DOI: 10.20998/2079-0023.2023.01.14
UDC 004.8

K. ONYSHCHENKO, Senior Lecturer at the Department of Software engineering, Kharkiv National university of Radio electronics, Kharkiv, Ukraine, e-mail: kostiantyn.onyshchenko@nure.ua, ORCID: https://orcid.org/0000-0002-7746-4570
Y. DANIIEL, Assistant at the Department of Software engineering, Kharkiv National university of Radio electronics, Kharkiv, Ukraine, e-mail: yana.daniiel@nure.ua, ORCID: https://orcid.org/0000-0002-3895-0744

USING LONG SHORT-TERM MEMORY NETWORKS FOR NATURAL LANGUAGE PROCESSING

The problem of emotion classification is a complex and non-trivial task of language interpretation due to the natural language structure and its dynamic nature. The significance of the study is in covering the important issue of automatic processing of client feedbacks, collecting opinions and trend-catching. In this work, a number of existing solutions for emotion classification problem were considered, having their shortcomings and advantages illustrated. The evaluation of performance of the considered models was conducted on emotion classification on four emotion classes, namely Happy, Sad, Angry and Others. The model for emotion classification in three-sentence conversations was proposed in this work. The model is based on smileys and word embeddings with domain specificity in state of art conversations on the Internet. The importance of taking into account the information extracted from smileys as an additional data source of emotional coloring is investigated. The model performance is evaluated and compared with language processing model BERT (Bidirectional Encoder Representations from Transformers). The proposed model achieved better performance at classifying emotions comparing to BERT (having F_1 score as 78 versus 75). It should be noted, that further study should be performed to enhance the processing by the model of mixed reviews represented by emotion class Others. However, modern performance of models for language representation and understanding did not achieve the human performance. There is a variety of factors to consider when choosing the word embeddings and training methods to design the model architecture.

Keywords: natural language processing, neural network, natural language, long short-term memory networks, text classification, emotional text analysis.

Introduction. Over the past few decades, the amount of text data produced by humanity has grown exceedingly. One of the reasons behind this fact is that we actively exchange information and publish our thoughts on websites and social media.

Such unstructured data is represented in arbitrary form and is often complemented by emojis, which makes it difficult for a computer to categorize and derive meaning from the source. On this basis, the challenge to teach computers to properly process the information arose.

Natural language processing (NLP) is widely used for text data analysis and classification [1].

The core aspects of language understanding include three parameters, which are morphology, semantics, and syntax. Morphology is the study of word or statement structure; semantics is the study of meaning or reference; syntax is the study of how words and morphemes combine to form larger units such as phrases and sentences [2].

Modern models consider all three parameters. The models are using data-driven approach through machine learning and deep learning.

The goal of this work is to consider existing solutions for text data processing in terms of emotional classification and propose the model that can solve such task.

Problem statement. The semantic evaluation problem of emotion classification will be considered in this work. The deep learning approach will be based on Keras and TensorFlow – the Python frameworks. These frameworks have ready to use functions for rapid optimization, model prediction, model tuning and many more. This will allow to achieve approximate state-of-art results with an original model architecture. BERT will also be implemented to obtain a state-of-art model in natural language understanding. The received results will be evaluated and compared [3].

The theoretical fundamentals of emotional classification models will be discussed. The practical details of the proposed model's training and BERT on unseen test data will be presented. The obtained results will be evaluated and compared. The differences between the models will be illustrated.

It should be considered that the proposed model will be implemented using Keras and TensorFlow. The trained models will be applied to a specific pre-defined dataset to solve the semantic evaluation problem of emotion classification. The models will be trained on four CPU cores. These conditions apply specific limitations on the scope of the project.

Analysis of existing solutions. Machine learning is a modelling approach focused on finding underlying patterns in a dataset. An algorithm with learning function is applied to rich dataset. The parameters of model are modified to enhance the predictive performance of a model. The unseen data is used to evaluate the trained model.

Recurrent Neural Network (RNN) is a class of artificial neural networks, where connections between nodes form a directed sequential graph. By this, the sequential nature of input can be considered when making output predictions [1].

A set of feedback weights contained in a hidden state vector is computed at every step in the sequence that pass information from earlier states. The ability to predict to which class the sequence belongs to is provided by the model reformulation. This will allow to incorporate new time dependency by using the final recurrent hidden state vector to make softmax probability predictions:

$$\hat{y} = \varphi(Vh_p), \quad (1)$$

where

$$h_p = \delta(Ux_p + Wh_{(p\text{-}1)}). \quad (2)$$

The parametrization W of the model is comprised of the weight matrices U, V and W. The hidden layer is dependent on earlier states.

The fig. 1 provides more information about the recurrence mechanism applied.

