RED: A Novel Dataset for Romanian Emotion Detection from Tweets

Alexandra Ciobotaru
Advanced Technologies Institute
Bucharest, Romania
alexandra.ciobotaru@dcti.ro

Liviu P. Dinu
University of Bucharest
Bucharest, Romania
liviu.p.dinu@gmail.com

Abstract

In Romanian language there are some resources for automatic text comprehension, but for Emotion Detection, not lexicon-based, there are none. To cover this gap, we extracted data from Twitter and created the first dataset containing tweets annotated with five types of emotions: joy, fear, sadness, anger and neutral, with the intent of being used for opinion mining and analysis tasks. In this article we present some features of our novel dataset, and create a benchmark to achieve the first supervised machine learning model for automatic Emotion Detection in Romanian short texts. We investigate the performance of four classical machine learning models: Multinomial Naive Bayes, Logistic Regression, Support Vector Classification and Linear Support Vector Classification. We also investigate more modern approaches like fastText, which makes use of subword information. Lastly, we fine-tune the Romanian BERT for text classification and our experiments show that the BERT-based model has the best performance for the task of Emotion Detection from Romanian tweets.

Keywords: Emotion Detection, Twitter, Romanian, Supervised Machine Learning

1 Introduction

Romanian language is a very little explored language in terms of Natural Language Processing (NLP) and Natural Language Understanding (NLU), but social media is full of data and people sharing their opinions about diverse topics, thus some opinion mining tools would be very useful.

Opinion mining is defined in Ravi (2015) as "the task of detecting, extracting and classifying opinions, sentiments and attitudes concerning different topics, as expressed in textual input". In NLP tasks, there are two major subcategories of opinion mining: Sentiment Analysis (SA) and Emotion Detection (ED). The main difference between these two is that sentiment analysis categorizes opinions between positive or negative, while emotion detection aims to extract the specific emotion a text gives to the reader.

English language is rich in NLP resources - we can see not only that there are many manually labelled datasets, with a large variety of emotions, but there also exist datasets with tweets labeled by emotion intensity (for example, WASSA dataset by Mohammad and Bravo-Marquez (2017) which was first presented at the 2017 Shared Task on Emotion Intensity). Detecting emotions can be a subjective task, even psychologists didn't agree upon what the main emotions of a human being are. Plutchik (1973) identified eight primary emotions: ecstasy, admiration, terror, amazement, grief, loathing, rage and vigilance, while Ekman (2005) said that the primary emotions are: joy, sadness, fear, disgust, surprise and anger. Inspired by the work of Mohammad and Bravo-Marquez (2017), we decided to collect tweets from Twitter and manually annotate them for four emotions: fear, anger, joy, sadness, and also add a neutral class. We show the quality of the overall dataset by computing statistic descriptors and finally we train machine learning classifiers in order to create the first qualitative Romanian tool for emotion detection.

The usefulness of having a tool that detects emotions from texts varies from exploring customer opinions in order to ensure business growth, to improving human-computer interactions (HCI) in applications like onboarding chatbots and personal digital assistants (Strapparava and Mihalcea (2008)).

2 Recent Works

It has been shown by Acheampong et al. (2020) that text based ED research studies have been given less

Proceedings of Recent Advances in Natural Language Processing, pages 291–300
Sep 1–3, 2021.
https://doi.org/10.26615/978-954-452-072-4_034

attention than other methods of ED, for instance multimodal emotion detection including speech, body language, facial expressions, and so on.

There are two principal ways of assessing the ED problem: by using rule construction techniques to identify emotive words and word combinations, and machine learning approaches, which can be supervised or unsupervised. While unsupervised methods learn from unstructured data and could provide important clustering insights in opinion mining tasks, supervised methods give better detection rates (Canales and Martinez-Barco (2014)), but only if there exist some quality labeled data to give as input to the classifier. Combining these two main approaches gives a third way of detecting emotions from text - the hybrid approach, which takes the advantages of rule-construction and amplifies them with the power of machine learning. The latter was used in this paper.

Alotaibi (2019) used the ISEAR dataset to train a supervised model of emotion detection with Logistic Regression. The ISEAR (International Survey on Emotion Antecedents and Reactions) dataset was first introduced in the work of Scherer and Wallbotth (1994) and contains 7666 emotional sentences labelled with 7 types of emotions: anger, disgust, fear, sadness, shame, joy and guilt. Abdel Razek and Frasson (2017) used the same dataset to test their dominant meaning approach in detecting emotions in chat messages.

For detecting emotions in tweets, Shah et al. (2019) proposed a hybrid approach consisting of lexical based approaches that use WordNet-Affect and EmoSenticNet with supervised classifiers trained on AIT-2018 dataset for English. This dataset was introduced in Semeval-2018 Task 1: Affect in Tweets by Mohammad et al. (2018) and consists of tweets annotated in 3 languages, for anger, fear, joy and sadness.

