Computational Linguistics and Intellectual Technologies:
Proceedings of the International Conference “Dialogue 2017”

Moscow, May 31—June 3, 2017

PART-OF-SPEECH TAGGING WITH RICH LANGUAGE DESCRIPTION

Anastasyev D. G. (daniil_an@abbyy.com),
Andrianov A. I. (andrew_an@abbyy.com),
Indenbom E. M. (eugene_i@abbyy.com)

ABBYY, Moscow, Russia

This paper deals with morphological parsing of natural language texts. We propose a method that combines comprehensive morphological description provided by ABBYY Compreno system and sophisticated machine learning techniques used by the state-of-the-art POS taggers. The morphological description contains information about possible grammatical values of a dictionary word that helps to identify a set of potential hypothesis for each word during the morphological analysis stage. To analyse out-of-vocabulary words we are building a number of most likely paradigms in the morphological model using the orthographic features of the analysed word. The proposed method helps to reduce the number of hypotheses using the context information of each word. We use Bidirectional LSTM classifier to handle the context information and to predict the most probable grammatical value. The ambiguous grammatical values obtained from morphological description are used as features for the classifier. Also, we use word embeddings and orthographic features to achieve better results.

Key words: pos-tagging, morphological analysis, lemmatization, machine learning, lstm

МОРФОЛОГИЧЕСКАЯ РАЗМЕТКА С ИСПОЛЬЗОВАНИЕМ ОБШИРНОГО ОПИСАНИЯ ЯЗЫКА

Анастасьев Д. Г. (daniil_an@abbyy.com),
Андрианов А. И. (andrew_an@abbyy.com),
Инденбом Е. М. (eugene_i@abbyy.com)

ABBYY, Москва, Россия

1. Introduction

Part-of-speech (POS) tagging is the task of assigning each word in the given text an appropriate grammatical value. The morphological analysis is an essential element of most NLP problems. It means that quality of their solutions highly depends on the quality of the POS tagging.

Most researches on POS tagging have focused on English. The evaluation of these models is typically based on Penn Treebank. The latest approaches have claimed to achieve more than 97.55% accuracy. Unlike the previous methods, the newest ones are usually designed to use as few morphological features as possible. Such solutions are more likely to have stable performance on different corpora and to have the ability to be trained on various languages.

The most well-known Russian POS taggers are mystem [Segalovich, 2003], TnT-Russian [Sharoff, Nivre, 2011], Tree-Tagger [Schmid, 1994]. In contrast to the modern English taggers, the mentioned algorithms mostly rely on morphological features. However, their comparison is difficult because of lack of standard morphological tagset and corpora for the Russian language for tagger evaluation.

This work aims to combine comprehensive morphological description provided by ABBYY Compreno system [Anisimovich et al., 2012] and quite sophisticated machine learning techniques used by the novel English POS taggers.

Its evaluation was performed during the RuMorphoEval-2017 competition which was designed to provide a standard tagset and corpus for taggers comparison purposes.

2. Proposed Method

2.1. Russian Morphological Model

The most notable distinction of the proposed approach is the usage of the rich morphological model of the Russian language. It consists of a vast number of morphological paradigms and extensive lexicon. The dictionary consists of about 240 thousand of lexemes which provide us more than 3.5 million of words.

Such a significant number of words could be stored quite compactly based on the information about the words' paradigms. These paradigms contain information about the grammatical value of dictionary word and its inflexion. Therefore, we can store in the dictionary only the lexemes and the paradigms and obtain the needed word by composing this information. Overall, there were identified more than three thousand Russian paradigms.

As a result, usually, the words in corpora can be found in the lexicon and analysed by the provided morphological model. However, this analysis is not unambiguous: most of the words are homonymous.

This ambiguity may take place between the words of the same lexeme. For instance, "стол" ("table") can be either nominative or accusative form. Also the ambiguity can appear between the words of different lexemes: "стекло" may be both noun ("glass") and verb ("to flow down").

