Russian sarcastic comments detection

Trenev Ivan, Chaplinskaya Nadezhda

December 2022

Abstract

This project proposes a solution to the problem of Russian sarcastic comments detection. A corpus of Russian-language comments was formed with binary markup: whether the comment is sarcastic or not. The problem was solved using three natural language processing approaches: TF-IDF and Logistic Regression, Recurrent Neural Networks, Transformers. A comparative analysis of the results of these approaches was carried out.

`https://github.com/123-39/NLP_Project`.

1 Introduction

Sarcasm is often used to convey, with the help of irony, the opposite of the literal meaning of the sentence. Previously, sarcasm was used almost exclusively in verbal communication due to the need for vocal connotations to certain words in a sentence in order to help the listener understand the irony. But with the advent of social media and the increase in text-based communication, the use of sarcasm in natural language has become more common in written form as well.

Sarcasm detection nowadays becomes one of the popular areas of sentimental text analysis. Correct identification of sarcasm by a machine is a necessary step not only in text communication between people, but also in the tasks of free communication between a machine and a person.

There are many works that offer approaches to learning to recognize sarcasm. Most of them use datasets of English comments and posts collected primarily from Reddit and Twitter. In Russian, the problem of defining sarcasm has not yet been solved. In this study, we prepared a Russian-language marked up corpus of sarcastic and ordinary comments, and presented and compared several trained models capable of recognizing sarcasm in Russian.

1.1 Team

This project was prepared by Trenev Ivan and Chaplinskaya Nadezhda. Approximate distribution of responsibilities is considered below.

Trenev Ivan dataset creating, utilizing the Transformer concept, utilizing the RNN concept.

Chaplinskaya Nadezhda dataset creating, data preprocessing and utilizing the Logistic Regression concept.

2 Related Work

The first robust algorithm used to detect sarcasm was developed in 2010 [1]. It was a semi-supervised sarcasm identification (SASI) algorithm for detecting sarcasm in Twitter and Amazon product reviews, using features based on patterns and punctuation in tweets. Researchers led by González-Ibáñez studied lexical and pragmatic features in unigram and dictionary-based tweets to classify sarcastic, positive, and negative tweets using two classifiers – support vector machine (SVM) and logistic regression (LogR) – in 2011 [2]. Lukin and Walker applied the Bootstrapping method to detect sarcasm and anger in online dialogue using pattern-based features in 2013 [3].

In 2014, a new multi-strategy ensemble learning approach (MSELA) was proposed to detect sarcasm in both English and Chinese social networks [4]. The scientists extracted different sets of features for English and Chinese texts, where the features of English sarcasm consisted of punctuation marks, lexical and syntactic features, and the features of Chinese sarcasm included thetorical, homophonic, construction features. In the same year Barbieri and his team determined the sarcasm of tweets by adding characteristics such as frequency, written-spoken, intensity, structure, sentiment, synonyms, and ambiguity [5].

A lot of articles devoted to the sarcasm recognition theme were published in 2015. Fersini and a team of researchers presented an ensemble approach, Bayesian Model Averaging (BMA) for detecting sarcasm and irony in microblogs, taking into account PoS tags [6]. A group of researchers led by Rajadesingan built a behavioral modeling framework for detecting sarcasm using a list of features depending on different forms of sarcasm [7]. A parse-based vocabulary generation algorithm (PBLGA) was proposed, on the task of recognizing sarcasm in tweets using hyperbole features and an NLP preprocessing method: PoS tagging [8]. Bamman and Smith used the binary logistic regression algorithm [9] for the task of recognizing sarcasm, while extralinguistic information was added: tweet characteristics, author characteristics, audience characteristics, environment characteristics.

In the same 2015, the task of detecting sarcasm was solved by Cocea and his team on a dataset of student tweets. Machine learning methods and n-grams, unigrams features were used along with additional features such as emotion label, polarity label, number of hashtags, etc. [10]. The sarcasm detection task has been reformulated as literal/sarcastic sense disambiguation (LSSD) by Ghosh and his team [11]. They explored twitter data using unsupervised methods and SVM classifier with a modified kernel.

