Recognition of Toxicity of Reviews in Online Discussions

Kristína Machová, Marian Mach, Matej Vasilko

Department of Cybernetics and Artificial Intelligence, Technical University of Košice, Letná 9, 04200 Košice, Slovakia, kristina.machova@tuke.sk, marian.mach@tuke.sk, matej.vasilko@student.tuke.sk

Abstract: The paper solves some problems belonging to the field of recognition of asocial behaviour in online space. Nowadays, it seems to be an important issue when we must question our way of dealing with the pandemic crisis. Social network users must deal with such unhealthy phenomena in online space as toxic comments and toxic troll authors that prevent constructive communication and knowledge sharing through the web space. We have proposed a new multimodal approach to social network analysis, which combines two methods, the first one to recognize toxic posts using machine learning and the second one to identify toxic authors in online space using sentiment analysis. The recurrent neural network was trained with different numbers of neurons in the hidden layers using three different types of hidden layers and optimizers along with various learning rates. Finally, the paper provides detailed results of extended experiments with deep learning models for recognition of toxic reviews, where a model generated by a combined deep learning architecture achieved accuracy over 0.9, and results of our novel approach to the detection of toxic troll reviewers achieving accuracy of 0.95. Our approach to troll recognition is based on a comparison of the sentiment related to the authors' posts to sentiment related to all comments of an online discussion.

Keywords: web mining; data analysis; recognition of toxic posts; detection of toxic reviewers; trolling; deep learning; sentiment analysis; short texts processing

1 Introduction

The information technologies of social web have created various services for web users, which help with easier access to information and learning possibilities. Nevertheless, there is the other side of the coin. We have gone from the age of information to the age of misinformation, offensive speech, trolling, fake news or reviews, etc. All these types of antisocial behaviour affect democracy in many countries and contribute to the polarization of a society (Tristan Harris – former expert on ethics of a design in Google, cofounder of Centre for Human Technologies). Many times, it is the trolling and spreading of toxic posts that try to manipulate opinions of users looking for answers in online space. So, information

technologies can cause mass chaos, rudeness, lack of trust, loneliness, society polarization, hacking into elections and other democratic procedures and more populism. Disinformation campaigns have proven several times in history as a possible tool to achieve a certain goal. Today, these campaigns are implemented in the internet environment. The Internet becomes the home ecosystem for trollism. Trolls use misinformation, aggressive words, verbal assaults to destroy constructive discussions and to create the illusion of wider support for a certain opinion or a certain political candidate, which can gain more votes in elections.

In this paper, we focus on the toxic offensive content on Twitter available on http://www.kaggle.com. We also created a new dataset for our approach to toxic troll-opponent recognition. The contributions of the paper are as follows:

- A new multimodal approach to social network analysis, which combines two methods for toxicity recognition, deep learning, and sentiment analysis.
- Extended experiments with deep learning architecture suitable for the problem of toxic post recognition.
- A novel method designed for recognition of toxic reviewers, focused on troll-opponent, based on the comparison of sentiment related to the examined author's comments to all comments of the online discussion.

2 Toxicity Recognition in Online Space

2.1 Toxicity of Texts

There are more approaches to the automatic detection of toxic texts, for example approach based on keywords, on metadata, or approach based on machine learning methods. *Keywords based approach* focuses on the recognition of toxic speech based on author dictionary analysis. *Metadata based approach* uses additional information from social media regarding authors of toxic texts as location and time of submission or viewed pages. *Machine learning approach* builds a model from short texts that have been labelled as toxic respectively nontoxic.

There are many effective machine learning algorithms for text data. For example, the work [1] uses TF-IDF weighting scheme, part-of-speech tags, and other linguistic features for representation of text inputs for support vector machine (SVM). This model failed in classifying offensive words used in a positive sense.

Work [2] focuses on Twitter data and analyses user and textual properties from different angles of abusive behaviour (hate speech, sexism vs. racism, bullying, sarcasm, etc.). They propose a deep learning architecture, which utilizes metadata and combines it with automatically extracted hidden patterns within the text of the

tweets. From more possibilities, they used only simple GRU (Gated Recurrent Unit) network architecture. In our paper, we have experimented with three various neural network architectures and their combinations.

Another neural network-based approach presented in work [3] uses the average results of 10 neural networks with different initializations of weights. They built ensemble classifier and tested it on a publicly available dataset. They found that ensemble models perform better on test sets compared to the mean of sub-models.

The work [4] uses a shallow Feed-Forward Artificial Neural Network with 100 hidden neurons for emotions recognition including a toxic speech, from speech activity detection using EEG data. The problem of neural networks is that they cannot be easy interpreted. Work [5] offers fast and understandable computation using neural networks.

In [6] multiple approaches for toxic comment classification are presented. Authors showed that more approaches combined into an ensemble have together higher F1-measures and especially outperforms basic approaches when there is high variance within the data. The ensemble containing deep neural networks are especially effective.

We have decided to use the machine learning, particularly deep learning to solve the problem of recognition of toxic comments.

2.2 Toxicity of Authors

Our approach to identification of the credibility of reviewers is focused on detection of a toxic reviewer, particularly troll-opponent based on the identification of opinion polarity of his comments. We are focusing on the troll who is in constant opposition, in which such a troll usually stands in a clear minority.

Trolls try to influence public opinion, or just have fun [7]. There are many kinds of trolls, for example: Provocative troll, Troll-opponent, Social-engineering troll, Interest troll, Professional troll, Satirical troll, or Troll using the Rick-rolling method [8]. “Rick-rolling” is a specific method when troll argues in the comments and, after several contributions, contributes to the discussion by referring to an article or video in the description that it is proof of the truth of his opinion. However, when user clicks on this link, an irrelevant window or video appears.

