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

Moscow, May 31—June 3, 2017

MORPHOBABUSHKA: SIMPLE AND FAST BASELINES YOUR GRANNY WOULD USE FOR PART-OF-SPEECH TAGGING OF RUSSIAN

Arefyev N. V. (nick.arefyev@gmail.com),
Ermolaev P. A. (ermolaev.p.a@yandex.ru)

Lomonosov Moscow State University, Moscow, Russia

The first shared task of MorphoRuEval-17 is to determine parts of speech and several grammatical categories such as case, number, gender, etc. for each word of text in Russian. We propose using NB-SVM over bag of character n-grams input representation to solve the task. Several methods are compared including CRF (Conditional Random Fields), SVM (Support Vector Machines) and NB-SVM (Naive Bayes SVM) and superiority of NB-SVM over other classifiers is shown.

The proposed model is the 5th best among 12 other models in the first shared task (per-token accuracy ranking). We also experimented with category grouping when a single classifier is used to determine several grammatical categories and showed that it improves the model performance even further.

Keywords: part-of-speech tagging, NB-SVM, CRF, multi-output classification

МОРФОБАБУШКА: ПРОСТЫЕ И БЫСТРЫЕ КЛАССИФИКАТОРЫ, КОТОРЫМИ ВОСПОЛЬЗОВАЛАСЬ БЫ ВАША БАБУШКА ДЛЯ МОРФОЛОГИЧЕСКОГО АНАЛИЗА РУССКОЯЗЫЧНЫХ ТЕКСТОВ

Арефьев Н. В. (nick.arefyev@gmail.com),
Ермолаев П. А. (ermolaev.p.a@yandex.ru)

Московский Государственный Университет
им. М. В. Ломоносова, Москва, Россия

1. Introduction

MorphoRuEval-17 (Sorokin, forthcoming) challenge consists of two shared tasks. The first one is to determine the part of speech and a number of grammatical categories (case, gender, number, etc.) for each word in a sentence. The second one is lemmatization.

There are various systems for russian language, such as Pymorphy, that allows to define grammatical categories and solve lemmatization problem, but unfortunately they don't have built-in solutions for resolving morphological ambiguity. Because of this, all possible variants of analysis are given for the word, which makes the practical application of such analyzers difficult.

The purpose of this article is to compare different approaches for resolving ambiguities. It can help selecting correct hypothesis among those returned by Pymorphy-like systems. So we focuse on the first task of MorphoRuEval-17 and don't perform lemmatization.

Part-of-speech tagging is an instance of sequence labeling task which is one of the fundamental tasks in Natural Language Processing. The difficulty of part-of-speech tagging lies in ambiguous and out-of-vocabulary words which require using context information as well as internal word structure. The most popular methods of sequence labeling are Hidden Markov Models (Brants, 2000) and Conditional Random Fields (Lafferty, 2001). Until recently implementing a good part-of-speech tagger required a lot of hand feature engineering, however recent successes in deep neural networks allowed to learn features necessary for the task during model training. First convolutional neural networks (dos Santos, 2014) and later recurrent neural networks (Plank, 2016) showed state of the art or near state of the art results in part-of-speech tagging learning from raw texts and using rather simple approaches compared to the previous state of the arts. Unfortunately neural networks require much more training data and computation resources to show good results. For instance, we tried convolutional neural networks similar to the ones in (dos Santos, 2014) for MorphoRuEval-17 tasks but experienced an order of magnitude larger training time compared to linear models (several hours instead of 5-10 minutes) and were not able to make them perform better than linear models before challenge deadline.

Usually grammatical information such as number for nouns or tense for verbs is often added to part-of-speech tags increasing the number of classes (for instance, plural and singular nouns are usually different classes). This reduces the task to standard multi-class classification where each example belongs to exactly one class. However, for morphologically rich languages this approach leads to the very large number of classes and very few examples for some of them. We treat the task as an instance of multi-output classification instead, i.e. for each grammatical category we train a separate multi-class classifier.

