Baltic J. Modern Computing, Vol. 10 (2022), No. 3, pp. 283–294
https://doi.org/10.22364/bjmc.2022.10.3.02

Comparison of Deep Learning Approaches for Lithuanian Sentiment Analysis

Jurgita KAPOČIŪTĖ-DZIKIENĖ[1,2], Askars SALIMBAJEVS[3,4],

[1] Tilde IT, Naugarduko str. 100, 03160 Vilnius, Lithuania,
[2] Vytautas Magnus University, Faculty of Informatics, Vileikos str. 8, 44404 Kaunas, Lithuania
[3] Tilde SIA, Vienibas str. 75a, 1004 Riga, Latvia,
[4] University of Latvia, Raina blvd. 19, 1050 Riga, Latvia

jurgita.dzikiene@tilde.lt, askars.salimbajevs@tilde.lv

Abstract. Sentiment analysis is one of the oldest Natural Language Processing problems, still relevant and challenging today. It is usually formulated and solved as a supervised machine learning problem. In this research, we are solving the three-class sentiment analysis problem for the non-normative Lithuanian language. The contribution of our research is related to applying the innovative BERT-based multilingual sentence transformer models to the Lithuanian sentiment analysis problem. For comparison purposes, we have also investigated traditional Deep Learning approaches, such as fastText or BERT word embeddings with the Convolutional Neural Network as the classifier. The best accuracy $\sim$0.788 was achieved with the purely monolingual model, i.e., fastText (trained on the very large and diverse Lithuanian corpus) and the Convolutional Neural Network (refined in various text classification tasks). The backbone of the second-best approach (reaching $\sim$0.762) is the multilingual sentence-transformer-based model, which is the trend in text classification tasks, especially for the English language.

Keywords: Sentiment analysis, monolingual vs. multilingual models, word vs. sentence embeddings, transformer models, the Lithuanian language

1 Introduction

Social media (Social networks, internet forums, internet comments, etc.) has become commonplace for spreading, sharing, and even forming opinions. Freely expressed thoughts play an important role in marketing (it helps managers to analyze customers' feedback or understand their further needs) and even politics (it can strongly impact political decisions) (Guille et al., 2013). Social listening, social monitoring, customer experience analytics, and other similar processes strongly rely on sentiment analysis as one of their analysis layers. Sentiment analysis is the NLP task that categorizes emotions of a given text by their type (positive, negative, neutral) and sometimes even intensity (very positive / negative, somewhat positive / negative, etc.).

Sentiment analysis is one of the oldest NLP tasks dating back to the 1990s. A comprehensive review of papers (covering different research topics and evolution in sentiment analysis) till 2016 is presented in Mäntylä et al. (2018). Recent research till 2022 and recent advances (covering more than 900 papers for 36 benchmarks and 73 datasets[5]) prove this problem is still very relevant and challenging today. The spectrum of applied methods covers rule-based (relying on the hand-crafted rules), dictionary-based (measuring a text sentiment based on the polarity of words found in sentiment dictionaries), traditional Machine Learning (ML) classifiers (e.g., Support Vector Machines – SVM, Naïve Bayes – NB, etc.), traditional (e.g., Convolutional Neural Networks – CNN, Long Short-Term Memory – LSTM method applied on a top of word2vec or GloVe) and innovative Deep Learning (DL) (e.g., BERT fine-tuning, BERT-based sentence transformers, etc.) approaches. The applied methods evolve over the years, but the pitfalls faced by researchers remain the same: sarcasm / irony or multipolarity due to users' subjectivity and therefore different understanding of the same context. However, as methods progress, the accuracy increases, which in turn, encourages further research in this direction.

Consequently, in this paper, we are solving a three-class (positive / negative / neutral) sentiment analysis problem for the Lithuanian language with various DL approaches. Our research is important as it covers a wide range of different traditional and innovative DL approaches, applied to the large, clean dataset.

2 Related work

In this overview, we skip all rule- and dictionary-based approaches, focusing only on the innovative and accurate techniques solving the sentiment analysis as the supervised text classification problem. There are many research papers whose results are contradictory, therefore it is difficult to conclude which method (or model) is the best one. Due to it, we are especially interested in comparative research as it is performed under the same experimental conditions: i.e., by using the same benchmark datasets, training / testing splits, evaluation metrics, etc. For this reason, it is necessary to mention the SemEval sentiment analysis competition that attracts many research teams from all over the world trying to offer the most accurate solution for the given dataset(s).