The most common and effective solutions for troll recognition are machine learning models and sentiment analysis. If the goal of the approach is to identify specific trolls and to reveal the real user behind a fake account, then machine learning methods is more suitable. By effectively analysing all the troll's comments and finding similar characteristics of the troll's account based on behavioural patterns, machine learning can bring an effective solution to the problem [9]. On the other hand, in some special cases, better approach is using sentiment analysis. In the work [10], two approaches were provided for sentiment classification of tweets of trolls

to detect them. First, only longest sentence in the tweet was analysed. The sentiment of this sentence determined the entire tweet's sentiment polarity. The authors assumed that the longest sentence probably represents the main points of the tweet. Second, the sentiment of all sentences in a tweet was analysed and the mean of their sentiment score determined the tweet's score.

In the work [11], sentiment analysis was applied to the social network Twitter and used to identify trolls as well as political activists against other parties in the Pakistani parliamentary elections, allowing experts to assess the degree of conflict between the different parties. For performing successful sentiment analysis and using it to recognize trolling behaviour, it is important to recognize important factors (metadata) of the given text first. These factors can be for example: the length of the text, the number of rough and offensive words in the text, number of capital only words, etc.

2.3 Multimodal Approach to Toxicity Recognition in Social Networks

Our analysis of social networks comments attempted to give social networks users information about toxicity, both about possible toxicity of comments as well as the credibility of authors of these comments. So, our approach is focusing on recognition of toxic posts containing various types of offensive speech defined in [12] and simultaneously on recognition of toxic reviewers called trolls. This approach is illustrated in Figure 1.

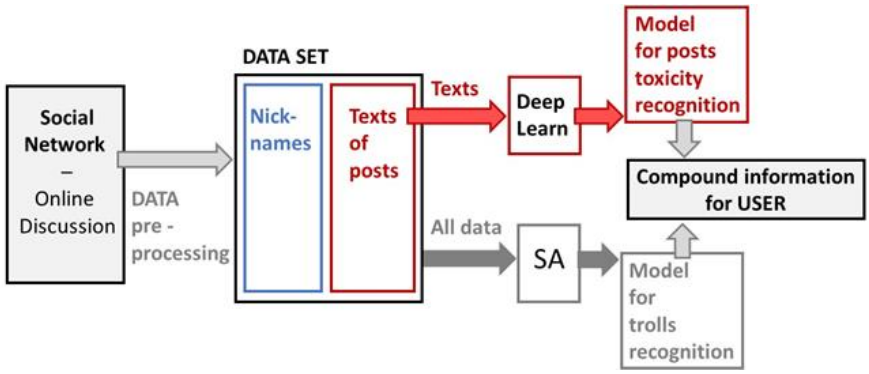

Figure 1

Multimodal approach to toxicity recognition combining the posts toxicity recognition and the recognition of toxic troll reviewers. A user is provided with compound information from two models.

The compound information for user consists of a combination of two predictions in four possibilities as follows:

1. toxic comment = true &toxic author = true (extremely suspicious),
2. toxic comment = true &toxic author = false (suspicious),

3. toxic comment = false &toxic author = true (suspicious),
4. toxic comment = false & toxic author = false (trustworthy).

The approach works with data from the given social network when data extraction is focused on short texts of an online discussion. The data are extracted and pre-processed into the form, which offers text data (texts of posts, tweets, comments) accompanied with data about authors of those texts (nicknames). Only the text data create input to the Deep Learn block where a model for recognition of toxic posts is generated using deep neural network learning. On the other hand, simultaneously the data are provided as input into SA model, but this block also needs data about authors (nicknames). The SA block performs sentiment analysis. It assigns value of polarity to each text when it is important to know who the author of the given text is and in that way the polarity value is assigned to each author. This information forms a base for training a model for toxic troll-opponent reviewer recognition. The building of this model is described in more detail in Section 7. The detailed structure of data processed by the two blocks DL (Deep Learn) and SA (Sentiment analysis) is illustrated in Figure 2. Blok DL represents using deep learning methods and block SA represents using lexicon-based sentiment analysis. All these methods are described in the following Sections 3.2 and 3.3.

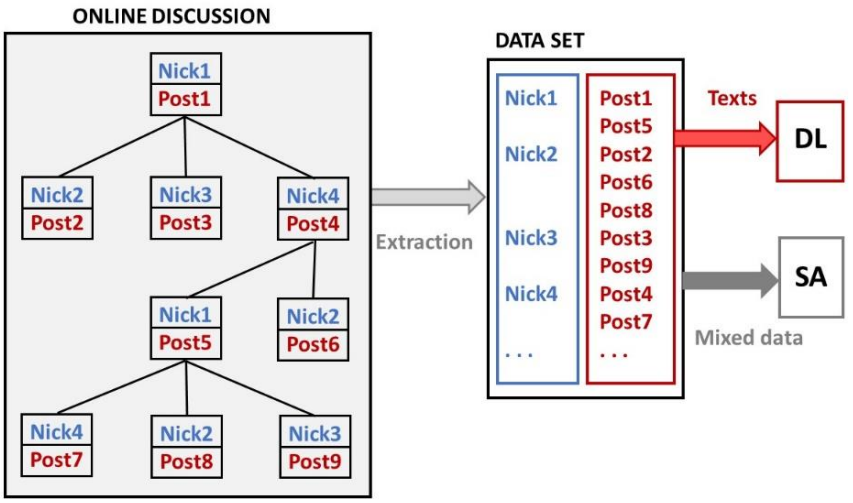

Figure 2

The data extraction from an online discussion and final structure of the data after pre-processing. The input for DL block contains only text data whereas block SA needs texts accompanied by information about their authors.

