Computing and Informatics, Vol. 36, 2017, 1127–1142, doi: 10.4149/cai_2017_5_1127

SENTIMENT ANALYSIS OF MICROBLOGS USING MULTILAYER FEED-FORWARD ARTIFICIAL NEURAL NETWORKS

Vladimir Despotovic, Dejan Tanikic

Technical Faculty in Bor
University of Belgrade
Vojske Jugoslavije 12
19210 Bor, Serbia
e-mail: vdespotovic@tfbor.bg.ac.rs

Abstract. Sentiment analysis aims to extract public opinion on a particular topic and microblogs, especially Twitter as the most influential platform, represent a significant source of information. The application to microblogs has to cope with difficulties, such as informal language with abbreviations, internet jargons, emoticons, hashtags that do not appear in conventional text documents. Sentiment analysis technique for microblogs based on a feed-forward artificial neural network (ANN) with sigmoid activation function is proposed in this paper and compared to machine learning approaches, i.e. Multinomial Naive Bayes, Support Vector Machines and Maximum Entropy. Experiments were performed on Stanford Twitter Sentiment corpus, a balanced dataset which contains noisy training labels weakly annotated using emoticons as sentiment indicators; and SemEval-2014 Task 9 corpus, an unbalanced dataset which contains manually annotated training examples. The obtained results show that ANN produces superior or at least comparable results to state-of-the-art machine learning techniques.

Keywords: Sentiment analysis, opinion mining, microblogs, Twitter, neural networks, machine learning

Mathematics Subject Classification 2010: 68-T50

1 INTRODUCTION

Sentiment analysis refers to the use of natural language processing, computational linguistics and machine learning techniques to determine the sentiment content from the written language. It is also known under the name of opinion mining, as it analyzes people's opinions, attitudes and emotions with respect to products, services, organizations, individuals, events or topics [1]. Since it can track a public opinion on a particular topic, it is widely used in business intelligence to benchmark products, services and consumer attitudes [2, 3], in politics to predict the outcome of elections [4] or determine the politician's reputation [5], in sociology [6] and psychology [7].

Sentiment analysis is considered as a classification problem, either at the document or sentence level [8]. This classification is usually binary where the polarity of a document is positive or negative, ignoring the fact that sentiment is not always clearly expressed. The rationale behind this is the assumption that there is less to learn from neutral examples comparing to the ones with clear positive or negative sentiment. However, in most of the polarity problems there is also a third, neutral class that should be taken into account. Moreover, the use of neutral training examples can improve distinction between positive and negative category [9]. Another way to deal with the neutral examples is to use a hierarchical approach where the document is first classified as neutral or polar, and in the second step sentiment polarity (positive or negative) is determined [10, 11].

The research on sentiment analysis goes in two main directions: the lexicon-based and the machine learning-based approaches [8]. The lexicon-based approaches rely on predefined lexicons of words, e.g. SentiWordNet [12] coupled with sentiment orientations to determine a sentiment of the sentence or the document under consideration. Their main advantage is the fact that they require no training data, and can be used across different domains [13]. However, the major drawback is a finite number of lexicon words, which may become a problem in dynamic environments such as Twitter or Facebook, where new terms, abbreviations and malformed words constantly emerge [14]. Alternatively, the machine learning based approaches can be used to train a sentiment classifier on a large set of labeled examples, which usually leads to better accuracy, but requires manual annotation. Moreover, it is domain dependent and performs poorly when applied to domains other than the one they were designed for [15].

In this paper we analyze a task of sentiment analysis for microblogs. We focus on Twitter, as the most popular and influential microblogging platform. Unlike standard text classification problems with large amount of text that can help gathering sufficient statistics, Twitter posts (tweets) have a limit of 140 characters, which is similar to a classification problem on a sentence level. However, the language in Twitter is rather informal, using many abbreviations (e.g. OMG, BTW, ASAP), internet jargons and contemporary spellings, emoticons, hashtags that do not appear in conventional text documents. Additionally, linguistic analysis in Twitter has to deal with often incorrect use of grammar and the lack of context. All this makes the task of sentiment analysis for microblogs very specific.

One of the first studies on sentiment analyses in Twitter was carried out by Go et al. [16] who created a large training dataset using only emoticons for filtering positive and negative sentiment, without the need for manual labelling. Although the obtained results using machine learning techniques, such as Support Vector Machine (SVM), Naive Bayes and Maximum Entropy justified the way the corpus was generated, the major drawback was ignoring the possibility of having the neutral sentiment. Pak and Paroubek [17] took into account the neutral tweets as well and studied the validity of Twitter for the sentiment analysis. They found that Conditional Random Fields performed the best among several tested machine learning algorithms. A hybrid method for classification of sentiment polarity is proposed by Zhang et al. [18], combining lexicon based approach with SVM. An excellent survey of techniques for Twitter Sentiment analysis is given in [19].