In section 2 of the paper we compare several approaches to part-of-speech tagging on MorphoRuEval-17 dataset. All our classifiers were trained from scratch, i.e. we used only training set provided by MorphoRuEval-17 challenge and didn't use any dictionaries (including provided), unlabeled datasets or other a priori knowledge. In section 3 we propose an extension allowing us in addition to part-of-speech tags to determine also grammatical categories (case, gender, number, etc.) which is necessary to solve

the first task of MorphoRuEval-17. We describe several tricks to obtain better classification results. The code to reproduce our best results is publicly available[1].

The main contributions of this paper are the following:

1. We proposed using NB-SVM model with bag of character n-grams input representation for POS-tagging and showed its superiority over linear SVM in this scenario.
2. We introduced scikit-learn compatible NB-SVM implementation for easier exploitation by NLP community.
3. We showed that it is beneficial to use a single classifier to jointly determine several grammatical categories (for instance, number and case).

2. Part-of-speech tagging

For part-of-speech tagging we tried two approaches: window-based and sentence-based classification. A window-based classifier treats each window (a target token to be classified with a fixed number of nearby tokens) as a separate example belonging to a single class (the part-of-speech tag of the target token). A sentence-based classifier receives the whole sentence and returns a sequence of classes (the part-of-speech tag for each token in the sentence). In theory, a sentence-based classifier working from left to right can benefit from knowing part-of-speech tags of previous tokens in the sentence while classifying the next word.

2.1. Window-based classification: NB-SVM classifier

We experimented with windows of sizes 1 (classification is based on target token only, no context is used), 3 (an example consists of the target token, one token to the left and one to the right) and 5. We didn't see significant improvements when changing window size from 3 to 5 so didn't try larger windows.

Each token inside a window was lowercased and vectorized using bag of character n-grams representation. To distinguish prefixes and suffixes from character n-grams occurred inside the token special symbols (^ and $) were added to each token as the first and the last character, this technique is usually referred as padding. Also we added additional features indicating token capitalization (lowercase, uppercase, etc.). Figure 1 shows an example of token vector when character bigrams and trigrams are used, this vector has thousands of elements so only some of them are shown. Finally we concatenated vectors for each token in the window to obtain vector representation of the window passed to the classifier.

...	^c	^ca	ca	cat	at	at$	t$	...	lower	upper	mixed
...	1	1	1	1	1	1	1	...	1	0	0

Figure 1. Vector representation of the token "cat" for (2–3)-grams. All vector elements not shown are zeros

[1] https://github.com/nvanva/MorphoBabushka

Several window-based classifiers including Logistic Regression, Multinomial Naive Bayes and Multilayer Perceptron were tried, however the best results were obtained with our implementation of NB-SVM classifier which we will describe in details.

NB-SVM classifier was first introduced for sentiment analysis and topic categorization in (Wang, 2012) paper and later with several variations showed excellent performance on IMDB movie reviews dataset exceeding all other single models including Recurrent Neural Networks and losing only when compared to ensemble models with NB-SVM as one of the classifiers in an ensemble (Mesnil, 2015). We have implemented NB-SVM classifier on top of scikit-learn library (Pedregosa, 2011) to use all advantages of this library including simple hyperparameter selection. Also we extended original NB-SVM allowing different scaling schemes for train and test set.

The main idea of NB-SVM is scaling input vectors before feeding them to the SVM classifier using feature-specific weights to obtain larger values for those features which are specific for one of the classes and smaller for those which occur uniformly across classes. Each feature value f_i is multiplied by

$$r_i = log\left(\frac{p_i/||p||_1}{n_i/||n||_1}\right)$$

where

$$p_i = \alpha + \sum_k I\{y^{(k)} = +\} f_i^{(k)}$$

$$n_i = \alpha + \sum_k I\{y^{(k)} = -\} f_i^{(k)}$$

are the sums of i-th feature values across positive or negative examples. The sums are smoothed by adding small α to eliminate zero denominators. These weights are essentially the feature weights learnt by Multinomial Naive Bayes model (MNB), hence the name NB-SVM.

Our implementation first trains MNB on training set, then uses learnt weights to rescale training set and trains linear SVM. We also added the possibility to binarize features or scale them to [0,1] interval before rescaling by MNB weights—these transformations can be done on training set, test set or both of them. We have found that no single transformation is optimal for all cases and best results can be achieved by selecting optimal transformation like other hyperparameters.