3 Used Methods

This first part of our Multimodal approach to analysis of social networks comments focuses on the recognition of a measure of posts texts toxicity using machine learning methods. For recognition of the measure of toxicity of authors a different approach was used, namely the sentiment analysis based on a lexicon.

3.1 Machine Learning Methods

Machine learning can get knowledge in the form of a model generalized from empirical data. Recently, mainly methods such as support vector machines, Bayesian networks or artificial neural networks (CNN or RNN) are successfully used for text processing.

Convolutional Neural Network (CNN) has topology defined by three types of layers: *convolutional layer*, *pooling layer*, and *fully connected layer*. The convolutional layer is represented by a set of kernels. We have used RNN (*Recurrent Neural Networks*) which are an excellent tool when working with the text. RNN treats each word of the sentence as a separate input at time *t*. The problem with recurrent neural networks is short-term memory. In case of long input, it has a problem shifting information from the past steps to the next. During backward promotion, the recurrent neural network has a problem with disappearing gradient [13]. To solve this memory problem, special types of RNN are created: LSTM (Long Short-Term Memory) and previously mentioned GRU. These networks use gates for the regulation of a flow of information.

3.2 Used Deep Learning Architecture

We have trained our models using LSTM, BiLSTM and GRU deep learning methods, which are described shortly in following for better understanding of the tested architectures.

LSTM (Long Short-Term Memory) [14] is an ordinary RNN, a neuron has only information relating to the entry and the past state. In LSTM, each cell contains a gate for Input, Output and Forget gate. The Forget gates use a sigmoidal function. The function returns a value between 0 and 1. A value closer to 0/1 means forget/remember. *BiLSTM (Bidirectional Long Short-Term Memory)* is a special type of LSTM, which can use context in both directions along the input sequence. It consists of two separate hidden layers, one used for the positive time direction (forward states) whereas the other one for negative time direction (backward states). *GRU (Gated Recurrent Unit)* represents a newer generation of recurrent neural networks. It is mostly like the LSTM network, but it uses only two gates, namely Reset gate and Update gate (represent Forget and Input gate in LSTM). The Reset gate decides how large part of past information should be forgotten [15].

In the process of neural networks training, we have used optimizing algorithms for setting and changing parameters of neural networks. We have experimented with three optimizers – ADAM, SGDM and RMSProp [16], [17]. *Adam (Adaptive Moment Estimation) optimizer* uses the first and second order approximations. The main point is that the gradient will not make large jumps so as not to accidentally avoid the minimum. However, this algorithm is computationally complex. *SGDM (Stochastic Gradient Descent Momentum) optimizer* represents two modifications of the simplest GD (Gradient Descent) optimization. The first modification of this optimizer is SGD (Stochastic GD) which tries to update network parameters more often than GD and therefore has faster convergence. The second modification SGDM convergence is softer, and fine. It speeds up convergence in the right direction and slows down convergence in the wrong direction. *RMSProp (Root Mean Square Propagation) optimizer* was developed as a stochastic method for mini-batch learning, which does not run after individual inputs but after groups of inputs. This method balances momentum and reduces jumps for large gradients.

3.3 Sentiment Analysis

The sentiment analysis represents mainly polarity of opinion analysis and emotions analysis. Solving polarity classification attempts to classify texts into three basic degrees of polarity: positive, neutral, and negative. The effectiveness of classification to opinion polarity relates on the way of negation processing (based on switching, shifting polarity and their combination) and intensification processing to determine the polarity of combinations of words.

The intensification processing attempts to identify the different degrees of positivity and negativity e.g., strongly negative, negative, fair, positive, and strongly positive. To apply sentiment analysis, it is necessary to estimate the strength of sentiment, which significantly changes the polarity of collocation, e.g., “surprisingly good”, “highly qualitative”, etc.

The sentiment analysis can be divided into three levels of investigation: document level analysis, sentence level analysis and analysis on the level of entity and aspects. In principle there are two approaches to sentiment analysis- machine learning approach and lexicon approach. For our purpose it is more suitable to select the lexicon approach. For high effectivity of lexicon approach, the quality is decisive, and it depends on the correct choice of words in the lexicon and their most accurate annotation – assigning to the degree of polarity. More about solving these problems of sentiment analysis can be found in [18].

4 Building a Deep Model for Toxicity Recognition

4.1 Data Analysis

The important step was to obtain the right dataset with sufficiently large number of examples for building a neural network. We were focusing on datasets available at https://www.kaggle.com/eldrich/hate-speech-offensive-tweets-by-davidson-et-al/, dealing with toxic offensive tweets. We have chosen the corpus of data containing the tweets that have been manually classified by CrowdFlower employees to the classes offensive and neutral, because we focused on toxic comments in the form of offensive language, which contained texts that can be considered racist, sexist, homophobic, or generally offensive.

Data were normalized, which is an important step for successful building a neural network. The use of normalization will cause that a convergence will not have a big range, which will ensure the possibility of optimization. First, the data were divided into two parts. The 70% were intended for learning and the rest 30% for testing. In the phase of data pre-processing, the following steps were provided:

- Separation of words of text by spaces
- Tokenization of words
- Transformation into lowercase letters
- Removing of diacritics of words
- Numeric indexation of words
- Conversion of a document into a sequence of indexes
- Justification of all texts to the same length of 35 tokens

Every annotator was familiar with the same definition of offensive speech. The context of tweets was also considered because of an occurrence of abusive word does not mean that the tweet is offensive. Each tweet in training set was labelled by three annotators. Final decision about class was formed as majority class between those votes. The test results were computed from numbers of true positive (TP), true negative (TN), false positive (FP), and false negative (FN) classifications of our models in comparison with labelling by CrowdFlower employees.