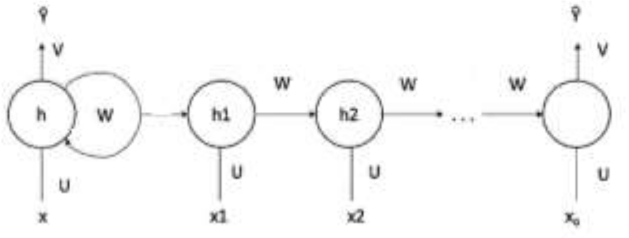

Fig. 1. The unfolding concept

New hidden state vector is computed at each step. The vector is trained to pass forward the most significant information for solving the problem through gradient descent with an appropriate loss function [3].

Long short-term memory (LSTM) networks have two enhancements comparing to previously considered RNNs [4]. The first enhancement is that each time step passes a hidden state vector and a local context vector to the next recurrent node. The second enhancement is that long short-term memory network contains a set of gating mechanisms (fig. 2). These mechanisms provide the ability to the model to decide which data to pass forward in recurrence. These enhancements allow the LSTM model to learn long-term dependencies in the sequence in a more stable way [5].

The gating mechanisms of LSTM contain an input gate, a context gate, a forgetting gate and an output gate, which are the activated matrix manipulations. The manipulations are based on gating weights optimally learned through training. The following relationships define the gates and their associated weights [1]:

$$f_p = \delta(x_p U^f + h_{p\text{-}1} W^f), \quad (3)$$

$$i_p = \delta(x_p U^i + h_{p\text{-}1} W^i), \quad (4)$$

$$o_p = \delta(x_p U^o + h_{p\text{-}1} W^o). \quad (5)$$

The functions are recurrent to the hidden state of the previous time step and the current input data.

A candidate $\tilde{c}_p$ for the context state $\tilde{c}_p$ is computed by the given formula:

$$c_p = \tanh(x_p U^c + h_{p\text{-}1} W^c). \quad (6)$$

The previous context information is filtered by the forgetting gate and the present information from input gate in an equation is filtered through the context gate form the context state.

The gating machinery provides an ability to determine which long-term and short-term data to filter and pass to the final output representation [5].

The hidden state representation of the sequence h_{p-1} is computed as a combination of the filtered output from the output gate and current context information:

$$h_p = o_p \tanh(c_p). \quad (7)$$

The received output state can be used for predictions. This output can be used in the same way as hidden states were used in RNN architecture equation (1) considered above [6].

On this basis, LSTM can represent complex sequences and make stable predictions.

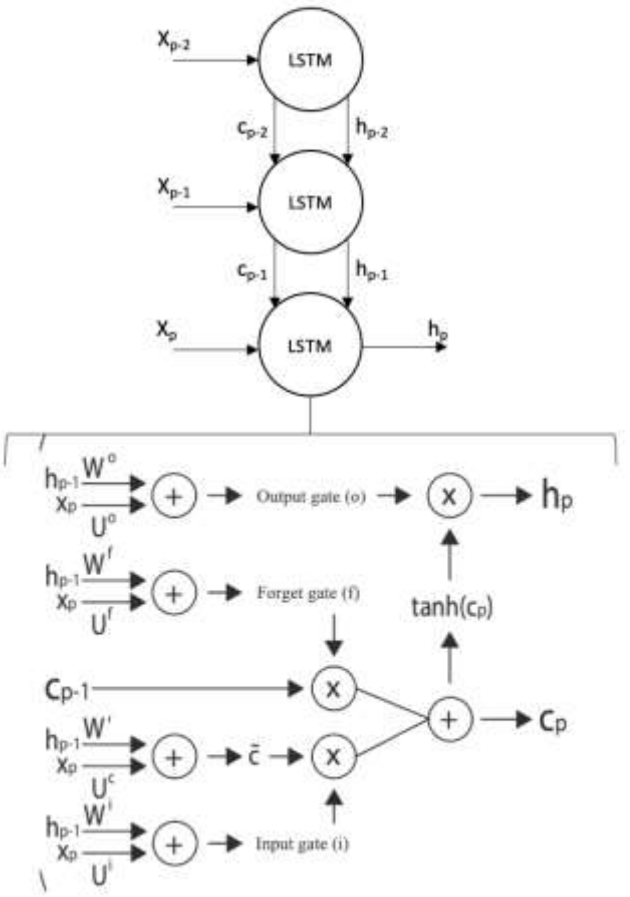

Fig. 2. The LSTM node

Bidirectionality. It should be considered that not all words can be predicted by the previous words in a sequence.

In the sequences like "bow tie" and "bow ring", "bow" is contextually dependent both on the previous and following word sequences. This is a challenge for regular recurrent networks. The solution for this challenge is bidirectionality, which means reversing direction of the sequence and feeding it to network. Both resulting hidden states are concatenated, which is a standard practice for many language models [7].

In fig. 3, the simple RNN from fig. 1 is extended to work bidirectionally.