Ghanbari-Adiv and Mosleh (2019) presented an ensemble classifier based on NLP techniques like Doc2Vec and 1500 k-Nearest Neighbor, Multilayer Perceptron and Decision Tree basic classifiers optimized with Parzen Tree Estimator (TPE), to detect emotions from ISEAR and OANC datasets, and also from an unstructured dataset of tweets from Crowdflower. Their results show an outstanding accuracy of 99.49% for regular sentences and 88.49% for irregular sentences.

Polignano et al. (2019) tried emotion detection in text using word-embeddings like Word2Vec, GloVe and FastText on their designed model, having as training datasets ISEAR, AIT-2018 and SemEval-2019 Task3 dataset. Their research concluded that FastText had slightly better performances.

Huang et al. (2019) investigated fine-tuning BERT (Bidirectional Encoder Representations from Transformers) on two datasets: EmotionLines with dialogues from Friends Sitcom, and EmotionPush containing Facebook messenger chats. Their model obtained a micro f-score of 0.815 on Friends and 0.885 on EmotionPush. Acheampong et al. (2020) fine-tuned BERT with a Bi-LSTM classifier and obtain an average f-score of 0.73 on the ISEAR dataset. Chiorrini et al. (2021) also investigated the use of BERT for both sentiment analysis and emotion detection in Twitter data, using WASSA dataset (Mohammad and Bravo-Marquez (2017)), a shorter version of AIT-2018 containing 6755 tweets annotated for: sadness, fear, anger and happiness.

For more exotic languages, however, researchers had to manually annotate sentences or words in order to have a high quality dataset to work on. For Arabic, Almanie et al. (2018) developed a dataset of 4000 emotional words, including emojis, which they used to classify real-time tweets into 5 types of emotions (happy, sad, angry, scared, surprised). Grover and Verma (2016) used a hybrid approach to detect emotions from Punjabi texts - first they consider a rule-based engine to detect if the sentence has an emotion or not, and then they applied Support Vector Machine (SVM) and Naïve Bayes (NB) classifiers to detect 6 emotions: happy, fear, anger, sadness, disgust and surprise. Jayakrishnan et al. (2018) created a corpus of manually labelled Malayalam texts (an Indian dialect) into emotions like: sad, happy, anger or fear. Further they used a SVM classifier to create emotion detection models.

For Romanian language, there were some studies conducted on speech datasets with annotated emotions. For instance, Feraru and Zbancioc (2015) presented a method for emotion detection in Romanian speech that use Largest Lyapunov exponent of the Mel-frequency energy bands, with SVM and WKNN (Weighted K-Nearest Neighbors) classifiers, trained on the SRoL dataset (developed by Feraru et al. (2010)). Franți et al. (2017) created a deep learning model of Convolutional Neural Networks (CNN) trained on a set of recordings in Romanian language. Pavaloi et al. (2014) used three sets of recordings in Romanian language, annotated

for positive and negative emotions, and trained models using k-NN and SVM classifiers. In terms of text data, Lupea et al. (2021) present an unsupervised clustering approach used to mine emotional patterns in Mihai Eminescu's poetry, based on the Romanian Emotion Lexicon created by Lupea and Briciu (2019) for feature extraction. In terms of sentiment analysis for Romanian language, Istrati and Ciobotaru (2021) created a dataset of tweets annotated for positive/negative sentiment and trained several classifiers on it, both classical and modern. There also exist some lexicon based approaches for Romanian sentiment analysis, like BabelSenticNet, a multilingual concept-level knowledge base described in Vilares et al. (2018).

Inspired by some of the works presented in this section, we created the first dataset of Romanian tweets labelled with five different emotions (anger, fear, sadness, hapiness and neutral), in order to obtain a solid tool for detecting emotions in texts.

3 Data

In this section we present in detail our novel dataset, RED (Romanian Emotion Detection).

3.1 Scrapping Process

We considered the work of Mohammad and Bravo-Marquez (2017) where they explained their methodology of creating the first annotated datasets for fear, anger, joy and sadness, and we create a similar dataset, but for the Romanian language. In our work we construct the same four classes of emotions and also add a neutral class, as it has been previously shown by Al-Rubaiee et al. (2016) the importance of having a neutral class when classifying sentiments or emotions. This way, the classifier will not be forced to classify information he wasn't trained to recognize, in one of the four classes created.

First, we create lists of *query words* correspondent to each of the class of emotions, which are:

- synonyms of the word defining the class - synonyms for "fear", "anger", "joy", "sadness" and "neutral", extracted from two synonym sources: an online dictionary[1] and the work of Bulgar (1995), and
- jargon and commonly used words that express a certain feeling matching the class of emotion for which the scraping process is conducted.

[1] https://www.sin0nime.com/dex

The total number of query words gathered for each class is detailed in Table 1. These main words are further expanded into their word families by adding prefixes and suffixes to their stem-words. Moreover, all morphological variants are generated for each resulted word. In this way, for each query-word a list or words is generated, and the resulted lists of words were further used for scraping tweets, using Snscrape[2] python library.

Tweets were gathered from Twitter in the time-frame: 1st of February 2020 - 1st of February 2021, and checked for Romanian language using langdetect[3] python library.