In SemEval-2019, 311 teams were trying to detect emotion classes (*happy*, *sad*, *angry*, and *others*) from the context of user&conversational agent interaction dialogs (Chatterjee et al., 2019). *Sad* and *happy* appeared to be the best and the worst recognized classes, respectively. The most common choice among the participated teams was the Bidirectional LSTM (BiLSTM) method which nowadays is considered the traditional DL approach. The winning team (Agrawal and Suri, 2019) combined lexical features (word, character n-grams) along with scores from the pre-trained DL-based models. These features were later combined and used to train the Light-GBM tree method.

[5] https://paperswithcode.com/task/sentiment-analysis

Moreover, this approach outperformed their other tested DL method (two-layered, recurrent, with skip connections and bidirectional cell and attention). The second-best team (Basile et al., 2019), applied an ensemble of four neural models (two types of BiLSTM networks, sentence encoder model, and BERT fine-tuning) by using the softmax output probabilities (16 features) from each of the four classes and all four models. These 16 features were later used to train several traditional ML classifiers: Naïve Bayes, Logistic Regression, SVM with normalization (SVM-n) and standardization (SVM-s), JRip, J48, Random Forest, and various meta-learners. The offered DL- and assembly-based approach with SVM-n achieved the best performance. The ensemble of the BERT fine-tuning and hierarchical LSTM with GloVe and ELMo embeddings (used to encode semantical and emotional context) remained in the third place (Huang et al., 2019). Unlike other teams, the authors noticed that most errors appear not by classifying the *happy* category, but *others*; however, they believe this problem can be solved with the two-staged classifier, i.e., by classifying the *others* category versus *non-others* and only then classifying *non-others*.

The SemEval-2020 sentiment analysis competition (Patwa et al., 2020) addressed the three-class (*positive*, *negative*, and *neutral*) problem with the code-switching. 61 and 28 teams solved the sentiment analysis tasks for tweets in Hinglish (Hindi-English) and Spanglish (Spanish-English) languages, respectively. BERT- or ensemble-based approaches were the most popular and successful among all participating teams. The best system for Hinglish (Liu et al., 2020) used the multilingual XLM-R transformer model (supporting 100 languages, including English and Hindi), which was further trained with the adversarial examples (acting as a regularizer and helping the trained network to generalize better). These adversarial examples were created using the gradient of the loss function. The second-best result on the Hinglish texts was also achieved with the XLM-R transformer, except that researchers haven't performed adversarial training (Srinivasan, 2020). The number one for Spanglish is the XLMs sentence embedding layer used with the CNN classifier (Ma et al., 2020). The ensemble-based approach (combining CNN, self-attention, and LSTM models) applied to the pre-processed text (stemmed, removal of stop words and character repetition in elongated words) achieved the second-best result on Spanglish tweets (Singh and Parmar, 2020).

With one year break, the sentiment analysis problem-solving returns to the SemEval-2022 competition with the 32 participating teams (Barnes et al., 2022). However, this time the task is a bit more complicated as it is a structural sentiment analysis problem that requires the detection of sentiment graphs (composed of the sentiment holder, target, and polarity of *positive* / *negative* / *neutral*). This task included five languages, i.e., Norwegian, Catalan, Basque, Spanish, and English. Despite the solving task being compound (with sentiment analysis / polarity detection as one of the components), the applied methods are similar. As presented in Barnes et al. (2022), the winner of the monolingual sub-track formulated the task as the dependency parsing approach and fine-tuned RoBERTa-Large (for English) and XLM-RoBERTA-Large (for non-English) for structural sentiment analysis. Despite the competition is still ongoing[6], it can be already seen that the vast majority of teams rely on the BERT-based sentence transformer models.