In 2016, Bouazizi and Otsuki presented a pattern-based approach using four feature sets for sarcasm detection: punctuation-related, mood-related, pattern-related, syntactic and semantic [12]. Also in this year Hadoop-based framework and big data approach was developed by Bharti et al. for real-time sarcasm detection based on parsing and PoS tagging [13].

Sarcasm recognition topic is still popular nowadays. A lot of other researches have been published since that time. Although there is actually no one of them to be about Russian sarcastic comments recognition.

3 Dataset

To build the corpus of Russian sarcastic and non-sarcastic comments we decided to use the text-based English news headlines corpus[1]. We have translated to Russian and then we have cleared all the samples. As the result we have got 9554 sarcastic and 10479 non-sarcastic translated samples.

But we had the idea that the corpus we want has to be more «Russian», has to have some exclusively Russian sarcastic statements that appears due to Russian mentality. That's why we decided to use also another corpus – the corpus of Russian jokes[2]. The corpus is huge: 130204 samples. These jokes we would like to consider as Russian sarcastic statements, because every Russian joke has Russian-sarcastic emotional background.

Finally, the corpus we made consisted of 5000 translated sarcastic samples and 5000 Russian jokes – as sarcastic comments (labelled 1), and 10000 translated non-sarcastic samples as non-sarcastic comments (labelled 0). Thus we have get the corpus of 20000 balanced comments (Fig. 1.)

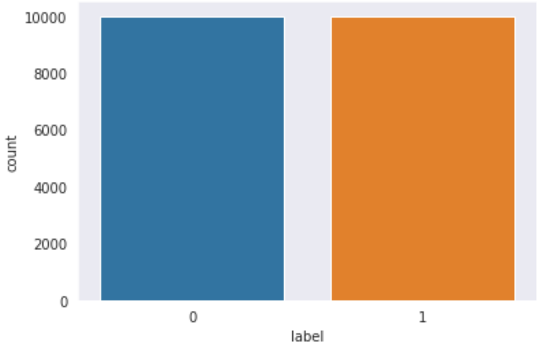


Figure 1: The histogram of sarcastic and non-sarcastic comments distribution in created corpus.

The split of the dataset to the train, valid and test samples is presented on the Tab. 1.

[1] A link to the website, where the text-based English news headlines corpus could be downloaded: here.

[2] A link to the website, where the corpus of Russian jokes could be downloaded: here.

	Train	Valid	Test
Comments	11200	2800	6000
Tokens	190400	36400	120000

Table 1: The distribution of samples for train, test and validation sets.

4 Model Description

We decided to use three train models: Logistic Regression, Recurrent Neural Network and Transformer.

4.1 Logistic Regression

As a simple approach for text classification, we have chosen logistic regression over TF-IDF embedding. There is no previous art on the problem we are presenting – Russian sarcasm detection, – so this approach is going to be considered as our baseline.

First we have preprocessed the data. The preprocessing consisted of removing links and removing text in brackets using regular expressions library – **re**. Also it was necessary to tokenize samples and remove stop-words, but these steps are included in TF-IDF realization.

Then **TfidfVectorizer** from the library **sklearn** was used to get a number for each word. The goal of using TF-IDF is to scale down the impact of tokens that occur very frequently in a given corpus and that are hence empirically less informative than features that occur in a small fraction of the training corpus.

The formula describing TF-IDF embedding:

$$x_{ik} = f_{ik} \log \frac{N}{n_i},$$

where i – word index, k – comment index, f_{ik} – word frequency in a comment, N – number of comments in the corpus, n_i – number of comments containing the word.

The resulting vectors of original comments samples became the inputs to the **LogisticRegression** model imported from **sklearn** library.

The decision boundary that distinguish two classes (sarcastic and ordinary comment) can be presented as equation:

$$d_\theta(x) = \theta^T x = 0,$$

where x – is the input vector of words after TF-IDF embedding, θ – the vector of weights of these words.

