Automatic Irony and Sarcasm Detection in Russian Sentences: Baseline Methods

Maksim Kosterin, Ilya Paramonov, Nadezhda Lagutina
P.G. Demidov Yaroslavl State University
Yaroslavl, Russia
makcost@gmail.com, ilya.paramonov@fruct.org, lagutinans@gmail.com

***Abstract*—The paper describes experiments performed on two sets of manually annotated data. The task of irony and sarcasm detection in Russian sentences was solved using baseline classifiers, i. e., BERT, Bi-LSTM, SVM, Random Forest, Logistic Regression. The best achieved F1-score for each classifier was 0.76, 0.73, 0.66, 0.64, 0.68 respectively. The results achieved by BERT and Bi-LSTM classifiers are comparable with the results from the articles describing the application of similar approaches for English language. Analysis of the results allowed to conclude that transferring the word context improves classification metrics and refinement of training data allows to improve the classifier's performance.**

I. Introduction

Automatic irony and sarcasm detection is a natural language processing (NLP) task that is usually considered as two-class classification (ironic and non-ironic) of single sentences or text fragments. This task is rather complicated due to the fact that irony and sarcasm can be expressed in a variety of ways, sometimes not obvious even for humans. However, there was a considerable progress achieved in this area recently, mainly due to the development of deep neural networks in application to NLP [1].

Despite the significant success in solving the task of irony and sarcasm detection for English, there is only a small amount of research devoted to the other languages. Partially this is caused by the lack of annotated corpora for this task in non-English languages. On the other hand, even for English the majority of annotated corpora are based on texts from Twitter, which reflect only a narrow and specific part of a natural language.

This paper is devoted to assessment of baseline methods of automatic irony and sarcasm detection in Russian sentences. By baseline methods we consider traditional classifiers (e.g., SVM or Random Forest), as well as BERT and Bi-LSTM neural networks that are often used to solve this task for texts in English. The reason we stick to these methods and do not consider more complex ones (e.g., ensemble classifiers or using linguistic features) is that the area of automatic irony and sarcasm detection is highly underexplored for Russian language [2] and it is beneficial to find out how efficient can be baseline methods before diving deeper.

The paper is structured as follows. Section II gives state-of-the-art on the topic of automatic irony and sarcasm detection. In Section III the collection and annotation of the corpora used in this research are described. Section IV expands on the methods used in experiments, whereas their results are given in Section V. The discussion of the results in comparison with state-of-the-art works for English language using the same methods is presented in Section VI. Conclusion summarizes the paper and highlights directions for future research.

II. State-of-the-art

The most widely used method to detect irony and sarcasm is to map each text to a vector of features with subsequent binary classification of such vectors. The feature vectors are usually constructed with the use of embeddings, primarily generated by Word2Vec, GloVe, BERT and its varieties. The systematic review [1] provides a good vision of the research landscape in this area until May 2021, however it only covers the works related to English language.

A. Agrawal et al. [3] used BERT and XLNets neural networks to automatically detect irony on the SemEval-2018 text corpus containing 3817 English tweets. The resulting F1-score was 70% and 74% for BERT and XLNets respectively. The same corpus was used by C. Turban and U. Kruschwitz [4], who utilized RoBERTa, which allowed to improve the F1-score up to 80%.

Within the framework of the Second Workshop on Figurative Language Processing the researchers proposed several solutions on the topic of the automatic sarcasm detection task with the use of BERT embeddings. The experiments have been conducted using two corpora of texts, extracted from the social networks Twitter and Reddit. Corpora contained 5000 and 4400 documents ranging from 20 to 1200 words respectively. A. Baruah et al. [5] showed the superiority of BERT in comparison with other classifiers, particularly LSTM and SVM. The resulting F1-score was 74.3% for Twitter and 65.8% for Reddit. Similar results with different versions of BERT (RoBERTa, spanBERT) were achieved by A. Kumar and V. Anand [6]. The F1-score was 77.2% for Twitter and 69.1% for Reddit. On average the results of participants of the Workshop were about 75% for Twitter and 5–6% less for Reddit.