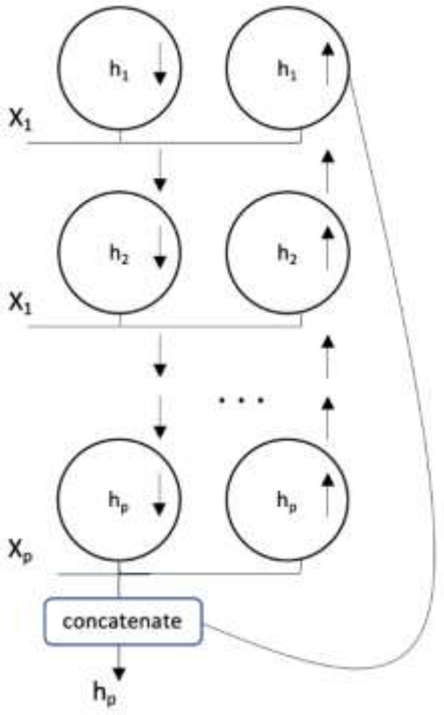

Fig. 3. Bi-directional RNN

The states h_p in fig. 3 can be extracted from regular nodes or LSTM. The network with the reverted states is a copy from the original network. The resulting hidden states are concatenated into form:

$$h_p = (\vec{h}_p, \overleftarrow{h}_p). \tag{8}$$

The final hidden state vector can be used similarly as for RNN through equation (8). The bidirectional models can be modelled by doubling the number of weights. This method is often applied to improve model representation and predictions due to its ability to add contextual information to language models [7].

Vector Representation of Language. Machine Learning models represented above have real valued vectors x_i. The language is represented as vectors which is an essential step in using neural networks as emotional classifiers. The words ware gathered in a vocabulary to form the bag of words [8]:

$$V = \{w_j : i \in 1, ..., N\}. \tag{9}$$

The phrase is divided into one-word hot vectors, which do not provide any extra information about the context. The vectors are also computationally hard to use as a single vector is a single word.

Word Embeddings. A more efficient solution is to pass a lower dimensional vector $x_j \in R_d, (d < N)$, which also contains language information about the word w_j. Less computational power is required to process such vector by the classificational model. This vector contains more language information than one-word hot vector. This type of lower dimensional representation is called word embedding [9].

Statistical language modelling is the other language feature. The words are used in conjunction with each other. This can extract more information about the semantic and syntactical use of a word through the context [10].

Principal Component Analysis (PCA) is a classic dimensional reduction technique, which is based on singular value decomposition of the co-occurrence matrix. This approach does not need linguistic rules to be fed to the system, which makes it unsupervised. It considers the text corpus in a whole, in different context, which also makes it computationally expensive.

This can be solved by using a feed forward neural network to predict the *n*-gram probabilities of the words in the vocabulary given by the context [10].

Trainable *d*-dimensional random initialized vector representation of each word and n previous words are used as a context. These vectors are concatenated and fed through several hidden layers. The output layer is a softmax probability over the vocabulary of all the probabilities to meet each word by the context words. The network is learning linguistically valuable information in the matrix of *d*-dimensional vectors while training to predict *n*-grams [11].

On this basis, words embedding can be used as the sole input to the other machine learning models for real NLP tasks, such as machine translation, and semantic sentence parsing. The fig.4 illustrates a scheme of such feed forward neural network.

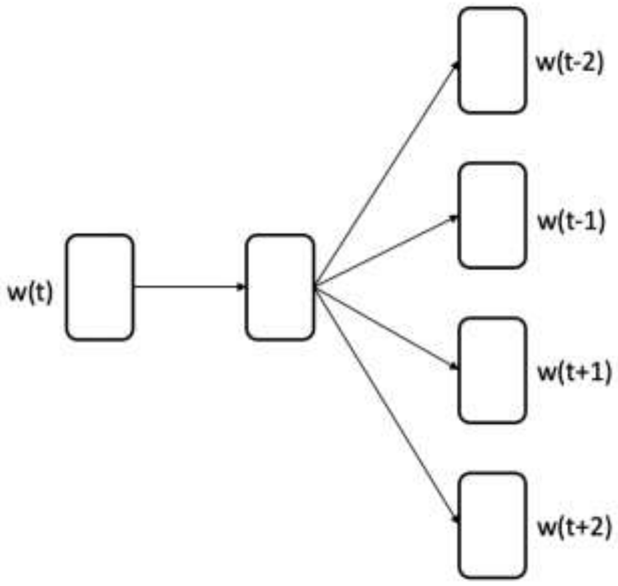

Fig. 4. Scheme of a feed forward neural network

It should be considered that this model is expensive to train. To increase the context window for the network to learn, there is a need to increase the number of input nodes, and the task of emotional classification requires a lot of context.

GloVe. The skip-gram model is another proposed RNN approach which is efficient on smaller datasets with rich context information. RNNs efficiently model rich context information due to the fact they have a sequence memory of the previously observed words. The model starts with the opposite probability comparing to the *n*-gram prediction, because this is the probability of different context words. The basis of this probability is a target word [12].