ANNs have recently attracted attention in sentiment learning. The automatic sentiment analysis system, named CIS-positive, was proposed in [20]. Ebert et al. present a combination of deep Convolutional Neural Networks (CNN) and SVM, where CNN was used to extract not only local features, but also context to predict sentiment, and its output is fed into SVM for better classification. A neural network based approach to combine the advantages of the machine learning and information retrieval techniques for sentiment classification in the blogosphere was used in [21]. A new Convolutional Neural Network architecture that fully uses joint text-level and image-level representation to perform multimedia sentiment analysis was performed by Cai and Xia [22]. Based on the idea of complementary effect of two representations as sentiment features, the proposed method exploits the internal relation between text and image in image tweets and achieves better performance in sentiment prediction. Socher et al. [23] proposed semi-supervised technique using recursive autoencoders for prediction of sentiment label distributions on a sentence-level. In [24] the authors use a deep convolutional neural network with two convolutional layers to extract relevant features at a character, word and sentence level in order to exploit from character to sentence level information (Character to Sentence Convolutional Neural Network CharSCNN). Ghiassia et al. [25] proposed a hybrid technique that uses n-gram approach to extract reduced set of features and dynamic artificial neural network (DAN2) for sentiment analysis task.

We propose the use of feed-forward neural network trained with scaled conjugate gradient back-propagation algorithm. Additionally, we compare our results to standard machine learning techniques used in semantic analysis, such as Multinomial Naive Bayes (MNB), Support Vector Machines (SVM) and Maximum Entropy (MaxEnt). We apply our approach to two datasets: the Stanford Twitter Sentiment corpus, which contains 1.6 million noisy labels (positive/negative) for tweets that were annotated automatically using only emoticons; and SemEval-2014 Task 9, which contains approximately 10 000 manually classified tweets into positive/negative/neutral categories, where the test dataset is a combination of Twitter, SMS and LiveJournal messages. The main motivation behind the choice of the datasets lays in their conceptual differences. While Stanford Twitter Sentiment corpus is a balanced dataset, SemEval-2014 Task 9 corpus is higly unbalanced with nonuniform distribution among classes. Moreover, SemEval-2014 Task 9 includes in the test dataset out-of-domain messages, which enables to test how the systems trained on Twitter data only will generalize on non-Twitter text messages (SMS and LiveJournal). On the other hand, Stanford Twitter Sentiment corpus is interesting since it is collected using only emoticons as a filter, avoiding the need for manual annotation, which is expensive and time-consuming task.

The paper is organized as follows. In Section 2 we present an approach for sentiment analysis of microblogs based on ANNs. More details about the datasets used in our experiments are given in Section 3. Section 4 describes an experimental setup, followed by the discussion on obtained results in Section 5 and concluding remarks in Section 6.

2 ARTIFICIAL NEURAL NETWORKS FOR SENTIMENT ANALYSIS

Multilayer feed-forward neural networks are one of the most widely used types of ANNs. They have a characteristic layered architecture, where the first layer is an input layer, the last layer is an output layer and the layers in between are known as hidden layers. Each layer is composed of one or more processing units called neurons, such that each neuron is connected to one or more other neurons by real-valued weights. Multiple layers of neurons with nonlinear transfer functions allow the network to learn nonlinear relationships between input and output vectors.

For our setup we define a two-layer feed-forward network, with sigmoid hidden and output neurons, as shown in Figure 1. The number of neurons in the input layer is equal to the number of detected tokens N in the training dataset. Hence, each input vector is an N-element vector with zeros if the token does not appear in the tweet and the number of token occurrences otherwise. Each output vector is a C-element vector, where C represents the number of target classes. X is the number of neurons in the hidden layer. Each neuron of the hidden layer calculates the argument of the transfer function a_i, in the following way:

$$a_i = x_1 w_{i,1} + x_2 w_{i,2} + \ldots + x_n w_{i,n} + b_i \tag{1}$$

where x_j is the j^{th} input element, $w_{i,j}$ are the weight coefficients between i^{th} hidden and j^{th} input element and b_i is a bias coefficient. The i^{th} neuron produces output:

$$y_i = f(a_i) = f\left(\sum_{j=1}^{n} x_j w_{i,j} + b_i\right). \tag{2}$$

Hidden and output neurons have either hyperbolic tangent sigmoid transfer function:

$$y = \text{tansig}(a) = \frac{2}{1+e^{-2a}} - 1 \tag{3}$$

or log-sigmoid transfer function:

$$y = \text{logsig}(a) = \frac{1}{1 + e^{-a}}. \tag{4}$$

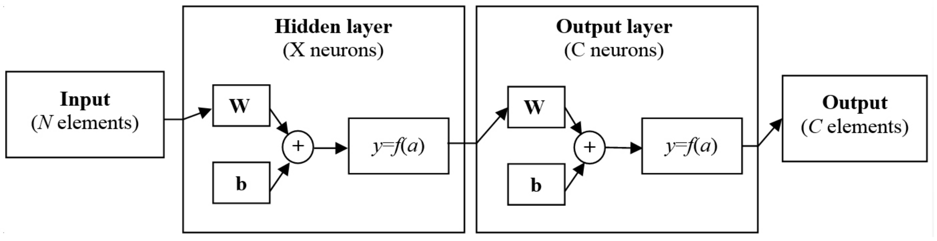

Figure 1. Structure of the proposed Artificial Neural Network

This output represents an input to the neurons of another layer, or an element of the neural network's output vector.