[6] https://competitions.codalab.org/competitions/33556#results

The same trends can be seen in the “Papers with code” leaderboard[7], tracking the progress of sentiment analysis research with various benchmark datasets. The best binary (*positive / negative*) sentiment analysis results on the Stanford Sentiment Treebank benchmark dataset are achieved with the offered Smart fine-tuned transformer model having the backbone of RoBERTa-Large (Jiang et al., 2020). For controlling high complexity and preventing the aggressive update, the authors use offered the Smoothness-inducing Adversarial Regularization technique and Bregman Proximal Point Optimization, respectively. Their method was able to outperform other BERT and RoBERTa models. The RoBERTa-Large model is also effective on the 5-class (*positive*, *somewhat positive*, *neutral*, *somewhat negative*, *negative*) sentiment analysis problem (Sun et al., 2020). The used transformer model is complemented with the additional layer (so-called interpretation layer) aggregating information for each text span (assigned with a specific weight and representing its contribution) and then their weighted combination is fed to the softmax function for the final prediction.

XLNet, i.e., another transformer model is effective on another benchmark dataset Yelp for the binary and 5-class sentiment classification problem (Yang et al., 2019). Same XLNet is also the second-best approach on the popular IMDb dataset; which is only slightly outperformed by the document embeddings with the cosine similarity approach (Thongtan and Phienthrakul, 2019).

Hence, transformer models are the most popular and the most effective for nowadays sentiment analysis problems. Moreover, despite the majority of research is done for the English language, many available transformer models are multilingual and therefore can be applied to other target languages (not only English). The research objective of our paper is the Lithuanian language, which is less resourced (because benchmark datasets are not available, the existing proprietary datasets contain noise, and are not annotated by multiple annotators); existing multilingual transformer models are not refined for the Lithuanian language; besides the Lithuanian language is more complicated compared to, e.g., English (based on the vocabulary size of the headwords, derivation system, morphology, and free word order in a sentence). All these factors make the sentiment analysis task more complicated; moreover, the achieved accuracy will likely be lower compared to similar experiments for English.

The sentiment analysis research for the Lithuanian language dates back to 2013 and covers traditional ML approaches (SVM, NB) (Kapočiūtė-Dzikienė et al., 2013) and later traditional DL (CNN, LSTM) approaches (Kapočiūtė-Dzikienė et al., 2019). Previous research lacks experiments with the most recent techniques, i.e, transformer models (and most importantly – sentence transformers). Consequently, this research is dedicated to overcoming this drawback. Besides, this research is performed with the new larger and cleaner sentiment analysis dataset, therefore enabling a more accurate comparative analysis of both traditional and innovative techniques.

[7] https://paperswithcode.com/task/sentiment-analysis

3 Methodology

3.1 Formal definition of the task

The sentiment analysis can be formulated as a supervised text classification problem and formally determined as follows.

Let $D = \{d_1, d_2, \ldots, d_n\}$ be a set of texts and $C = \{c_1, c_2, \ldots, c_m\}$ be a set of sentiments (classes). We have a closed-set classification problem with *m* limited to 3 (*positive*, *negative*, and *netural*). Besides, we are solving a single-label classification problem, where each $d_i \in D$ can be attached to only one $c_j \in C$ (we do not allow multipolarity of the text).

Let function η be a classification function that maps each text to its correct sentiment: $D \to C$. Let $D^L \subset D$ be a gold-standard training dataset containing instances, i.e., paired texts with their sentiments $< d_i, c_j >$.

Let Γ be a classification method that could learn an approximation of η: how to predict a sentiment label for each incoming text. This research aims to offer an effective method Γ that could achieve high sentiment analysis accuracy on unseen instances (for this reason we will test our method on the dataset D^T, where $D^L \cap D^T = \emptyset$).

3.2 The dataset

The sentiment analysis as described in Section 3.1 is a supervised text classification task, that requires a labeled dataset. In our experiments, we have used a dataset containing internet comments collected from various Lithuanian internet portals from 2012 to 2021. These internet comments are rather short ($\sim$21.6 tokens per comment) and represent the spoken non-normative Lithuanian language. They also cover different domains about business, technologies, culture, science, sports, health, etc.