GloVe is the other unsupervised learning algorithm for obtaining vector representations for words. The difference of GloVe from the skip-gram and feed forward word embedding is the fact that GloVe is trained on a loss function. The loss function considers both local co-occurrences from the *n*-length context windows and global count-based co-occurrence probabilities from the text corpus. This allows to encode more of the language features comparing to PCA. This ability is provided since including only local context data does not give enough information to features about the frequency the words occur in rare contexts [12]. The loss function has slightly richer context information embedded in the vectors comparing to skip-gram model.

BERT. Pre-trained word embeddings in language are easy to use and flexible. Their drawback is that one vector can represent only one word. This means homonyms or phrasal verbs, like ‘key’ or ‘get’ lose their language information [14]. This can be mitigated by using RNN or LSTM to carry information over sequences to create a context for a single word, but these networks cannot provide rich context data to predict context-heavy language constructions.

One of the proposed approaches to solve this issue is BERT (Bi-Directional Representations from Transformers). In its core, the transformer relies on attention mechanism for its representation to improve the contextual awareness [14].

The weighted representation of all hidden state vectors is trained at the same time with the recurrence relations. This allows the network to be aware of the previous hidden states and do not rely solely on the final hidden state representation when making predictions.

The transformers apply attention mechanism to underlying word vectors from the original sequence to extract relevant language features. This is called text encoding. Since transformers do not use any recurrent relations, this allows to conduct training in parallel [5].

On this basis, BERT model is applicable to solve NLP tasks. BERT pre-trained weights can be downloaded via the Internet and used for transfer learning. This will allow to compare BERT with the other models in the task of emotion classification.

BERT is a sequence-to-sequence language representtation model, which is efficient for NLP tasks. A sequence of language information $X = (I_o, ..., I_n)$ is used as inputs and outputs as contextualized vector representation $H = (h_o, ..., h_n)$ of the elements of the input sequence.

During the pre-processing phase, the model splits words into word-parts. The placeholder tags are inserted before the sequence and after sentences, however this does not allow input vectors to directly correspond to the underlying words in the sequence [13].

Due to the design of BERT model, the output presentation h_o becomes a distributed representation of the underlying sequence. This sequence is similar to the final hidden state of RNN. A classification model can be obtained when adding an extra hidden layer to the BERT model and activating this model via a softmax function [9]:

$$\hat{y} = \varphi(Vh_0). \qquad (10)$$

Encoders, which are stacking nodes, are the other part of this language representation framework. Encoders are used to create encoded text representation (fig. 5).

Every encoder layer abstracts language patterns from an input sequence. It allows to form more complicated patterns as the information flows to the further layers. The first encoder layer L receives language inputs and the last encoder layer outputs the final encoded language information [15]:

$$\begin{gathered} H_1 = \text{Encoder}(X) \\ \ldots \\ \hat{y} = \varphi(Vh_0). \end{gathered} \qquad (11)$$

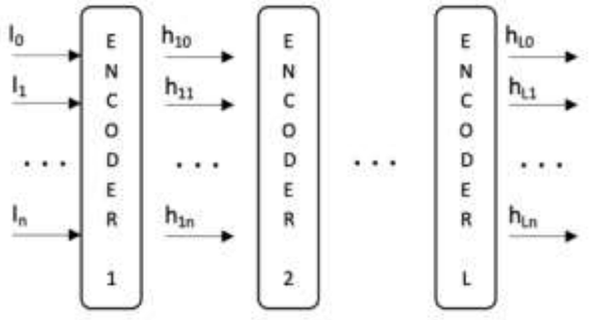

Fig. 5. Layers of the model

The first vector h of the final H representation is the basis for the classification task. Two sub-processes form each encoder layer. As the first step, the inputs to the encoder are passed through a multi-head self-attention layer. This layer uses a series of matrix manipulations to extract language information from the data inputs. The definition of this process is multi-head self-attention (fig. 6).

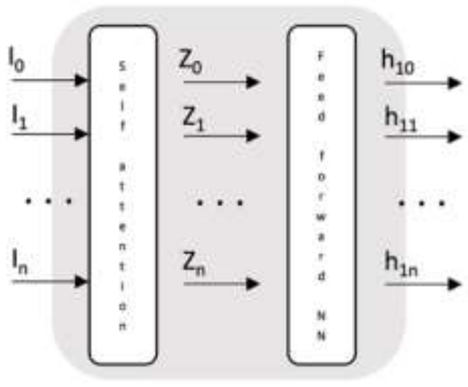

Fig. 6. The encoder dissection

The second sub-layer receives the outputs of the first layer after these outputs are residually connected and normalized. The second sub-layer retrieves the most valuable information from the attention layer. This data is residually connected and normalized yet another time [15]. After this, the outputs are sent onwards to the following encoder layer.

Theoretical propositions. This section is dedicated to proposition of a model for classifying the last phrase of three-turn conversations into any of pre-defined emotions. These emotions are Happy, Angry, Sad and Others. The context is defined by the previous two sentences. The datasets are classified beforehand by the emotion they represent, since model training should be performed on high-quality data. The dataset examples (data samples) of three-turn conversations and their corresponding labels can be seen in table 1. These data samples may include emojis and smileys as separate complete response options with emotional coloring.