We also experiment with three-layer feed-forward neural networks, by introducing a second hidden layer with sigmoid transfer function. However, one should be very careful with increasing the number of hidden layers, since it also increases training time and the danger of overfitting, which can lead to poor generalization for the test dataset. Using two hidden layers exacerbates the problem of local minima, which can have extreme spikes even when the number of weights is much smaller than the number of training cases.

Instead of adopting the traditional gradient descent method, the network is trained using the scaled conjugate gradient (SCG) back-propagation algorithm, to speed up the convergence. Back-propagation is used to calculate derivatives of performance with respect to the weight and bias variables. Although this routine usually requires more iterations to converge compared to other conjugate gradient algorithms, the number of computations in each iteration is significantly reduced because no line search is performed. By using a step size scaling mechanism, SCG avoids a time consuming line search per learning iteration, which makes the algorithm faster than other second-order algorithms (e.g. conjugate gradient with line search, Broyden-Fletcher-Goldfarb-Shanno quasi-Newton algorithm) [26]. Training stops either when the maximum number of epochs is reached or the network performance on the validation set fails to improve for predefined number of epochs in a row.

3 DATASETS

3.1 Stanford Twitter Sentiment Corpus

The Stanford Twitter Sentiment corpus contains tweets collected via Twitter API using only emoticons as a filter. The advantage of this way of collection of training examples is the fact that no manual labeling is required. However the accuracy is questionable since emoticons might not always represent the actual sentiment of the tweet; therefore the labels cannot be considered as ground truth. The training set contains 1.6 million tweets, 800 000 with positive and 800 000 with negative emoticons. Neutral or objective tweets are not taken into account, which makes this a binary classification task.

The test data is manually annotated and consists of 177 negative tweets and 182 positive tweets. Not all the test data includes emoticons. The test dataset is collected by searching Twitter API with specific queries including consumer products (bing, kindle2, ...), companies (aig, at & t, ...), and people (Bobby Flay, Warren Buffet, ...) [16].

3.2 SemEval-2014 Task 9 Corpus

Corpus was collected within the SemEval-2014 Task 9 competition that was a part of the International Workshop on Semantic Evaluation. Training data is actually the same as in SemEval-2013 task 2 competition. The tweets were gathered on a range of topics using Twitter API, including entities (e.g., Gadafi, Steve Jobs), products (e.g. Kindle, Android phone) and events (e.g. Japan earthquake, NHL playoffs) over a one-year period from January 2012 to January 2013. Keywords and Twitter hashtags were used to identify messages relevant to the selected topic [27]. We also added a development set to a training set, that was intended to be used as evaluation dataset at the development-time, which left us with 9 353 manually annotated messages for training (not all the messages were available at the time of download), out of which 3 455 have positive, 1 423 negative and 4 475 neutral sentiment.

The test tweets had different topics from training and spanned later periods. The complete test dataset combines five test-sets collected for SemEval-2013 and SemEval-2014 competitions: Twitter 2013, SMS 2013, Twitter 2014, Twitter Sarcasm 2014, LiveJournal 2014 and contains 8 055 manually annotated messages in total. The idea of having a test set composed of messages from different domains is to find out how the systems trained on Twitter data only will generalize on out-of-domain text messages.

4 EXPERIMENTAL SETUP

While the task for the Stanford Twitter Sentiment corpus is a binary classification task and determines the polarity of the sentiment (positive or negative), the SemEval-2014 Task 9 corpus also includes a third, neutral class representing messages without a clear sentiment. This makes the task a multi-classification problem. Additionally, the task is harder since the training dataset is unbalanced and the test dataset beside Twitter includes SMS and LiveJournal messages, which were not used in the training process.

The same feature extraction is applied to all machine learning techniques used in our experiments. We use word counts as features, where feature function equals zero if the word does not appear in the tweet and the number of word occurrences otherwise. We also experiment with bigrams. The feature reduction process is used to decrease the dimensionality of the feature vectors and includes the following steps:

Removing infrequent words: removing words that appear less than n_{min} number of times;

Removing stop words: removing words from the stop-words dictionary that add no sentiment value (articles, some prepositions, etc.);

Stemming: reducing distinct words to their root form (stem). We use Porter stemmer [28] that identifies word suffixes and strips them off, leaving only the base word form (e.g. *connect*, *connected*, *connection*, *connecting* are all reduced to a word stem *connect*);

Using bigrams: a set of two consecutive words is added to the set of features. This can be beneficial for detecting negations (e.g. while word *good* has a positive sentiment, *not good* will obviously define a negative sentiment, which can be captured by a bigram);

Removing hyperlinks: hyperlinks are very often in tweets, but they add no sentiment value; therefore they are replaced by the token *URL*;

Removing usernames: Twitter usernames are often used to direct the messages using @ character as a prefix (@*username*). All the tokens starting with @ character are replaced by the token *ATUSER*;

Removing hashtags: hash character # is used in Twitter as a prefix to a word and enables to search for tweets that have a common topic. Hashtags are removed, leaving only the root word, which may contain information important for determining sentiment;

Removing repeated letters: language in Twitter is rather casual and often contains repeated letters to emphasize the word (e.g. *gooooooood*). All the letters occurring more than two times in a word are replaced by two occurrences;

Replacing emoticons: we replace the positive emoticons by the common word that defines positive sentiment (e.g. *good*) and the negative emoticons by the common negative sentiment word (e.g. *bad*);

Removing non-word characters: punctuations, numbers and white spaces are removed.