4.2 Neural Network Topology

The possibilities for simulating neural networks are currently extensive. We have used the Deep Learning Toolbox available in Matlab, namely Keras-like. We worked with the version of Matlab R2020a, because in this version GRU network topology is available. We have initialized deep learning of our models with the following neural network topology:

- *Input layer* – was represented by two layers "sequenceInputLayer" and "wordEmbeddingLayer". We have selected these layers because they are suitable specifically for text data processing when Deep Learning Toolbox is used.
- *Hidden layer* – consisted of three hidden layers available in Matlab that can be used for work with text data, namely: *GRU, LSTM* and *BiLSTM.*
- *Output layer* – consisted of two layers "softmaxLayer" and "classificationLayer". "SoftmaxLayer" represents an activation function and also takes care of the loss computing and "classificationLayer" splits output to classes: Offensive and Neutral.

The previously specified function on the output layer – "Softmax" offers the output value from [-1,1] range. We experimented with different types of the hidden layer in the network, several optimizers, and with parameters, namely values for the learning parameter, and the number of neurons in the hidden layers. From the beginning, the number of neurons in the hidden layer was the same, 70 neurons. The network was learned within 5 epochs. The number of iterations in each epoch was different but containing approximately 170 iterations.

The Figure 3 illustrates our most successful topology (according to Table 13) combining more various hidden layers. The first is GRU with 100 neurons and the second is BiLSTM with 112 neurons.

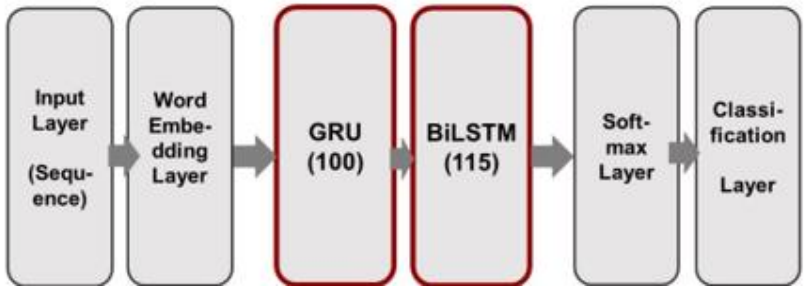

Figure 3
The topology of one of the tested neural networks deeply learned, when the hidden layer contained GRU network as the first layer followed by BiLSTM layer

5 Testing Basic Deep Learning Models for Toxicity Recognition

5.1 Experiments with LSTM

At first, we experimented with LSTM neural networks because they need not a precise tuning of parameters. LSTM networks work well over a wide range of parameters. At the beginning we found the optimal numbers of neurons in a hidden layer. We started with the Adam optimizer that represented the best choice when starting with the learning parameter $\delta = 0.1$ as a commonly used value (Table 1).

Table 1 shows that the best results were achieved using 85 neurons in the hidden layer. So, we used 85 neurons in the hidden layer for next experiments with three mentioned optimizers Adam, RMSProp and SGDM. The results of these experiments are presented in Table 2. Table 2 shows that the highest accuracy of 0.906 was achieved using LSTM network and optimalization algorithm SGDM with the learning parameter $\delta = 0.5$. This combination was used for training the model for recognition of toxic tweets. The resting results of this model are presented in Table 3.

Table 1
Testing results of LSTM network with the learning parameter $\delta = 0.1$

Number of Neurons	**Accuracy**	**Loss**
40	0.825	0.50
55	0.816	0.54
70	0.825	0.53
85	**0.843**	**0.48**
100	0.814	0.56
115	0.809	0.61
130	0.811	0.55

Table 2
Testing results of LSTM network using Adam, RMSProp and SGDM optimizers

Optimizers	**ADAM**		**RMSProp**		**SGDM**	
Learning Parameter δ	**Accuracy**	**Loss**	**Accuracy**	**Loss**	**Accuracy**	**Loss**
0.01	**0.874**	0.52	**0.883**	0.41	0.774	0.52
0.05	0.864	0.52	0.881	0.41	0.890	0.52
0.10	0.843	0.48	0.850	0.47	0.887	0.48
0.20	0.755	0.66	0.777	0.82	0.898	0.66
0.50	0.730	1.07	0.719	1.43	**0.906**	1.07

Table 3
Results of deep classifier based on LSTM network (85 neurons) using SGDM optimizer and learning parameter δ = 0.5 in the form of contingent table with numbers of true classifications (represented by the main diagonal) and false classifications to classes Offensive and Neutral Tweets. The rightest column contains values of Precision and the values of Recall for all classes are in the bottom row. The table also contains the value of F1-rate for Offensive class.

	Offensive	**Neutral**	**Precision**
Offensive	**1807**	26	**0.986**
Neutral	75	**390**	0.839
Recall	**0.960**	0.938	**$F1_{Off}$=0.973**

5.2 Experiments with GRU

Similarly, as in experiments with LSTM, in experiments with GRU we first have found the optimal numbers of neurons in the hidden layer (see Table 4). In this experiment, we have used the SGDM optimizer as the most successful in experiments with LSTM and the learning parameter δ = 0.1as a commonly used value.