All gathered tweets were scrapped from public accounts. Still, to protect confidentiality and anonymity of Twitter users, we removed usernames and also all proper nouns from tweets in the final dataset, using preprocessing techniques described in Section 3.3.

3.2 Annotation Process

The annotation process involved 3 annotators: Annotator 1 and Annotator 2, native Romanian speakers who decided over the same tweets, and Annotator 3, a psychologist, also native Romanian speaker, who tipped the scales regarding the tweets where Annotator 1 and Annotator 2 did not agree upon.

The result of the scraping process for each query word was a spreadsheet containing all the tweets found in the mentioned time-frame, having in their composition at least one word expanded from the query word. These spreadsheets were shuffled and manually checked by Annotator 1 if tweets indeed represented the emotion conveyed by the class they were scrapped for. Maximum 50 tweets were kept for each query word, in order not to bias the classification process. An important rule for annotating was to annotate tweets that clearly expressed the researched emotion.

After the work of Annotator 1 was done, a number of approximately 1000 tweets resulted per class. In order the create a high quality dataset, these tweets were double-checked by Annotator 2 in order to make sure that the tweets indeed represent the emotion labelled by the first annotator. The two annotators disagreed upon 223 tweets for Anger, 251 for Fear, 309 for Joy, 279 for Sadness and 210 for Neutral class. These selected tweets were

[2] https://github.com/JustAnotherArchivist/snscrape

[3] https://pypi.org/project/langdetect/ version 1.0.8

checked by Annotator 3, who gave his verdict regarding their conveyed emotion. Further, tweets were checked for duplicates and shuffled. The final number of remaining tweets is shown in Table 1. It can be observed that classes remain balanced after the quality check involved in the annotation process.

Table 1: Number of query words and final number of labelled tweets per class

Class name	Query words	Labelled tweets
Anger	35	807
Fear	25	778
Joy	32	876
Sadness	29	781
Neutral	24	805

3.3 Dataset Preprocessing

In order to have a high quality dataset, suitable for machine training, we performed some text preprocessing by removing unnecessary information from tweets that could potentially bias the classification:

- usernames in the form of @username;
- hiperlinks;
- hashtag sign (#), but the hashtag word was preserved, as it can contain relevant information for the classification problem;
- artefacts like &amp and \n;
- proper nouns, using Named Entity Recognition pipeline for Romanian from spacy[4] (ro_core_news_sm).

The first four preprocessing steps were performed using regex. Emoticons, emojis, as well as all punctuation marks were left untouched by preprocessing techniques, as they convey emotions per se.

Final dataset was created by gathering all labelled tweets and shuffling them all. Further, the dataset was split into 3237 tweets for training, 405 tweets for validation and 405 tweets for testing, and this split was used for training all classifiers presented in Section 4.

3.4 Dataset Analysis

In Figure 1 we show the distribution of tokens per tweets for each class of emotions, in order to make sure all five classes have approximately the same distribution. As it can be seen, Fear, Joy and Neutral have approximately the same distributions, while Anger differs the most.

In Table 2 we compute descriptive statistics using R function aggregate. It can be seen that the longest tweet pertains to the Joy class, with 75 tokens, while the majority of tweets lie under 35 tokens, as the highest median is of 30 tokens, belonging to the Anger class. For this class it can also be observed that it has the highest mean an median, which could mean that Twitter users express anger using longer phrases in Romanian. Lowest mean and median are observed in Sadness and Neutral classes, which could mean that Twitter users express sadness and neutral feelings using less words in Romanian.

Table 2: Descriptive statistics of token distributions in tweets.

Emotion	Min	Median	Mean	Max
Anger	2.00	30.00	31.97	68.00
Fear	2.00	21.00	24.18	70.00
Joy	2.00	21.00	25.27	75.00
Neutral	1.00	11.00	21.34	67.00
Sadness	1.00	16.00	28.36	74.00

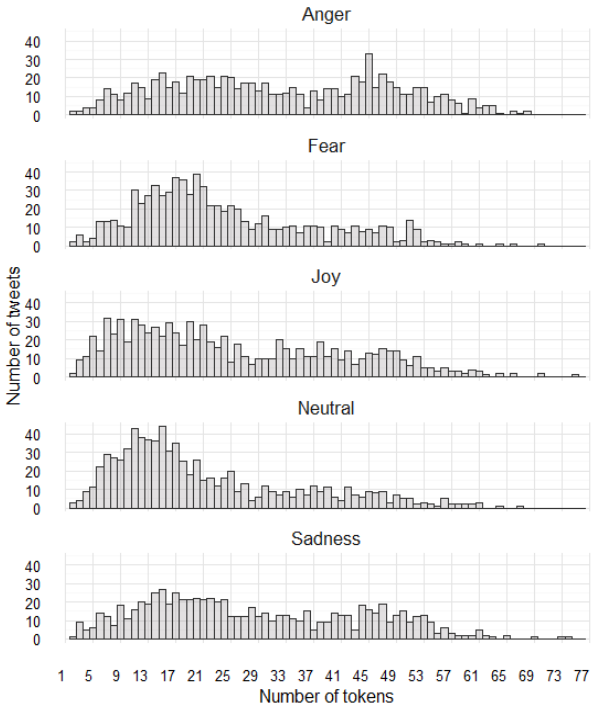

Figure 1: Distribution of tokens per tweet for each class of emotions.