Finally, we estimate the mutual information between features that are still present after the feature reduction is carried out [29], sort the features in descending order according to the mutual information and use only the subset of features that has the largest influence on sentiment classification.

5 EXPERIMENTAL RESULTS AND DISCUSSION

We use the following baseline machine learning techniques: Multinomial Naive Bayes, Support Vector Machines and Maximum Entropy, to compare with a multilayer feed-forward artificial neural network for a task of sentiment analysis of microblogs. A linear kernel is used for SVM, having in mind large number of features. In this case there is no need to map data to a higher dimensional space, that is, the nonlinear mapping does not improve the performance. In case there are more than two classes, multiclass classification is implemented using one-against-one approach, where one SVM is constructed for each pair of classes [30]. Classification is performed according to the maximum voting criterion; the unknown entry is assigned to the class with the highest number of votes. We used LIBSVM library for SVM implementation[1] [31]. The limited-memory quasi-Newton optimization algorithm (L-BFGS) was used to determine the optimal weights in MaxEnt. In our work we used MaxEnt implementation given in [32][2].

As a performance measure we use a balanced F-score, which is a harmonic mean of precision and recall

$$\text{F-score} = \frac{2 \cdot \text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}. \tag{5}$$

Precision is a function of its correctly classified examples (true positives) and examples misclassified as positives (false positives), while recall is a function of true positives and its misclassified examples (false negatives) [33], as in:

$$\text{precision} = \frac{\text{true positives}}{\text{true positives} + \text{false positives}},$$
$$\text{recall} = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}}.$$

The same feature extraction is applied to all machine learning techniques used in our experiments, as explained in Section 4. We removed all infrequent words that appeared less than $n_{min} = 7$ times (the optimal number is empirically obtained). Stemming was omitted, since it was shown that it does not contribute to performance, even reduces it in some cases, mainly due to over-stemming errors. Finally, we applied feature reduction based on mutual information to decrease the dimensionality of the feature vectors. We empirically determined that leaving only a subset of 55 % of all the features that has the largest influence on classification was a good trade-off between performance and computational complexity.

A list of ten most discriminative features based on mutual information criterion for SemEval 2014 task 9 corpus is given in Table 1. Note that adding the bigrams resulted in only one bigram feature (cannot wait) in the list of 10 most influential

[1] Available from http://www.csie.ntu.edu.tw/∼cjlin/libsvm
[2] Available from http://www.cs.grinnell.edu/∼weinman/code/

	1	2	3	4	5	6	7	8	9	10
Unigram	*good*	*love*	*url*	*great*	*happy*	*bad*	*excited*	*ur*	*fuck*	*wait*
Uni+Bigram	*good*	*love*	*url*	*great*	*happy*	*cannot wait*	*bad*	*excited*	*fuck*	*ur*

Table 1. Ten most discriminative features in SemEval 2014 task 9 corpus based on mutual information criterion

ones. On the other hand, 32 % of 100 most discriminative words were found to be bigrams. However, while some of them are clearly good indicators of sentiment polarity (e.g. *looking forward*, *happy birthday*), contribution of the most of them (e.g. *excited for*, *forward to*, *wait for*, *to see*...) is questionable compared to the influence of single words (unigrams) it consists of. Also note that token *url* which replaces hyperlinks during the feature reduction process is highly ranked (3rd place) in this list, suggesting that hyperlinks have an important influence on sentiment classification.

We first present the results obtained on a Stanford Twitter Sentiment corpus (STS). For the training we randomly selected the subset of 50 000 tweets, 25 114 of them being positive and 24 886 of them being negative, making it a balanced training dataset. The training examples in STS are not manually labelled, but only weakly annotated using emoticons as semantic indicators for sentiment, which makes this a distant supervision task. Due to the lack of human supervision, such heuristically derived training labels may be noisy [34]. Obviously, learning sentiment classifiers from noisy labels may decrease the overall performance [35]. However, the results obtained on this corpus justify the automatic annotation of training samples, avoiding the need for expensive manual labelling.

