UDC 004.056
doi:10.31799/1684-8853-2020-1-2-14

Articles

Generalized approach to sentiment analysis of short text messages in natural language processing

E. V. Polyakov[a], *Post-Graduate Student, orcid.org/0000-0002-6143-1408*
L. S. Voskov[a], *PhD, Tech., Professor, orcid.org/0000-0002-4008-8807, lvoskov@hse.ru*
P. S. Abramov[a], *Student, orcid.org/0000-0002-7079-0613*
S. V. Polyakov[b], *Post-Graduate Student, orcid.org/0000-0002-2151-6570*
[a]*National Research University «Higher School of Economics», 20, Myasnitskaya St., 101000, Moscow, Russian Federation*
[b]*Moscow Aviation Institute (National Research University), 4, Volokolamskoe Sh., 125993, Moscow, Russian Federation*

Introduction: *Sentiment analysis is a complex problem whose solution essentially depends on the context, field of study and amount of text data. Analysis of publications shows that the authors often do not use the full range of possible data transformations and their combinations. Only a part of the transformations is used, limiting the ways to develop high-quality classification models.* ***Purpose:*** *Developing and exploring a generalized approach to building a model, which consists in sequentially passing through the stages of exploratory data analysis, obtaining a basic solution, vectorization, preprocessing, hyperparameter optimization, and modeling.* ***Results:*** *Comparative experiments conducted using a generalized approach for classical machine learning and deep learning algorithms in order to solve the problem of sentiment analysis of short text messages in natural language processing have demonstrated that the classification quality grows from one stage to another. For classical algorithms, such an increase in quality was insignificant, but for deep learning, it was 8% on average at each stage. Additional studies have shown that the use of automatic machine learning which uses classical classification algorithms is comparable in quality to manual model development; however, it takes much longer. The use of transfer learning has a small but positive effect on the classification quality.* ***Practical relevance:*** *The proposed sequential approach can significantly improve the quality of models under development in natural language processing problems.*

Keywords *— natural language processing, machine learning, deep learning, vectorization, modeling, preprocessing, automatic machine learning, transfer learning.*

For citation: Polyakov E. V., Voskov L. S., Abramov P. S., Polyakov S. V. Generalized approach to sentiment analysis of short text messages in natural language processing. *Informatsionno-upravliaiushchie sistemy* [Information and Control Systems], 2020, no. 1, pp. 2–14. doi:10.31799/1684-8853-2020-1-2-14

Introduction

One of the most important research areas in machine learning is natural language processing [1] which solves the problems related to linguistic data, i. e. any textual content from Internet pages, books, forums or social networks. Determining the emotional coloring of a text (Sentiment Analysis) is a problem of algorithmically telling a negative statement from a positive one, whether it is an online article, comment on news or any other text written by any author. The main difficulty of sentiment analysis is that problems solved for one area cannot be successfully transferred to a different one [2]. As their basic component, sentiment analysis systems use a dictionary of words and phrases common in a particular area, which can significantly differ even from a dictionary for an adjacent area. As an example of such differences, we compared the distribution of words in two data sets (Fig. 1).

The first set of data is the visitors' feedback at the IMDb (Internet Movie Database) website [3] for movie search; the second one is the customers' feedback at the Amazon online store [4]. After analyzing the distribution of words, you can find the differences in their dictionaries. The IMDb dictionary differs from the Amazon one in almost every component. Another feature are collocations. While in the Amazon reviews bigrams look quite informative, the IMDb reviews need trigrams to be informative. In other words, a combination of bigrams and trigrams gives a better understanding in the case of IMDb, while for the Amazon case bigrams are enough. Thus, each sentiment analysis problem is individual, requiring a separate study. This makes sentiment analysis an important research field.

Related works

Despite a relatively large number of works devoted to comparing approaches and methods for sentiment analysis, most of them study only some aspects of solving the problem [5–12]. For example, in [5], the authors compare the approach based on the lexical algorithm from the Apache Hadoop architecture and Stanford coreNLP library with the implementation of recursive neural networks. The

■ ***Fig. 1.*** Distribution of words, bigrams and trigrams in IMDb and Amazon data sets

researchers classify the sentiments of the reviews left at a zoological forum. However, their paper avoids the comparison of sequential approaches traditionally used in natural language processing. In [6], a mood analysis system was proposed for enterprise software developers. The authors limited themselves to using only a set of vectorizers, classic machine learning models, and a preprocessing stack. The proposed SentiSW tools showed acceptable recognition quality. In [11], a BAM model was proposed for mood classification. A novel bi-level attention approach was used, which provided results of high quality. The data used were short texts in Chinese. The authors compared a large number of vectorizers and deep learning models, but did not consider the stages of data preprocessing and hyperparameter setting. Another work [12] discusses an approach based on the resolution of ambiguities. The authors try to find the best algorithm by heuristic selection of models, using different preprocessing techniques and various deep learning models. They applied the learning transfer approach, but did not consider hyperparameter optimization. Table 1 shows a comparative analysis of the stages of experiments conducted by the authors of the above works and the stages of our work.

We propose a generalized approach to sentiment analysis of short text messages, which consists in sharing all the main stages in the development of forecast models: intelligence analysis, basic solution, vectorization, preprocessing, modeling and hyperparameter tuning. The combined use of these stages allows you to get better results. For the experiments, we chose the Python programming lan-