The GRU network achieved the best result of accuracy = 0.913 using 100 neurons in the hidden layer. So, in following experiments 100 neurons in the hidden layer was used in combination with Adam optimizer, RMSProp optimizer and SGDM optimizer. The results of these experiments are presented in Table 5.

Table 5 shows that the highest accuracy = 0.913 was achieved using GRU network and optimalization algorithm SGDM with the learning parameter δ = 0.10. An interesting finding was the fact that GRU network using RMSProp optimizer had higher loss. The simpler GRU network achieved worse results generally, than LSTM one. The best GRU neural network was used for training a model for recognition of toxic tweets. Table 6 (contingent table) contains numbers of true classifications (at main diagonal) and false classifications to classes Offensive and Neutral Tweets.

The results which were achieved in recognition of the Offensive tweets were comparable with results of LSTM model and little bit better (F1rate = 0.978).

Table 4
Testing results of GRUnetwork with the learning parameter δ = 0.1

Number of Neurons	**Accuracy**	**Loss**
40	0.898	0.29
55	0.902	0.29
70	0.902	0.28
85	0.899	0.30
100	**0.913**	**0.27**
115	0.898	0.29
130	0.893	0.29

Table 5
Testing results of GRU network using Adam, RMSProp and SGDM optimizers

Optimizers	**ADAM**		**RMSProp**		**SGDM**	
Learning Parameter δ	**Accuracy**	**Loss**	**Accuracy**	**Loss**	**Accuracy**	**Loss**
0.01	**0.865**	0.56	0.859	0.50	0.781	0.54
0.05	0.853	0.46	**0.897**	**0.42**	0.902	0.32
0.10	0.825	0.63	0.720	1.28	**0.913**	**0.27**
0.20	0.722	1.19	0.617	2.37	0.902	0.30
0.50	0.713	2.52	0.640	3.40	0.899	0.31

Table 6
Results of deep classifier based on **GRU** network (100 neurons) using **SGDM** optimizer. The table also contains the value of F1-rate for Offensive class.

	Offensive	**Neutral**	**Precision**
Offensive	**1858**	34	**0.982**
Neutral	50	**382**	0.884
Recall	**0.974**	0.918	**$F1_{Off}$=0.978**

5.3 Experiments with BiLSTM

In this case experiments with the number of neurons in the hidden layer (see Table 7) showed that BiLSTM network achieved the best result of accuracy = 0.899 using 115 neurons in the hidden layer. Obviously, the number of neurons in hidden layer has not overly significant influence on the results of accuracy and the results of loss.

Table 7
Testing results of BiLSTM network with the learning parameter δ = 0.1

Number of Neurons	**Accuracy**	**Loss**
40	0.884	0.34
55	0.881	0.34
70	0.890	0.34
85	0.898	0.31
100	0.892	0.32
115	**0.899**	**0.30**
130	0.887	0.35

Based on these results, in following experiments, 115 neurons in the hidden layer were used in combination with Adam optimizer, RMSProp optimizer and SGDM optimizer. The results are presented in Table 8. The highest accuracy = 0.899 was achieved using BiLSTM network and SGDM optimizer with the learning parameter δ = 0.10. This network was used for training a model for recognition of toxic tweets.

Table 9 contains numbers of true and false classifications to classes Offensive and Neutral tweets. The precision of recognition of Offensive tweet is excellent again (0.989) but recall of recognition of Offensive tweets is lower than recognition of Neutral ones.

By comparing the three models considering the measure F1 rate, we get as the best model the deep model learned using GRU network. All tests in this section also proved the SGDM optimizer to be the best solution for deep learning model building for the task of recognition of offensive comments. Our results are better than those in Waseem and Hovy (F1 = 0.739 using n-grams) on the same dataset [19].

Table 8
Testing results of BiLSTM network using Adam, RMSProp and SGDM optimizers

Optimizers	**ADAM**		**RMSProp**		**SGDM**	
Learning Parameter δ	**Accuracy**	**Loss**	**Accuracy**	**Loss**	**Accuracy**	**Loss**
0.01	**0.870**	**0.50**	0.867	0.44	0.774	0.66
0.05	0.865	0.50	**0.868**	0.45	0.894	0.33
0.10	0.810	0.63	0.839	0.73	**0.899**	**0.30**
0.20	0.704	0.89	0.764	1.71	0.884	0.33
0.50	0.765	3.09	0.466	4.96	0.895	0.36

Table 9
Results of deep classifier based on BiLSTM network (115 neurons) using **SGDM** optimizer

	Offensive	**Neutral**	**Precision**
Offensive	**1808**	20	0.989
Neutral	102	**396**	0.795
Recall	0.947	0.952	**$F1_{Off}$=0.968**

6 Experiments with Combined Neural Networks

In our experiments, we aimed not only at finding the best values of neural networks parameters, but also at finding the best combination of different neural networks (GRU, LSTM, BiLSTM) and the best ordering of these networks in the hidden layer. We tried to improve our results by learning deep model based on a combination of hidden layers of basic neural networks. Each combination of layers uses the number of neurons and the optimizer according to the best result of the basic networks from the previous section. The learning parameter cannot be selected separately for each layer, so we tried several options. As an example, we present results of networks combined from two hidden layers, where the first layer is GRU and the second layer is LSTM. We worked with the assumption that the order of the layers is important. In the first GRU layer, 100 neurons and SGDM optimizer were

used. In the second layer SGDM optimizer was used too and the number of neurons in the case of LSTM was 85, but in the case of BiLSTM 115 neurons were used.