	Number of Neurons in the Hidden Layer					
One Hidden Layer	**10**	**20**	**50**	**100**	**200**	**# Features**
Unigram (tansig)	19.50 %	20.33 %	21.45 %	21.45 %	22.56 %	2 646
Unigram (logsig)	**18.94** %	20.06 %	20.61 %	20.06 %	21.45 %	2 646
Uni+Bigram (tansig)	23.40 %	23.40 %	24.79 %	24.23 %	25.35 %	8 861
Uni+Bigram (logsig)	**21.45** %	23.68 %	22.68 %	25.07 %	22.01 %	8 861
Two Hidden Layers	**10/10**	**20/20**	**50/50**	**100/100**	**200/200**	**# Features**
Unigram (tansig)	19.50 %	21.45 %	21.17 %	21.45 %	22.01 %	2 646
Uni+Bigram (tansig)	22.84 %	23.68 %	25.63 %	23.40 %	22.56 %	8 861

Table 2. Classification error using proposed ANN for various neural network architectures for Stanford Twitter Sentiment corpus

Table 2 shows the classification error obtained using two-layer feed-forward ANN (one hidden layer) with different number of neurons in the hidden layer ranging from 10 to 200, and two sigmoid transfer functions: hyperbolic tangent sigmoid and log-sigmoid. We also present results with three-layer ANN, by introducing a second hidden layer with sigmoid transfer function. The best result obtained using unigram and a combination of unigram and bigram features is highlighted in Table 2. Note

		MNB	SVM	MaxEnt	ANN
Unigram	F-score	79.11 %	79.94 %	76.88 %	**81.06** %
	Learning time	1.51 s	748.21 s	15.58 s	155.40 s
Uni+Bigram	F-score	76.04 %	78.83 %	79.39 %	78.55 %
	Learning time	5.58 s	1 476.93 s	113.18 s	576.10 s

Table 3. Performance for Stanford Twitter Sentiment corpus using proposed machine learning techniques

that increasing the number of neurons did not bring the expected improvement in performance; hence there is no need for using large neural networks which can be memory and time consuming. The smallest overall error obtained using only 10 neurons in the hidden level equals 18.94 % for unigram word features. Adding bigrams did not bring any improvement, moreover it increases dimensionality of the input space, leading to overfitting. Log-sigmoid transfer function have shown superior performance over hyperbolic tangent sigmoid function. Adding the second hidden layer was not beneficial. Not only it increases complexity, but also leeds to poor generalization for test data samples, due to overfitting.

The criterion to stop the training of ANN was either a maximum number of epochs (set to 1 000) or a network performance (measured as average cross-entropy error) on the validation vectors that fails to improve for 10 epochs in a row. In practice, maximum number of epochs was rarely reached. For the best ANN configuration the training stopped after 297 epochs.

The best performing ANN configurations for unigram and a combination of unigram and bigram features were chosen for comparison with other machine learning techniques (see Table 3). We conclude that the best overall performance in terms of F-score equals 81.06 % and is obtained using the proposed ANN with unigram features. This is over 1 % higher compared to the second best algorithm – SVM. It is interesting to note that adding bigrams improved the scores only for MaxEnt algorithm, making it the best performing algorithm in that case.

We also compare learning (training) times of tested algorithms for STS corpus (Table 3). Proposed ANN requires significantly less time for learning compared to the second best performing SVM, but it is still an order of magnitude slower compared to MaxEnt. One of the problems affecting SVMs is a large set of support vectors that is usually needed to form their output function, making it complex and computationally expensive for real-time applications [36]. The support vectors are always a subset of the data in the SVM algorithm, whereas in ANN the network structure (the number of hidden layers and the number of neurons in the hidden layer) is fixed *a priori*, hence the complexity of ANN can be controlled by keeping the number of hidden nodes small. All three algorithms require more time to train in comparison to MNB, primarily due to the optimization problem that needs to be solved in order to estimate the parameters of the model. Nevertheless, MNB provides competitive results when the number of training samples is sufficiently large (as in STS corpus).

	Number of Neurons in the Hidden Layer					
One Hidden Layer	**10**	**20**	**50**	**100**	**200**	**# Features**
Unigram (tansig)	36.13 %	36.67 %	36.45 %	36.44 %	35.88 %	1 456
Unigram (logsig)	35.75 %	36.11 %	35.94 %	35.92 %	**35.61** %	1 456
Uni+Bigram (tansig)	35.61 %	35.53 %	35.84 %	35.94 %	35.65 %	3 179
Uni+Bigram (logsig)	**35.03** %	35.39 %	35.70 %	35.21 %	35.68 %	3 179
Two Hidden Layers	**10/10**	**20/20**	**50/50**	**100/100**	**200/200**	**# Features**
Unigram (tansig)	36.21 %	35.80 %	36.55 %	36.23 %	36.04 %	1 456
Uni+Bigram (tansig)	36.57 %	36.49 %	36.90 %	36.64 %	36.51 %	3 179

Table 4. Classification error using proposed ANN for various neural network architectures for SemEval 2014 task 9 corpus

		MNB	SVM	MaxEnt	ANN
	F-score	60.07 %	64.93 %	64.61 %	64.39 %
Unigram	Learning time	1.04 s	121.57 s	6.20 s	568.40 s
	F-score	61.23 %	65.40 %	**65.79** %	64.97 %
Uni+Bigram	Learning time	1.52 s	163.60 s	10.18 s	91.18 s

Table 5. Performance for SemEval 2014 task 9 corpus using proposed machine learning techniques