■ ***Table 1***. Comparative analysis of the stages

Research	Intelligence data analysis	Base line	Vectorization/ vectorizer comparison	Preprocessing/ preprocessing comparison	Modeling/model comparison		Transfer learning	Tuning hyperparameters	Automatic machine learning
					Classical machine learning	Deep learning			
Sentiment analysis in a cross-media analysis framework [5]	Partially	No	No	No	The lexical algorithm from Apache-Hadoop	Stanford coreNLP, RNTN	Partially	No	No
Entity-level sentiment analysis of issue comments [6]	Yes	No	TF-IDF, Doc2Vec	Sumbols, Stop-Words, Punctuation, Marks, Tokenization, Stemming	Random Forest, Bagging, Gradient Boosting Tree, Naive Bayes, Ridge Regression, Linear Support, Vector Machine	No	No	No	No
Bi-level attention model for sentiment analysis of short texts [11]	Partially	No	BoW, MCNN, GloVe, Word-2Vec, FastText, T-WAM	No	SVM	Bi-GRU, AttBiLSTM, TMN, T-WAM, BAM, MCNN, RCNN, VDCNN	BERT	No	No
Thai comments sentiment analysis on social networks with deep learning approach [12]	Partially	No	No	Usernames, Sumbols, Phrase Substitution, Character Replacement	No	NB, CNN, CNN+ATNN, BLSTM, BLSTM+ATTN, CNN+-BLSTM, CNN+-BLSTM+ATTN, BGRU, BGRU+ATTN,CNN+BGRU, CNN+BGRU+ATTN	ULMFiT	No	No
This paper	Yes	Yes	BoW, TF-IDF, Word-2Vec, Doc2Vec	Lower Case, Stop-Words, Sumbols, Lemmatization, Stemming, Tokenization	Logistic Regression, Random Forest, Linear SVC, Gradient Boosting	FFNN, SimpleRNN, LSTM, GRU, BidirectionalSimpleRNN, BidirectionalLSTM, BidirectionalGRU, Convalution1D	GloVe, BERT	Yes	Yes

guage, along with the main libraries scikit-learn, Keras, Gensim and Plotly. We used Ubuntu Linux 19.04 operating system, 32 GB of RAM, swap file of size 64 GB, a video card Nvidia GeForce RTX 2060 / PCIe / SSE2 / 6 GB, and a processor Intel Core i7-8700 CPU / 3200 GHz x12.

Research methodology

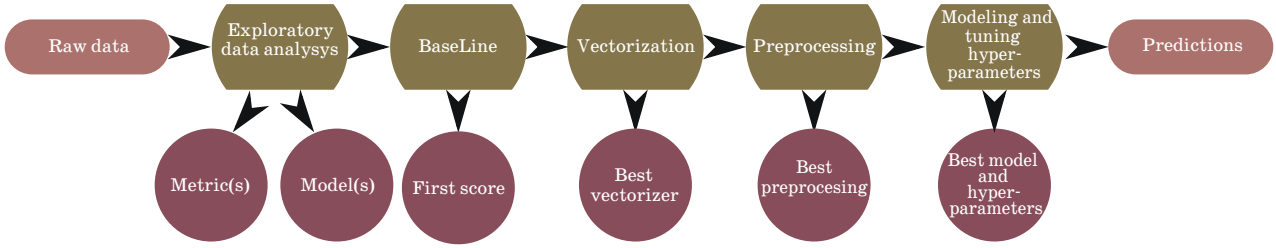
At each stage of our research, we will consistently apply the basic existing approaches to data processing, conduct experiments and evaluate the results obtained. The conceptual scheme of the research is shown in Fig. 2.

The exploratory analysis is carried out to study the nature of the data; according to its results, an assumption is made about the choice of metric and the set of models suitable for the problem. The basic solution is the first solution that gives us an understanding of what quality of the model can be obtained without any sophisticated data manipulations. At the vectorization stage, we sequentially train a set of commonly used vectorizers in order to identify the most suitable one. The preprocessing stage can either significantly improve or worsen the final model; therefore, it is necessary to conduct experiments with preprocessing of various types and to identify the best one.

After the preprocessing stage, the stage of modeling and hyperparameter tuning is performed in order to increase the effect of the previous stages by fine-tuning of the vectorizer model and classification model. Table 2 sums up and briefly describes the development stages in the proposed generalized approach.

Exploratory data analysis

Exploratory data analysis assumes obtaining information about the nature of the data in order to



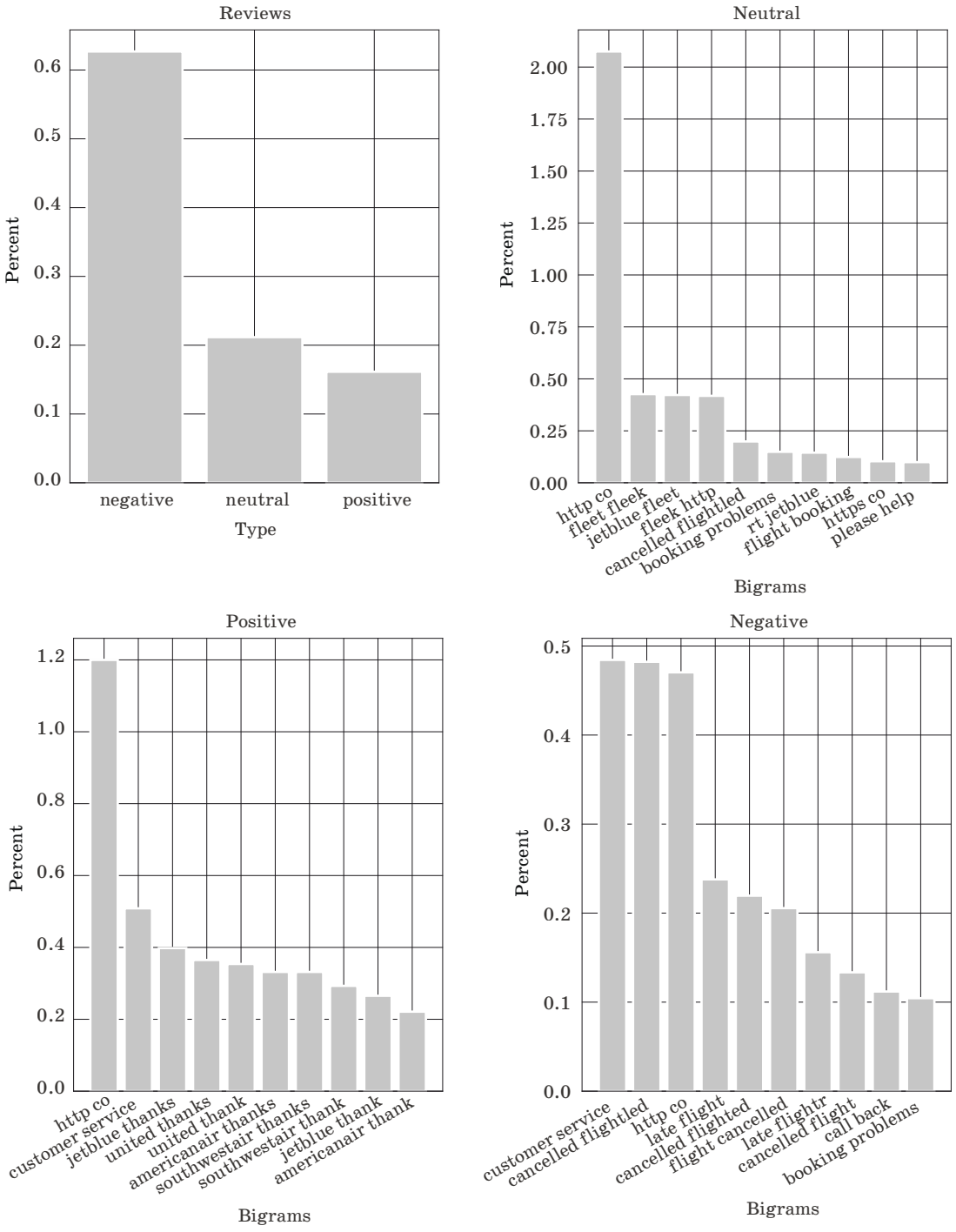
■ ***Fig. 2.*** Conceptual scheme of the research