2.2. Sentence-based classification: CRF

An alternative to window-based classification is sentence-based classification when a classifier accepts the whole sentence as a single example and returns a class for each token in the sentence. A sentence-based classifier can benefit from learning dependencies between classes of nearby tokens (for instance, it is more probable that an adjective is followed by as noun than a verb) and using classes of previous tokens to classify the next one.

For sentence-based classification we trained Conditional Random Fields (CRF) model (Lafferty, 2001). For CRF we tried to take the same features as for NB-SVM, except for the beginning/end of the word, instead of special symbols, we always added for the word and its neighbors several (1, 2 and 3) first and last letters.

We have implemented CRF classifier using sklearn-crfsuite[2]. It's a thin CRFsuite (Okazaki N., 2007) wrapper which provides interface similar to scikit-learn.

2.3. Memory baseline

The simplest model we used as baseline memorizes classes assigned to each token in training set and returns the most frequent class of the given token. If the token didn't occur in the training set, the most frequent class overall is returned (NOUN in our case). We want to stress that this technique for dealing with out-of-vocabulary word used in memory baseline only. Other approaches we tried are based on character n-grams, not words, so they don't suffer from this problem.

The only preprocessing we did was lowercasing which improved performance a bit.

2.4. Experiments

Since there was no separate shared task for part-of-speech tagging in MorphoRuEval-17, we report here results of our own evaluation. We used official train/test split of Gikrya dataset and didn't use additional datasets or any other resources. The models were trained and evaluated on all 13 parts of speech occurred in training set instead of only 7 officially evaluated in MoprhoRuEval-17 (results are much better when measured only on 7 parts of speech, but this is quite non-standard evaluation scheme). We report accuracy on test set which is the proportion of correctly classified tokens. For each classifier we selected the best regularization and for NB-SVM we also selected the best scaling scheme using 3 fold cross-validation on training set.

Table 1. Accuracy on POS-tagging. NB-SVM (no padding) doesn't add special symbols (^ and $) to the token. NB–SVM (no caps) doesn't use capitalization features

accuracy	model
0.93	Memory baseline
0.97	CRF
0.979	NB-SVM (no padding)
0.98	Tf-idf + linear SVM
0.981	linear SVM
0.983	NB-SVM (no caps)
0.983	NB-SVM

Table 1 shows the results of NB-SVM compared to CRF, linear SVM over pure bag of character n-grams representation, linear SVM with a more traditional tf-idf scaling and memory baseline. Window of size 5 and n-grams of size from 1 to 5 were used for all classifiers. NB–SVM is the best model for POS-tagging improving results by 0.2% (10% error reduction) compared to linear SVM with no scaling. Tf-idf scaling doesn't help to improve

[2] http://sklearn-crfsuite.readthedocs.io/en/latest/contributing.html

accuracy and MNB scaling in NB-SVM helps probably because the latter takes dependencies between classes and features occurrences into account. Regarding features importance we can see that padding tokens to distinguish between same character n-gram occurred as prefix, postfix or inside the token helps, but capitalization features don't.