We tested the same machine learning algorithms for SemEval 2014 task 9 corpus using the unigram word features, and a combination of unigram and bigram features. Note that this training dataset is unbalanced, since the class distribution is not uniform among classes. Table 4 shows the classification error obtained using different number of neurons in the hidden layer ranging from 10 to 200. The smallest error obtained using ANN with logsig activation function and 200 neurons in the hidden level equals 35.61 % for unigram word features. Contrary to STS corpus, adding bigrams outperforms unigrams slightly, leading to the smallest obtained error of 35.03 % for 10 neurons in the hidden level, at the cost of increased number of features (3 179 features for bigram compared to 1 456 features for unigram case). As in STS corpus, adding the second hidden layer deteriorates performance, due to overfitting. Number of epochs needed to stop the training was 270 epochs for the best ANN configuration.

Comparison to standard machine learning techniques used for sentiment analysis is given in Table 5. The best performing ANN configurations for unigram and a combination of unigram and bigram features were chosen from Table 4. Proposed ANN significantly outperforms Multinomial Naive Bayes, however the F-scores are slightly lower compared to Maximum Entropy and Support Vector Machines for the unigram case. Adding the bigrams improves the results for all classifiers, leading to 65.79 % as the best overall F-score for SemEval dataset obtained using MaxEnt classifier. Proposed neural network outperforms MNB by approximately 3 % and has a performance comparable to SVM and MaxEnt.

Regarding the learning time, similar conclusions can be drawn as in the case of STS corpus. Note that learning time for ANN with unigram features is very high (568.4 s) since the best ANN configuration has 200 neurons in the hidden level; however almost the same performance can be obtained using only 10 neurons, with significatly lower computational complexity and training time reduced to 37.55 s. Adding the bigrams increases the number of features and leads to training time of an ANN with 10 neurons in the hidden level which is equal to 91.18 s, lower than SVM but still higher than MNB and MaxEnt.

6 CONCLUSIONS

A two-layer feed-forward artificial neural network is proposed for a task of sentiment analysis of Twitter messages, giving comparable results, or in some cases even outperforming state-of-the-art machine learning techniques, such as Support Vector Machines and Maximum Entropy. This proves that ANNs can be a promising tool for determining sentiment polarity from microblogs.

It is interesting to note that adding the bigrams to single word (unigram) features improves the performance only for SemEval 2014 dataset, while in the case of Stanford Twitter Sentiment corpus the results dropped for all the tested algorithms, except the Maximum Entropy. The reason might be the size of the lexicon; in smaller lexicons there are higher chances of discovering stable phrases constructed of bigrams that can contribute significantly to overall performance. On the other hand, adding bigrams can drastically increase dimensionality of the input space, leading to overfitting. Highly discriminative bigrams indeed do exist, but their number is much smaller compared to non-informative ones. Although they would be able to improve the classification, their contribution is weak in comparison to a contribution of unigram features. Selecting only the discriminative bigrams which are less likely to be noisy and will not increase the dimensionality significantly might lead to better effects of adding bigram or higher-order n-gram features.

In summary, our results indicate that it is not possible to make a straight choice of the best machine learning method for the task of sentiment analysis of microblogs. Indeed, the choice will depend on the underlying learning problem. ANNs have shown to be very robust to noisy input data, making them a good choice when manual annotations are not available (as in STS corpus). MaxEnt are, on the other hand, sensitive to noisy data, but perform better when training data are highly unbalanced (as in SemEval 2014 task 9 corpus), and probably are the best trade-of between performance and computational complexity. SVMs are competitive in terms of performance in all our experiments, but large number of support vectors makes them computationally slow in both training and test phase, which is especially important for real-time applications. MNB is a good candidate only in the presence of very large training datasets, due to computational efficiency.

REFERENCES

[1] Liu, B.: Sentiment Analysis and Opinion Mining. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers, 2012.

[2] Plaza, L.—Carillo de Albornoz, J.: Sentiment Analysis in Business Intelligence: A Survey. In: Colomo-Palacios, R., Varajão, J., Soto-Acosta, P. (Eds.): Customer Relationship Management and the Social and Semantic Web: Enabling Cliens Conexus. Chapter 14. IGI Global, Hershey, PA, 2012, pp. 231–252.

[3] Azevedo, A.—Santos, M. F.: Integration of Data Mining in Business Intelligence Systems. IGI Global, Hershey, PA, 2015, doi: 10.4018/978-1-4666-6477-7.

[4] Bakliwal, A.—Foster, J.—van der Puil, J.—O'Brien, R.—Tounsi, L.—Hughes, M.: Sentiment Analysis of Political Tweets: Towards an Accurate Classifier. NAACL Workshop on Language Analysis in Social Media, Atlanta, GA, 2013, pp. 49–58.

[5] Mejova, Y.—Srinivasan, P.—Boynton, B.: GOP Primary Season on Twitter: "Popular" Political Sentiment in Social Media. Proceedings of the Sixth ACM International Conference on Web Search and Data Mining, New York, NY, 2013, pp. 517–526, doi: 10.1145/2433396.2433463.