In the beginning, all collected texts were manually annotated by two annotators. Ambiguous instances (i.e., texts on which polarity annotators could not agree) were filtered out from the dataset. The prepared dataset was shuffled and split into training and testing. Moreover, part of the dataset ($\sim$40% of texts, all from the training split) that contained the most complicated / ambiguous instances (as denoted by the first two annotators) was re-examined by the third annotator, who could see the previously attached sentiment labels and had to remove instances to which labels he disagreed. The client requested to create as accurate a sentiment analysis tool as possible and it was possible only by minimizing the subjectivity impact on the training dataset. No pre-processing was performed, except lowercasing if the uncased vectorization models were applied. The final size of the dataset can be seen in Table 1.

3.3 Applied approaches

The goal of our experimental investigation is to offer the most accurate supervised ML Γ method (defined in Section 3.1) for our sentiment analysis task. During the experimental investigation we have explored the following approaches described in the subsections below.

Table 1. Statistics about the sentiment analysis dataset used in our experiments

	All dataset	Training split	Testing split
positive	4,748	3,943	805
negative	9,627	7,755	1,872
neutral	4,606	3,807	799
In total	18,981	15,505	3,476

3.3.1 FastText + CNN We have used the Convolutional Neural Network (CNN) method as the text classifier which architecture and hyper-parameter value set (as presented in Figure 3 in Kapočiūtė-Dzikienė et al. (2020)) was already optimized before in various text classification tasks for the Lithuanian language. The CNN method is suitable for our solving task because it can search for patterns (i.e., word n-grams) in texts.

However, the CNN method cannot be applied to the text directly, therefore for the text vectorization, we have used the fastText embeddings. The fastText embeddings are the word-level vectorization technique that leans on how to vectorize character n-grams and later uses these n-gram vectors to represent words (by sliding through the word and summing its n-gram vectors: e.g., having 5-gram embeddings <senti>, <entim>, <ntime>, <timen>, <iment> the word <sentiment> would be composed). The advantage of fastText embeddings is that they can vectorize unseen or misspelled words (at least some of incoming character n-grams can be found); moreover, their vectors are rather close to their correct equivalents. It is an advantage when vectorizing the Lithuanian non-normative texts.

In our research, we have used two different fastText embedding models, i.e., free pre-trained *Facebook* model (*cc.lt.300.vec* [8]) and *Tilde*. Both models are monolingual (trained only on the normative Lithuanian texts), follow the same approach, and produce the same length vectors equal to 300. The only difference is in the amount of text that was used for fastText model training: *Facebook* uses Wikipedia only, but *Tilde* had far more texts crawled from various Lithuanian portals.

3.3.2 BERT-w + CNN In this approach, the CNN method (as the classifier) was applied on top of the BERT (Bidirectional Encoder Representations from Transformers) Devlin et al. (2019) output (which acted as the word-level text vectorizer). The BERT model has inner mechanisms to disambiguate words-homographs because 1) it learned how to predict masked words from their broader context (feature acquired during the masked language modeling phase); 2) it can determine the word-related context/sentences (feature acquired during the next sentence prediction phase). This BERT ability is especially important for the ambiguous Lithuanian language, which has even more ambiguity problems in the non-normative texts (especially due to missing diacritics).

In this approach the BERT vectorizes an input text (maintaining the word order), then word vectors are concatenated into a single longer vector (*text vector length* =

[8] https://fasttext.cc/docs/en/crawl-vectors.html

word vector length * *number of words per text*) which is later fed into the CNN model. In our experiments, we set a *number of words per text* parameter to 30 (as the average length of the text in our dataset is $\sim$21.6).

In our experiments, we have used 2 pre-trained multilingual models, i.e., *bert-base-multilingual-cased* and *bert-base-multilingual-uncased* sensitive and insensitive to letter casing, respectively. The texts in the dataset were lowercased before applying the *uncased* model. Both BERT models support the Lithuanian language; and their architecture has 12 stacked encoder layers, 12 attention heads, and 768 hidden layers (more details on the model page[9]). As the classifier, we have used the CNN method with the same architecture and hyper-parameter value set as in the *fastText + CNN* approach.