In Logistic Regression model words' weights are fitting to minimize the convex cost function $J(\theta)$:

$$J(\theta) = -\frac{1}{n} \sum_{i=1}^{n} (y^i \log(h_\theta(x^i)) + (1 - y^i) \log(1 - h_\theta(x^i))),\ y^i \in \{0,\ 1\},$$

where $h_\theta(x)$ is logistic function

$$h_\theta(x) = \frac{1}{1+e^{-d_\theta(x)}} = \frac{1}{1+e^{-\theta^T x}}.$$

4.2 Recurrent Neural Network

The next step in our study was the use of a recurrent neural network (RNN) — class of artificial neural network where connections between nodes form a directed graph along a sequence. This allows it to exhibit dynamic temporal behavior for a time sequence.

As a prepaid embedding, it was attempted to use a relatively good quality, and most importantly, lightweight glove model «navec»[3].

Using the glove model, we will build a dictionary and tokenize (converting text strings into sequences of numbers) every comment. In this case, the size of the vocabulary was limited to 30,000 words.

4.3 Transformer

The transformer-based language models have been showing promising progress on a number of different natural language processing (NLP) benchmarks. The combination of transfer learning methods with large-scale transformer language models is becoming a standard in modern NLP. Therefore, the next stage of our research was the use of pre-trained transformers of the BERT type.

The rubert-tiny2 (sentence encoder model) model was chosen for classification. Since at the moment, among the Russian-language models of the sentence encoder, this model wins in terms of the balance of speed and quality

As a result, the pre-trained tokenizer and model were loaded from huggingface[4]. For classification, it was necessary to add a fully connected layer, the number of inputs of which is the internal dimension of the embedding of the network, and the output is the number of classes for classification

5 Experiments

5.1 Metrics

In this project the F1-score was used as metric to evaluate the results of each model.

F1-score is calculated as the harmonic mean of the Precision and Recall:

$$F1_{score} = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall}.$$

[3] navec glove model: here.

[4] rubert-tiny2 model: here.

Here the Precision shows the number of true positive (TP) outcomes from the entire set of positive model answers (TP + FP):

$$Precision = \frac{TP}{TP + FP}.$$

The Recall shows the number of true positive (TP) outcomes from the entire set of positive actual samples (TP + FN):

$$Recall = \frac{TP}{TP + FN}.$$

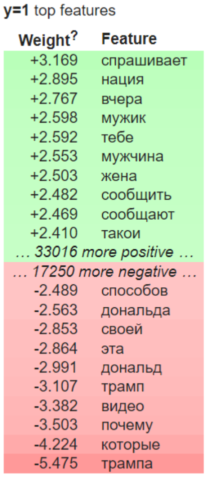
5.2 Experiment Setup

5.2.1 Logistic Regression

For **TfidfVectorizer** we have set the parameter **stop words** as Russian stop-words list that was downloaded from **nltk** library. The parameter **ngram range** was set to (1,1) which means we are using only only unigrams. Also we set the parameter **lowercase** to the value True to make all the words being written in lowercase. The received TF-IDF embeddings are between [0,1].

For **LogisticRegression** we have set the parameter **max iter** to 10000 and the parameter **n jobs** to -1, which means we want to use all available processors (CPU cores).

The final weights received by the Logistic Regression method can be presented in the descending order with the help of **eli5** library. You could see them on the Fig. 2. The positive weights make a bigger contribution to the sarcastic labelling, when the negative weights make a bigger contribution to the non-sarcastic labelling.



y=1 top features

Weight?	Feature
+3.169	спрашивает
+2.895	нация
+2.767	вчера
+2.598	мужик
+2.592	тебе
+2.553	мужчина
+2.503	жена
+2.482	сообщить
+2.469	сообщают
+2.410	такои
… 33016 more positive …	
… 17250 more negative …	
-2.489	способов
-2.563	дональда
-2.853	своей
-2.864	эта
-2.991	дональд
-3.107	трамп
-3.382	видео
-3.503	почему
-4.224	которые
-5.475	трампа

Figure 2: The final weights distribution in Log-Reg method.

5.2.2 Recurrent Neural Network

RNN classifier consists of:

1. Bidirectional GRU (Gated recurrent units). 3 hidden layers of size 100 with dropout coefficient 0.1.
2. Linear layers with dropout coefficient 0.1;
3. Sigmoid activation function.

Params:

1. Loss: CrossEntropyLoss().
2. Optimizer: Adam() (learning rate=$1e-3$).
3. Number epoch: 10.
4. Weight decay: $1e-6$.
5. Batch size: 64.