This task is solved by creation of neural network architecture. BERT is used as a state-of-art language representation model. The proposed model and BERT are compared in terms of their performance on the given problem.

The pre-processing is divided into two steps. The first step is data cleaning, which stands for defining and fixing data irregularities. The second step is tokenization, which is a mathematical mapping of text data into a suitable input for classification models.

Table 1 – Data sample

id	Turn 1	Turn 2	Turn 3	
112	It's cool meme	Ha-ha, yes	⍰	Happy
154	I don't think it is a good idea	You better think twice	No, I won't!	Angry
186	In my hometown there are many chestnuts	Where is it?	In Kyiv	Others

Data cleaning provides an ability to the model to clarify the text content and boost the coverage. The coverage is measured by pre-trained embeddings as the number of words for which pre-trained embeddings exist divided by the total amount of unique words in the problem. Since our dictionary is problem-dependent and GloVe embedding matrix is generalized, it is not certain that the embedding matrix has a vector representation of a given word. Thus, data cleaning is recurrently applied to boost the coverage.

In the given problem, data cleaning is used to decompose abbreviations into their initial state and replace symbol emojis with Unicode emojis. This allows to improve sentence representation in terms of their meaning.

After the data cleaning, tokenization is applied. The words are vectorized and prepared for passing into classification models. The words are separated and mapped to unique numbers for further decoding. Keras, which is used to solve the problem, has this as a built-in feature.

As a result, tokenized texts are mapped with embedding matrices which represent the sequences in a vector space. The vector represented sequences are passed to LSTM models. The inputs have to be of equal length to allow to train the model weights to recognize features at the same positions in different sequences. To solve this, the vectors are zero padded to the same length, which is constrained to 100 words per sentence.

The fig. 7 illustrates the word length distribution of the training data.

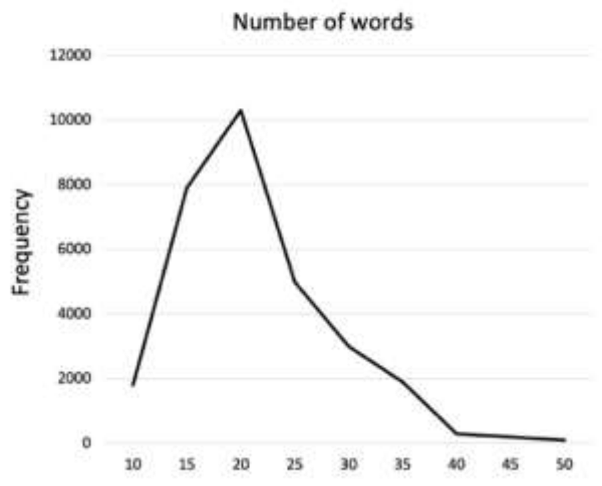

Fig. 7. Word lengths per sentence

Each sentence in a conversation is a matrix with elements of padded vectors representing the sentences. The corresponding labels are one-hot vectors. The index of the element 2 represents the emotion the training example is connected to. The indexes are as follows: 0 – Angry, 1 – Sad, 2 – Happy, 3 – Others. The given dataset is ready for further processing.

Proposed model. The proposed model (fig. 8) is built as follows. The standard LSTM outputs straight to a prediction layer and is filtered through the softmax function. The result is a probability vector. We use GloVe embeddings, trained on the text corpus represented by the data from Wikipedia. The 100-dimensional word embeddings and 128-dimensional LSTM layer with built in dropout are used by the model.

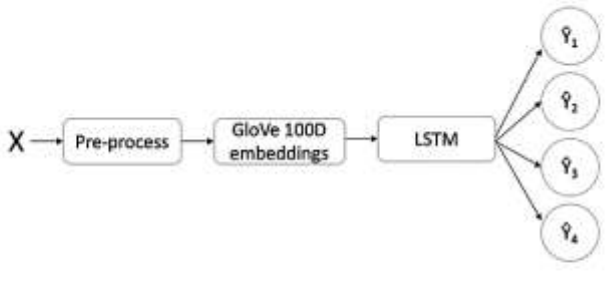

Fig. 8. The proposed model architecture

The model passes several improvements, which allow us to define features with high impact on model performance. This impact is quantitatively represented by F_1 score. Such features are isolated and used to create simple networks.

The first improvement is to separate the phrases of the conversation and pass them to LSTM nodes. Thus, the model is provided by a clearer vision of sentence differences.

The second improvement is to extract emojis and smileys from text for further separate processing. The emojis are represented with Emoji2Vec embeddings, which provides new language information to the model. Emoji2Vec is a 300-dimensional representation of Unicode smileys. Further concatenation of information contained in emojis with the data from the rest of the model is passed through the final dense layer. This allows the model to weight the relative importance of smileys for emotion classification. The rectified linear unit is the activation function in this dense layer.