3.3.3 BERT-s + FFNN In this approach, the text vectorization is performed with the BERT-based sentence-level (instead of the word-level) transformer models. The output of such vectorization is the fixed-length vector representing an input text as a whole (thus, boundaries between words are lost). These methods are more adjusted to focus on the semantics of texts instead of the semantics of words and their order. Due to this reason, they seem more suitable for the Lithuanian language, having relatively free word order in a sentence, which becomes especially challenging when dealing with non-normative texts which also have other issues (typos, abbreviations, missing diacritics, etc.). In our experiments, we have tested DistilUSE, XLM (Conneau and Lample, 2019), RoBERTa (Liu et al., 2019), DistilBERT (Sanh et al., 2019), and LaBSE (Feng et al., 2020) approaches and their modifications, specifically, 6 pre-trained multilingual transformer models: *distiluse-base-multilingual-cased-v2*, *xlm-r-distilroberta-base-paraphrase-v1*, *xlm-r-bert-base-nli-stsb-mean-tokens*, *LaBSE*, *distilbert-multilingual-nli-stsb-quora-ranking*, *paraphrase-xlm-multilingual-v1*. More details about these models can be found on the sentence transformers models' page[10].

The Feed Forward Neural Network (FFNN) (dense, fully-connected layer) connects the BERT sentence transformer output with classes: weights of this layer are adjusted during the sentiment analysis model's training phase.

3.3.4 Summary It is important to notice that all applied approaches represent very different research directions and therefore are interesting from the research perspective. *FastText + CNN* is a completely monolingual approach, whereas the other two approaches use multilingual models for text vectorization. *FastText + CNN* and *BERT-w + CNN* both perform word-level vectorization, whereas *BERT-s + FFNN* – the sentence level vectorization. *FastText + CNN* differs from *BERT-w + CNN* and *BERT-s + FFNN* as it does not use transformer models. *FastText + CNN* and *BERT-w + CNN* approach both use CNN as the classifier and therefore are representatives of the traditional DL; whereas *BERT-s + FFNN* is the representative of the innovative purely transformer-based DL technique.

[9] https://huggingface.co/models
[10] https://huggingface.co/sentence-transformers

4 Experiments and results

During the experimental investigation, we have tested all approaches (presented in Section 3.3) on the dataset (Section 3.2). The performance of each method was evaluated using the *accuracy* and *macro F1-score* metrics using the *sklearn.metrics* python library. The obtained values were averaged in 3 runs, the confidence intervals were calculated using the *scipy.stats* with 95% of the confidence interval.

A method is considered reasonable and suitable for the solving task if it's accuracy exceeds both *random* (eq. 1) and *majority baselines* (eq. 2) (where $P(c_j)$ is a probability of class c_j), calculated from the training split and equal to ~0.38 and ~0.5, respectively. When comparing different approaches, it is important to determine if the differences between results are statistically significant. For this reason, we have applied the McNemar test with 95% confidence using *statsmodels.stats.contingency_tables* library. This test was performed for all pairs of methods, considering all runs within the same experiment.

$$random_{baseline} = \sum P^2(c_j) \tag{1}$$

$$majority_{baseline} = max(P^2(c_j)) \tag{2}$$

The calculated averaged *accuracy* values with the confidence intervals are presented in Fig. 1. We do not present *F1-score* values as they demonstrate the same trend as *accuracy*. The figure presents results above $majority_{baseline}$=0.5 (which is larger of the baselines). Table 2 summarizes between which methods differences in results are statistically significant / insignificant.

Table 2. This table presents tested methods (No.1 – No.10) and determines between which differences in results are statistically significant / insignificant. Notations *y* and *n* stand for yes / significant and no / insignificant, respectively

No.		1	2	3	4	5	6	7	8	9
1	FastText + CNN: Facebook	-	-	-	-	-	-	-	-	-
2	FastText + CNN: Tilde	y	-	-	-	-	-	-	-	-
3	BERT-w + CNN: bert-base-multilingual-cased	y	y	-	-	-	-	-	-	-
4	BERT-w + CNN: bert-base-multilingual-uncased	n	y	y	-	-	-	-	-	-
5	BERT-s + FFNN: distiluse-base-multilingual-cased-v2	n	y	y	n	-	-	-	-	-
6	BERT-s + FFNN: xlm-r-distilroberta-base-paraphrase-v1	n	y	y	n	n	-	-	-	-
7	BERT-s + FFNN: xlm-r-bert-base-nli-stsb-mean-tokens	n	y	y	n	n	n	-	-	-
8	BERT-s + FFNN: distilbert-multilingual-nli-stsb-quora-ranking	n	y	y	n	n	n	y	-	-
9	BERT-s + FFNN: paraphrase-xlm-r-multilingual-v1	n	y	y	n	n	n	n	n	-
10	BERT-s + FFNN: LaBSE	n	y	y	y	y	n	n	y	n