4 Models

In this section we describe the tried models in order to generate the first supervised emotion detector for Romanian short texts. First, we create a benchmark

[4] https://spacy.io/models/ro

composed of classical machine learning models, and second we try two more modern approaches, which make use of word embeddings: one that uses Facebook's fastText word embeddings and a fine-tuned BERT classifier.

4.1 Classical Machine Learning Models

Classical machine learning models can sometimes have good results for modern problems. Thus, before diving into more modern solutions we first investigate Support Vector Classification (SVC), Linear Support Vector Classification (LinearSVC), Logistic Regression and Multinomial Naive Bayes (MultinomialNB) algorithms.

For feature extraction we use Term Frequency - Inverse Document Frequency (*tf-idf*) to represent tweets in vector form, which is a measure that tries to estimate the importance of tokens in the dataset by computing two statistics: *term frequency* (the number of appearances of a word in the whole dataset - see Eq.1) and *inverse document frequency* (the number of tweets in relation to the number of tweets containing the word - see Eq.2), as explained in Sammut (2010).

$$tf = \frac{\text{No. Of Word Appear. in Tweet}}{\text{No. of Words in Tweet}} \quad (1)$$

$$idf = \frac{\text{No. Of Tweets}}{\text{No. of Tweets with Word}} \quad (2)$$

The final result of the *tf-idf* is obtained by multiplying Eq.1 and Eq.2 (AlZoubi et al. (2020)):

$$tf - idf = tf \times idf \quad (3)$$

Practically, we convert tweets into a matrix of *tf-idf* features using TfidfVectorizer[5], with the following characteristics:

- sublinear_df parameter is set to True in order to use a logarithmic form for term frequency, because it seems unlikely that twenty occurrences of a term in a document truly carry twenty times the significance of a single occurrence (Manning Christopher (2008)). Thus, Eq. 3 becomes:

$$wf - idf_{t,d} = wf_{t,d} \times idf_t \quad (4)$$

  where:

$$wf_{t,d} = \begin{cases} i + log tf_{t,d} & if\ tf_{t,d} > 0 \\ 0 & otherwise \end{cases} \quad (5)$$

- *min_df* parameter is set to 5, being the minimum number of tweets a word must be present in to be kept in the feature vector;
- ngram_range parameter is set to (1, 2) to indicate that we want to consider both unigrams and bigrams in our vector representation.

For label encoding we used Sklearn LabelEncoder[6].

4.2 FastText Based Model

FastText[7] is an open-source library, developed by Facebook AI Research lab with the purpose of text classification and representation. As Bojanowski et al. (2016) described in their work, fastText creates word representations based on the skipgram model, where each word is represented as a bag of character n-grams. A vector representation is associated to each character n-gram, words being represented as the sum of these representations. Using character level information helps capture the meaning of shorter words and allows the embeddings to map suffixes and prefixes.

The character n-gram selection is done using a sliding window between the minimum value of the character n-gram and the maximum value of the character n-gram. The word is stored in memory like the sum of character n-grams. For classification, word representations are averaged into a text representation to form a hidden variable, which is in turn fed to a linear classifier (Joulin et al. (2016)).

4.3 BERT Based Model

Another modern approach for text encoding is using pretrained vector representations. BERT (Bidirectional Encoder Representations from Transformers), introduced by Devlin et al. (2019) in their work, is considered state-of-the-art in many natural language processing tasks that use language representation. We fine-tune BERT model in order to obtain an emotion detection classifier.

BERT-base model contains an encoder with 12 Transformer blocks, 12 self-attention heads, and the hidden size of 768 (Chi et al. (2019)). To use a pre-trained BERT model, we first need to convert the input data into an appropriate format so that each sentence can be sent to the pre-trained model in order to obtain the corresponding embedding.

[5] https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html

[6] https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.LabelEncoder.html

[7] https://fasttext.cc/

For this task we use HuggingFace's transformers package, and in particular the tokenizer for Romanian and the BERT pretrained model for Romanian[8] described by Dumitrescu et al. (2020). The Romanian model was trained on three Romanian corpuses: OPUS, OSCAR and Wikipedia.

BERT tokenizer adds special tokens to the input text, converts all tokens into their corresponding IDs in the model, and adds the attention mask. The attention mask is a vector of 1 and 0 which tells the model which tokens should be taken into consideration and which should not. The resulted vectors are used to train the model.

5 Experiments and Results

In this section we present the results obtained on the test set for the previously described models.

5.1 Classical Machine Learning

The investigated algorithms are SVC, LinearSVC, Logistic Regression and Multinomial NB. For training we use sklearn off-the-shelf functions, without any other parameter modification. The accuracies we obtained for each model are shown in Table 3.