[6] Kale, A.—Karandikar, A.—Kolari, P.—Java, A.—Joshi, A.: Modeling Trust and Influence in the Blogosphere Using Link Polarity. Proceedings of the International Conference on Weblogs and Social Media (ICWSM 2007), Boulder, CO, 2007.

[7] Razavi, A. H.—Matwin, S.—De Koninck, J.—Amini, R. R.: Dream Sentiment Analysis Using Second Order Soft Co-Occurrences (SOSCO) and Time Course Representations. Journal of Intelligent Information Systems, Vol. 42, 2014, No. 3, pp. 393–413.

[8] Medhat, W.—Hassan, A.—Korashy, H.: Sentiment Analysis Algorithms and Applications: A Survey. Ain Shams Engineering Journal, Vol. 5, 2014, No. 4, pp. 1093–1113, doi: 10.1016/j.asej.2014.04.011.

[9] Koppel, M.—Schler, J.: The Importance of Neutral Examples for Learning Sentiment. Computational Intelligence, Vol. 22, 2006, No. 2, pp. 100–109, doi: 10.1111/j.1467-8640.2006.00276.x.

[10] Wilson, T.—Wiebe, J.—Hoffmann P.: Recognizing Contextual Polarity in Phrase-Level Sentiment Analysis. Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing, Vancouver, Canada, 2005, pp. 347–354, doi: 10.3115/1220575.1220619.

[11] Pandey, V.—Iyer, C.: Sentiment Analysis of Microblogs. Technical Report No. CS229, Stanford University, 2009.

[12] Esuli, A.—Sebastiani, F.: SentiWordNet: A Publicly Available Lexical Resource for Opinion Mining. Proceedings of the 5th Conference on Language Resources and Evaluation, Genoa, Italy, 2006, pp. 417–422.

[13] Thelwall, M.—Buckley, K.—Paltoglou, G.: Sentiment Strength Detection for the Social Web. Journal of American Society for Information Science and Technology, Vol. 63, 2012, No. 1, pp. 163–173, doi: 10.1002/asi.21662.

[14] Saif, H.—Fernandez, M.—He, Y.—Alani, H.: SentiCircles for Contextual and Conceptual Semantic Sentiment Analysis of Twitter. In: Presutti, V., d'Amato, C., Gandon, F., d'Aquin, M., Staab, S., Tordai, A. (Eds.): The Semantic Web: Trends and Challenges (ESWC 2014). Springer, Cham, Lecture Notes in Computer Science, Vol. 8465, 2014, pp. 83–98.

[15] Aue, A.—Gamon, M.: Customizing Sentiment Classifiers to New Domains: A Case Study. Proceedings of the International Conference on Recent Advances in Natural Language Processing, Borovets, Bulgaria, 2005.

[16] Go, A.—Bhayani, R.—Huang, L.: Twitter Sentiment Classification Using Distant Supervision. Technical Report No. CS224N, Stanford University, Stanford, CA, 2009.

[17] Pak, A.—Paroubek, P.: Twitter as a Corpus for Sentiment Analysis and Opinion Mining. In: Chair, N. C. C., Choukri, K., Maegaard, B., Mariani, J., Odijk, J., Piperidis, S., Rosner, M., Tapias, D. (Eds.): Proceedings of the 7th Conference on International Language Resources and Evaluation, Valletta, Malta, 2010, p. 19.

[18] Zhang, L.—Ghosh, R.—Dekhil, M.—Hsu, M.—Liu, B.: Combining Lexicon-Based and Learning-Based Methods for Twitter Sentiment Analysis. Technical Report No. HPL-2011-89, HP Labs, 2011.

[19] Martínez-Camara, E.—Martín-Valdivia, M.—Urena-Lopez, L.—Montejo-Raez, A.: Sentiment Analysis in Twitter. Natural Language Engineering, Vol. 20, 2014, pp. 1–28, doi: 10.1017/S1351324912000332.

[20] Ebert, S.—Vu, N. T.—Schutze, H.: CIS-Positive: Combining Convolutional Neural Networks and SVMs for Sentiment Analysis in Twitter. Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015), Denver, Colorado, 2015, pp. 527–532, doi: 10.18653/v1/S15-2088.

[21] Chen, L. S.—Liu, C. H.—Chiu, H. J.: A Neural Network Based Approach for Sentiment Classification in the Blogosphere. Journal of Informetrics, Vol. 5, 2011, pp. 313–322, doi: 10.1016/j.joi.2011.01.003.

[22] Cai, G.—Xia, B.: Convolutional Neural Networks for Multimedia Sentiment Analysis. In: Li, J., Ji, H., Zhao, D., Feng, Y. (Eds.): Natural Language Processing and Chinese Computing. Lecture Notes in Computer Science, Vol. 9362, 2015, pp. 159–167.

[23] Socher, R.—Pennington, J.—Huang, E. H.—Ng, A. Y.—Manning, C. D.: Semi-Supervised Recursive Autoencoders for Predicting Sentiment Distributions. Proceedings of the Conference on Empirical Methods in Natural Language Processing, Stroudsburg, PA, 2011, pp. 151–161.