To deal with the ambiguity, we need to use context information. In the next sections, we are going to describe the method used to choose the correct analysis.

2.2. Unknown Words Processing

Despite the size of the provided lexicon, there are many out of vocabulary words in the texts. The most obvious examples of such words are named entities and neologisms.

To lemmatize such words, we use the following technique. We are constructing a set of pseudo-forms—hypothetical analyses of the given word. Then we are sorting them by their quality—the probability that the word is in such paradigm.

During the first step, we have to obtain all pairs of stem and paradigm conformed to our language model. As a result, we are going to receive the grammatical value of the word and its inflexion.

The stem of the word is its part without ending. All potential endings of the word can be found in the language description. Moreover, for each ending we can collect all possible paradigms—that is all paradigms where such ending occurs. Therefore, we get few stems with a limited number of paradigms agreed with the stem according to the language model.

Thus, we build a set of hypotheses—more than half of thousand in average. The next step is their ranking. The key element of the sorting is the usage of N-gram statistics of suffixes of word's stems. It based on the assumption that new words should contain patterns similar to some fragments of existing ones. Then it is likely to find these patterns in the suffixes of dictionary words.

Therefore, we should prefer forms that maximise the following function:

$$Q(form) = \mathrm{P}(paradigm(form), suffix(form))$$

Such probability can be estimated by corpora information.

Still, to improve the ranking, we should use the context information in a similar way as in the case of dictionary words.

2.3. Features

In our model following features are incorporated.

Grammatical value. Obviously, the information about the grammatical values of context words is vital in determining the grammatical value of the analysed word. We store these grammatical values of a word in the vector of size equals to the overall number of available grammemes. It means that each component of this vector corresponds to some grammeme.

However, as was mentioned in section 2.1, practically each morphological analysis contains some homonymy. So we write into the vector the estimated probability of each grammeme. The probability is calculated using the sum of frequencies of the morphological forms contains such grammeme.

For instance, consider the word "стул" ("chair"). It is a nominative form with frequency equals to $1.03{\cdot}10^{-6}$ or accusative form $8.15{\cdot}10^{-7}$ frequency. Thi leads us to the quality of the accusative grammeme calculated as

$$\frac{8.15 \cdot 10^{-7}}{8.15 \cdot 10^{-7} + 1.03 \cdot 10^{-6}} \approx 0.4417$$

Ambiguity classes' probabilities. Another type of features is probabilities of predicted classes (i.e. word's possible grammatical values). Such features set soft constraints on predictions. The probability is proportional to max frequency of form obtained by morphological analysis of the word.

Punctuation. The binary feature that corresponds to whether particular punctuation mark appears in the particular position in the word's surrounding.

Word's case type. The binary feature that says whether the word has proper, or upper, or lower capitalization.

Suffixes. The binary feature: whether the word has such suffix. Suffixes with length up to 3 were used during the developing of the model. To reduce the dimensions of the feature space, suffixes with low frequency were pruned: we collected 35 one-letter suffixes, 507 two-letters suffixes and 2316 three-letters suffixes with considerably large frequency.

Word Embeddings. 250-dimension dense vector corresponding to some word. The word embeddings technique has proved to be very effective in various NLP-tasks. There is a number of state-of-the-art English POS-taggers which utilise the power of the technique.

2.4. Learning Model

Predicted Classes. We enumerated grammatical values encountered in the train set. It appears to be slightly less than three hundred different categories of grammatical values. Hence, we can formulate the aim of the learning algorithm as a multiclass classification between the obtained grammatical values.

In this paper, to use a context of the analysed word, we take advantage of the Bidirectional Long-Short Term Memory neural networks (BiLSTM) [Hochreiter and Schmidhuber, 1997].

LSTM Classifier. LSTM is a variant of recurrent neural networks (RNNs). The RNNs use the information from the previous predictions to choose the label of the current input. Such architecture suits to POS-tagging better than traditional neural networks. However, it was proved that RNNs suffer from the gradient vanishing problem [Bengio et al., 1994]. It means that the ordinary recurrent network is aware only about the inputs from the short-period, but the information from more time steps is vanishing. LSTMs use gating mechanism to deal with the problem. It helps to the network to explicitly model long-term dependencies.