■ ***Table 2.*** Generalized approach development stages

Input	Stage	Process	Output
Raw data	Exploratory data analysis	Visual data analysis, selection based on analysis of metrics and models	Metric(s), model(s)
Raw data, metric(s), model(s)	BaseLine	Training of a simple classification model(s) and a simple vectorizer on basic settings	Base model, base score
Raw data, metric(s), base model	Vectorization	Cyclic training of vectorizer models and the basic classification model at basic settings	Best vectorizer, vectorized data, base model
Best vectorizer, vectorized data, base model, metric(s)	Preprocessing	Cyclic data conversion and training on the basic model and the best vectorizer on the basic settings	Best preprocessing, best vectorizer, vectorized data, base model
Best vectorizer, best preprocessing, vectorized data, base model, metric(s)	Modeling	Cyclic training of different models with the best preprocessing and with the best vectorizer at basic settings	Best model, best vectorizer, best preprocessing, vectorized data
Best model, best vectorizer, best preprocessing, vectorized data, metric(s)	Tuning hyper-parameters	Setting hyperparammeters for vectorizers and classification models	Customized vectorizer model, customized classifier model

■ ***Table 3***. Twitter users' reviews

Tonality	Text
Positive	1. Thank you for the response, we got it resolved at the counter. 2. ... I love you. Air travel doesn't get easier.
Neutral	1. You must follow me in order for me to send you a direct message if that is what you meant. 2. ha, ha not a make or break for me either way!
Negative	1. I agree but per the captain this issue happened before boarding & we all sat in the plane for almost 2 hrs 2. no it weighed 45.5 and it was the only checked bag



■ ***Fig. 3***. Key statistics on the word distribution in the US airline Twitter comment data set

choose metric and a set of models most appropriate for solving the problem. For the experiments, we chose a benchmark data set containing 14.640 short messages left in Twitter by American airline customers [13]. Examples of the reviews from Twitter users are presented in Table 3.

Statistics of the word distribution in the reviews (Fig. 3) show that the distribution of classes in the samples is not balanced: about 60% of the messages are negative, 23% are neutral, and 17% are positive. For comparison, it is common to use such metrics as Accuracy, Precision or Recall which identify the prediction accuracy separately and independently for each class [14]. The aim of our study is to compare the ability of a model to generalize on the base of a sequence of dependent stages, not by a separate class. Therefore, as a metric for comparison, the ROC-AUC metric [15] was taken which takes into account the actual imbalance of the classes and reflects the stability of the model.

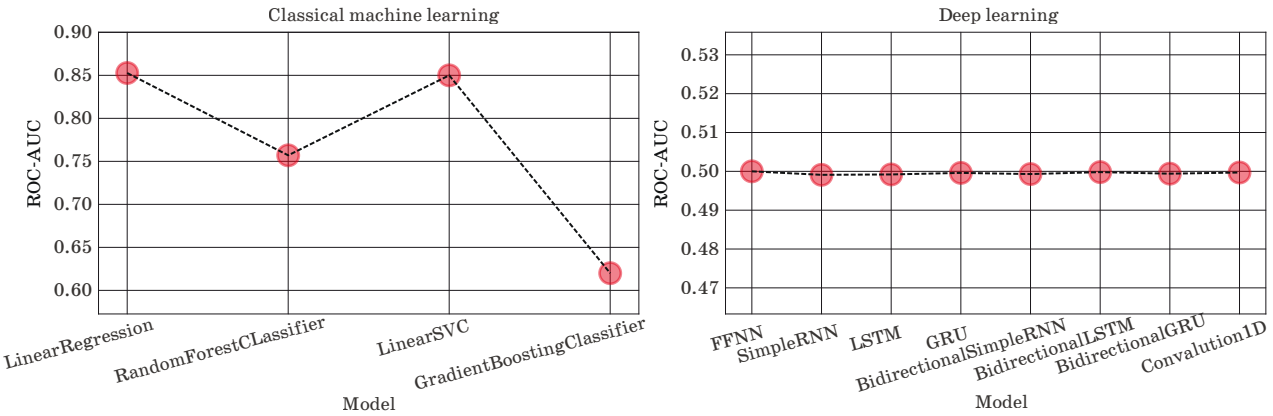
In natural language processing problems, both classical machine learning and deep learning are used. The choice of algorithms depends on the volume and complexity of the data. Often, when the data is huge in volume and the structure is complex, deep learning does the job better. However, determining the data complexity is a difficult task, strongly depending on the experience of the data processing specialist. The 14.640 tweets we used in our research cannot be considered as either small or large amount of data to make a straightforward decision on the choice of an algorithm. We will conduct a comparative study of using both classical algorithms and deep learning in order to demonstrate that deep learning provides the best result with our generalized approach.