5.2.3 Transformer

Params:

1. Loss: CrossEntropyLoss().
2. Optimizer: AdamW() (learning rate=$2e-5$).
3. Sheduler: get_linear_schedule_with_warmup().
4. Number epoch: 10.
5. Batch size: 64.
6. Max length: 512.

6 Results

The confusion matrix for the Logistic Regression approach is presented in a Fig. 3.

The confusion matrix for the Recurrent Neural Network approach is presented in a Fig. 4.

The confusion matrix for the Transformer approach is presented in a Fig. 5.

The evaluations (F1-score) of presented methods on the testing set are showed in a Tab. 2.

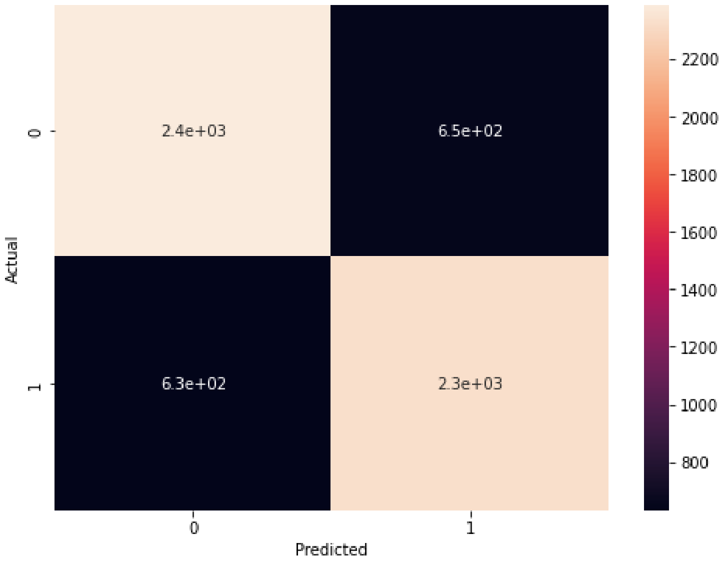


Figure 3: The confusion matrix for Log-Reg method.

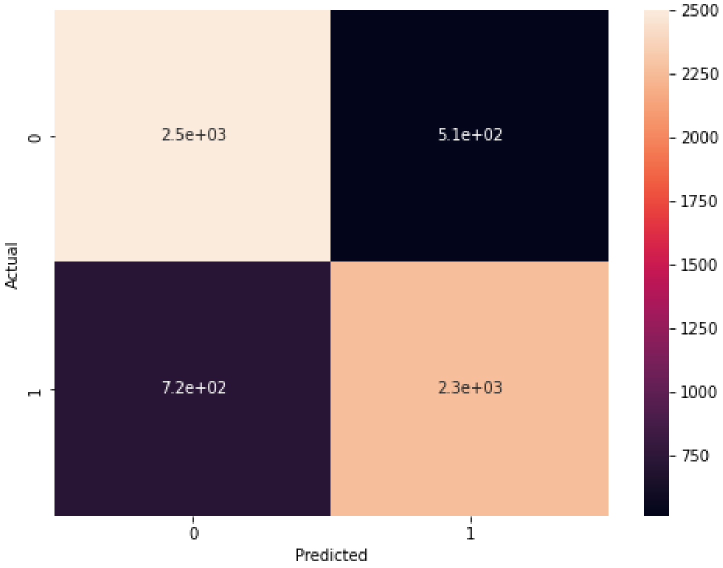


Figure 4: The confusion matrix for RNN method.