It should be mentioned that practically all research on the task of automatic irony and sarcasm detection in English are based on corpora from Twitter or Reddit automatically annotated through the distant supervision approach [7]. It utilizes API of corresponding social networks and hashtags, e.g., #sarcasm, #irony, #sarcastic, #not and others, as the markers of containing irony or sarcasm in the corresponding texts. This approach is very successful due to high popularity of Twitter in English-speaking world and simplicity of getting and automatic processing high amounts of data. However, for Non-English languages this approach faces some troubles. For example, P. Golazizian et al. [8] mentioned that automatic annotation of Persian texts from Twitter is difficult, because of the absence of appropriate hashtags. Similar troubles are inherent for Russian: the way, in which hashtags like #irony and #sarcasm are used in Russian texts from Twitter, does not allow to make a robust automatic annotation [9]. Meanwhile, manual annotation is very rare in modern research because of high complexity and labor costs and used only if it is not possible to annotate a corpus automatically.

Some researchers use various lexical features of text to detect sarcasm or supplement classical embeddings with them to improve the quality of detection. Just lexical features (interjections, punctuation marks, capital letters, intensifiers, elongated words) are used by V. Govindan and V. Balakrishnan [10], whose best results were achieved using Random Forest with accuracy of 78.74%. It is worth mentioning that this is one of the few works where the corpus was annotated by experts in linguistics. W. Chen et al. [11] used a combination of sentiments and incongruity (a contradiction between conveyed sentiments and context semantics) to detect sarcasm. Classification was done using concatenated vectors that consist of sentiment features determined by LSTM and incongruity features being a result of the semantics analysis of sentences. The achieved F1-score was 73.85% for the Reddit corpus and 77.19% for the IAC-V2 dataset constructed from texts of political debates.

Unlike English, there is only a small amount of research devoted to automatic irony and sarcasm detection in non-English languages. A. Wadhawan [12] used AraBERT—a BERT model for Arabian language and achieved F1-score of 72% for a corpus of tweets. Participants of the SemEval-2022 contest showed even worse results [13]. Similar contest was conducted for Spanish for the task of automatic detection of humor [14], the best F1-score was 71%. Linguistic features and traditional classifiers were used by Z. B. Nezhad and M. A. Deihimi [15] for automatic detection of sarcasm in Persian language with the best result of 80% achieved by SVM. In another research for the same language [8] the researchers manually annotated a corpus of 3000 tweets and detected irony using multi-layer neural network with accuracy of 83%.

There are practically no research on the automatic detection of irony and sarcasm in Russian [2]. The most close one is devoted to detection of humor [16]. The authors used Word2Vec embeddings in combination with lexical features, such as POS tags, average word length and punctuation. The classification was made by SVM, the best achieved F1-score was 88%. The authors manually annotated a corpus of 100000 texts containing jokes from social networks that constitute a positive class and stories, news and proverbs that constitute a negative class. It is most likely that high results were caused by significant stylistic differences between the classes, which would not remain in case of more homogeneous texts.

III. Corpora

In this research experiments were conducted using two datasets constructed from OpenCorpora (http://opencorpora.org), an open corpus of Russian news, analytical and opinion articles.

Dataset A was built upon a subset of texts from the OpenCorpora corpus annotated by a group of 14 volunteers into two classes: ironic and non-ironic. The ironic class also includes sentences with sarcasm, which can be considered a form of irony [17]. Annotation procedure was conducted in two rounds. Every round each volunteer was given 2000 randomly selected sentences to determine if a sentence is ironic without knowing its context. Sentences were distributed in such a way that every sentence was assigned to 2 reviewers. On average each reviewer marked 4.7% of sentences as ironic. The results were then combined into a single dataset. 1672 sentences labeled as ironic by at least one reviewer were picked from the combined set as positive samples and 1672 negative samples were randomly selected from a larger set of sentences labeled as non-ironic. In total, this dataset consists of 3344 sentences separated into two equally-sized classes. The length of sentences varies from 4 up to 66 or 106 tokens for ironic and non-ironic class respectively, with average of 16 tokens.