Basic solution

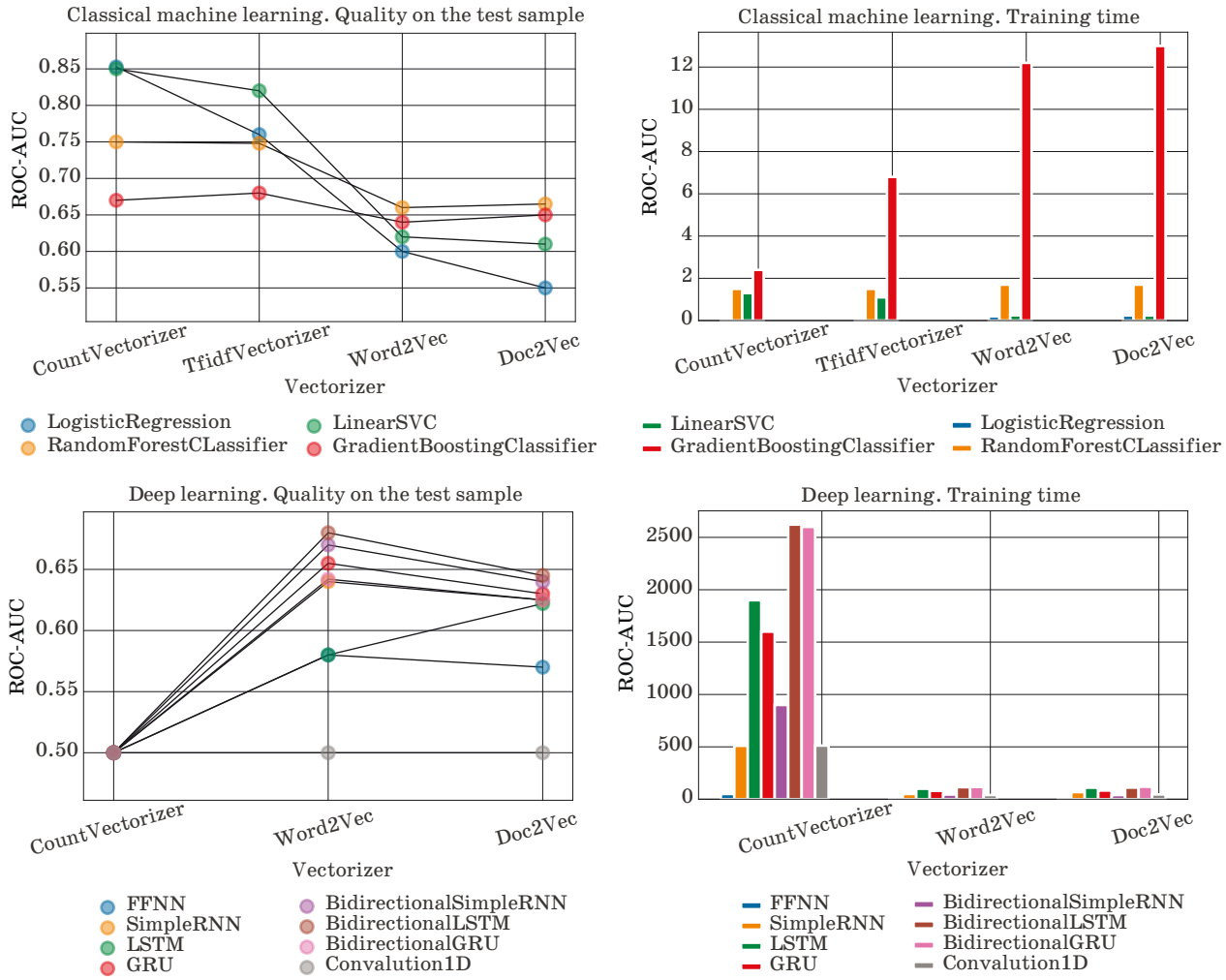
To get a basic solution, we have trained the vectorizer model and the classification models for both classical algorithms and deep learning. All the training was carried out with the default parameters set up by the developers of the *Keras* library, on 10 eras for deep learning and for classical algorithms, and with default settings for the scikit-*learn* library. The basic solution with classical machine learning and deep learning algorithms is shown in Fig. 4. Evaluation by ROC-AUC metric shows the quality of the basic solution and the machine time spent in the experiment: 0.12 s for CountVectorizer and LogosticRegression, and 11.83 s for FFNN and CountVectorizer. We will use the obtained values of the ROC-AUC metric in the future to evaluate the next steps when improving the model. The resulting low quality (0.5) of the basic model (see Fig. 4) for neural networks using deep learning as compared to the classical machine learning algorithms (0.85) does not mean that the model is unsuitable; the model may be too simple to generalize the data and should be improved. Improving the model by increasing its complexity is a way to improve the classification quality, which will be demonstrated below.

Vectorization

To improve the model, you can use various vectorizers which convert text data into numbers. We use four types of vectorizers with different approaches to text encoding: CountVectorizer (Bag of Words) [16], TfidfVectorizer [17], Word2Vec, and Doc2Vec [18]. These vectorizers were chosen because statistically they are more likely to provide a higher vectorization quality, being commonly used to solve problems of this kind. An improved model with vectorization showed that the basic solution found by the classical algorithms remained the best; however, algorithms based on neural networks significantly improved their performance from 0.5 to 0.68. The best model of the previous step was changed from FFNN to BidirectionalLSTM, and



■ ***Fig. 4.*** Basic solution using classic machine learning and deep learning algorithms



■ ***Fig. 5.*** Vectorizer selection

the vectorizer was changed to Word2Vec. The improved model had a higher complexity which led to a longer training time: it increased from 11.83 s to 43.7 min (Fig. 5).

Data preprocessing

In our work, we used the most popular types of preprocessing (Table 4): reduction to lower case (lc), deletion of stop words (sw), deletion of characters (sum), lemmatization (lemm), stemming (stem), and combinations thereof [19]. First, those preprocessing types are used which least affect the text structure, followed by those which have a greater effect on it. Processing with the name "dum" means no processing. The preliminary data processing (Fig. 6) provided only a slight increase in quality for LogisticRegression and CountVectorizer from 0.8527 to 0.8577 relative to the previous stage and the basic solution.