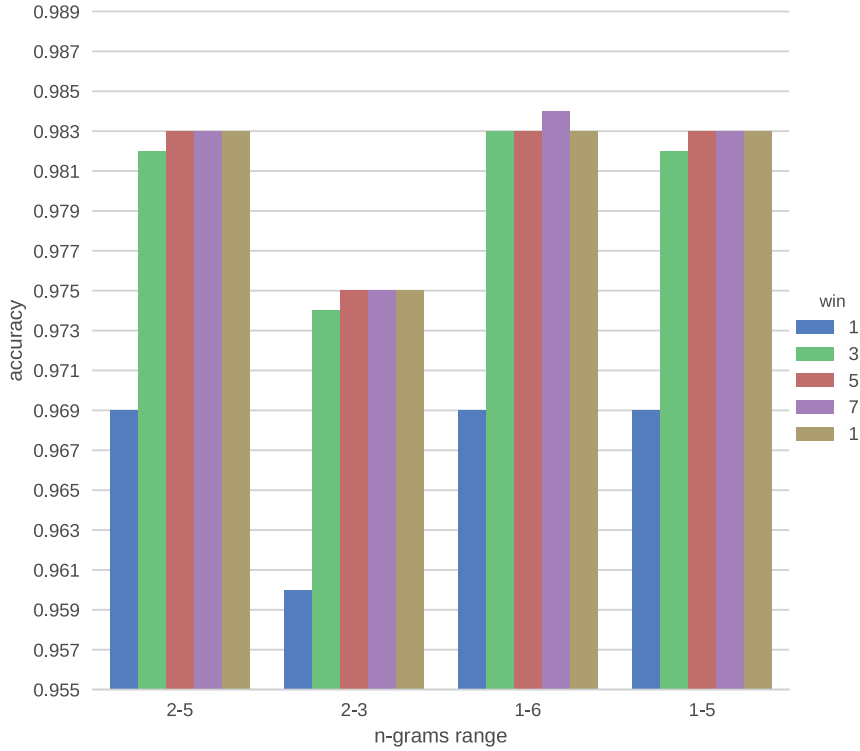

Figure 2. Accuracy of NB-SVM on POS-tagging w.r.t. window and n-grams sizes

The results can be affected not only by the classifier but also by the features used. Figure 2 shows the classification accuracy for NB-SVM depending on window size and n-grams size used to form input representations. We can see that using window of size 1 (no context) is a bad idea—context does matter. However, using larger context than one token to the left and one token to the right helps little (at most 0.1% improvement when increasing window size from 3 to 5 and from 5 to 7). Using only character bigrams and trigrams seems not enough: using character n-grams with n from 1 to 5 improves accuracy by 1%, but adding also 6-grams improves accuracy only by a small margin (0.1%).

3. Multi-output extension of part-of-speech tagging

To solve the first shared task of MorphoRuEval-17 not only parts of speech but also grammatical categories (case, number, tense, etc.) were required. The simplest

solution often used for morphologically not-so-rich languages is to use each possible combination of part of speech and grammatical attributes as a separate class, however this showed very poor accuracy in our preliminary experiments since the number of possible classes becomes very large and most of them have very few examples.

We treated the task as multi-output classification, when a classifier has fixed number of outputs each indicating a class of the input example according to its own criteria (grammatical category). The classes for each output are disjoint (for instance, this corresponds to impossibility for a token to be both a noun and a verb, or to be both in nominative and dative case). Disjointness of classes for each output distinguishes multi-output classification from multi-label classification, in the latter when all combinations of classes are possible.

3.1. Category grouping

One of the tricks we tried is using single output for several grammatical categories. For instance, we can group number and gender and have a single output with 6 classes instead of two different outputs with 2 and 3 classes. In theory, it can help if some classes are not linearly separable (remember that we use linear classifiers).

For instance Figure 3a shows an example of non-linearly separable classes s and p (singular and plural) which are transformed into six linearly separable classes (Figure 3b) after grouping number and gender grammatical categories.

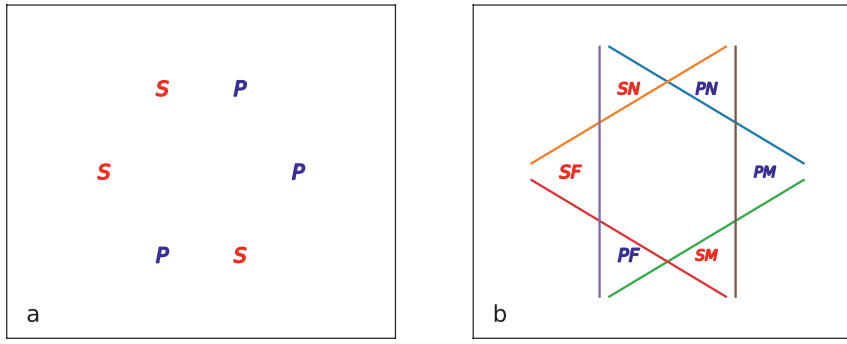

Figure 3. Joint prediction of several grammatical categories can transform into non-linearly separable dataset (3a) and linearly separable (3b). Here S means singular, P—plural, M—masculine, F—feminine, N—neuter.