Meanwhile, the LSTM's hidden state stores information only about the previous words. To obtain the data from both the previous and the next words, we use the Bidirectional LSTM architecture. Its basic idea is to combine two LSTMs—forward and backward—and concatenate their output. Such a simple solution has been proven to be effective in the POS-tagging and similar tasks.

In this work, we decided to use a two-layer Bidirectional LSTM. During the development of the model, the additional layer gave obvious improvements in the performance of the LSTM on the validation set. However, it should be noted that such improvement may not be necessary for the practical usage. With the extra layer, both the train and the prediction time increases twofold. Moreover, the size of the network grows up. As a result, the usage of the second layer does not seem to be mandatory.

Additional Layers. Furthermore, we add a hidden Dense layer with ReLU activation on top of the LSTMs. This layer should help to handle the nonlinearity of the problem. The ReLU activation function is designed to deal with gradient vanishing problem. To connect this layer with the LSTM, we use the TimeDistributed wrapper from the keras library. This wrapper is used to apply the Dense layer to each word in the sentence separately.

Output Layer. The output layer is also wrapped by the TimeDistributed layer and it uses softmax activation function to output the probabilities for each considered grammar value.

Input Layers. We use few distinct input layers. First of all, we have a Grammemes input layer. It receives morphological features—the grammatical value, ambiguity classes' probabilities and the word's case type—and information about punctuation. Overall, we collected 617 different features. It is much lesser than the number of features used in most of the state-of-the-art classifiers. The main reason to such small set is the specificity of neural networks: we hardly can train a network on a large and sparse feature set. On the other hand, the features obtained by morphological analysis seem to be strong enough to rely on them.

As stated in section 2.3, we also utilise word embeddings technique. So we have another input layer to perform it. The model with both word embeddings and morphological features dramatically outperformed the model that uses only the morphological features.

We have considered the usage of the suffix features. We apply them using the embedding technique: for each suffix length we create a separate input layer and pass the input to the embeddings layer.

To reduce the dimensions of word embeddings' and suffixes' features, we implement a preprocessing to each analysed word. We substitute by a star ('*') all letters that do not belong to Russian alphabet or number, punctuation and symbols Unicode character categories. We replace each digit by zero ('0'). Finally, we convert each word to lower case. Such normalization leads to the reduction of the number of possible different word and suffix types.

Regularization. For the regularisation proposes we use Dropout technique [Srivastava et al., 2014]. We apply dropout to the Embedding layer, to the output of the LSTMs and inside the LSTM layers. Also, we utilise Batch Normalization [Ioffe, 2015] for the hidden Dense layer. This method helps to achieve faster learning speed and higher overall accuracy.

Optimizer. As an optimisation algorithm we have chosen the Adam optimizer [Kingma and Ba, 2014].

Summary. We implemented our model using the keras library[1] on theano backend [Bergstra et al., 2010].

The Fig. 1 illustrates the basic structure of our neural network with parameters corresponded to the keras parameters.