Table 3: Comparison between four classical machine learning models

Model	Accuracy
LinearSVC	82.96%
LogisticRegression	78.77%
MultinomialNB	76.79%
SVC	76.30%

According to the results, LinearSVC has the best accuracy, 82.96%, second comes Logistic Regression with an accuracy of 78.77%, third MultinomialNB with 76.79% accuracy and fourth SVC with an accuracy of 76.30%. For LinearSVC, the best among the classical models analyzed, we compute the confusion matrix and the normalised confusion matrix and show our results in Appendix, Figure 2. Normalization of the confusion matrix is useful in the case of not such perfectly balanced classes, to be able to visual interpret which class is being misclassified the most. It can be seen in the normalized confusion matrix in Appendix, Figure 2, that Sadness class is being missclassified most frequently, and the Neutral class is being classified correctly most often.

We also analyze precision, recall and f-score for each class, which are computed in Table 4, using classification_report[9] method. In accordance with the confusion matrix, Neutral class has the highest precison, recall and F-score scores (not surprisingly, these values are also equal), while Sadness class has lowest scores.

Table 4: Classification report for Linear SVC

Emotion	Precision	Recall	F-score	Support
Joy	0.83	0.78	0.80	89
Fear	0.79	0.85	0.82	72
Anger	0.85	0.84	0.85	83
Neutral	0.89	0.89	0.89	81
Sadness	0.78	0.80	0.79	80
Macro avg	0.82931	0.83095	0.82972	405

Combining the results from Appendix, Figure 2 and Table 4 we see that out of 80 tweets labelled with Sadness, the model correctly classified only 64, 8 being classified as Joy, and out of 89 tweets labelled with Joy, the model correctly classified only 69, 7 being classified as Sadness. This means that the model confuses Joy with Sadness and vice-versa, which is counter-intuitive.

5.2 FastText Based Model

To train the fastText based model we used the train_supervised method from fasttext library, with the word n-grams parameter set to 2, because for the classical machine learning models considered previously we have also used word n-grams of 2 during encoding. We conducted a set of four experiments for the fastText based model, two experiments using pretrained vectors for the Romanian language, and another two experiments where we let the model autotune on the validation set, without taking into consideration pretrained word vectors. Results are shown in Table 5, along with the hyperparameters used for each model, which are:

- *lr* - learning rate;
- *vectors* - fastText pretrained vectors for Romanian were trained on Common Crawl and Wikipedia using CBOW with position-weights, in dimension 300, with character n-grams of length 5, a window of size 5 and 10 negatives[10];
- *ws* - size of the context window;
- *wNgms* - maximal length of word n-gram;
- *epoch* - number of epochs used for training;

[8] https://huggingface.co/dumitrescustefan/bert-base-romanian-cased-v1

[9] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.classification_report.html

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

- *tune* - time set for autotuning on the validation file;
- *minn* - minimum length of character n-gram;
- *maxn* - maximum length of character n-gram.

As seen in Table 5, models trained without making use of pretrained vectors for Romanian had better performances than the models trained with these vectors, but this can be due to the fact that experiments without pretrained vectors were done using hyperparameter autotuning on the validation set, and might not be necessarily related to using pretrained vectors or not. At the time these experiments were conducted, train_supervised method from fastText library didn't allow autotuning of hyperparameters using pretrained vectors.

Table 5: Hyperparameters and performance of tried fastText models

Model	lr	vectors	ws	wNgms	epoch	minn	maxn	tune	Acc.
Model1	0.001	yes	2	2	1000	-	-	-	0.733
Model2	0.001	yes	2	2	1000	2	3	-	0.795
Model3	-	no	-	2	-	1	3	600s	0.795
Model4	-	no	-	2	-	-	-	600s	0.847

For Model 4 we compute the confusion matrix (Appendix, Figure 3), both classic and normalized. We can see from the normalized confusion matrix that the model has an overall best performance on the Anger class, and worst performance on the Neutral class. But if we examine performance in detail, we see in the classification report (Table 6) that although Anger class has the highest recall, it lacks precision, and while the Neutral class has high precision, it lacks recall. Also, analyzing the confusion matrix we observe many high values of missclassification, for each class: from 89 tweets labelled Joy the model classified 6 as Sadness; from 72 tweets labelled Fear, the model classified 7 as Anger; from 83 tweets labelled as Anger, the model classified 6 as Sadness; from 81 tweets labelled Neutral the model classified 8 as Joy, and from 80 tweets labelled Sadness the model classified 6 as Anger. We can conclude that the fastText based model may have an overall good performance taking into account all classes at once, but behaves poorly in discrete mode.

5.3 BERT Based Model

To train the BERT model we preprocessed data as explained in Section 4.3, using the tokenizer.encode_plus method from Huggingface. We fine-tuned BERT model by adding a classifier for the task of emotion detection, comprised of:

Table 6: Classification report for fastText Model 4

Emotion	Precision	Recall	F-score	Support
Joy	0.89	0.85	0.87	89
Fear	0.88	0.85	0.87	72
Anger	0.78	0.88	0.82	83
Neutral	0.90	0.79	0.84	81
Sadness	0.80	0.86	0.83	80
Macro avg	0.85170	0.84666	0.84742	405

- a drop-out layer with probability 0.3. As explained in Hinton et al. (2012), drop-out layers are used for regularization and preventing the co-adaptation of neurons;
- a fully connected layer that applies a linear transformation to data;
- a transformation of the output using *Softmax* function.