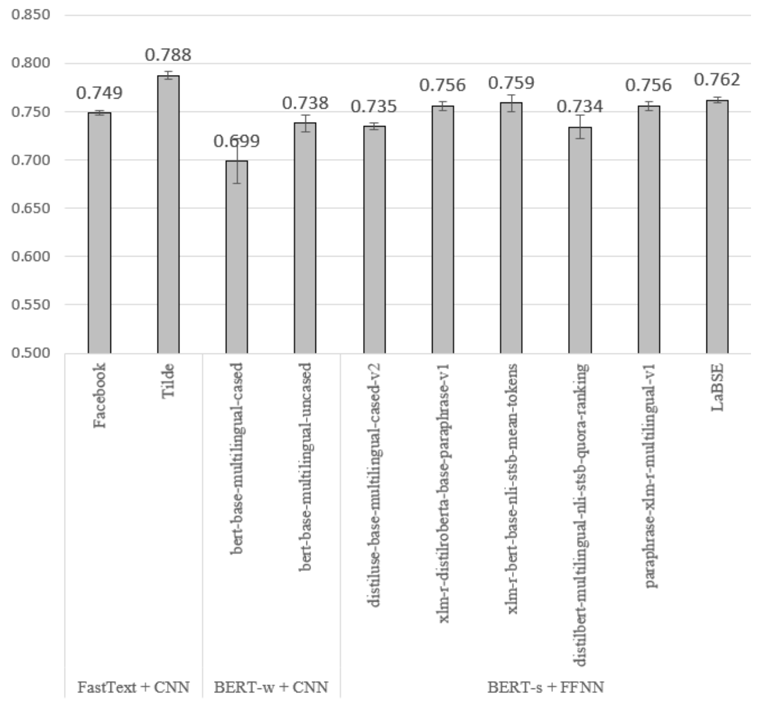

Fig. 1. Averaged accuracy values + confidence intervals obtained with different approaches on the data testing split

5 Discussion

Zooming into the results (Fig. 1 and Table 2) allows us to conclude that all results are reasonable because exceed random and majority baselines.

The best results (i.e., ∼0.788) are achieved with the *FastText + CNN* approach and *Tilde* fastText word embedding model; moreover, differences from all other results are statistically significant. The deeper analysis revealed that the *negative*, *positive*, and *neutral* classes were predicted with ∼0.90, ∼0.84, ∼0.47 of the accuracy, respectively. The *FastText + CNN* approach with the *Facebook* embeddings model demonstrates slightly lower performance compared to 4 of 6 *BERT-s + FFNN* models. This allows us to conclude that in general any *FastText + CNN* approach cannot be immediately considered as the best choice for the sentiment analysis tasks unless it uses the comprehensive and refined embedding model. It is important to notice that the *Tilde* model was trained on >10 times larger and more diverse Lithuanian corpus compared to *Facebook*.

The results of the *BERT-w + CNN* approach strongly depend on the embedding model: the difference between the *cased* and *uncased* model is not only statistically significant but also rather big, i.e., ∼0.4. However, the poor performance of the cased-sensitive model is explainable. Under case-sensitive settings, there are more words; more words mean more sparseness in models. Whereas in our sentiment analysis task case sensitivity does not play an important role: the meaning of the context/words is much more important.

If we compare *FastText + CNN* with *Facebook* word embedding model and *BERT-w + CNN* with the *uncased* model, the monolingual *Facebook* embedding model seem to slightly outperform *multilingual BERT*; however, these differences are not statistically significant.

The results of *BERT-s + FFNN* approach are very controversial and cover the range from ∼0.734 to ∼0.762, therefore if choosing this approach, it is very important to correctly select the pre-trained sentence transformer model: for our solving task *LaBSE* is the most accurate solution, therefore it is recommended. However, its performance does not differ significantly from the *FastText + CNN* with *Facebook* word embeddings approach. The *fastText* embeddings have mechanisms to cope with the non-normative texts; besides, they are monolingual models specifically adjusted to the Lithuanian language, therefore such a result is logically explainable.