As Dataset A was annotated by volunteers, its labels were inaccurate. That is why Dataset B was constructed. It contains a subset of Dataset A, which sentences were validated by an expert. Out of 1672 positive sentences, 964 were labeled as true positive, 708 as false positive. Dataset B thus contains 964 positive samples that were validated as ironic, and 964 negative samples that were randomly selected from all non-ironic sentences of Dataset A, for a total of 1928 sentences.

The reason for using both Dataset A and Dataset B in experiments is to determine whether deep neural models like BERT that are naturally "hungry" for training data would favor scarce data with better quality over better quantity. It also allows us to understand how much of an increase in classifier's performance can be achieved by validating the data we feed it.

IV. Methods

In this research we use a range of classifiers to measure their performance on the task of automatic irony and sarcasm detection. This range includes BERT, Bi-LSTM, SVM, Random Forest and Logistic Regression. The choice of these classifiers for our experiments stems from our goal to determine the best baseline methods for detecting irony and sarcasm in Russian texts. They were selected solely based on how well they were able to perform on their own for the task of sarcasm detection in English texts. Because of that, we, for example, did not consider hybrid models that pair multiple classifiers in a single pipeline. Additionally, we investigated whether the use of pre-trained word embeddings lead to increasing the methods' performance.

A. RuBERT

In this paper we use RuBERT—a BERT-based model for Russian language provided by DeepPavlov [18]. It was trained on a set of texts from the Russian part of Wikipedia and news outlets. Just like the regular BERT-base model, this model has 12 transformer blocks, the output vector size of 768 and the number of self-attention heads is 12. Unlike the original model, it is case-sensitive and has 180M parameters instead of 110M ones. The classification layer of the model is a fully-connected layer with the softmax activation function. The model is fine-tuned using the Adam optimizer with the learning rate of $2 \cdot 10^{-5}$ and the binary cross-entropy function for calculating loss.

Since this research relies on BERT implementation from J. Devlin et al. [19], the classification layer in built using TensorFlow 1.15 functional API (https://www.tensorflow.org). Hence, RuBERT's model for TensorFlow is used.

B. Bi-LSTM

In this research Bi-LSTM classifier is implemented in two variants. Both of those were built using the sequential model from Keras API (https://keras.io) for TensorFlow 2.0.

The first one is built as a regular bidirectional LSTM classifier with self-trainable embedding layer. It is shown in Fig. 1. The Model's pipeline starts with an encoder layer (1) that takes a raw text sequence, tokenizes it into words and produces vectors of word indices padded to 110 elements. The next layer is a trainable embeddings layer (2) that converts the sequences of word indices to sequences of 300-dimensional vectors. The embedding layer is followed by a bidirectional LSTM with one forward LSTM layer (3) and one backward LSTM layer (4) with 110 units each. Bi-LSTM layer's output is fed to a fully-connected layer (5) with 110 units that converts Bi-LSTM output to a single 110-dimensional vector using the Rectified Linear Units (ReLU) activation function. After that a 110-dimensional vector is converted to a single prediction score.

The model is trained using the Adam optimizer with the learning rate of 10^{-4} and the binary cross-entropy function for calculating loss. Since this model operates directly on a text sequence, there is no separate preprocessing stage outside of the encoding step (1).

The second Bi-LSTM model uses a pre-trained word embeddings dictionary provided by the spaCy 3 *ru_core_news_lg* model (https://spacy.io/models/ru#ru_core_news_lg). The Bi-LSTM model described above was modified in such a way that the text encoding step was moved from the model pipeline to the preprocessing stage. The modified structure is shown in Fig. 2. As a result, the model directly takes word indices while the Embedding layer (1) is initialized with a precompiled embedding matrix where word indices are matched against 300-dimensional word embeddings. Other than that, this model follows the same structure as the first variant.