[1] From keras library: https://github.com/fchollet/keras/

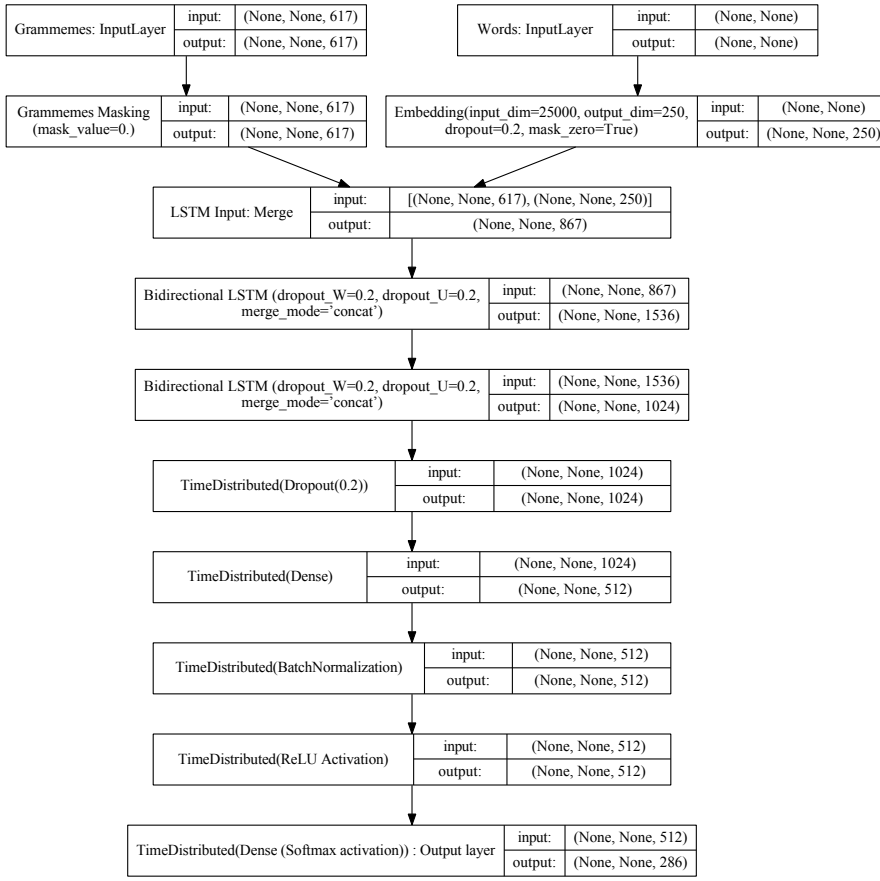

Fig. 1. Structure of the neural network

3. Model Development

The model was trained during participation in the MorphoRuEval-2017 competition[2]. In this competition the multiclass accuracy is used as the metric.

3.1. Tagset

The competition used slightly modified Universal Dependencies tagset[3].

Our morphological description is based on other tags, so we wrote a converter from our grammatical values to the required tagset. The convertor's mapping sets the

[2] https://github.com/dialogue-evaluation/morphoRuEval-2017

[3] http://universaldependencies.org/ru/feat/all.html

one-to-many relationship between our grammemes and the grammemes in the Universal Dependencies. It means that in some cases we convert our grammatical value to a few Universal Dependencies grammatical values with frequencies equal to the frequency of the initial grammatical value.

The converter is used in two ways. First of all, it is applied to obtain the set of ambiguity classes' probabilities. It seems acceptable to have an additional ambiguity due to the conversion process. Besides, we used the converter to train our model on additional corpora that were tagged with our tagset.

3.2. Training Data

As an additional corpora, we used a subset of Russian Wikipedia and parallel corpus of translated English novels. The Wikipedia corpus contains more than 3 million tokens. From the corpus of novels, we extracted subcorpus with about 30 million tokens. We used ABBYY Compreno system to perform tagging of the texts.

Besides, we used GICR texts with Universal Dependencies tagset[4]. This corpus consists of about one million tokens. It contains sentences from different social media sources.

3.3. Sentences padding

We use LSTMs to be able to deal with the whole sentence during classification stage. However, this neural network requires a three-dimensional tensor as an input. Due to inequality of the sentence lengths, we are not able to store the train data in one tensor without any changes. One of such possible changes is padding method: we choose the maximum sentence length and pad (i.e. add zeros to) all shorter sentences.

In addition to padding, we use a masking mechanism: we restrict the network from training on the padded elements.

The padding may drastically expand the size of the train data: with large maximum sentence length, we would usually waste memory on the zeros in the short sentences. To reduce the usage of memory, we divided all sentences into a few groups with different length: the sentences with up to 6, from 7 to 14, from 15 to 25, from 26 to 40 and more than 40 words.

