ISSN 1870-4069

An Open-Source Lemmatizer for Russian Language based on Tree Regression Models

Iskander Akhmetov[1,2], Alexander Krassovitsky[1], Irina Ualiyeva[1], Alexander Gelbukh[3], Rustam Mussabayev[1]

[1] Institute of Information and Computational Technologies, Almaty, Kazakhstan

[2] Kazakh-British Technical University, Almaty, Kazakhstan

[3] Instituto Politécnico Nacional, CIC, Mexico City, Mexico

`i.akhmetov@ipic.kz, www.gelbukh.com`

Abstract. In this article, we consider the problem of supervised morphological analysis using an approach that differs from industry spread analogs. The article describes a new method of lemmatization based on the algorithms of machine learning, in particular, on the algorithms of regression analysis, trained on the open grammatical dictionary of Russian language. Comparison of obtained results was performed with existing alternative applications that are used nowadays for addressing lemmatization problems in NLP problems for Russian language. The proposed method shows some potential for further development as it has comparable quality but uses relatively simple machine learning algorithm and at the same time is not rule based involving no manual work. The source code for our lemmatizer is publicly available.

Keywords: lemmatization, text normalization, supervised machine learning, decision tree regression models.

1 Introduction

A common problem in the analysis of texts is a large feature space that corresponds to the dictionary used in text vectorizers (90–200 thousand attribute entities). A common approach to reduce vector space is to normalize texts. It shows considerable success in reducing vector space in cases when a relatively small amount of text available in datasets leads to more balanced and accurate models. In addition to dimensionality reduction of Vector Models it also reduces the size of the index, which speeds up all text processing operations.

Normalization, namely, word lemmatization is a one of the main text preprocessing steps needed in many downstream NLP tasks. The lemmatization is a process for assigning a lemma for every word form. A lemma is the canonical (normal, dictionary) form of the lexeme. For instance, in Russian language, the normal form of a noun corresponds to the form in the nominative case in the singular (for example, “сестры” *sestry* ‘sisters’ → “сестра” *sestra* ‘sister’), for adjectives, the normal form will correspond to the nominative case, masculine, singular (“сильными” *sil'nymi* ‘by strong’) → “сильный” *sil'nyj* ‘strong’), verbs have the normal form corresponding to the infinitive (“бегут” *begut* ‘they run’) → “бежать” *bezhat'* ‘to run’) [1]. A number of approaches exists for lemmatization [2–4], employing specific language morphological rules hardcoding (rule based approach), simple dictionary methods, and up to the contemporary deep learning methods.

Morphologically rich languages are difficult to implement lemmatization, because in addition to ambiguous morphology exception rules the semantics of words is highly dependent on the attached prefixes, affixes and suffixes. Our machine learning task is complicated by that fact that Russian language is influenced by a number of essential attributes related to the internal complexity of this natural language [5].

Two popular morphological analyzers for the Russian language are the pymorphy2 [6] and MyStem [7], the comparison with which is carried out in this article. MyStem is a tool for morphological data acquisition for Russian languages, pymorphy2 is a morphological analyzer for Russian and Ukrainian languages. Both of them are freely available for non-commercial and limited commercial use. MyStem is based on a dictionary, automatically converted to trie a structure.[1] Pymorphy2 is based on OpenCorpora dictionaries [8]. Both of them are based on manually elaborated set of heuristic rules, and on corpus statistics to eliminate extra morphological variants and obtain morphology of a wide lexical coverage.

In our work, the lemmatization is treated by building tree regression models [9], i.e., by supervised automatic learning with decision trees that are constructed corresponding to language grammatical features. A number of regression models have been compared by training on a well-built dictionary. Our method is a direct supervised approach of building word lemma regressor. In principle, this approach may be applied to any language, that captures the property of high variability inside its syntactic forms. Our approach estimates the possibility of computing syntactic models using only datasets in the form of wordform–lemma dictionaries.

This paper presents a comparative analysis of the lemmatization with Pymorphy2, MyStem and a publicly available implementation[2] of the method presented in this paper. For testing purposes lemma data set from is obtained by parser of ABBYY Compreno [10]. The ABBYY tool is taken as a gold standard of comparison approach, because nowadays is considered as state of art for the industrial techs. The dataset contains 225 publications taken from the Kazakhstan news portal tengrinews.kz marked by this parser. Our lemmatization procedure can be used in various scenarios; however, it is currently considered useful mainly as a preprocessing of Russian-language media

[1] A *trie* is a prefix tree and a special data structure used in information retrieval (IR) tasks [14].

