments can be seen in the table below.

Features	Unlabeled Attachment Score	
	Corpus A	Corpus B
L: treetagger CPOS: tagset NO-UNK: True MODEL-TYPE: basic	64.69 %	76.38 %
L: treetagger CPOS: tagset NO-UNK: False MODEL-TYPE: basic	64.31 %	75.48 %
L: pymorphy CPOS: tagset MODEL-TYPE: basic	65.45 %	76.49 %
M: pymorphy L: pymorphy CPOS: spliced MODEL-TYPE: basic	64.06 %	77.05 %

The results above show that the combination of features that performs best varies slightly when evaluated on corpus A and on corpus B. The results are better on both corpora when detailed part of speech tags are omitted. When using lemmas from TreeTagger in conjunction with coarse part of speech tags from the TreeTagger tagset, the results on both corpora are better when "<unknown>" lemmas are replaced with lemmas from Pymorphy. The best performing combination of features on Corpus B, however, use both morphological features and lemmas from Pymorphy and coarse part of speech tags that come from splitting the detailed part of speech tags from the TreeTagger tagset.

9 Final Revisions

10 Future Work

The work described above has resulted in a small annotated 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 are likely to be inconsistencies in the actual annotation of the training data which lead to noise that will have a large effect on the parser, given the small amount of training data. Thus, one possibility for future direction is to try an active learning method based on parser confidence to search for other Wikipedia articles that are most likely to help increase the unlabeled attachment score.