3.4. Word Embeddings

As a baseline, we used randomly uniformly initialized embeddings for the first 5000 most frequent words with output dimension equals to 250. Surprisingly, the pretrained word embeddings (about 470 thousand words and 200-dimension output vector) had not given any enhancement. We decided to use randomly initialized embeddings of the 25 thousand most frequent words only.

---

[4] https://github.com/dialogue-evaluation/morphoRuEval-2017/blob/master/GICRYA_texts.zip

3.5. Model Training

To evaluate the quality of the model, we divided the GICR data into train and validation set at a ratio of 2 to 1.

The model development was performed in the following way. Firstly, we have experimented on the GICR texts to obtain optimal network architecture and the best set of hyperparameters. Then we have used the additional corpora to improve the achieved results. Also, we have experimented with extra layers and increased number of neurones on the additional data.

During the first stage, we found out the effectiveness of the network's architecture described in Fig 1. We achieved 96.31% accuracy on the validation set.

The additional suffixes features increased the accuracy up to 96.41%.

The usage of the extra Wikipedia subcorpus helped to improve the classifier quality to 96.78%. At the same time, the model trained on the Wikipedia only managed to achieve only 93.24%. The reason for such poor quality seems to be the case of the known fact: the accuracy of tagger trained on one text genre drops dramatically on other genres [Giesbrecht and Evert, 2009].

To deal with the problem, we applied the technique known as fine tuning. We used the weights of the model pretrained on the Wikipedia to initialize weights of the model and trained the model on GICR. That led to 97.43% accuracy.

We exploited this method to train the model on our novels subcorpus. The model trained on the novels subcorpus only was able to reach 95.36% accuracy. Fine tuning of this model on the GICR texts gave 97.78% accuracy, which is our best result on the validation set.

The Table 2 summarises the performance of the model achieved by usage of different train sets.

Table 2. Accuracies of the model trained on different corpora achieved on the validation set

Model	Train corpus	Accuracy on the validation set
Basic model	GICR	96.31%
+ suffix features	GICR	96.41%
+ suffix features	Wiki	93.24%
+ suffix features	GICR + Wiki	96.78%
+ suffix features	pretrained on Wiki, trained on GICR	97.43%
+ suffix features	Novels	95.36%
+ suffix features	pretrained on Novels, trained on GICR	**97.78%**

4. Evaluation

The evaluation was performed on three different genres of texts: fiction texts[5], news texts[6] and social networks texts[7].

4.1. Achieved Results

Our system received the following results:

Table 1. Performance of the model evaluated on MorphoRuEval-2017 test data

genre	accuracy by tokens	# tokens	# correct tokens	accuracy by sentences	# sentences	# correct sentences
fiction	97.45%	4,042	3,939	81.98%	394	323
news	97.37%	4,179	4,069	87.71%	358	314
social	96.52%	3,877	3,742	81.34%	568	462

The accuracy by sentences metric shows the fraction of sentences where each word was tagged correctly.

The degradation of performance on the social media texts should be the case of the genre differences between the train and test sets. Besides, the design of our algorithm leads to better performance on texts with proper spelling and good grammar. Frequent misspellings in the social media text limit the ability of the method to use the lexicon.

4.2. Errors Analysis

Table 3 shows the most frequent mistakes that our algorithm made during the MorphoRuEval-2017 competition. The "Number of occurrences" column shows frequency of the correct tag in the test selection, the "Number of error" column shows the number of cases when another tag was mistakenly predicted.

Table 3. Frequencies of the most common errors made by our system

Correct tag	Number of occurrences	Predicted tag	Number of errors
Nominative	2,650	Accusative	60
Accusative	1,644	Nominative	37
Plural	2,777	Singular	28
Nominative	2,650	Genitive	19
DET	656	PRON	14
PRON	1,133	DER	11

[5] From magazines.russ.ru

[6] From lenta.ru

[7] From vk.com

About 30% of all mistakes are the result of the ambiguity between nominative and accusative cases. The architecture of our network was designed to deal with such ambiguity by usage of the whole context of the word. However, even LSTM networks cannot perform well on long dependencies (which is a case of the gradient vanishing described in the 2.4 section). On the other hand, the system tends to follow the agreement between the tag of noun and its modifiers. Therefore, the incorrectly chosen tag of a noun usually leads to errors additional errors in the predicted grammatical value of the modifiers.