[2] A working demo of our lemmatizer can be found at http://isa1.pythonanywhere.com/, and the source code is available on GitHub at https://github.com/iskander-akhmetov/Lemmatization-of-Russian-Language-by-Tree-Regression-Models/.

texts. The motivation to develop this lemmatizer is because the same entities of Russian language used in the media of some countries, such as Kazakhstan, are partially different from the entities of Russian language used in the media of Russian Federation.

2 Data and Methods

2.1 Dataset

For training models, the open grammatical dictionary of Russian language (ODICT) [11] was used, consisting of more than two million word forms and their lemmas. Examples of the dictionary entries are shown in Table 1.

Table 1. ODICT example entries.

Word	Lemma
елям (*yelyam* ‘to pine trees’)	ель (*yel’* ‘pine tree’, noun)
требовали (*trebovali* ‘they required)	требовать (*trebovat’* ‘to require, verb)
фактических (*fakticheskih* ‘of factual’, plural)	фактический (*fakticheskiy* ‘factual’, adjective)

To test the method, the corpus of the Kazakhstan news portal tengrinews.kz was taken, including 225 publications (20621 words). All publications were parsed via the ABBYY parser. To test for accuracy of the regression models open-corpora dataset [8] was used.

2.2 Method

Vectorization of words is performed character-by-character into a vector of fixed length 30 (feature space) and values as an order of a letter in the Russian alphabet with following zeros. After vectorization of various word forms and their initial forms obtained from the open dictionary, two arrays of vectors were obtained, which were randomly divided into training and test samples in a ratio 67 to 33. The resulting arrays were fitted into corresponding regression models. The following regression models were used: Decision Tree, Random Forest, Extra Tree, and Bagging from sklearn Python library [12].

We use tree-based methods for regression. These involve stratifying or segmenting the predictor space into a number of simple regions. We divide the predictor space—that is, the set of possible values for $X_1, \dots, X_p$—into J distinct and non-overlapping regions, $R_1, \dots, R_J$. For every observation that falls into the region R_j, we make the same prediction, which is simply the mean of the response values for the training observations in R_j. The goal is to find boxes $R_1, \dots, R_J$ that minimize the RSS, given by

$$\sum_{j=1}^{J} \sum_{i \in R_j} (y_i - \hat{y}_{R_j})^2,$$

where $\hat{y}_{R_j}$ is the mean response for the training observations within the j-th box.

Bagging, random forests, and boosting use trees as building blocks to construct more powerful prediction models. Each of these approaches involves producing multiple trees, which are then combined to yield a single consensus prediction.

3 Evaluation

We compared four variants of our regressor lemmatizers with the Pymorphy2 and MyStem. The performance of the lemmatizers was evaluated using the accuracy metric, roughly, the proportion of correct answers given by a lemmatizer. Table 2 presents the results for all regressor lemmatizers for four testing sets.

Table 2. Accuracy of regressor lemmatizers and available alternatives.

Regressor tagger	Cross-validation (5 folds, 2,337,988 words)	Train / Test split 67% / 33% (23,37,988 words)		ABBYY corpus check (20,621 words)	Open-Corpora check (347,409 words)
		Train	Test		
Pymorphy	–	–	–	0.8181	0.8967
MyStem	–	–	–	0.7250	0.8208
DecisionTree	0.3466	0.7296	0.6788	0.7561	0.7088
RandomForest	0.5991	0.8035	0.7562	0.3623	0.3556
ExtraTrees	0.6697	0.8759	0.8096	0.7544	0.6840
BaggingRegressor	0.6006	0.8045	0.7571	0.3682	0.3569

As it can be seen from Table 2, Extra Tree achieves 88% / 81%, Random Forest Regressor achieves 80% / 76%, Decision Tree Regressor 73% / 68% and Bagging Regressor, 80% / 76% on the train / test accuracy scoring. Experiments with variations on hyper parameters of the computation algorithms have shown that their optimization (i.e., a search for optimal values of tree depth and maximal splitting size) does not give essential improvement.