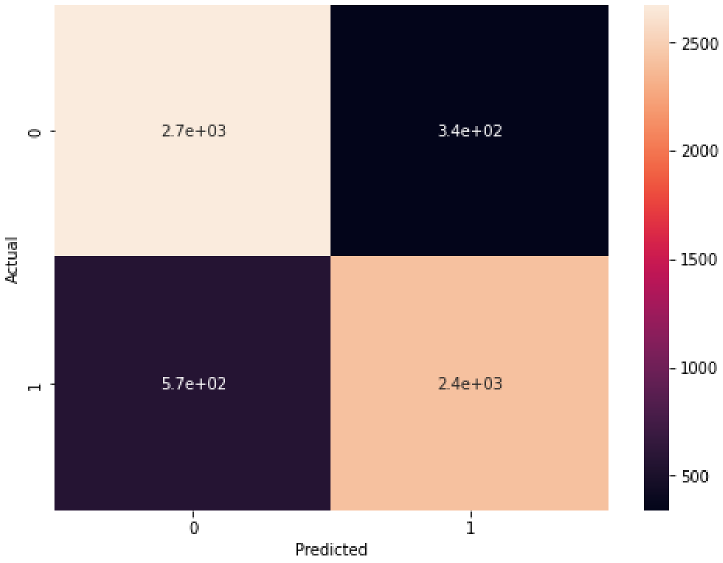


Figure 5: The confusion matrix for Transformer method.

Method	F1-score
Log-Reg	0.7868
RNN	0.7940
Transformer	0.8478

Table 2: The F1-score for each method.

7 Conclusion

This project proposes a solution to the problem of Russian sarcastic comments detection. A corpus of Russian-language comments was formed, using the translated binary labelled English sarcastic comments and Russian jokes which were considered as additional sarcastic comments. The problem was solved using three natural language processing approaches: TF-IDF and Logistic Regression, Recurrent Neural Networks, Transformers. A comparative analysis of the results of these approaches showed that the third approach – Transformers – has the higher score of detection, than Recurrent Neural Networks which showed the result close to the score of the Logistic Regression.

In respect that the presented model is the first model capable to detect Russian sarcasm, the received results are quite encouraging for the further researches connected to this field.

References

[1] Dmitry Davidov, Oren Tsur, Ari Rappoport *Semi-supervised recognition of sarcastic sentences in twitter and amazon* // Proceedings of the Fourteenth Conference on Computational Natural Language Learning, Association for Computational Linguistics, 2010.

[2] Roberto González-Ibánez, Smaranda Muresan, Nina Wacholder *Identifying sarcasm in Twitter: a closer look* // Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2, Association for Computational Linguistics, 2011.

[3] Stephanie Lukin, Marilyn Walker *Really? well. apparently bootstrapping improves the performance of sarcasm and nastiness classifiers for online dialogue* // Proceedings of the Workshop on Language Analysis in Social Media, 2013.

[4] Peng Liu, et al. *Sarcasm detection in social media based on imbalanced classification* // International Conference on Web-Age Information Management, Springer International Publishing, 2014.

[5] Francesco Barbieri, Horacio Saggion, Francesco Ronzano *Modelling sarcasm in twitter, a novel approach* ACL 2014, 2014.

[6] Elisabetta Fersini, Federico Alberto Pozzi, Enza Messina *Detecting irony and sarcasm in microblogs: The role of expressive signals and ensemble classifiers* // Data Science and Advanced Analytics (DSAA), 2015.

[7] Ashwin Rajadesingan, Reza Zafarani, Huan Liu *Sarcasm detection on twitter: A behavioral modeling approach* // Proceedings of the Eighth ACM International Conference on Web Search and Data Mining, ACM, 2015.

[8] Santosh Kumar Bharti, Korra Sathya Babu, Sanjay Kumar Jena *Parsing-based sarcasm sentiment recognition in Twitter data* // 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM). IEEE, 2015.

[9] David Bamman, Noah A. Smith *Contextualized sarcasm detection on twitter* // Ninth International AAAI Conference on Web and Social Media, 2015.

[10] Nabeela Altrabsheh, Mihaela Cocea, Sanaz Fallahkhair *Detecting sarcasm from students' feedback in Twitter* // Design for teaching and learning in a networked world, Springer International Publishing, 2015.

[11] Debanjan Ghosh, Weiwei Guo, Smaranda Muresan *Sarcastic or not: word embeddings to predict the literal or sarcastic meaning of words* // Procceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Accociation for Computational Linguistics, 2015.

[12] Mondher Bouazizi, Tomoaki Otsuki *A pattern-based approach for sarcasm detection on twitter* // IEEE Transl., 2016.

[13] S.K. Bharti, et al. *Sarcastic sentiment detection in tweets streamed in real time: a big data approach* // Digital Communications and Networks, 2016.