BERT is used due to its applicability in wide range of problems. The model is hosted on TensorFlow hub. TensorFlow is a Google pre-trained machine learning repository.

It provides an ability to download an implementation which has to perform the following: the number of encoded layers, the number of heads in the multi-head attention, the number of dense layers in the feed forward network and the number of used training samples.

On the contrary to GloVe embeddings, BERT uses word-piece embeddings and follows different pre-processing and tokenizing which can be downloaded with the ready-to-use pre-trained model. This allows us not to perform the pre-processing step of the sentences but directly pass them to BERT.

Experimental results. The table 2 illustrates the comparative micro-averaged F_1 results for the models on emotion classification. The performance of the models was considered by evaluation metric on Happy, Angry and Sad emotions. The model performance on emotion class Others was ignored in this evaluation.

The LSTM model achieved overall micro-averaged F_1 as 0.616. This model required minimal data pre-processing and GloVe word embeddings. The word embeddings were trained on the Wikipedia data. It can be seen in the table, that the model achieved the worst results on Happy conversations and the best results on Angry conversations. The shortcoming of the model is the inability to distinguish the Happy, Angry and Sad emotion classes from the emotion class Others.

Table 2 – F_1 scores

Model	Emotion F_1			Micro F_1
	Happy	Sad	Angry	
LSTM	0.523	0.601	0.724	0.616
BERT	0.687	0.799	0.771	0.752
SS-LSTM	0.556	**0.818**	0.784	0.719
Proposed Model	**0.784**	0.767	**0.811**	**0.787**

BERT model achieved better results in macro F_1 comparing to LSTM due to the presence of extra context information. The Angry emotion class was predicted as 0.771 F_1 points. The Sad emotion class was also predicted better in F_1 points comparing to LSTM. The proposed model F_1 scores are shown in the table. The model has shown the best results on Angry and Happy emotion classes.

The confusion matrix of the proposed model is shown in table 3. The matrix illustrates the distribution between the emotion labels in the test dataset. The most confusion comes from distinguishing the Others emotion class from Happy, Angry and Sad emotion classes. Angry and Sad emotion classes are never evaluated by the model as Happy and vice versa. The confusion between predicting Angry and Sad labels is present rarely. On this basis, the focus should be put on prediction improvement among Others emotion class and the other three emotion classes.

Table 3 – Confusion Matrix

TRUE	Predicted			
	Others	Happy	Sad	Angry
Others	4103	101	80	94
Happy	63	215	4	1
Sad	40	0	212	8
Angry	46	0	6	334

The idea of whether the early stopping effectiveness on outfitting prevention is got by looking at the loss function over the training steps. The decrease of training loss is present over the training steps. The validation loss is decreasing by reaching the minimum after three epochs and is rising afterwards (fig. 9).

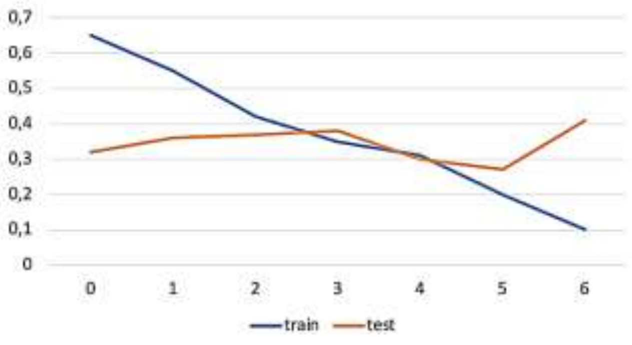

Fig. 9. Training and Validation Loss

The explanation behind the given tendency is the different distribution of emotions in the training dataset and validation dataset. This can be seen in table 4.

Table 4 – Emotion Class Distribution

	Others	Happy	Sad	Angry	**Size**
Train	0.441	0.980	0.211	0.194	29884
Validation	0.793	0.045	0.054	0.052	2467
Test	0.882	0.039	0.061	0.044	5804

The loss from the training set updates the gradients. The gradient updates change the validation loss in other ways. The amount of data points in the validation set is fewer comparing to the training set, which contributes to higher variance in the effects of updating the loss. On this basis, it is important to use early stopping after 3 epochs since the improvements in training loss are not contributing into improvements in validation loss after this epoch.

Conclusions. The problem of emotion classification is a complex and non-trivial task of language interpretation. The existing generic solutions for text data processing were considered in this work, having their shortcomings and advantages illustrated.

The evaluation of performance of the considered models was conducted. New model for emotion classification was proposed. The model is based on 128-D LSTM neural network combined with 100-D GloVe embeddings and BERT as a state-of-art language representation model.

The proposed model shows better results comparing to generalized model with transfer learning (having F_1 score as 78 versus 75). It should be noted that the proposed model needs further improvements to better capture ambiguous or mixed emotional expression differences represented by Others emotion class. It also showed slightly lower results on Sad emotion class compared to SS-LSTM model (having Emotion F_1 score as 0.767 versus 0.818). However, the proposed model still overpowers all considered models in micro F_1 score.