During the preprocessing step a text sequence is tokenized into words that are used to build the model's vocabulary. Each word in the vocabulary is assigned an index based on how often the word appears in the dataset. After the vocabulary has been built, the embedding matrix is created by assigning a pre-trained word embedding to every word index in a vocabulary. If a word does not have a corresponding word embedding in the dictionary, it is assigned a zero-filled vector. Once the embedding matrix is initialized, word tokens in input sequences are replaced with their word indices. Finally, the embedding matrix is passed to the Embedding layer (1) of our model as a constant initializer for the layer's vectors. The proposed model is then trained on the prepared data.

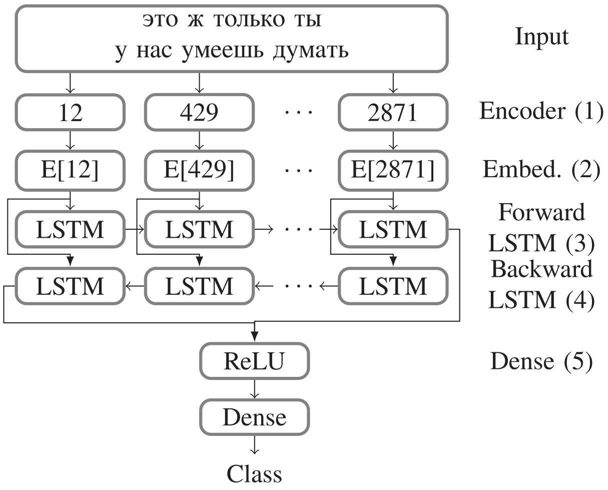

Fig. 1. Bi-LSTM without dictionary

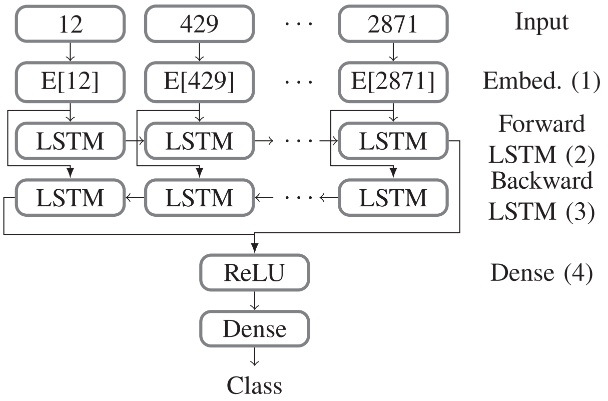

Fig. 2. Bi-LSTM with dictionary

C. SVM, Random Forest, Logistic Regression

The traditional machine learning algorithms used in this research are: Support Vector Machine, Random Forest and Logistic Regression. These algorithms, like Bi-LSTM models, utilize word embeddings as their input values. The transformation from a text sequence to a concatenation of embeddings is done during the input preprocessing step as shown in Fig. 3. At first, word tokenisation is performed on the input text sequence. Each word then matched with the corresponding word embedding from the dictionary. Finally, word embed-

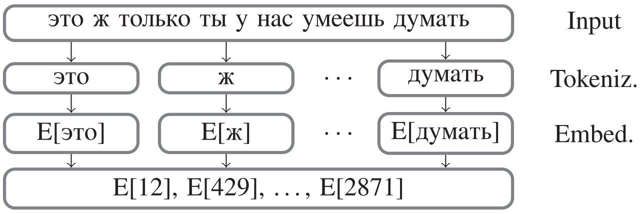
это ж только ты у нас умеешь думать
Input
это
ж
· · ·
думать
Tokeniz.
E[это]
E[ж]
· · ·
E[думать]
Embed.
E[12], E[429], . . . , E[2871]

Fig. 3. Input pipeline for traditional classifiers

dings are concatenated into a single numeric sequence in the same order that the corresponding words appear in the original text sequence. The resulting vector then passed directly into the classifier.

Scikit-learn [20] is the machine learning library of choice for our experiments with traditional classifiers. It provides an out-of-the-box implementation for a wide variety of machine learning algorithms. For the SVM classifier, the *sklearn.svm.SVC* class with the "linear" kernel was chosen. Regularization parameter was set to 1.0. Classification with Random Forest algorithm was done through the *sklearn.ensemble.RandomForestClassifier* class. As for the Logistic Regression, we used *sklearn.linear_model.LogisticRegression* with the *max_iter* parameter set to 10000.