The mistakes in the determination of the number of nouns are also quite frequent—around 11% of all errors. For example, word "спортсменки" may be singular and in the genitive case ("sportswoman") or plural and in the nominative case ("sportswomen").

Another type of common errors is connected with distinguishing between some determiners and pronouns. Word "его" can have either "он" ("he") or "его" ("his") lemma and be either pronoun or determiner.

However, the fraction of such errors seems to be insignificantly low compared to the number of occurrences of the correct tags. The resolution of the ambiguity between the nominative and accusative cases seems to be the main issue of the algorithm.

4.3. Model Parameters Comparison

Using the test data provided by organisers we tested our model with different parameters.

Table 4 summarises the received results. The Accuracy columns contain the information about accuracies by tokens and by sentences.

Table 4. Comparison of performance of different models

Model	Fiction Accuracy	News Accuracy	Social Accuracy
Emb(5000)-1LSTM(768)-Dropout(0.2)-WithSuffixes	92.75% / 59.90%	94.52% / 55.59%	92.03% / 60.39%
Emb(5000)-1BiLSTM(768)-Dropout(0.2)-WithSuffixes	94.95% / 69.54%	97.01% / 75.70%	94.30% / 71.30%
Emb(5000)-1BiLSTM(768)-Dense(768)-Dropout(0.2)-WithSuffixes	95.35% / 71.83%	97.20% / 76.82%	94.66% / 73.94%
Emb(5000)-1BiLSTM(768)-2BiLSTM(512)-Dense(768)-Dropout(0.2)-WithoutSuffixes	**95.62%** / **74.11%**	97.37% / 77.65%	94.97% / 74.65%
Emb(5000)-1BiLSTM(768)-2BiLSTM(512)-Dense(768)-Dropout(0.2)-WithSuffixes	95.57% / 73.10%	97.37% / 78.77%	95.13% / 74.47%
Emb(5000)-1BiLSTM(768)-2BiLSTM(512)-Dense(768)-Dropout(0.5)-WithSuffixes	95.30% / 73.35%	**97.54%** / **79.89%**	**95.15%** / **75.00%**
Emb(50000)-1BiLSTM(768)-2BiLSTM(512)-Dense(768)-Dropout(0.2)-WithSuffixes	95.27% / 71.57%	97.03% / 76.54%	95.00% / 74.65%
Final Variant	97.45% / 81.98%	97.37% / 87.71%	96.52% / 81.34%

The names of models reflect the architecture of the used network. “Emb” parameter shows the number of words in the embeddings layer. The following parameters show the types of layers and numbers of neurones in them. The last parameter indicates whether the suffix features were incorporated.

All models except the last one were trained on GICR corpus only. The last (“Final variant”) refers to the model described in section 3.5.

The model with single LSTM layer shows the worst tagging quality. Obviously, the context information received from the left context only is not sufficient for proper tagging.

Clearly, the system gains from additional layers: the next two models with a single Bidirectional LSTM layer perform worse than more complicated models. On the other hand, larger embeddings layer (the Emb(50000)-model) also leads to poor accuracy. It can be explained by the lack of train data: we should use much bigger corpus to train such embeddings.

The increase in the dropout values helps to achieve a little better accuracy. Besides, the suffix features give an improvement in the model’s performance.

It should be noted that the model with single Bidirectional LSTM layer and only 5 thousand words in embeddings achieves good enough results in comparison with our final model while it is 2.5 times smaller. For some applications, such model could be more plausible than large but accurate one.