3.2. Experiments

To evaluate the performance of our models in the first shared task we used the official MorphoRuEval-17 training / development sets[3] split and the official evalua-

[3] available at: https://github.com/dialogue-evaluation/morphoRuEval-2017

tion script. For all of our models we report per-token accuracy on development set measured by us using the official script. Additionally for those of our models that were submitted to the challenge we report per-token and per-sentence accuracies averaged over three test sets measured by the challenge organizers and used for the final participants ranking. It worth mentioning that unlike our POS-tagging evaluation in paragraph 2.4 the official script checks classification correctness not for all tokens but only for some of them belonging to several parts of speech and not for all grammatical categories but only several of them depending on the part of speech.

The initial results that we have submitted are shown in Table 3. Dev accuracy was measured by us using the official development set, test accuracy shows the official results on test set measured by the challenge committee. Test accuracy was reported only for those models, that were submitted for the challenge. Features are the same as they were with POS-tagging, all categories were classified separately. Similarly to POS-tagging the best results were achieved by NB-SVM classifier. Per-token NB-SVM accuracy was the 5th among other models, per-sentence accuracy—the 7th.

Next we tried to improve the accuracy of the best model using the category grouping. In our experiments with groupings, we decided to try combinations represented in Table 2. These combinations were chosen for practical reasons. The evaluation of all possible combinations would take a long time.

Table 2. Influence of category grouping on NB-SVM accuracy

grouping	number of outputs	accuracy
—	10	0.922
Gender+Number+Case, VerbForm+Mood+Tense	6	0.926
Gender+Number	9	0.923
Number+Case	9	0.928
VerbForm+Mood+Tense	8	0.922

Table 2 shows accuracy of NB-SVM on each group. For each category group (including consisting of a single category only) optimal hyperparameters were chosen separately using 3-fold cross-validation on training set which ensures the best possible classifier performance (that's why we obtained better results without category grouping compared with Table 3). As we can see, the most successful group is "Number+Case". So the correct grouping gives +0.6% to the accuracy.

Table 3. Reported results for Multi-output extension of part-of-speech tagging

classificator	dev accuracy (per token)	test accuracy (per-token/per-sentence)
NB-SVM	0.921	0.901 / 0.481
CRF	0.913	0.892 / 0.456
Memory baseline	0.742	0.724 / 0.138
NB-SVM (grouping—Number+Case)	0.928	—

4. Error Analysis

For error analysis we trained separate NB-SVM classifier for each of the 10 grammatical categories and used all of their values, not only officially evaluated. We used window of size 5, n-grams of size from 1 to 5 and selected best regularization and train / test scaling schemes individually for each classifier using 3-fold cross-validation on train set. Then we analyzed classifiers' performance using the official development set.

Table 4 shows performance of NB-SVM for different grammatical categories. In addition to accuracy and error rate for each category we report support (the number of tokens in the development set used for evaluation) and error count (the number of tokens misclassified by the corresponding classifier). It should be emphasized that error counts in table 4 may not sum to the total number of misclassified tokens (for instance, the same token can have both case and gender misclassified).

Table 4. Performance of NB-SVM for different grammatical categories

	accuracy	error number	error rate	support
Pos	0.983	4,537	0.017	270,264
Number	0.984	2,298	0.016	142,411
Case	0.927	8,117	0.073	110,967
Gender	0.979	2,262	0.021	107,544
VerbForm	0.999	31	0.001	39,083
Mood	0.998	64	0.002	30,170
Tense	1.000	0	0.000	31,227
Variant	1.000	0	0.000	3,810
NumForm	1.000	0	0.000	925
Degree	0.999	60	0.001	40,608

The situation when the classifier returned some value for a certain grammatical category and those tokens which didn't have this category in the gold standard was not considered as an error by the MorphoRuEval-17 official evaluation script. For instance, returning some gender tag for plural adjectives or case tag for verbs was not penalized. Hence during evaluation for each category we ignored those tokens which didn't have this category (in the gold standard) which explains different support for each category. This also means that the error number, not the accuracy of individual classifiers affects the overall performance most. For instance, the accuracy for Pos category is higher than for Gender category, but the support and the error number is also higher, so it can be more effective to improve Pos classifier first.