V. Results

The experiment results for Datasets A and B are shown in Tables I and II respectively. All the experiments were conducted using 5-fold cross-validation.

It follows from the results that the neural network models using transfer learning in the form of pre-trained word embeddings (i.e., BERT, Bi-LSTM with vocabulary, SVM, Random Forest and Logistic Regression) outperform those that self-train their own embeddings during the training step (Bi-LSTM with trainable embeddings). This shows that utilizing contextual language model alone can increase the efficiency of a classifier.

As expected, BERT shows the best overall results for both datasets by achieving F1-score of 0.74 on Dataset A and 0.76 on Dataset B. This can be attributed to both applying transfer learning as a powerful tool in contextualizing word embeddings and utilizing attention mechanism to "emphasize" the most impactful words for expressing the ironic intent. Bi-LSTM model that uses pre-trained word embeddings achieves F1-scores of 0.71 and 0.73 for Datasets A and B respectively, thus falling slightly behind BERT by 3%, but pulling far ahead of Bi-LSTM with self-trainable word embeddings. The improvement over Bi-LSTM model with self-trained embeddings comes solely from introducing an element of transfer learning since both models are almost identical in their structure and training parameters.

Traditional machine learning classifiers (SVM, Random Forest and Logistic Regression) also managed to outperform Bi-LSTM model with self-trainable word embeddings once again due to using pre-trained word embeddings. Their F1-scores are 0.62, 0.64, 0.66 on Dataset A and 0.66, 0.64, 0.68 on Dataset B for SVM, Random Forest and Logistic Regression respectively.

Results shown by both BERT and Bi-LSTM baseline models serve a good-enough starting point for future experiments as there are many available options for their improvement into more complex models. BERT can be scaled by adopting a bigger model such as RuBERT large (https://huggingface.co/sberbank-ai/ruBert-large), or a model tuned to a specific task. Bi-LSTM can be expanded by adding new layers and introducing attention mechanism. There is also an approach of combining both models into a single ensemble model that has been popular in the recent works [1].

The results achieved on Dataset B tend to be slightly higher by about 2-4% than on Dataset A except for Random Forest. This is a direct consequence of Dataset B having better quality of data due to the undergoing validation procedure. The increase in metrics achieved by the classifiers comes notwithstanding the decrease in the size of Dataset B compared to Dataset A. Variance of results between a higher quality dataset and a lower quality dataset allows us to draw two conclusions. First, as B. Moores and V. Mago [25] noted, it is hard to objectively compare results shown in different research as they may be reliant on entirely different sets of data ranging not only by quality, but also by genre. For example, it can be far easier to detect ironic intent in a social media post than in a news article. Thus, an approach of utilising the same dataset across different experiments should

TABLE I. Experiment results on Dataset A

Metric Method	Precision	Recall	F1-score
RuBERT	0.73	0.74	0.74
Bi-LSTM with voc.	0.75	0.69	0.71
Bi-LSTM without voc.	0.57	0.64	0.60
SVM	0.62	0.63	0.62
Random Forest	0.61	0.67	0.64
Logistic Regression	0.66	0.66	0.66

TABLE II. Experiment results on Dataset B

Metric Method	Precision	Recall	F1-score
RuBERT	0.73	0.79	0.76
Bi-LSTM with voc.	0.74	0.72	0.73
Bi-LSTM without voc.	0.67	0.62	0.64
SVM	0.66	0.66	0.66
Random Forest	0.61	0.68	0.64
Logistic Regression	0.68	0.68	0.68