Table 10 illustrates the results of experiments when GRU layer is the first layer, and the second layer is represented by LSTM or BiLSTM. The number of neurons is ***n.*** In the experiments with combining neural networks, we have concentrated on Accuracy and Loss as measures of efficiency, which are currently most common in works using deep learning to solve problems.

Table 10

Testing results of deep network layers combinations with GRU as the first layer

Combination	**Parameter δ**	**Accuracy**	**Loss**
GRU (n=100) & LSTM (n=85)	0.05	0.869	0.36
GRU (n=100) & LSTM (n=85)	0.10	0.894	0.33
GRU (n=100) & LSTM (n=85)	0.20	0.902	0.30
GRU (n=100) & LSTM (n=85)	0.50	0.905	0.28
GRU (n=100) &BiLSTM (n=115)	0.05	0.864	0.36
GRU (n=100) &BiLSTM (n=115)	0.10	0.902	0.32
GRU (n=100) &BiLSTM (n=115)	**0.20**	**0.909**	**0.29**
GRU (n=100) &BiLSTM (n=115)	0.50	0.902	0.29

Tables 11 and 12 represent results of experiments with LSTM or BiLSTM layer used as the first layer. Table 11 shows that in case when LSTM layer is the first layer and the second layer is GRU, the learning parameter $\boldsymbol{\delta} = 0.2$ is preferred. When the BiLSTM layer is the second after LSTM, the learning parameter $\boldsymbol{\delta} = 0.5$ is preferred to obtain best results. Table 12 shows that BiLSTM layer is not suitable to be the first layer because the results are worse than those presented in Tables 10 and 11. Finally, we can conclude that neural networks combining more hidden layers did not achieve better results than basic networks, but difference is not high. Selection of best achieved results are presented in Table 13.

Tables 10-13 contain result of experiments with various possible combination of LSTM, GRU and BiLSTM networks. In these experiments, the best number of neurons in hidden layer was used according to former experiments – 85 for LSTM, 100 for GRU and 115 for BiLSTM. Within these experiments, parameter **δ** was changed to achieve the highest accuracy and the smallest loss. Table 13 is summarization of three best combinations according to achieved results. Particularly, the combination GRU (n=100) with BiLSTM (n=115) with Accuracy 0.909 is the best combined solution.

The accuracy of each combined network was nearly 0.91. The result of all these experiments showed that the best solution for training the model for toxic comments recognition is the basic GRU hidden layer with 100 neurons using SGDM optimizer.

Table 11
Testing results of deep network layers combinations with LSTM as the first layer

Combination	**Parameter δ**	**Accuracy**	**Loss**
LSTM (n=85) & GRU (n=100)	0.05	0.775	0.63
LSTM (n=85) & GRU (n=100)	0.10	0.876	0.36
LSTM (n=85) & GRU (n=100)	**0.20**	**0.906**	**0.29**
LSTM (n=85) & GRU (n=100)	0.50	0.774	0.63
LSTM (n=85) &BiLSTM (n=115)	0.05	0.774	0.63
LSTM (n=85) &BiLSTM (n=115)	0.10	0.843	0.48
LSTM (n=85) &BiLSTM (n=115)	0.20	0.876	0.36
LSTM (n=85) &BiLSTM (n=115)	**0.50**	**0.906**	**0.29**

Table 12
Testing results of deep network layers combinations with BiLSTM as the first layer

Combination	**Parameter δ**	**Accuracy**	**Loss**
BiLSTM (n=115) & GRU (n=100)	0.05	0.877	0.63
BiLSTM (n=115) & GRU (n=100)	0.10	0.766	0.36
BiLSTM (n=115) & GRU (n=100)	0.20	0.772	0.29
BiLSTM (n=115) & GRU (n=100)	0.50	0.774	0.63
BiLSTM (n=115) & LSTM (n=85)	0.05	0.870	0.63
BiLSTM (n=115) & LSTM (n=85)	0.10	0.859	0.48
BiLSTM (n=115) & LSTM (n=85)	0.20	0.791	0.36
BiLSTM (n=115) & LSTM (n=85)	0.50	0.750	0.29

Table 13
Combinations with the best results achieved

Combination	**Parameter δ**	**F1**	**Accuracy**	**Loss**
GRU (n=100) &BiLSTM (n=115)	0.20	0.974	0.909	0.29
LSTM (n=85) & GRU (n=100)	0.20	0.971	0.906	0.29
LSTM (n=85) &BiLSTM (n=115)	0.50	0.971	0.906	0.29

Table 14
Comparison of our approach with other approaches based on published results

	Methods	**F1**	**Accuracy**
Wasem et al. [19]	n-grams	0.739	0.690
Davidson et al. [1]	Tf-idf + SVM	0.910	0.841
D'Sa et al. [23]	BiLSTM	0.919	0.858
Maslej/Kresnakova et al. [24]	BiLSTM + CNN	0.690	0.897
Our best network	GRU(SGDM)	0.978	0.913
Our best combination	GRU(100) + BiLSTM(115)	0.974	0.909

Table 14 contain a comparison of our proposed method with the results of other related works. We can see that neural networks give better result than classic machine learning approaches (n-gams, tf-idf, svm). Using neural networks, our models gave better results than the other mentioned works.

7 An Approach to Recognition of Toxic Reviewer

Our approach to an identification of the toxicity of reviewers is focused on the detection of troll reviewers of the opponent type based on the identification of polarity of their comments using methods of sentiment analysis.