There are many decisions applicable for solving the emotion classification problem. There is a variety of factors to consider when choosing the word embeddings and training methods to design the model architecture.

To date, machine learning models did not achieve the human performance in terms of language representation and emotion classification. The confusion is present in labeling complex sentence structures. Such nuances are explained by the language structure and its dynamic nature.

However, the area of capturing the language nuances is constantly evolving and new approaches are created every day.

References

1. Graves A., Schmidhuber J. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural networks*. 2005. Vol. 18(5). P. 602–610.
2. Srivastava N., Hinton G., Krizhevsky A., Sutskever I., Salakhutdinov R. Dropout: a simple way to prevent neural networks from overfitting. *Journal of machine learning research*. 2014. Vol. 15. P. 1929–1958.
3. Daniiel Y., Onyshchenko K. Implementation of Recursive Deep Learning Algorithms for Natural Language Processing. *Information systems and technologies 2021*. Kharkiv-Odesa, 2021. P. 141–145.
4. Afanasieva I., Golian N., Hnatenko O., Daniiel Y., Onyshchenko K. Data exchange model in the internet of things concept. *Telecommunications and radio engineering.* 2019. Vol. 78(10). P. 869–878.
5. Collobert R., Weston J., Bottou L., Karlen M., Kavukcuoglu K., Kuksa P. Natural language processing (almost) from scratch. *CoRR*. 2011. P. 201–244.
6. Joulin A., Grave E., Bojanowski P., Mikolov T. Bag of tricks for efficient text classification. *CoRR*. 2016. P. 11–32.
7. Otter D.W., Medina J.R., Kalita J.K. A survey of the usages of deep learning in natural language processing. *CoRR*. 2018. P. 112–123.
8. Allen J. Natural language understanding. 2nd ed. Redwood City, CA, USA: Benjamin-Cummings Publishing Co., Inc., 1995. 512 p.
9. Gupta U., Chatterjee A., Srikanth R., Agrawal P. A sentiment-and-semantics-based approach for emotion detection in textual conversations. *In neu-ir: the SIGIR 2017 workshop on neural information retrieval*. 2017. P. 21–28.
10. Jurafsky D., Martin J.H. *Speech and language processing: an introduction to natural language processing, computational linguistics, and speech recognition*. Upper Saddle River, NJ, USA: Prentice Hall PTR, 2000. 1044 p.
11. Christopher M., Schütze H. *Foundations of statistical natural language processing. Cambridge*. MA, USA: MIT Press, 1999. 718 p.
12. Yao L., Guan Y. An improved LSTM structure for natural nanguage processing. *2018 IEEE International Conference of Safety Produce Informatization (IICSPI)*. Chongqing, China, 10–12 December 2018. 2018. P. 565–569.
12. Korti S. S., Kanakaraddi, S. G. Depression detection from Twitter posts using NLP and Machine learning techniques. *2022 Fourth International Conference on Emerging Research in Electronics, Computer Science and Technology (ICERECT)*. Mandya, India. 2022. P. 1–6.
13. Sharma D., Dhiman C., Kumar D. Automated Image Caption Generation Framework using Adaptive Attention and Bi-LSTM. *IEEE Delhi Section Conference (DELCON)*. New Delhi, India. 2022. P. 1–5.
14. Aziz A. A., Diamal E. C., Ilyas R. Paraphrase Detection Using Manhattan's Recurrent Neural Networks and Long Short-Term Memory. *6th International Conference on Electrical Engineering, Computer Science and Informatics (EECSI)*. Bandung, Indonesia, 2019. P. 432–437.
15. Yang H., Feng Y. Authoritative Prediction of Website Based on Deep Learning. *IEEE Fourth International Conference on Big Data Computing Service and Applications (BigDataService)*. Bamberg, Germany, 2018. P. 208–212.

References (transliterated)