TABLE III. COMPARISON OF RESULTS WITH OTHER RESEARCH

Reference	Dataset	Classifier	Precision	Recall	F1-score	Dataset size	Dataset annotation method	
							Manual	Self-annotated
this work	Dataset A	RuBERT	0.73	0.74	0.74	3344	✓	
		Bi-LSTM	0.75	0.69	0.71			
		SVM	0.62	0.63	0.62			
	Dataset B	RuBERT	0.73	0.79	0.76	1928	✓	
		Bi-LSTM	0.74	0.72	0.73			
		SVM	0.66	0.66	0.66			
[5]	Reddit	BERT	0.66	0.66	0.66	4400		✓
	Twitter	BERT	0.74	0.75	0.74	5000		✓
		Bi-LSTM	0.67	0.67	0.67			
		SVM	0.68	0.68	0.68			
[6]	Reddit	RoBERTa-large	0.69	0.70	0.69	4400		
		BERT-large	0.68	0.68	0.68			✓
	Twitter	RoBERTa-large	0.77	0.77	0.77			
		BERT-large	0.76	0.77	0.76	5000		✓
[21]	Reddit	BERT	-	-	0.62	4400		✓
		XLNet	-	-	0.54			
	Twitter	BERT	-	-	0.75	5000		✓
		LSTM	-	-	0.67			
		Bi-LSTM	-	-	0.66			
		XLNet	-	-	0.68			
[22]	Reddit	RoBERTa-large	0.72	0.72	0.72	4400		✓
	Twitter	RoBERTa-large	0.77	0.77	0.77	5000		✓
[23]	Twitter	Bi-LSTM + CNN	0.89	0.91	0.89	6000	✓	
		Bi-LSTM	0.70	0.69	0.69			
[24]	IAC	LSTM	0.67	0.82	0.73	4692		✓
		SVM	0.66	0.67	0.66			
	Twitter	LSTM	0.77	0.75	0.76	25991		✓
		SVM	0.66	0.66	0.66			

be preferred where applicable. However, it is not yet possible for a research focused on Russian language due to the lack of widely available text corpora. Second, the data quality contributes to the classification result just as much as its amount. This is especially worth noting in fields that lack widely available sets of data to conduct experiments on.

VI. COMPARISON WITH RESEARCH FOR ENGLISH LANGUAGE

A comparison of the achieved results can be drawn with similar works on automatic irony and sarcasm detection for English texts (see Table III). Research submitted as a part of the Second Workshop on Figurative Language Processing [5], [6], [21], [22] show similar or slightly higher results while operating on a bigger corpora and taking into consideration message context. The listed submissions were chosen for comparison because they, just like our work, utilize only baseline models.

A. Baruah et al. [5] achieved the best F1-score of 74.4% and 65.8% with BERT classifier for Twitter and Reddit datasets respectively. F1-score for Twitter was achieved by using the message itself plus its context—a message to which the classified message was in response to. For Reddit only the message itself was used.

A. Kumar and V. Anand [6] reported their best F1-score with RoBERTa-large as 77.4% for Twitter and 69.9% for Reddit while considering message context. However, when operating only on the message itself (i.e., without using context) they only managed to achive F1-score of 67.5% and 63.2% on Twitter and Reddit respectively.

A. Avvaru et al. [21] conducted experiments using BERT on a variable number of conversation sentences being taken into consideration. The best F1-score they achieved is 75.2% for Twitter and 62.1% for Reddit with 7 and 5 conversation sentences passed into the classifier respectively. The paper does not mention whether experiments have been done without considering context, however with minimal amount of context (3 conversation sentences) F1-score is 71.0% and 60.3% for Twitter and Reddit.

T. Dadu and K. Pant [22] showed the result similar to the two previous works. Considering context, they were able to achieve F1-score of 77.2% for Twitter and 71.6% for Reddit with RoBERTa-large classifier. Without context the results are 75.2% and 67.9% for Twitter and Reddit.

This comparison shows that generally context plays a significant role in the ability of a classifier to detect irony or sarcasm. In the papers that use classifiers very similar to ours, the results of irony and sarcasm detection with consideration of context clearly surpass our results. However, the results achieved without considering context are usually lower, thus falling behind the results shown in this research.

Another comparison can be made to research [23], [24]. Their approach is different to the ones submitted to the Second Workshop on Figurative Language Processing, but it is similar to the approach we chose for Bi-LSTM classifier.