Table 4 shows that most errors are introduced by the Pos and Case classifiers and the Case classifier is responsible for roughly half of the errors.

Table 5 shows performance for each of the classes for the four most problematic categories. Also we show misclassification matrices for Pos and Case categories in fig. 4, 5. For the Case classifier more than half of the errors come from misclassifying accusative case as nominative and vice versa. The errors of the Pos classifier are more diverse, the most common one is misclassifying particles as conjunctions (14% of the errors).

Table 5. Performance of NB-SVM w.r.t. to classes for 4 most error-prone categories

	precision	recall	f1-score	support
Pos=ADJ	0.98	0.97	0.98	24,113
Pos=ADP	1.00	1.00	1.00	24,573
Pos=ADV	0.96	0.96	0.96	16,498
Pos=CONJ	0.93	0.96	0.94	16,211
Pos=DET	0.96	0.96	0.96	10,442
Pos=H	0.96	0.96	0.96	651
Pos=INTJ	0.91	0.87	0.89	257
Pos=NOUN	0.99	0.99	0.99	60,271
Pos=NUM	0.98	1.00	0.99	2,855
Pos=PART	0.97	0.91	0.94	10,208
Pos=PRON	0.97	0.97	0.97	19,742
Pos=PUNCT	1.00	1.00	1.00	45,360
Pos=VERB	1.00	1.00	1.00	39,083
Number=Plur	0.98	0.96	0.97	38,009
Number=Sing	0.99	0.99	0.99	104,402
Case=Acc	0.91	0.84	0.87	25,389
Case=Dat	0.97	0.93	0.95	7,112
Case=Gen	0.93	0.96	0.95	25,615
Case=Ins	0.97	0.97	0.97	10,070
Case=Loc	0.98	0.97	0.98	9,791
Case=Nom	0.90	0.94	0.92	32,990
Gender=Fem	0.99	0.98	0.98	35,574
Gender=Masc	0.97	0.98	0.98	48,054
Gender=Neut	0.98	0.97	0.98	23,916