The cross validation showed descent result for the Decision tree algorithm and low results for the rest of the algorithms tested, but on train/test examination the results improved significantly. The rationalization behind this can be that cross validation performed on large datasets leaves testing on relatively large amount of unseen data performed several times (5 times in our case for the number of folds used) and if it happens to randomly select a difficult test set even for one time it can spoil the average significantly.

After comparing the results the models showed on ABBYY and OpenCorpora datasets, we saw that Decision Tree algorithm showed comparable results to ExtraTrees algorithm, but the obtained models differ 10 times in size: 450 MB for Decision Tree and 4.5 GB for ExtraTrees. Therefore, it was decided to use Decision Tree model for its relative simplicity and smaller size.

The reason for failure of RandomForest and Bagging Regressors on ABBYY and OpenCorpora tests is open question, but can be due to the difficulties of these algorithms with outliers and thus low robustness. Nevertheless, both of the failed algorithms are relatively hard to interpret and we leave it for further discussion and upcoming research.

Each feature receives weights according to its contribution to computed lemma. See Fig. 1, where the axes X and Y mean feature (letter position number in a word) and the computed weight, correspondingly.

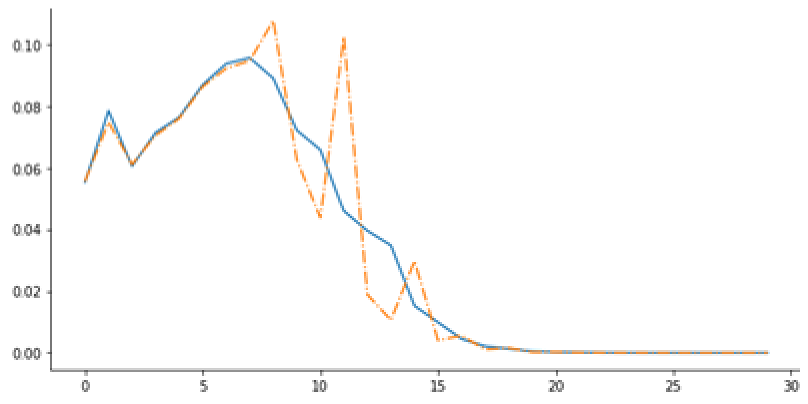

Fig. 1. The weights for features distribution (feature importance) reflect Russian word morphology (prefix, root, suffixes, etc.) are shown for Random Forest (dotted line) and Decision Tree (continuous line) regression models.

As can be seen from the figure, the model attributes more weight and thus account for importance of the beginning and especially the middle of the word - in Russian language it is usually the stem part [15]. Low significance of the letters in the positions 10 through 30 can be explained by the fact that the average word length in Russian is 10 letters and words of length more than 15 symbols are very rare.

Table 3. Comparison with alternative lemmatizers.

Lemmatizer	ABBYY corpus check (20621 words)	OpenCorpora check (347409 words)
Decision Tree Lemmatizer	**0.7561**	**0.7088**
Pymorphy2	0.8181	0.8967
MyStem	0.7250	0.8208

Despite the act that the proposed algorithm was less accurate than Pymorphy2 on ABBYY test and was left behind on the OpenCorpora test by both Pymorphy2 and MyStem, it is based on a relatively simple machine learning technique and ancient algorithm involving not much computational resources and still eliminates lots of manual rule hardcoding workload.

3.1 Error Analysis

In order to evaluate the performance of the method, the authors' lemmatizer was compared with the MyStem and Pymorphy2 lemmatizers, using the ABBYY parser to provide the testing data set. The number of wrongly lemmatized words is compared and shown by Venn diagram for these three lemmatizers by using the ABBYY test dataset; see Fig. 2.