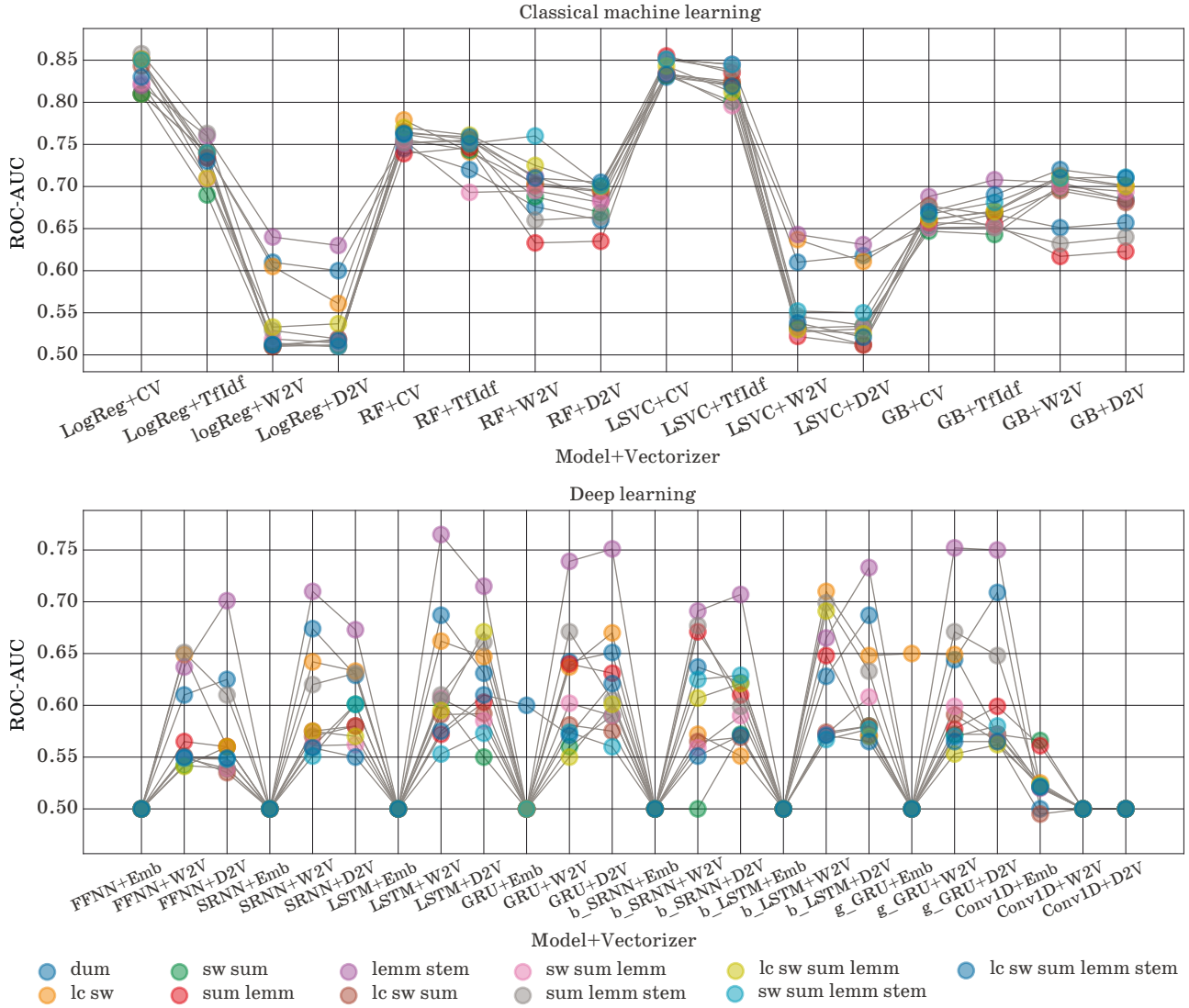
The training time increased from 0.12 to 0.80 s. With deep learning, we managed to obtain more significant results, increasing the ROC-AUC from 0.6818 to 0.7649. This time, the model with LSTM layers and such preprocessing as lemmatization or stemming instead of BidirectionalLSTM showed better results, which reduced the training time from 43.7 min down to 27.16 min, respectively.

Modeling and hyperparameter optimization

The previous steps allowed us to choose important joint components: the metric, vectorizer, preprocessing parameters and model. The stage of modeling and hyperparameter optimization is aimed at improving the quality of the resulting model achieved at the previous stages. We use one of the HyperOpt smart tuners [20] based on the Bayesian optimizer in order to avoid a complete

■ ***Table 4.*** Preprocessing examples

Preprocessing type	Preprocessing example
dum	I agree but per the captain this issue happened before boarding & we all sat in the plane for almost 2 hrs
lc	I agree but per the captain this issue happened before boarding & we all sat in the plane for almost 2 hrs
sw	Agree per captain issue happened boarding & sat plane almost 2 hrs
sum	I agree but per the captain this issue happened before boarding we all sat in the plane for almost 2 hrs
lemm	I agree but per the captain this issue happened before boarding & we all sat in the plane for almost 2 hrs
stem	I agre but per the captain this issu happen befor board & we all sat in the plane for almost 2 hrs
all	Agre per captain issu happen board sat plane almost 2 hrs
tokens	[‘I’, ‘agree’, ‘but’, ‘per’, ‘the’, ‘captain’, ‘this’, ‘issue’, ‘happened’, ‘before’, ‘boarding’, ‘&’, ‘we’, ‘all’, ‘sat’, ‘in’, ‘the’, ‘plane’, ‘for’, ‘almost’, ‘2’, ‘hrs’]