Training was done using Cross Entropy loss function with an AdamW optimizer for the learning rate. Although BERT authors have some recommendations in their paper, Devlin et al. (2019), we opted for a batch size of 8 and 5 epochs, because our 16GB GPU card couldn't handle a bigger batch size, and in 5 epochs the model already reached its highest accuracy. The maximum utterance length was set to 100.

For fine-tuning BERT we chose Cross Entropy loss function and Softmax activation function, as these are usually used with multiclass clasification problems.

The confusion matrices, both simple and normalized, are shown in Appendix, Figure 4. It can be observed on the normalized confusion matrix that Sadness class has been missclassified most often, while Joy and Fear have best accuracies per class. On the simple confusion matrix we can see that missclassifications don't have such high values like on the other tried models' confusion matrices, none exceeding 5 wrong classifications.

The overall accuracy for the BERT based model is 90.37%, fact that can also be observed in the confusion matrix, by summing up the values on the diagonal and dividing to the whole number of samples (405 tweets in the test set).

This model's classification report is presented in Table 7. The macro averaged precision, recall and f-score are very similar, and to the overall accuracy of the model as well. Best precision, recall and F-score belong to the Joy class, worst precison and f-score to Anger class, and worst recall to Sadness class.

Table 7: Classification report for BERT based model

Emotion	Precision	Recall	F-score	Support
Joy	0.95	0.93	0.94	89
Fear	0.93	0.89	0.91	72
Anger	0.85	0.89	0.87	83
Neutral	0.87	0.91	0.89	81
Sadness	0.90	0.86	0.88	80
Macro avg	0.90437	0.90354	0.90366	405

5.4 Comparisons

Retrospectively, although LinearSVC had a high overall accuracy, it confused Joy and Sadness, but had good performance on the Neutral class. On the other hand, the fastText based model had worst performances on the Neutral class, and performed best when classifying Anger, but had an overall better accuracy than LinearSVC. Lastly, our BERT-based model outperforms all the other models considered, as seen in Table 8, where we aggregate all our results and compare models using accuracy, macro averaged precision, recall and F-score. We can see that the BERT-based model came out best regarding all measures, fastText based model came out second, and classical LinearSVC on the last place. A probable explanation for such good results is that pre-trained BERT learned contextual relations between words and fine-tuning the model makes use of these relations when classifying.

Table 8: Comparison between created ED models

Model	Accuracy	Precision	Recall	F-score
BERT	90.37%	90.44%	90.35%	90.37%
fastText	84.70%	85.17%	84.67%	84.74%
LinearSVC	82.96%	82.93%	83.10%	82.97%

6 Conclusions and Future Works

In this article, we presented our novel dataset for emotion detection, the first dataset of its kind for the Romanian language. We researched the state-of-the-art and created a benchmark of machine learning models in order to obtain an automatic emotion detector from tweets, having the purpose of being used in real-life tasks, adjacent to the field of opinion mining.

Although the dataset is not very large, it provided enough data to generate a text emotion detection model with an accuracy of 90.37%. In the future, we plan to enlarge this dataset with more tweets per class, an action that most probably will increase accuracy of the models. Also, we plan on adding more classes of emotions, to generate an even more fine-grained dataset for detecting emotions in Romanian content.

7 Acknowledgments

We would like to thank Nicu Ciobotaru and Ioana Alexandra Răducanu for their help with the annotation process, Ligia Maria Bătrînca for proof reading and suggestions, as well as the anonymous reviewers for their time and valuable comments.

We acknowledge the support of a grant of the Romanian Ministry of Education and Research, CCCDI—UEFISCDI, project number 411PED/2020, code PN-III-P2-2.1-PED-2019-2271, within PNCDI III.

References

Mohammed Abdel Razek and Claude Frasson. 2017. Text-based intelligent learning emotion system. *Journal of Intelligent Learning Systems and Applications*, 09:17–20.

Francisca Acheampong, Chen Wenyu, and Henry Nunoo-Mensah. 2020. Text-based emotion detection: Advances, challenges and opportunities.

Hamed Al-Rubaiee, Renxi Qiu, and Dayou Li. 2016. The importance of neutral class in sentiment analysis of arabic tweets. *International Journal of Computer Science and Information Technology*, 8:17–31.

Tahani Almanie, Alanoud Aldayel, Ghaida Alkanhal, Lama Alesmail, Manal Almutlaq, and Ruba Althunayan. 2018. Saudi mood: A real-time informative tool for visualizing emotions in saudi arabia using twitter. pages 1–6.

Fahad Alotaibi. 2019. Classifying text-based emotions using logistic regression. *VAWKUM Transactions on Computer Sciences*, pages 31–37.

Omar AlZoubi, Saja Al-Tawalbeh, and Mohammad AL-Smadi. 2020. Affect detection from arabic tweets using ensemble and deep learning techniques. *Journal of King Saud University - Computer and Information Sciences*.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5.