While D. Jain et al. [23] proposed a hybrid model classifier consisting of Bi-LSTM and CNN, they also provided metrics for each part of the model independently. For English language Bi-LSTM layer uses GloVe word embeddings as an input. In conjunction those two layers are able to reach F1-score of 89.0%, at the same time Bi-LSTM layer with embeddings alone only shows F1-score of 69.4%, which is lower that we achieved using this approach.

D. Ghosh et al. [24] conducted experiments with feature-based SVM and embedding-based LSTM. When using message context, they achieved F1-score of 73.3% on IAC corpus with LSTM classifier; without context performance dropped by about 3.6%. SVM shown F1-score of 66.1% on the same corpus without context, while losing 3.5% of its performance when introducing context. On the Twitter corpus the results were 76.3% for LSTM and 65.7% for SVM with context. Removing context reduces performance by 1.8% and 1.2% respectively, which isn't a significant change. Overall, the results even considering context are comparable to the ones presented in this research.

VII. Conclusion

In this research an evaluation of baseline classifiers have been performed for the task of irony and sarcasm detection in Russian language. We examined the performance of five different classifiers (BERT, Bi-LSTM, SVM, Random Forest, Logistic Regression) on two manually created datasets of Russian texts. BERT classifier powered by RuBERT model showed the best result by achieving F1-score of 0.76 on the verified dataset, with Bi-LSTM taking the second place with F1-score of 0.73.

It was also shown that using pre-trained word embeddings leads to an increase in classifier performance: Bi-LSTM's F1-score improved from 0.64 to 0.73 when introducing embeddings. Since using pre-trained word embeddings lead to an increase in model metrics, a further potential improvement in applying transfer learning for irony detection would be to switch to transfering sentiment knowledge from specialized vocabularies [26].

Also our experiments showed that datasets annotated by volunteers may not be entirely accurate, therefore a disparity in results can often be seen when comparing the same classifier's performance in different research. However, it is worth mentioning that refinement of training data can improve the classifier's performance.

For our future research we are aiming to adopt a larger BERT or BERT-derived model such as RuBERT-Large (https://huggingface.co/ai-forever/ruBert-large) or RuROBERTA-Large (https://huggingface.co/ai-forever/ruRoberta-large). The plan is to build an ensemble of BERT and Bi-LSTM that will improve individual model's performance by pipelining their outputs.

Another direction for improvement we are looking into is introducing sentence context in the training step. As multiple works referenced in this research shown, providing context to the classifier leads to an increase in performance. By introducing leading and trailing context for each sentence we may be able to further improve our results.

Acknowledgment

The reported study was funded by the grant of Russian Science Foundation No. 23-21-00495 (https://rscf.ru/en/project/23-21-00495/).

References

[1] A.-C. Băroiu and Ș. Trăușan-Matu, "Automatic sarcasm detection: Systematic literature review," *Information*, vol. 13, no. 8, p. 399, 2022.

[2] S. Smetanin, "The applications of sentiment analysis for Russian language texts: Current challenges and future perspectives," *IEEE Access*, vol. 8, pp. 110 693–110 719, 2020.

[3] A. Agrawal, A. K. Jha, A. Jaiswal, and V. Kumar, "Irony detection using transformers," in *2020 International Conference on Computing and Data Science (CDS)*. IEEE, 2020, pp. 165–168.

[4] C. Turban and U. Kruschwitz, "Tackling irony detection using ensemble classifiers," in *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, 2022, pp. 6976–6984.

[5] A. Baruah, K. Das, F. Barbhuiya, and K. Dey, "Context-aware sarcasm detection using BERT," in *Proceedings of the Second Workshop on Figurative Language Processing*, 2020, pp. 83–87.

[6] A. Kumar and V. Anand, "Transformers on sarcasm detection with context," in *Proceedings of the second workshop on figurative language processing*, 2020, pp. 88–92.

[7] I. A. Farha, S. Wilson, S. Oprea, and W. Magdy, "Sarcasm detection is way too easy! an empirical comparison of human and machine sarcasm detection," in *Findings of the Association for Computational Linguistics: EMNLP 2022*, 2022, pp. 5284–5295.