5. Conclusion

We have developed a POS-tagging model for Russian that can achieve high accuracy. Our system showed the best results on the MorphoRuEval-2017 competition. The degradation of its performance on some genres seems to be reasonably insignificant. Our model takes advantage of vast morphological description and modern machine learning techniques. Such approach seems likely to bring improvements in the quality of NLP-analysis systems based on the morphological analysis.

References

1. *Anisimovich K. V., Druzhkin K. Ju., Minlos F. R., Petrova M. A., Selegey V. P., Zuev K. A.* (2012), Syntactic and semantic parser based on ABBYY Compreno linguistic technologies. Computational linguistics and intellectual technologies: Proceedings of the International Conference “Dialog 2012”. Vol. 2, pp. 91–103
2. *Bengio Y., Simard P., Frasconi P.* (1994), Learning long-term dependencies with gradient descent is difficult. IEEE Transactions on Neural Networks, Vol. 9, Issue 2, pp. 157–166.
3. *Choi J.* (2016), Dynamic Feature Induction: The Last Gist to the State-of-the-Art, Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (NAACL 2016), San Diego, CA, pp. 271–281.
4. *Georgiev G., Zhikov V., Osenova P., Simov K., Nakov P.* (2012), Feature-rich part-of-speech tagging for morphologically complex languages: application to Bulgarian, Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics, Avignon, France, pp. 492–502.

5. *Giesbrecht E., Evert S.* (2009), Is Part-of-Speech Tagging a Solved Task? An Evaluation of POS Taggers for the German Web as Corpus, Proceedings of the Fifth Web as Corpus Workshop (WAC5), pp. 27–35.
6. *Hochreiter S., Schmidhuber J.* (1997), Long Short-Term Memory, Neural Computation, Vol. 9, Issue 8, pp 1735–1780.
7. *Ioffe S., Szegedy Ch.* (2015), Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift, available at: https://arxiv.org/abs/1502.03167.
8. *Kingma D., Ba J.* (2014), Adam: A Method for Stochastic Optimization, available at: https://arxiv.org/abs/1412.6980.
9. *Ma X., Hovy Ed.* (2016), End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF, available at: https://arxiv.org/abs/1603.01354
10. *Schmid H.* (1994), Probabilistic Part-of-Speech Tagging Using Decision Trees. Proceedings of International Conference on New Methods in Language Processing, Manchester, UK.
11. *Segalovich I.* (2003), A Fast Morphological Algorithm with Unknown Word Guessing Induced by a Dictionary for a Web Search Engine. Proceedings of the International Conference on Machine Learning; Models, Technologies and Applications, Las Vegas, Nevada, USA.
12. *Selegey D., Shavrina T., Selegey V., Sharoff S.* (2016) Automatic morphological tagging of Russian social media corpora: training and testing, Computational linguistics and intellectual technologies: Proceedings of the International Conference "Dialog 2016", pp. 589–604
13. *Sharoff S., Nivre J.* (2011), The proper place of men and machines in language technology: Processing Russian without any linguistic knowledge. Computational Linguistics and Intellectual Technologies: Proceedings of the International Conference "Dialog 2011" [Komp'yuternaya Lingvistika i Intellektual'nye Tekhnologii: Trudy Mezhdunarodnoy Konferentsii "Dialog 2011"], Bekasovo, pp. 591–605.
14. *Srivastava N., Hinton G., Krizhevsky A., Sutskever I., Salakhutdinov R.* (2014), Dropout: A simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research, Vol. 15 Issue 1: pp. 1929–1958.
15. *Theano Development Team* (2016), Theano: A Python framework for fast computation of mathematical expressions, available at: https://arxiv.org/abs/1605.02688.
16. *Toutanova K., Klein D., Manning C., Singer Y.* (2003), Feature-Rich Part-of-Speech Tagging with a Cyclic Dependency Network. In Proceedings of HLT-NAACL 2003, pp. 252–259
17. *Zalizniak A.* (1977) Russian Grammar Dictionary [Grammaticheskii Slovar' Russkogo Iazyka. Russki Iazyk].