Gheorghe Bulgar. 1995. *Dicționar de sinonime*. Palmyra.

Lea Canales and Patricio Martinez-Barco. 2014. Emotion detection from text: A survey. pages 37–43.

Sun Chi, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019. *How to Fine-Tune BERT for Text Classification?*, pages 194–206.

Andrea Chiorrini, Claudia Diamantini, Alex Mircoli, and Domenico Potena. 2021. Emotion and sentiment analysis of tweets using BERT. In *Proceedings of the EDBT/ICDT 2021 Joint Conference*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *Association for Computational Linguistics*, Proceedings of NAACL-HLT:4171–4186.

Stefan Dumitrescu, Andrei-Marius Avram, and Sampo Pyysalo. 2020. The birth of Romanian BERT. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4324–4328. Association for Computational Linguistics.

Paul Ekman. 2005. *Basic Emotions*, volume 99, pages 45 – 60.

Monica Feraru and Marius Zbancioc. 2015. Emotion recognition using lyapunov exponent of the mel-frequency energy bands. *Proceedings of the 2014 6th International Conference on Electronics, Computers and Artificial Intelligence, ECAI 2014*, pages 19–22.

S. M. Feraru, H. Teodorescu, and M. Zbancioc. 2010. SRoL - Web-based Resources for Languages and Language Technology e-Learning. *Int. J. Comput. Commun. Control*, 5:301–313.

Eduard Franți, Ioan Ispas, Voichita Dragomir, Monica Dascălu, Elteto Zolta, and Ioan Cristian Stoica. 2017. Voice based emotion recognition with convolutional neural networks for companion robots. *Romanian Journal of Information Science and Technology*, 20(3):222–240.

Fereshteh Ghanbari-Adiv and Mohammad Mosleh. 2019. Text emotion detection in social networks using a novel ensemble classifier based on parzen tree estimator (tpe). *Neural Computing and Applications*, 31.

Sheeba Grover and Amandeep Verma. 2016. Design for emotion detection of punjabi text using hybrid approach. In *2016 International Conference on Inventive Computation Technologies (ICICT)*, volume 2, pages 1–6.

Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2012. Improving neural networks by preventing co-adaptation of feature detectors. *CoRR*, abs/1207.0580.

Yen-Hao Huang, Ssu-Rui Lee, Mau-Yun Ma, Yi-Hsin Chen, Ya-Wen Yu, and Yi-Shin Chen. 2019. EmotionX-IDEA: Emotion BERT - an Affectional Model for Conversation. *CoRR*, abs/1908.06264.

Lucian Istrati and Alexandra Ciobotaru. 2021. Automatic monitoring and analysis of brands using data extracted from Twitter in Romanian. *Springer: Lecture Notes in Networks and Systems (to appear)*.

R Jayakrishnan, Greeshma Gopal, and M Santhikrishna. 2018. Multi-class emotion detection and annotation in malayalam novels. pages 1–5.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*.

Mihaiela Lupea and Anamaria Briciu. 2019. Studying emotions in romanian words using formal concept analysis. *Computer Speech Language*, 57:128–145.

Mihaiela Lupea, Anamaria Briciu, and Elena Bostenaru. 2021. Emotion-based hierarchical clustering of romanian poetry. *Studies in Informatics and Control*, 30(1):109–118.

Hinrich Schütze Manning Christopher, Prabhakar Raghavan. 2008. *Introduction to information retrieval*, volume 12. Cambridge University Press, Cambridge, England.

Saif Mohammad and Felipe Bravo-Marquez. 2017. Emotion intensities in tweets. pages 65–77.

Saif Mohammad, Felipe Bravo-Marquez, Mohammad Salameh, and Svetlana Kiritchenko. 2018. SemEval-2018 task 1: Affect in tweets. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 1–17, New Orleans, Louisiana. Association for Computational Linguistics.

Ioan Pavaloi, Adrian Ciobanu, Mihaela Luca, Elena Musca, Anca Ignat, and Tudor Barbu. 2014. A study on positive and negative emotion automatic recognition in speech.

Robert Plutchik. 1973. The nature of emotions. pages 810–817.

Marco Polignano, Pierpaolo Basile, Marco de Gemmis, and Giovanni Semeraro. 2019. A comparison of word-embeddings in emotion detection from text using bilstm, cnn and self-attention. pages 63–68.

Kumar Ravi. 2015. A survey on opinion mining and sentiment analysis: Tasks, approaches and applications. *Knowledge-Based Systems*, 89:14–46.

Webb Geoffrey I. (Eds.) Sammut, Claude. 2010. *Encyclopedia of Machine Learning*. Springer, US, Boston, MA.

K.R. Scherer and G. Wallbotth. 1994. Evidence for universality and cultural variation of differential emotion response patterning. *Journal of Personality and Social Psychology*, 66:310–328.

Faisal Shah, Abdus Reyadh, Asif Shaafi, Sifat Ahmed, and Fatima Sithil. 2019. Emotion detection from tweets using ait-2018 dataset.

Carlo Strapparava and Rada Mihalcea. 2008. Learning to identify emotions in text. pages 1556–1560. ACM.