In our experiments, we have also tested multilingual word and sentence transformer model versions that are not specifically refined for the Lithuanian language. Despite it, new improved versions constantly appear on the huggingface.co site and have a big potential to surpass simpler approaches. One of the promising research directions could be training such a monolingual transformer model on the huge Lithuanian corpora and then adjusting such model for the sentiment analysis problems.

The practical value of this research is also important: the created sentiment analysis model was integrated into the system available online [11]. Despite the achieved acceptable accuracy (i.e., ∼0.788 which is rather high for the three-class sentiment analysis problem), in the future, we are planning to seek solutions on how to increase it even further. Research should cover two important directions: the search for even more advanced methods and more annotated data.

6 Conclusions

In this research, we were solving the three-class (*positive*, *negative*, and *neutral*) sentiment analysis problem for the non-normative Lithuanian language. For this, we were using the manually annotated dataset of internet comments (∼19 thousand texts in total). With this dataset, different traditional and innovative DL classification techniques were investigated. The CNN classifier was applied on top of fastText (2 tested monolingual models) or BERT (2 multilingual models) word embeddings. The dense layer classifier processed the output of the BERT-based sentence transformer models (6 multilingual models).

The best results (reaching ∼0.788 of the accuracy and significantly better compared to all other tested methods) were obtained with a purely monolingual approach, i.e., fastText and CNN, where the fastText word embeddings were trained on the very large and exhaustive Lithuanian corpus and the CNN classifier had the architecture tuned and refined in various text classification tasks for the Lithuanian language. The second-best

[11] https://ekalba.lt/nuomoniu-analize/

result was achieved with the innovative DL approach which used the existing multilingual *LaBSE* sentence transformer model as the vectorization and the dense layer on top of it as the classifier, of which weights were adjusted during the training stage.

In the future, we are planning further improve the sentiment analysis accuracy. It could cover the exhaustive error analysis; further data collection and annotation; aspect-based sentiment analysis that could help us to cope with the multipolarity in longer texts by identifying several sentiments related to different aspects.

Acknowledgements

The sentiment analysis tool was developed within the framework of the project “Development of Lithuanian language resources information system (E. kalba)” (No. 02.3.1-CVPA-V-527-01-0006) and co-financed by the European Regional Development Fund.

The research has also been supported by the European Regional Development Fund within the research project “AI Assistant for Multilingual Meeting Management” No. 1.1.1.1/19/A/082.

References

Agrawal, P., Suri, A. (2019). Nelec at semeval-2019 task 3: Think twice before going deep, *Proceedings of the 13th International Workshop on Semantic Evaluation*, pp. 266–271.

Barnes, J., Oberländer, L. A. M., Troiano, E., Kutuzov, A., Buchmann, J., Agerri, R., Øvrelid, L., Velldal, E. (2022). Semeval-2022 task 10: Structured sentiment analysis, *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022), Seattle. Association for Computational Linguistics*.

Basile, A., Franco-Salvador, M., Pawar, N., Štajner, S., Chinea-Ríos, M., Benajiba, Y. (2019). Symantoresearch at Semeval-2019 Task 3: Combined Neural Models for Emotion Classification in Human-Chatbot Conversations, *Proceedings of the 13th International Workshop on Semantic Evaluation*, pp. 330–334.

Chatterjee, A., Narahari, K. N., Joshi, M., Agrawal, P. (2019). Semeval-2019 Task 3: Emocontext Contextual Emotion Detection in Text, *Proceedings of the 13th international workshop on semantic evaluation*, pp. 39–48.

Conneau, A., Lample, G. (2019). Cross-Lingual Language Model Pretraining, *in* Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., Garnett, R. (eds), *Advances in Neural Information Processing Systems*, Vol. 32, Curran Associates, Inc.

Devlin, J., Chang, M.-W., Lee, K., Toutanova, K. (2019). ”BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”, *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)*, Association for Computational Linguistics, Minneapolis, Minnesota, pp. 4171–4186.