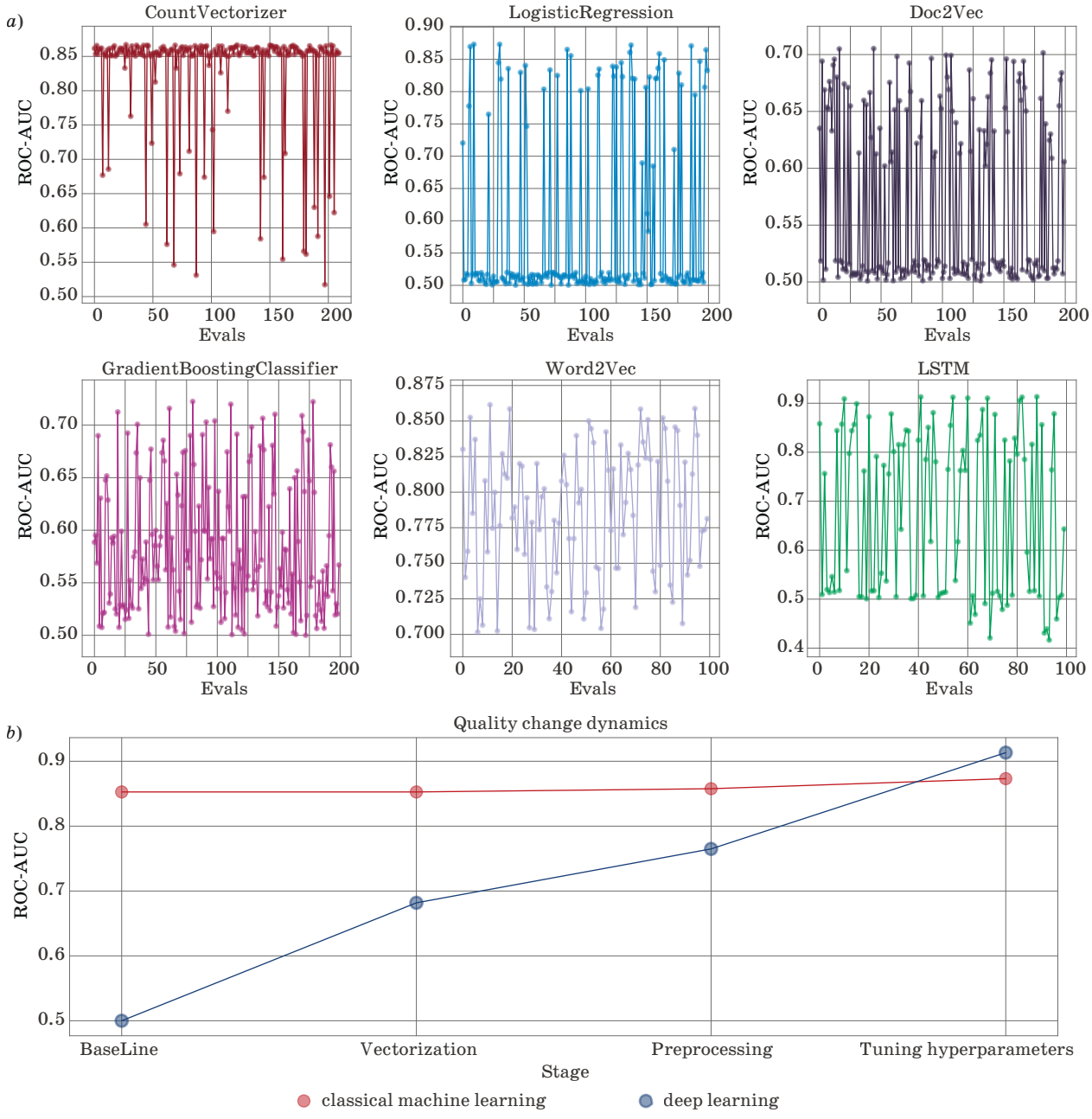
■ ***Fig. 6.*** Data preprocessing results

enumeration of parameters and significantly reduce the execution time for this stage.

First, we sequentially tune the vectorizers and models for classical algorithms, and then tune the parameters for deep learning in the same way. As a result of tuning the hyperparameters for CountVectorizer, the ROC-AUC grew from 0.8577 to 0.8671, while with the LogisticRegression tuning, the increase was up to 0.8732 (Fig. 7, *a*). The modeling time increased from 0.80 s to 32.8 min. Tuning Word2Vec for deep learning increased the ROC-AUC from 0.7649 to 0.8615, while tuning a neural network on LSTM [21] further increased the ROC-AUC up to 0.9133. The modeling time increased from 27.16 min to 79.3 h. For deep learning, we tuned such parameters as the activation functions, range of hidden layers/neurons, optimizers and level of thinning, in order to reduce the effect of retraining.

The application of the proposed generalized approach to sentiment analysis of short text messages provided the best result (0.9133) on sets for which the classical algorithms showed a good result in the basic solution (0.85) and significant improvement from one stage to another (Fig. 7, *b*).

The results of the hyperparameter optimization, the number of the iterations for the best parame-



■ ***Fig. 7.*** Hyperparameter tuning: *a* — iterations of the tuning; *b* — effect of the stages on the ROC-AUC results

■ ***Table 5***. Default hyperparameters

Model	Default hyperparameters
CountVectorizer	analyzer: 'word', binary: False, max_df: 1.0, max_features: None, ngram_range: (1, 1)
LogisticRegression	C: 1.0, class_weight: None, dual: False, fit_intercept: True, max_iter: 100, penalty: 'l2', tol: 0.0001
Doc2Vec	alpha: 0.0001, dbow_words: 0, dm: 1, dm_concat: 0, dm_mean: None, epochs: 10, max_vocab_size: None, min_count: 100, negative: 5, ns_exponent: 0.75, vector_size: 40, window: 1
GradientBoosting-Classifier	learning_rate: 0.1, loss: 'deviance', max_depth: 3, max_features: None, min_impurity_decrease: 0.0, min_samples_leaf: 1, min_samples_split: 2, min_weight_fraction_leaf: 0.0, subsample: 1.0
Word2Vec	alpha: 0.025, cbow_mean: 1, hs: 0, iter: 5, max_num_words: 100, max_seq_length: 10, max_vocab_size: None, min_count: 5, negative: 5, ns_exponent: 0.75, sample: 0.001, sg: 0, size: 100, window: 5
LSTM	activation: 'relu', units: 32, optimizer: 'RMSprop', epochs: 3, dropouts: 0.2

■ ***Table 6***. Hyperparameter optimization results

Model	Intervals/best options	Iteration	ROC-AUC
CountVectorizer	analyzer ('word', 'char', 'char_wb'): 'char_wb' | binary (True, False): False | max_df (0.2...1): 0.6 | max_features (100...500000): 28100 | ngram_range (1...14): (1, 10)	61	0.8671
LogisticRegression	C (0.1...0.9): 0.6241 | class_weight ('balanced', None): 'balanced' | fit_intercept (True, False): True | max_iter (1...1000): 801 | penalty ('l1', 'l2'): 'l2' | tol (0.0001...1000): 1.0011	142	0.8732
Doc2Vec	alpha (0.0001...10): 7.0001 | dbow_words (1, 0): 0 | dm (1, 0): 1 | dm_concat (1, 0): 1 | dm_mean (1, 0): 1 | epochs (1...100): 41 | max_vocab_size (None, 100...20000): 12100 | min_count (100...2000): 100 | negative (5...21): 7 | ns_exponent (0...1.0): 0.15 | vector_size (40...1000): 800 | window (1...20): 8	128	0.7053
GradientBoosting-Classifier	learning_rate (0.01...10): 0.61 | loss ('deviance', 'exponential'): 'exponential' | max_depth (2...20): 19 | max_features (None, 1...100): None | min_impurity_decrease (0...10): 1.6 | min_samples_leaf (1...20): 8 | min_samples_split (2...10): 5 | min_weight_fraction_leaf (0...0.5): 0.1 | subsample (0.001...1): 0.64	104	0.7225

ters, as well as the selection ranges are presented in Table 6. The default hyperparameters for the models are presented in Table 5.