David Vilares, Haiyun Peng, Ranjan Satapathy, and Erik Cambria. 2018. Babelsenticnet: A commonsense reasoning framework for multilingual sentiment analysis. pages 1292–1298.

Appendix

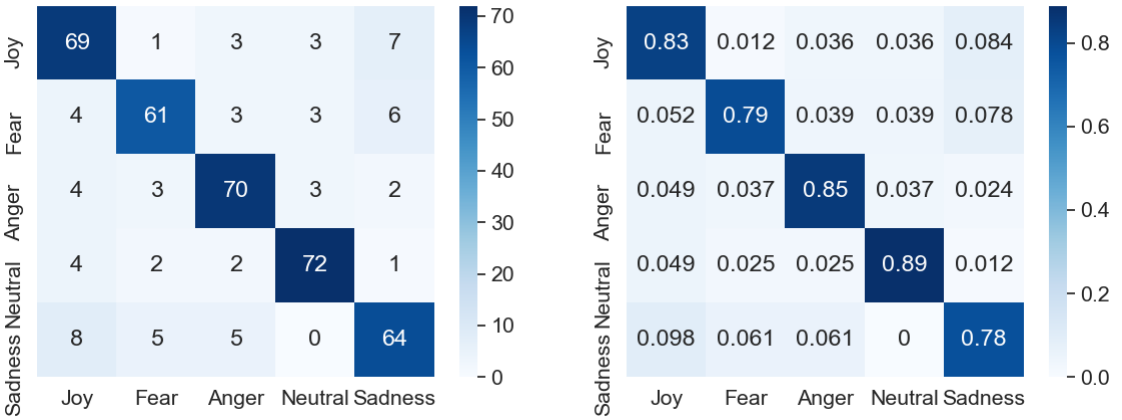

Figure 2: Confusion matrix for linear SVC model.

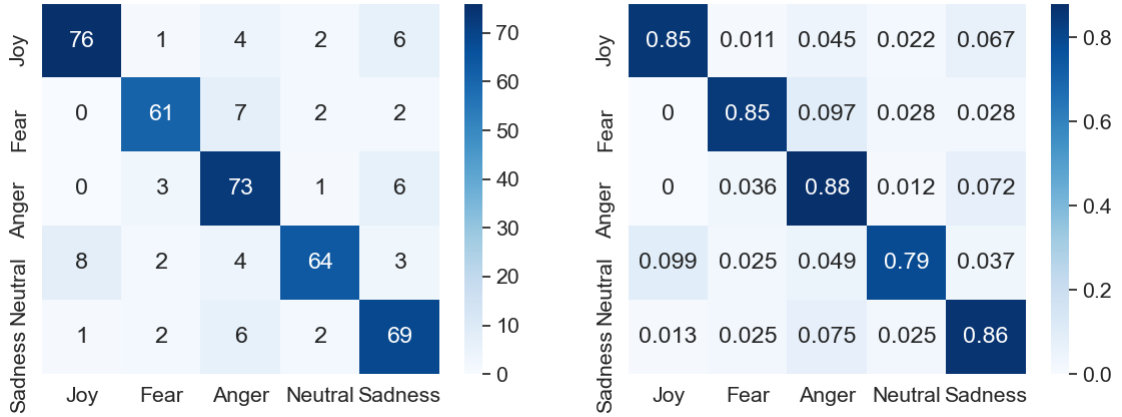

Figure 3: Confusion matrix for fastText based model.

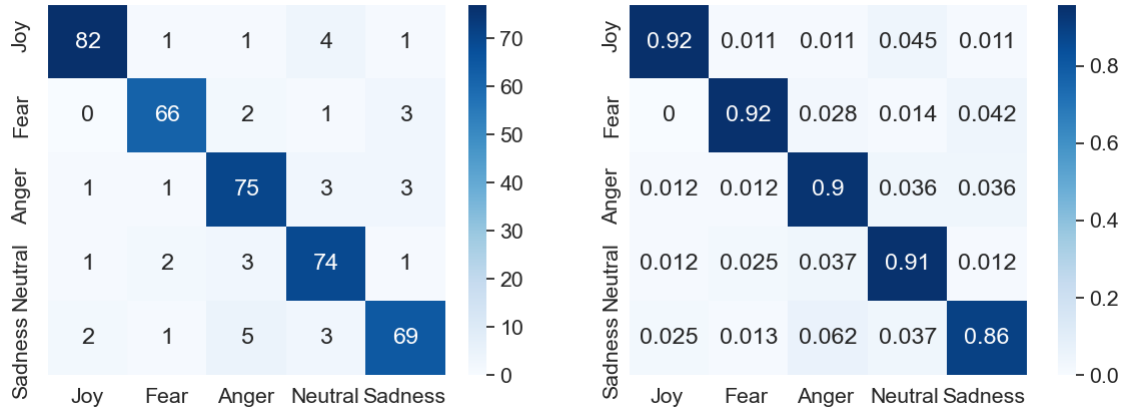

Figure 4: Confusion matrix for BERT based model.