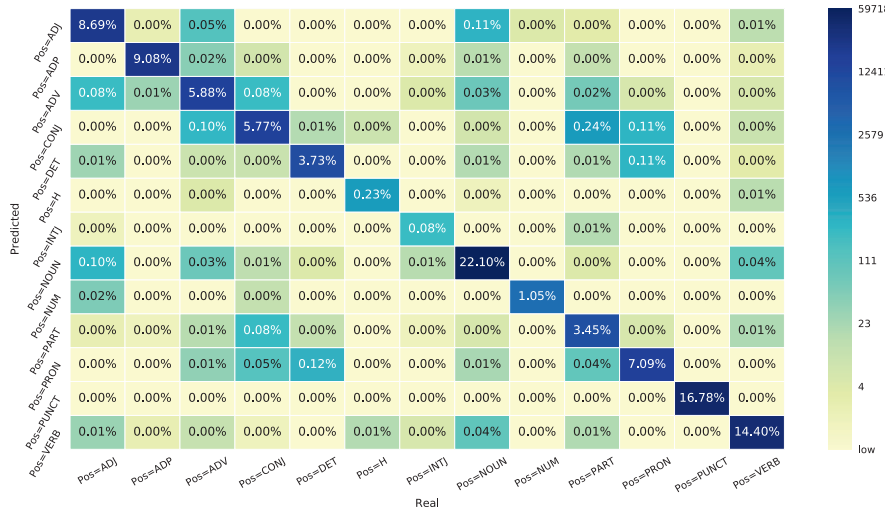

Figure 4. Misclassification matrix for part-of-speech tags

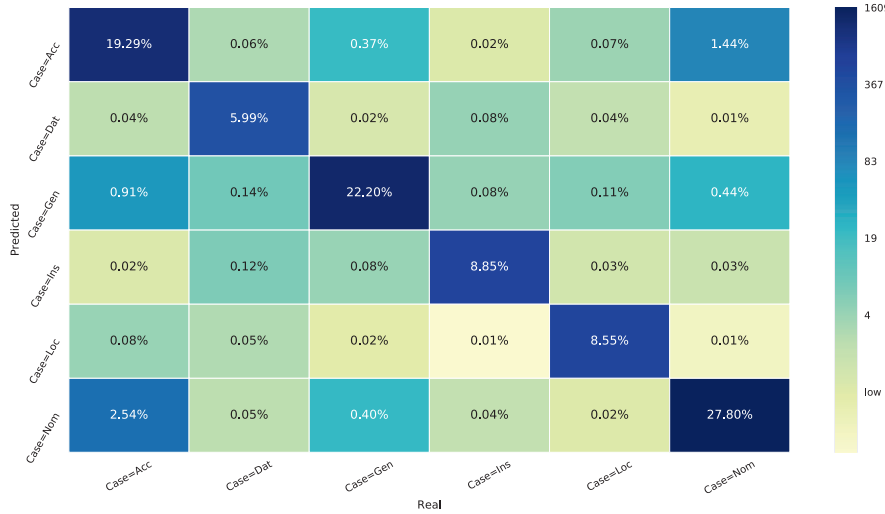

Figure 5. Misclassification matrix for case

5. Conclusions and future work

Part-of-speech tagging is well developed task, but it still has some places for improvement. As far as we know we are the first who proposed using NB-SVM with character n-grams representation for POS-tagging and showed that it outperforms other linear classifiers in the first shared task of MorphoRuEval-17 challenge, moreover

it was in 5 best models. Also for this task we showed that it can be advantageous to use single classifier to jointly determine several grammatical categories instead of using separate classifier for each category. Error analyses showed that the most promising direction is to improve Case classifier, for example, to increase its prediction score for Nominative and Accusative.

For the future work it will be interesting to try Recurrent Neural Networks which showed state-of-the-art results for POS-tagging of English and several other languages. Using large unlabeled corpora for unsupervised pretraining is also very promising technique because it can significantly improve classification of rare and out of vocabulary words.

References

1. *Brants T.* (2000), Tnt—a statistical part-of-speech tagger. In Proceedings of the 6th Applied NLP Conference, ANLP-2000, April 29—May 3, 2000, Seattle, WA.
2. *Lafferty J., McCallum A., Pereira F.* (2001), Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In Proceedings of the Eighteenth International Conference on Machine Learning, Williamstown, USA, pp. 282–289.
3. *Mesnil, G., Mikolov, T., Ranzato, M., Bengio, Y.* (2015), Ensemble of Generative and Discriminative Techniques for Sentiment Analysis of Movie Reviews. Submitted to the workshop track of ICLR 2015, available at: https://arxiv.org/abs/1412.5335.
4. *Okazaki N.* (2007), CRFsuite: a fast implementation of Conditional Random Fields (CRFs), available at: http://www.chokkan.org/software/crfsuite/
5. *Plank B., Søgaard A., & Goldberg Y.* (2016), Multilingual part-of-speech tagging with bidirectional long short-term memory models and auxiliary loss. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL), Berlin, Germany.
6. *Pedregosa F., Varoquaux G., Gramfort A., Vincent M., Bertrand T., Grisel O., Blondel M., Prettenhofer P., Weiss R., Dubourg V., Vanderplas J., Passos A., Cournapeau D.* (2011), "Scikit-learn: Machine Learning in Python". Journal of Machine Learning Research. 12: pp. 2825–2830.
7. *Santos C., Zadrozny B.* (2014), Learning character-level representations for part-of-speech tagging. In ICML. In Proceedings of the 31 st International Conference on Machine Learning, Beijing, China, 2014. JMLR: W&CP vol. 32., pp. 1818-1826.
8. *Sorokin, A., Shavrina, T., Lyashevskaya, O., Bocharov, V., Alexeeva, S., Droganova, K., Fenogenova, A.* (forthcoming). MorphoRuEval-2017: an evaluation track for the automatic morphological analysis methods for Russian. In Computational linguistics and intellectual technologies. Proceedings of International Workshop Dialogue'2017, Moscow.