Other methods

Automatic machine learning. There are libraries which allow you to automate the processes of model development [22–24]. The most famous one is TPOT [22]. This library uses a genetic algorithm to optimize machine learning pipelines.

The library supports the following stages: selection of features, preprocessing, construction of features, model selection and hyperparameter optimization. We have compared the results of using the library with the results obtained for all the manual development stages and obtained a comparable ROC-AUC quality which was 0.8724 versus 0.8732 when manually setting up the model. The process took a rather long time, 13.6 h. Unfortunately, for neural networks there are only limited versions of libraries for automatic learning. For example, Auto-Keras [24] supports optimization only for two-dimensional convolutional neural networks, which rules out using these methods for natural language processing problems.

Transfer learning. There are models trained on a large amount of data using huge computing resources. They can be used as ready-made vectorizers without prior training, for example, GloVe [25], ELMo [26], ULMFit [27] or BERT [28]. In our case, the best vec-

torizer was Word2Vec which we replaced by GloVe (a 100-dimensional vector, pre-trained on Twitter messages) and BERT (768-dimensional one). The quality grew up to 0.9232 and 0.9269, respectively.

The discussion of the results

We proposed a generalized approach to sentiment analysis of short text messages and compared it with the classical machine learning algorithms. The comparison showed that the proposed approach provided the best result and showed a significant improvement from one stage to another (Table 7). Using automatic machine learning for classical algorithms is comparable in quality to manual model development, but takes a much longer time. Transfer learning with standard settings slightly improves the model quality.

The aim of this work was to study how a complete generalized approach with various stages affects the model development, as compared to the approaches with selective or partial implementation of stages. We did not make any comparisons with the works whose authors also did sentiment analysis of short text messages using the same set of data, but we have presented the results demonstrating the improvement of the model as a whole, from one stage to another, as shown in Fig. 7, *b*.

Conclusion

The paper is devoted to a generalized approach to sentiment analysis of text messages left by customers of American airlines in a microblog. We discussed various approaches, comparing their methods of data preprocessing, vectorization, modeling and hyperparameter tuning, based on classical algorithms or deep learning algorithms. The results showed that the use of the stages studied in this work has a significant effect, being able to improve the quality by approximately 5%, compared to the classical algorithms. The latest SOTA solutions and learning transfer technology are discussed, too.

Financial support

The research was supported by the Academic Fund Program at the National Research University Higher School of Economics (HSE) in 2019–2020 (grant 19-04-022) and by the Russian Academic Excellence Project 5-100.

■ ***Table 7***. Comparison of the generalized approach

Stage	Classical machine learning	Deep learning	ROC-AUC		Time	
			ML	DL	ML	DL
BaseLine	CountVectorizer+ LogisticRegression	CountVectorizer(Embedding)+FFNN	0.8527	0.5	0.12 s	11.83 s
Vectorization	CountVectorizer+ LogisticRegression	BidirectionalLSTM+ Word2Vec	0.8527	0.6818	0.12 s	43.7 min
Preprocessing	CountVectorizer+ LogisticRegression+Stemming	BidirectionalLSTM+ Word2Vec+Stemming	0.8547	0.7392	0.28 s	37.3 min
Preprocessing combinations	CountVectorizer+ LogisticRegression+ Remove Characters+Lemmatization+ Stemming	LSTM+Word2Vec+ Lemmatization+Stemming	0.8577	0.7649	0.80 s	27.16 min
Tuning hyperparameters	CountVectorizer+LogisticRegression+ Remove Characters + Lemmatization+Stemming +Settings	LSTM+Word2Vec+ Lemmatization+ Stemming+Settings	0.8732	0.9133	32.8 min	79.3 h
Automatic machine learning	TPOT	—	0.8724	—	13.6 h	—
Transfer learning	—	LSTM+GloVe+ Lemmatization+Stemming+ Best hyperparameters	—	0.9232	—	37 min
Transfer learning	—	LSTM+BERT+ Lemmatization+ Stemming+ Best hyperparameters	—	0.9269	—	2 h