7.1 Approach to Detection of the Troll Opponent

The section presents an implemented approach to the detection of the troll-opponent, who is based on the analysis of sentiment and determining the opinion polarity of texts of comments of selected reviewers in online discussions. Our previous application LBSApso (Lexicon-based Sentiment Analysis Using the Particle Swarm Optimization) [18] was used to label individual comments by the value of opinion polarity, which was used for final prediction of troll-opponent class. We assume that the troll-opponent will contribute comments that will have negative polarity or will have the opposite polarity as the rest of comments of an online discussion. From the rating of all comments of the same reviewer, an average of the polarity values of his comments is computed. Subsequently, it is necessary to compute the average of polarity values of all comments of the entire online discussion as well because polarity of comments of the given reviewer is compared to the polarity of the whole discussion.

Reviewers who did not contribute a significant number of comments to the discussion cannot be considered as a troll-opponent, even if the polarity of their posts is significantly different from the polarity of the entire discussion. For those reviewers with the number of comments to the online discussion over a given threshold, the average polarity value of all their comments is calculated. This average value forms the input of further processing – comparison with polarity of the whole discussion. The polarity value of the entire online discussion is reduced by the part of its value that belongs to the reviewer against whose polarity the entire discussion polarity is compared. So, also the entire polarity must be computed for each comparison with the newly investigated reviewer again. In this way, each reviewer is compared to the average polarity of the rest of the entire discussion without his affecting the whole discussion polarity.

During further processing, the calculated difference between the polarity of the reviewer and the polarity of the whole discussion is used. If this difference value exceeds a predetermined threshold, then the given reviewer is classified to “TRUE” (Troll) class. If the difference is below the threshold, then the reviewer is classified

to "FALSE" (Non-troll) class. This Threshold is set to 2 experimentally, considering that polarity values are integers from [-3, +3]. Too small threshold value would lead to overestimated numbers of identified trolls (higher FP – false positive), although some reviewers would not be troll-opponents. On the other hand, too high threshold value would mean that few, if any, reviewers would be identified as troll (higher FN – false negative). Finally, the difference between two sums of polarity values (polarity of texts of a reviewer and polarity of the other texts) must be greater than the given threshold for a reviewer to be recognized as a troll-opponent.

The novelty of our approach is in using sentiment analysis based on lexicon. There are two basic approaches to the sentiment analysis, particularly based on a lexicon or based on machine learning methods. While previous works used the machine learning approach, we use the dictionary approach that can better detect dictionary typical for the troll-opponent.

7.2 Experiments

The new approach to the recognition of troll-opponent was implemented in the programming language Java in the development environment IntelliJ. For comments download the service ExportComments was used. This service was focused on Twitter, Facebook, Instagram, YouTube and TikTok. The extracted data contained nicknames, comments from all hierarchical levels of a discussion and also numbers of "I like" within mentioned social networks and was saved as an xlsx file. In this way, we have created a training set containing the posts of 100 reviewers. The dataset contained texts in the Slovak language. This dataset is available at http://people.tuke.sk/kristina.machova/ as Useful links in Research "Dataset for detection of the Troll-opponent" (http://people.tuke.sk/kristina.machova/useful/).

We have annotated the data manually, after deep analysis of all extracted comments to be able to compute the values of measures of effectiveness presented in Table 14. This annotation was discrete in the form of the class label Troll/Not-troll. We have tested our new approach on this training data set using service Online Confusion Matrix. The tests results are presented in Table 15.

Table 15

Testing results of new approach to troll-opponent recognition

Measure of effectiveness	**Value**
Recall = Sensitivity	0.833
Selectivity = Specificity	**0.966**
Precision = Positive predictive value	0.769
Negative predictive value	**0.977**
Accuracy	**0.950**
F1 score	0.800
Matthews correlation coefficient	0.772

The values of accuracy (0.950), selectivity (0.966) and negative predictive value (0.977) are very high. On the other hand, precision (0.769), recall (0.833) and F1 score (0.800) are also good, so we can state that our approach is promising. The second part of the multimodal analysis (recognition of toxic reviewers called trolls) gives comparable results with the first part of this analysis (toxic comments recognition). So, our multimodal analysis is balanced.

Conclusions

The aim of this work was to create a neural network model that can classify toxic content of online discussions with sufficient accuracy. As mentioned above, our model of deep neural network GRU achieved the best performance F1=0.978 using SGDM optimizer. The best optimizer was SGDM for all types of networks (LSTM, GRU and BiLSTM). Combined neural networks achieved more than 90% effectiveness. The best combination was GRU (n=100) layer as the first layer and BiLSTM (n=115) as the second layer. Some of experiments with LSTM layer were published in [20] but in this work presented experiments have been significantly extended by experiments with GRU layers, BiLSTM layers and with combined models of deep neural networks. The best deep model can be successfully used for recognition of the offensive content.

Our approach to troll-opponent recognition based on comparison of polarity of opinions of a given reviewer and polarity of all discussions achieved also very good results. It is a novel approach and seems to be quite usable.

The multimodal analysis is based on these two models and can be used to build a web service, which provides users of a social platform with detailed information about a given comment as one selection from: the comment is toxic, but its author is not; the comment is not toxic, but its author is toxic; both are toxic; neither of them is toxic.

For future, we would like to recognize the offensive speech from different types of monitored data. We would like to consider an image data, supplementing the text of comments to distinguish toxic emotions [21]. Another approach to online texts processing is processing them as texts streams to enrich this processing on the level of dynamic processing using adaptive bagging method for data streams processing [22]. Interesting future research could be to examine the possibility of increasing effectiveness of our model using BERT [23] for creation the embedding layer [24].