Feng, F., Yang, Y., Cer, D., Arivazhagan, N., Wang, W. (2020). Language-agnostic BERT Sentence Embedding, *CoRR* **abs/2007.01852**.

Guille, A., Hacid, H., Favre, C., Zighed, D. A. (2013). Information Diffusion in Online Social Networks: A Survey, *SIGMOD record* **42**(2), 17–28.
https://hal.archives-ouvertes.fr/hal-00848050

Huang, C., Trabelsi, A., Zaiane, O. R. (2019). ANA at SemEval-2019 Task 3: Contextual Emotion Detection in Conversations through Hierarchical LSTMs and BERT, *Proceedings of the 13th International Workshop on Semantic Evaluation*, pp. 49–53.

Jiang, H., He, P., Chen, W., Liu, X., Gao, J., Zhao, T. (2020). SMART: Robust and Efficient Fine-Tuning for Pre-trained Natural Language Models through Principled Regularized Optimization, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 2177–2190.

Kapočiūtė-Dzikienė, J., Damaševičius, R., Woźniak, M. (2019). Sentiment Analysis of Lithuanian Texts Using Traditional and Deep Learning Approaches, *Computers* **8**(1), 4.

Kapočiūtė-Dzikienė, J., Krupavičius, A., Krilavičius, T. (2013). A Comparison of Approaches for Sentiment Classification on Lithuanian Internet Comments, *Proceedings of the 4th Biennial International Workshop on Balto-Slavic Natural Language Processing*, pp. 2–11.

Kapočiūtė-Dzikienė, J., Balodis, K., Skadiņš, R. (2020). Intent Detection Problem Solving via Automatic DNN Hyperparameter Optimization, *Applied Sciences* **10**(21).

Liu, J., Chen, X., Feng, S., Wang, S., Ouyang, X., Sun, Y., Huang, Z., Su, W. (2020). Kk2018 at semeval-2020 task 9: Adversarial training for code-mixing sentiment classification, *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pp. 817–823.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, V. (2019). RoBERTa: A Robustly Optimized BERT Pretraining Approach. cite arxiv:1907.11692.
http://arxiv.org/abs/1907.11692

Ma, Y., Zhao, L., Hao, J. (2020). XLP at SemEval-2020 Task 9: Cross-Lingual Models with Focal Loss for Sentiment Analysis of Code-Mixing Language, *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pp. 975–980.

Mäntylä, M. V., Graziotin, D., Kuutila, M. (2018). The Evolution of Sentiment Analysis—a Review of Research Topics, Venues, and Top Cited Papers, *Computer Science Review* **27**, 16–32.

Patwa, P., Aguilar, G., Kar, S., Pandey, S., Pykl, S., Gambäck, B., Chakraborty, T., Solorio, T., Das, A. (2020). Semeval-2020 Task 9: Overview of Sentiment Analysis of Code-Mixed Tweets, *Proceedings of the fourteenth workshop on semantic evaluation*, pp. 774–790.

Sanh, V., Debut, L., Chaumond, J., Wolf, T. (2019). DistilBERT, a Distilled Version of BERT: Smaller, Faster, Cheaper and Lighter, *CoRR* **abs/1910.01108**.
http://arxiv.org/abs/1910.01108

Singh, A., Parmar, S. P. S. (2020). Voice@ SRIB at SemEval-2020 Tasks 9 and 12: Stacked Ensembling Method for Sentiment and Offensiveness Detection in Social Media, *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pp. 1331–1341.

Srinivasan, A. (2020). MSR India at SemEval-2020 Task 9: Multilingual Models Can Do Code-Mixing Too, *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pp. 951–956.

Sun, Z., Fan, C., Han, Q., Sun, X., Meng, Y., Wu, F., Li, J. (2020). Self-Explaining Structures Improve NLP Nodels, *arXiv preprint arXiv:2012.01786* .

Thongtan, T., Phienthrakul, T. (2019). Sentiment Classification Using Document Embeddings Trained with Cosine Similarity, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pp. 407–414.

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. R., Le, Q. V. (2019). XLNet: Generalized Autoregressive Pretraining for Language Understanding, *Advances in neural information processing systems* **32**.

Received July 25, 2022 , accepted July 28, 2022