1. Graves A., Schmidhuber J. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks*. 2005, vol.18(5), pp.602–610.
2. Srivastava N., Hinton G., Krizhevsky A., Sutskever I., Salakhutdinov R. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research.* 2014, vol. 15, pp. 1929–1958.
3. Daniiel Y., Onyshchenko K. Implementation of Recursive Deep Learning Algorithms for Natural Language Processing. *Information Systems and Technologies 2021*. Kharkiv-Odesa, 2021, pp. 141–145.
4. Afanasieva I., Golian N., Hnatenko O., Daniiel Y., Onyshchenko K. Data exchange model in the internet of things concept. *Telecommunications and Radio Engineering.* 2019, vol. 78(10), pp. 869–878.
5. Collobert R., Weston J., Bottou L., Karlen M., Kavukcuoglu K., Kuksa P. Natural language processing (almost) from scratch. *CoRR*, 2011, pp. 201–244.
6. Joulin A., Grave E., Bojanowski P., Mikolov T. Bag of tricks for efficient text classification. *CoRR*, 2016, pp. 11–32.
7. Otter D.W., Medina J.R., Kalita J.K. A survey of the usages of deep learning in natural language processing. *CoRR*, 2018, pp. 112–123.
8. Allen J. *Natural Language Understanding*. 2nd ed. Benjamin-Cummings Publishing Co., Inc. Redwood City, CA, USA, 1995. 512 p.
9. Gupta U., Chatterjee A., Srikanth R., Agrawal P. A sentiment-and-semantics-based approach for emotion detection in textual conversations. *In neu-ir: the SIGIR 2017 workshop on neural information retrieval*. 2017, pp. 21–28.
10. Jurafsky D., Martin J.H. *Speech and language processing: an introduction to natural language processing, computational linguistics, and speech recognition*. Upper Saddle River, NJ, USA, Prentice Hall PTR, 2000. 1044 p.
11. Christopher M., Schütze H. *Foundations of Statistical Natural Language Processing.* MIT Press, Cambridge. MA, USA, 1999. 718 p.
12. Yao L., Guan Y. An Improved LSTM Structure for Natural Language Processing, *2018 IEEE International Conference of Safety Produce Informatization (IICSPI).* Chongqing, China. 2018, pp. 565–569.
12. Korti S. S., Kanakaraddi, S. G. Depression detection from Twitter posts using NLP and Machine learning techniques, *2022 Fourth International Conference on Emerging Research in Electronics, Computer Science and Technology (ICERECT)*. Mandya, India. 2022, pp. 1–6.
13. Sharma D., Dhiman C., Kumar D. Automated Image Caption Generation Framework using Adaptive Attention and Bi-LSTM, *IEEE Delhi Section Conference (DELCON)*. New Delhi, India, 2022, pp. 1–5.

14. Aziz A. A., Diamal E. C., Ilyas R. Paraphrase Detection Using Manhattan's Recurrent Neural Networks and Long Short-Term Memory, *6th International Conference on Electrical Engineering, Computer Science and Informatics (EECSI)*, Bandung, Indonesia. 2019, pp. 432–437.
15. Yang H., Feng Y. Authoritative Prediction of Website Based on Deep Learning. *IEEE Fourth International Conference on Big Data Computing Service and Applications (BigDataService)*. Bamberg, Germany, 2018, pp. 208–212.

Received 10.05.2023

УДК 004.8

К. Г. ОНИЩЕНКО, Харківський національний університет радіоелектроніки, старший викладач кафедри програмної інженерії, м. Харків, Україна, e-mail: kostiatyn.onyshchenko@nure.ua, ORCID: https://orcid.org/0000-0002-7746-4570
Я. Д. ДАНІЄЛЬ, Харківський національний університет радіоелектроніки, асистент кафедри програмної інженерії, м. Харків, Україна, e-mail: yana.daniiel@nure.ua, ORCID: https://orcid.org/0000-0002-3895-0744

ВИКОРИСТАННЯ МЕРЕЖ ДОВГОТРИВАЛОЇ ПАМ’ЯТІ ДЛЯ ОБРОБКИ ПРИРОДНОЇ МОВИ

Проблема класифікації емоцій є складним та нетривіальним завданням інтерпретації мови через структуру природної мови та її динамічний характер. Актуальність дослідження полягає в охопленні важливої проблеми автоматичної обробки відгуків клієнтів, збирання думок та виявлення тенденцій. У цій роботі розглянуто ряд існуючих рішень для проблеми класифікації емоцій, де продемонстровано їхні недоліки та переваги. Оцінка продуктивності розглянутих моделей була проведена на класифікації емоцій чотирьох класів: Happy, Sad, Angry та Other. У цій роботі запропоновано модель для класифікації емоцій в трирядкових розмовах. Модель базується на емодзі та представленнях слів зі специфікою області сучасних розмов в Інтернеті. Досліджується важливість врахування інформації, отриманої зі емодзі як додаткового джерела даних з емоційним забарвленням. Оцінено продуктивність моделі та порівняно її з мовною моделлю BERT (Bidirectional Encoder Representations from Transformers) для класифікації емоцій. Запропонована модель показала кращу продуктивність у класифікації емоцій порівняно з BERT (з F_1-оцінкою 78 порівняно з 75). Слід зазначити, що потрібні додаткові дослідження для поліпшення обробки моделлю змішаних відгуків, що представлені класом емоцій "Other". Однак, сучасна продуктивність моделей для представлення та розуміння природної мови не досягла рівня людини. Є різноманітні фактори, які необхідно враховувати при виборі представлень слів та методів навчання для проектування архітектури моделі.

Ключові слова: обробка природної мови, нейронна мережа, природна мова, мережі довготривалої пам’яті, текстова класифікація, емоційний аналіз тексту.

Повні імена авторів / Author's full names

Автор 1 / Author 1: Онищенко Костянтин Георгійович, Onyshchenko Kostiantyn
Автор 2 / Author 2: Данієль Яна Дмитрівна, Daniiel Yana