[24] Dos Santos, C.—Gatti, M.: Deep Convolutional Neural Networks for Sentiment Analysis of Short Texts. Proceedings of the 25th International Conference on Computational Linguistics, Dublin, Ireland, 2014, pp. 69–78.

[25] Ghiassia, M.—Skinnerb, J.—Zimbraa D.: Twitter Brand Sentiment Analysis: A Hybrid System Using n-Gram Analysis and Dynamic Artificial Neural Network. Journal of Expert Systems with Applications, Vol. 40, 2013, No. 16, pp. 6266–6282.

[26] Moller, M. F.: Neural Networks. Vol. 6, 1993, pp. 525–533, doi: 10.1016/S0893-6080(05)80056-5.

[27] Nakov, P.—Kozareva, Z.—Ritter, A.—Rosenthal, S.—Stoyanov, V.—Wilson, T.: Semeval-2013 Task 2: Sentiment Analysis in Twitter. Proceedings of the

7th International Workshop on Semantic Evaluation, Association for Computational Linguistics, Atlanta, GA, 2013, pp. 312–320.

[28] Porter, M. F.: An Algorithm for Suffix Stripping. In: Jones, K. S., Willett, P. (Eds.): Readings in Information Retrieval. Morgan Kaufmann Publishers Inc., San Francisco, CA, 1997, pp. 313–316.

[29] Pohjalainen, J.—Rasanen, O.—Kadioglu, S.: Feature Selection Methods and Their Combinations in High-Dimensional Classification of Speaker Likability, Intelligibility and Personality Traits. Computer Speech and Language, Vol. 29, 2015, No. 1, pp. 145–171, doi: 10.1016/j.csl.2013.11.004.

[30] Milgram, J.—Cheriet, M.—Sabourin, R.: "One Against One" or "One Against All": Which One is Better for Handwriting Recognition with SVMs? Tenth International Workshop on Frontiers in Handwriting Recognition, La Baule, France, 2006.

[31] Chang, C. C.—Lin, C. J.: LIBSVM: A Library for Support Vector Machines. ACM Transactions on Intelligent Systems and Technology, Vol. 2, 2011, No. 3, Article No. 27.

[32] Weinman, J.—Lidaka, A.—Aggarwal, S.: Large-Scale Machine Learning. In: Hwu, W.-M. W. (Ed.): GPU Computing Gems Emerald Edition. Chapter 19. Morgan Kaufmann Publishers, Burlington, MA, 2011, pp. 277–291.

[33] Sokolova, M.—Japkowicz, N.—Szpakowicz, S.: Beyond Accuracy, F-Score and ROC: A Family of Discriminant Measures for Performance Evaluation. Advances in Artificial Intelligence, Lecture Notes in Computer Science, Vol. 4304, 2006, pp. 1015–1021, doi: 10.1007/11941439_114.

[34] Bross, J.: Aspect-Oriented Sentiment Analysis of Customer Reviews Using Distant Supervision Techniques. Ph.D. Thesis, Free University Berlin, Germany, 2013.

[35] Speriosu, M.—Sudan, N.—Upadhyay, S.—Baldridge, J.: Twitter Polarity Classification with Label Propagation over Lexical Links and the Follower Graph. Proceedings of the EMNLP First Workshop on Unsupervised Learning in NLP, Edinburgh, Scotland, 2011, pp. 53–63.

[36] Romero, E.—Toppo, D.: Comparing Support Vector Machines and Feedforward Neural Networks with Similar Hidden-Layer Weights. IEEE Transactions on Neural Networks, Vol. 18, 2007, pp. 959–963, doi: 10.1109/TNN.2007.891656.

Vladimir Despotovic received his Dipl. Ing. and Ph.D. degrees in electrical engineering from the University of Nis, Faculty of Electronic Engineering, Serbia, in 2003 and 2012, respectively. Currently he is working as Assistant Professor at the University of Belgrade, Technical Faculty in Bor. Previously he was engaged as postdoctoral researcher at the University of Paderborn, Department of Communications Engineering, Germany. His main research interests include statistical signal processing, machine learning, natural language processing, spoken language understanding and fractional calculus. He is a member of IEEE and ISCA (International Speech Communication Association). He was a recipient of Erasmus Mundus Action 2 postdoctoral fellowship, OeAD scholarship for a research carried out at the Vienna University of Technology, Vienna, Austria, and SAIA scholarship for a research carried out at the Technical University of Kosice, Kosice, Slovakia.

Dejan Tanikic received his B.Sc., M.Sc. and Ph.D. degrees in mechanical engineering in 1998, 2004 and 2009, respectively, from the University of Nis, Faculty of Mechanical Engineering, Serbia. He works as Associate Professor at the University of Belgrade, Technical Faculty in Bor. His main research interests include soft computing systems (neural networks, fuzzy logic, genetic algorithms, etc.) and their application in various engineering fields. He serves as Vice Dean of the Technical Faculty in Bor since 2017, and Head of Department of Electromechanical Engineering since 2012.