Acknowledgement

This work was partially supported by the Slovak Grant Agency of the Ministry of Education and Academy of Science of the Slovak Republic undergrant no. 1/0685/21 and The Slovak Research and Development Agency undergrant no. APVV-16-0213.

References

[1] Davidson, T.; Warmsley, D.; Macy, M.; Weber, I.: Automated Hate Speech Detaction and the Problem of Offensive language. In Proceedings of the 11th International AAAI Conference on Web and Social Media, ICWSM 2017, Association for the Advancement of Artificial Intelligence (www.aaai.org), March 2017, pp. 512-515

[2] Founta, A. et al.: A Unified deep learning architecture for abuse detection. InProc. 10th ACM Conf. on Web Science, 2019, pp. 105-114

[3] Zimmerman, S. Fox; C. Kruschwitz, U.: Improving hate speech detection with deep learning ensembles. In Proc. of the LREC 2018, Language Resources and Evaluation Conference, 2018, pp. 2546-2553

[4] Kosturova, M.; Juhár, J.: EEG-based Speech Activity Detection. In ActaPolytechnicaHungarica, 2021, Vol. 18, No. 1, pp. 65-77

[5] Alvarez, K et al.: Towards Fast and Understandable Computations: Which "And" and "Or"Operations Can Be Represented by the Fastest Neural Networks? Which Activations Functions Allow Such Representations? In Acta Polytechnica Hungarica, 2021, Vol. 18, No. 2, pp. 27-45

[6] Van Aken, B., et al.: Challenges for toxic comment classification: An in-depth error analysis. 2018, arXiv preprint arXiv:1809.07572

[7] Birkbak, A.: Into the wild online: Learning from internet trolls. InFirst Monday, 2018, Vol. 23, pp. 5-7

[8] Berghel, H.; Berleant, D.: The Online Trolling Ecosystem. In Computer2018, IEEE Computer Society, pp. 44-51

[9] Myhailov, T.; Gregoriev, G.; Nakov, P.: Finding opinion manipulation trolls in news community forum. In Proceedings of the 19th Conference on Computational Language Learning, Beijing, China, July 30-31, 2015, pp. 310-314

[10] Jansson, F.; Casselryd, O.: Troll detection with sentiment analysis and nearest neighbour search. Examensarbete Inom Teknik, Stockholm, June 2017, pp. 1-22

[11] Dollberg, S.: The Metadata Troll Detector, ETH, Zὺrich, January 2015, pp. 1-14

[12] Sap, M.; Card, D.; Gabriel, S.; Yejin, C.; Smith, N.: The Risk of Racial Bias in Hate Speech Detection. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, July 28 – August 2, 2019, Association for Computational Linguistics, 2019, pp. 1668-1678, DOI:10.18653/v1/P19-1163

[13] Graves, A.: Supervised sequence labelling. Supervised sequence labelling with recurrent neural networks, Springer, Berlin, Heidelberg, 2012, pp. 5-13

[14] Sak, H.; Senior, A.; Beaufays, F.: Long Short-term Memory Recurrent Neural Network Architectures for Large Scale Acoustic Modeling. In Proceedings of the Annual Conference of the International Speech Communication Association – INTERSPEECH-2014, 2014, pp. 338-342

[15] Dey, R.; Salem, F.: Gate-variants of Gated Recurrent Unit Neural Networks. In Proceedings of the MWSCAS 2017 – 60th IEEE International Midwest Symposium on Circuits and Systems, 2017, pp. 1597-1600

[16] Kingma, D. P.; Ba, J. A.: A method for stochastic optimization. In Proceedings of the International Conference on Learning Representations, San Diego, 2015

[17] Ruder, S. An overview of gradient descent optimization algorithms. In Computer science – ArXiv, 2016, Vol. 1609, pp. 1-14

[18] Machová, K.; Mikula, M.; Gao, X.; Mach, M.: Lexicon-based sentiment analysis using the particle swarm optimization. In Electronics, 2020, Vol. 9, No. 8, pp. 1-22

[19] Wassem, Z.; Hovy, D.: Hateful Symbols or Hateful People? Predictive Features for Hate Speech Detection on Twitter. In Proceedings ot the NAACL Research Workshop, San Diego, California, USA, 2016, pp. 88-93

[20] Machova, K.; Suchanic, D.; Maslej-Kresnakova, V.: Recognition of Hate or Offensive Tweets in the Online Communities. In Proceedings ot the ICETA 2020 – 18th IEEE International Conference on Emerging eLearning Technologies and Applications, Stary Smokovec, High Tatras, Slovakia, pp. 1-6

[21] Turabzadeh, S.; Meng, H.; Swash, R.M.; Pleva, M.; Juhar, J.: Facial Expression Emotion Detection for Real-time Embedded Systems. In Technologies, 2018, Vol. 6, No. 17, pp. 1-18

[22] Sarnovsky, M.; Marcinko, J.: Adaptive Bagging Methods for Classification of Data Streams with Concept Drift. In Acta Polytechnica Hungarica, 2021, Vol. 18, No. 3, pp. 47-63

[23] d'Sa, A. G.; Illina, I.; Fohr, D.: Bert and fasttext embeddings for automatic detection of toxic speech. In. Proceedings of the International Multi-Conference on“Organization of Knowledge and Advanced Technologies”(OCTA),Tunis, February 6-8, 2020, IEEE, pp. 1-5

[24] Maslej-Krešňáková, V., et al.: Comparison of deep learning models and various text pre-processing techniques for the toxic comments classification. In. Applied Sciences, Vol. 10, No. 23 (2020), https://doi.org/10.3390/app10238631