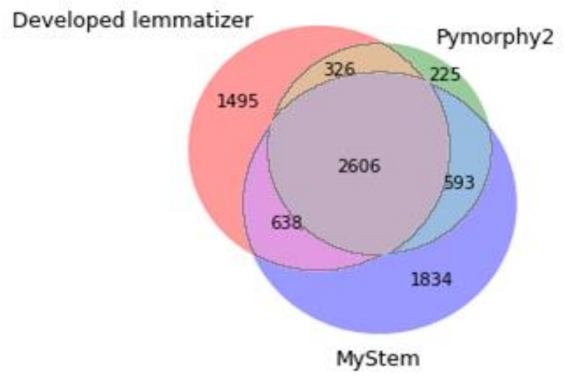

Fig. 2. The total number of errors and number of mutual errors in the testing dataset (20621 words) for our Developed lemmatizer (Decision Tree Regressor), MyStem and pymorphy2 are shown.

All three lemmatizers share 2,606 errors. The largest number of errors peculiar only to special lemmatizer belongs to MyStem (1,834 errors), which is followed by the Developed lemmatizer based on the Decision Tree algorithm (1,495 errors) and Pymorphy2 has mere 225 unique errors.

4 Conclusions

Decision tree regressors is not a silver bullet in machine learning, yet it can be a good tool in modelling language models in cases when it is too complicated to compose thousands of different rules manually. Our approaches estimate the possibility of computing syntactic models using datasets in the form of wordform–lemma dictionaries.

Number of experiments shown the developed new lemmatizer is able to solve the problem of lemmatization (especially for specific text topics), although it needs further training. Experiments can be continued for the corpus with a large number of publications and with the study of the speed of the algorithms.

Acknowledgment. This research is conducted within the framework of the grant No. BR05236839 "Development of information technologies and systems for stimulation of personality's sustainable development as one of the bases of development of digital Kazakhstan."

References

1. Smirnov, I.V.: Introduction to natural language analysis. RUDN University, Institute for Systems Analysis RAS, Moscow (2014)
2. Dave, R., Balani, P.: Survey paper of different lemmatization approaches. In: International Journal of Research in Advent Technology Science and Technology. Special Issue: First International Conference on Advent Trends in Engineering, Science and Technology, ICATEST 2015, pp. 366–370 (2015)
3. Ozturkmenoglu, O., Alpkocak, A.: Comparison of different lemmatization approaches for information retrieval on Turkish text collection. In: INISTA 2012, International Symposium on Innovations in Intelligent Systems and Applications (2012)
4. Plisson, J., Lavrac, N., Mladenić, D.D.: A rule based approach to word lemmatization. Proc. 7th International Multiconference on Information Society (2004)
5. Plungjan, A., Ljashevskaja, O.N., Sichinava, D.V.: On modern Russian language corpus morphological standard. Nauchno-tehnicheskaja informacija (Scientific and Technical Information). Series 2, Informacionnye processy i sistemy (Information processes and systems), pp. 2–9 (2005)
6. Korobov, M.: Morphological analyzer and generator for Russian and Ukrainian languages. In: Communications in Computer and Information Science, pp. 330–342 (2015)
7. Segalovich, I.: A fast morphological algorithm with unknown word guessing induced by a dictionary for a web search engine. In: MLMTA 2003, International Conference on Machine Learning: Models, Technologies and Applications (2003)
8. Otkrytyj korpus (OpenCorpora), http://opencorpora.org/. Accessed 21 Jan 2019
9. Geurts, P., Ernst, D., Wehenkel, L.: Extremely randomized trees. Mach. Learn. **63**(04), pp. 3–42 (2006)
10. ABBYY Company, https://www.abbyy.com/ru-ru/science/technologies/compreno/. Accessed 21 Jan 2019
11. Zaliznjak, A. A. Open grammar dictionary of the Russian language. http://odict.ru. Accessed 21 Jan 2019
12. Bird, S., Klein, E., Loper, E.: Natural language processing with Python. 2nd edn. O'Reilly (2017)
13. James, G., Witten, D., Hastie, T., Tibshirani, R.: An Introduction to Statistical Learning: with Applications in R, Springer Texts in Statistics, Springer (2013)
14. De La Briandais, R.: File searching using variable length keys. In: IRE-AIEE-ACM'59 (Western), Papers Presented at March 3–5, 1959, Western Joint Computer Conference. Association for Computing Machinery, pp. 295–298 (1959)
15. Sidorov, G.O.: Lemmatization in automatized system for compilation of personal style dictionaries of literary writers. In: Word of Dostoyevsky, Moscow, Russia, Russian Academy of Sciences, pp. 266–300 (1996)