References

1. *Natural language processing*. Available at: https://becominghuman.ai/a-simple-introduction-to-natural-language-processing-ea66a1747b32 (accessed 11 August 2019).
2. **Ikoro V., Sharmina M., Malik K., and Batista-Navarro R.** Analyzing sentiments expressed on Twitter by UK Energy Company consumers. *2018 Fifth International Conference on Social Networks Analysis, Management and Security (SNAMS)*, Valencia, 2018, pp. 95–98.
3. *Large movie review dataset*. Avalible at: http://ai.stanford.edu/~amaas/data/sentiment/ (accessed 11 August 2019).
4. *Amazon reviews for sentiment analysis*. Available at: https://www.kaggle.com/bittlingmayer/amazonreviews (accessed 11 August 2019).
5. **Woldemariam Y.** Sentiment analysis in a cross-media analysis framework. *2016 IEEE International Conference on Big Data Analysis (ICBDA)*, Hangzhou, 2016, pp. 1–5.
6. **Ding J., Sun H., Wang X., and Liu X.** Entity-level sentiment analysis of issue comments. *2018 IEEE/ACM 3rd International Workshop on Emotion Awareness in Software Engineering (SEmotion)*, Gothenburg, 2018, pp. 7–13.
7. **Kumar M., and Bala A.** Analyzing Twitter sentiments through big data. *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)*, New Delhi, 2016, pp. 2628–2631.
8. **Sabra K., Zantout R., El-Abed M., and Hamandi L.** Sentiment analysis: Arabic sentiment lexicons. *2017 Sensors Networks Smart and Emerging Technologies (SENSET)*, Beirut, 2017, pp. 1–4.
9. **Vanaja S., and Belwal M.** Aspect-level sentiment analysis on E-commerce data. *2018 International Conference on Inventive Research in Computing Applications (ICIRCA)*, Coimbatore, 2018, pp. 1275–1279.
10. **Alshari E. M., Azman A., Doraisamy S., Mustapha N., and Alkeshr M.** Effective method for sentiment lexical dictionary enrichment based on Word2Vec for sentiment analysis. *2018 Fourth International Conference on Information Retrieval and Knowledge Management (CAMP)*, Kota Kinabalu, 2018, pp. 1–5.
11. *Bi-level attention model for sentiment analysis of short texts*. Avalible at: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=\&arnumber=8807106\&isnumber=6514899 (accessed 11 August 2019).
12. **Piyaphakdeesakun C., Facundes N., and Polvichai J.** Thai comments sentiment analysis on social networks with deep learning approach. *2019 34th International Technical Conference on Circuits/Systems, Computers and Communications (ITC-CSCC)*, JeJu, Korea (South), 2019, pp. 1–4.
13. *Twitter US airline sentiment*. Avalible at: https://www.kaggle.com/crowdflower/twitter-airline-sentiment (accessed 11 August 2019).
14. *Top 15 evaluation metrics for classification models*. Avalible at: https://www.machinelearningplus.com/machine-learning/evaluation-metrics-classification-models-r/ (accessed 11 August 2019).
15. **Brown Christopher D., Davis Herbert T.** Receiver operating characteristic curves and related decision measures: a tutorial. *Chemometrics and Intelligent Laboratory Systems*, 2006, vol. 80, pp. 24–38.
16. *A Gentle introduction to the bag-of-words model*. Available at: https://machinelearningmastery.com/gentle-introduction-bag-words-model/ (accessed 11 August 2019)
17. *Tf–idf*. Avalible at: http://www.tfidf.com/ (accessed 11 August 2019).
18. *Efficient estimation of word representations in vector space*. Avalible at: https://arxiv.org/abs/1301.3781v3 (accessed 11 August 2019).
19. *All you need to know about text preprocessing for NLP and Machine Learning*. Avalible at: https://www.kdnuggets.com/2019/04/text-preprocessing-nlp-machine-learning.html (accessed 11 August 2019).
20. *Hyperopt. Distributed asynchronous hyperparameter optimization in Python*. Avalible at: http://hyperopt.github.io/hyperopt/ (accessed 11 August 2019).
21. **Hochreiter S., Schmidhuber J.** Long short-term memory. *Neural Computation*, 1997, vol. 9, no. 8, pp. 1735–1780.
22. **Randal S. Olson, Ryan J. Urbanowicz, Peter C. Andrews, Nicole A. Lavender, La Creis Kidd, and Jason H.** Automating biomedical data science through tree-based pipeline optimization. *Applications of Evolutionary Computation*, 2016, pp. 123–137.
23. *Auto-Sklearn*. Avalible at: https://automl.github.io/auto-sklearn/master/ (accessed 11 August 2019).
24. *Auto-Keras: An efficient neural architecture search system*. Avalible at: https://arxiv.org/abs/1806.10282v3 (accessed 11 August 2019).
25. *GloVe: Global vectors for word representation*. Avalible at: https://nlp.stanford.edu/projects/glove/ (accessed 11 August 2019).
26. *Extending a parser to distant domains using a few dozen partially annotated examples*. Avalible at: https://arxiv.org/abs/1805.06556v1 (accessed 11 August 2019).
27. *Universal language model fine-tuning for text classification*. Avalible at: https://arxiv.org/abs/1801.06146v5 (accessed 11 August 2019).
28. *BERT: Pre-training of deep bidirectional transformers for language understanding*. Avalible at: https://arxiv.org/abs/1810.04805v2 (accessed 11 August 2019).

УДК 004.056
doi:10.31799/1684-8853-2020-1-2-14

Исследование обобщенного подхода к решению задач анализа настроений коротких текстовых сообщений в задачах обработки естественного языка

Е. В. Поляков[а], аспирант, orcid.org/0000-0002-6143-1408, epolyakov@hse.ru
Л. С. Восков[а], канд. техн. наук, профессор, orcid.org/0000-0002-4008-8807, lvoskov@hse.ru
П. С. Абрамов[а], студент, orcid.org/0000-0002-7079-0613, psabramov@edu.hse.ru
С. В. Поляков[б], аспирант, orcid.org/0000-0002-2151-6570, s.polyakov@mai.ru
[а]Национальный исследовательский университет «Высшая школа экономики», Мясницкая ул., 20, Москва, 101000, РФ
[б]Московский авиационный институт (национальный исследовательский университет), Волоколамское ш., 4, Москва, 125993, РФ

Введение: определение тональности текста — сложная проблема, решение которой существенно зависит от контекста, области исследования и объема текстовых данных. Проведенный анализ публикаций показывает, что авторы в своих работах не используют полный спектр возможных преобразований над данными и их комбинаций. Используется только некоторая часть преобразований, что не позволяет в полной мере разрабатывать модели высокого качества классификации. **Цель:** разработка и исследование обобщенного подхода к построению модели, который заключается в последовательном прохождении этапов разведочного анализа, получения базового решения, векторизации, предобработки, настройки гиперпараметров и моделирования. **Результаты:** сравнительные эксперименты, проведенные с применением обобщенного подхода для классических алгоритмов машинного обучения и глубокого обучения к решению задачи анализа настроений коротких текстовых сообщений в области обработки естественного языка, показали динамику роста качества классификации от этапа к этапу. Для классических алгоритмов такой рост качества был незначительным, но для глубокого обучения прирост качества на каждом этапе в среднем составил 8 %. Проведение дополнительных исследований показало, что использование автоматического машинного обучения, в котором применяются классические алгоритмы классификации, сопоставимо по качеству с ручной разработкой модели, однако занимает намного больше времени. Использование переноса обучения оказывает небольшой, но положительный эффект на качество классификации. **Практическая значимость:** предложенный последовательный подход позволяет существенно повысить качество разрабатываемых моделей в задачах обработки естественного языка.

Ключевые слова — обработка естественного языка, машинное обучение, глубокое обучение, векторизация, моделирование, предварительная обработка, автоматическое машинное обучение, перенос обучения.

Для цитирования: Polyakov E. V., Voskov L. S., Abramov P. S., Polyakov S. V. Generalized approach to sentiment analysis of short text messages in natural language processing. *Информационно-управляющие системы*, 2020, № 1, с. 2–14. doi:10.31799/1684-8853-2020-1-2-14

For citation: Polyakov E. V., Voskov L. S., Abramov P. S., Polyakov S. V. Generalized approach to sentiment analysis of short text messages in natural language processing. *Informatsionno-upravliaiushchie sistemy* [Information and Control Systems], 2020, no. 1, pp. 2–14. doi:10.31799/1684-8853-2020-1-2-14