[8] P. Golazizian, B. Sabeti, S. A. A. Asli, Z. Majdabadi, O. Momenzadeh, and R. Fahmi, "Irony detection in Persian language: A transfer learning approach using emoji prediction," in *Proceedings of The 12th Language Resources and Evaluation Conference*, 2020, pp. 2839–2845.

[9] T. Zefirova and N. Loukachevitch, "Irony and sarcasm expression in Twitter," *EPiC Series in Language and Linguistics*, vol. 4, pp. 45–49, 2019.

[10] V. Govindan and V. Balakrishnan, "A machine learning approach in analysing the effect of hyperboles using negative sentiment tweets for sarcasm detection," *Journal of King Saud University-Computer and Information Sciences*, vol. 34, no. 8, pp. 5110–5120, 2022.

[11] W. Chen, F. Lin, X. Zhang, G. Li, and B. Liu, "Jointly learning sentimental clues and context incongruity for sarcasm detection," *IEEE Access*, vol. 10, pp. 48 292–48 300, 2022.

[12] A. Wadhawan, "AraBERT and Farasa segmentation based approach for sarcasm and sentiment detection in Arabic tweets," in *Proceedings of the Sixth Arabic Natural Language Processing Workshop*, 2021, pp. 395–400.

[13] I. A. Farha, S. V. Oprea, S. Wilson, and W. Magdy, "SemEval-2022 task 6: iSarcasmEval, intended sarcasm detection in English and Arabic," in *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*, 2022, pp. 802–814.

[14] R. Ortega-Bueno, F. Rangel, D. Hernández Farıas, P. Rosso, M. Montes-y Gómez, and J. E. Medina Pagola, “Overview of the task on irony detection in Spanish variants,” in *Proceedings of the Iberian languages evaluation forum (IberLEF 2019), co-located with 34th conference of the Spanish Society for natural language processing (SEPLN 2019). CEUR-WS. org*, vol. 2421, 2019, pp. 229–256.

[15] Z. B. Nezhad and M. A. Deihimi, “Sarcasm detection in Persian,” *Journal of Information and Communication Technology*, vol. 20, no. 1, pp. 1–20, 2021.

[16] A. Ermilov, N. Murashkina, V. Goryacheva, and P. Braslavski, “Stierlitz meets SVM: humor detection in Russian,” in *Artificial Intelligence and Natural Language: 7th International Conference, AINL 2018, St. Petersburg, Russia, October 17–19, 2018, Proceedings 7*. Springer, 2018, pp. 178–184.

[17] Sarcasm / Merriam-Webster.com dictionary. [Online]. Available: https://www.merriam-webster.com/dictionary/sarcasm

[18] Y. Kuratov and M. Arkhipov, “Adaptation of deep bidirectional multilingual transformers for Russian language,” *arXiv preprint arXiv:1905.07213*, 2019.

[19] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of deep bidirectional transformers for language understanding,” in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 2019, pp. 4171–4186.

[20] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, “Scikit-learn: Machine learning in Python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[21] A. Avvaru, S. Vobilisetty, and R. Mamidi, “Detecting sarcasm in conversation context using transformer-based models,” in *Proceedings of the second workshop on figurative language processing*, 2020, pp. 98–103.

[22] T. Dadu and K. Pant, “Sarcasm detection using context separators in online discourse,” in *Proceedings of the Second Workshop on Figurative Language Processing*, 2020, pp. 51–55.

[23] D. Jain, A. Kumar, and G. Garg, “Sarcasm detection in mash-up language using soft-attention based bi-directional LSTM and feature-rich CNN,” *Applied Soft Computing*, vol. 91, p. 106198, 2020.

[24] D. Ghosh, A. R. Fabbri, and S. Muresan, “The role of conversation context for sarcasm detection in online interactions,” *arXiv preprint arXiv:1707.06226*, 2017.

[25] B. Moores and V. Mago, “A survey on automated sarcasm detection on Twitter,” *arXiv preprint arXiv:2202.02516*, 2022.

[26] S. Zhang, X. Zhang, J. Chan, and P. Rosso, “Irony detection via sentiment-based transfer learning,” *Information Processing & Management*, vol. 56, no. 5, pp. 1633–1644, 2019.