Sentiment Analysis on Social Media for Albanian Language

Roland Vasili^{1*}, Endri Xhina², Ilia Ninka², Dhori Terpo¹

¹Department of Mathematics, Informatics and Physics, Eqrem Çabej University, Gjirokastra, Albania

²Department of Informatics, University of Tirana, Tirana, Albania

Email: *rvasili@uogj.edu.al

How to cite this paper: Vasili, R., Xhina, E., Ninka, I. and Terpo, D. (2021) Sentiment Analysis on Social Media for Albanian Language. *Open Access Library Journal*, 8: e7514.

<https://doi.org/10.4236/oalib.1107514>

Received: May 12, 2021

Accepted: June 25, 2021

Published: June 28, 2021

Copyright © 2021 by author(s) and Open Access Library Inc.

This work is licensed under the Creative Commons Attribution International License (CC BY 4.0).

<http://creativecommons.org/licenses/by/4.0/>

Open Access

Abstract

The recent advances in technology and particularly, the rising prominence of social media platforms have made it possible to express our emotions through electronic means, which have led to the creation of large collections of unstructured textual documents. These collections can be saved and potentially studied with many modern technologies like Text Mining, Machine Learning and Natural Language Processing to obtain new knowledge from them. Sentiment Analysis is a field of Natural Language Processing that focuses on extracting sentiment from text. Moreover, as a Text Mining technique expresses the ability to track the subjective opinion of a text produced by an entity. The purpose of this paper is to test and review different approaches in Sentiment Analysis for messages in the Albanian language found on Twitter. Additionally, we compare the results among different methods and note the challenges that arise while finally we suggest future directions for further research. This paper's research was conducted as follows: the data was pre-processed, before being converted from text to vector representation using a range of feature extraction techniques such as Bag-of-Words, TF-IDF, Word2Vec, and Glove. We study the performance of sentiment classification techniques from three main approaches: traditional machine learning, lexicon-based and deep learning approach. For model evaluation, since they were trained in unbalanced data, we used not only classical evaluation criteria such as Accuracy, Specificity, Precision, and Recall but more appropriate criteria such as F-measure, Balanced Accuracy, and Matthews Correlation Coefficient (MCC). According to all these criteria, our experiments revealed that LSTM based RNN with Glove as a feature extraction technique provides the best results with $F\text{-score} = 87.8\%$, followed by Logistic Regression.

Subject Areas

Artificial Intelligence

Keywords

Sentiment Analysis, Sentiment Lexicon, Text Mining, Machine Learning, Twitter

1. Introduction

Until the end of the 20th century, emotion analysis was the subject only of related disciplines like Psychology and Cognitive Sciences. It has changed as a consequence of the exponential growth of social media and the widespread commercial and social interest in it. Today, social media has become a very popular tool for everyday communication of Internet users [1]. Furthermore, the recent advances in technology have made it possible to express our emotions through electronic means and in our mother tongue. Because of easy accessibility and free format of messages, Internet users tend to abandon traditional communication tools and rapidly increasing the number of social media users. Many messages appear daily in micro-blogging media such as Facebook¹, Instagram², Twitter³ and all these have led to the creation of large collections of unstructured textual documents which we can save and potentially study. With this rapid development of social networks (social media) on the internet, individuals and organizations use public opinion to make decisions [2]. Among these media, research on the Twitter platform has also received great interest. Their main concern is the automatic extraction of opinion, or the emotional polarities. Although it seemed like a simple classification problem, in practice it proved to be more difficult, for many reasons, such as the lack of a syntactic and grammatical structure that directly characterizes the category, the diversity of words for the same emotion, and the lack of a classification of behaviors that characterize an emotion. The way we approach the text can affect the levels of analysis. Regardless of the level at which emotion classification is applied, in general, the techniques for its implementation fall into three categories: 1) machine learning techniques [3] [4]; 2) lexicon-based techniques [5] [6]; 3) and hybrid [7] [8] [9]. Many studies employ well-known Machine Learning algorithms for sentiment analysis, transforming the sentiment extraction problem into a classification problem [10]. Datasets from Twitter messages that have already been annotated, usually manually, are used to train classifiers, which are then used to extract emotions. The second category sustain in sentiment lexicons, depend heavily on linguistic resources including a sentiment lexicon composed of pairs of words and its polarity values.

Although such linguistic resources built for many languages [11]-[16], the same cannot be said for the Albanian language. The Albanian language is an In-

¹<https://www.facebook.com/>

²<https://www.instagram.com/>

³<https://twitter.com/>

do-European language, mostly spoken in Albania, Kosovo, and in other parts of the Balkans by about 7.6 million Albanians. It has distinctive morphological and lexical features, a large alphabet, and very rich in polysemantic words, but all these are not enough to stimulate the generation of linguistic resources, which help us to analyze emotive texts [17].

▪ The purpose and outline of this work

Present research is a work in progress. The aim is to build a SA System for Albanian language for studying building and evaluating models with final objectives re-implementing the state of the art in SA for texts written in Albanian language, using Python and a variety of tools (Python modules) for NLP and TM tasks. In this part we try to test and review different approaches in Sentiment Analysis for messages in the Albanian language found on Twitter, and for this we have focused on the most prevalent technologies of the last five years. For this purpose, different approximations for Sentiment Analysis in the Albanian language are presented with techniques based primarily on Machine Learning and Word Embeddings using an annotated dataset by Mozetič *et al.* in [18] for the Albanian language, and secondarily in sentimental lexicons utilizing a dictionary created by Yanqing Chen and Steven Skiena in 2014 for 81 languages, including the Albanian language [19]. In our approach, we conduct text-level sentiment analysis, in which the entire user-generated comment (tweet) is analyzed and classified into the appropriate category, and we use firstly, supervised machine learning and deep learning techniques in which experts annotate user-generated textual data and secondly a lexicon-based technique as a baseline. Finally, useful model results and predictions are also evaluated providing us more information and, in the meantime, potential future further research ideas on this topic.

The structure of the paper is as follows: Section 2 presents theoretical background for SA; Section 3 presents existing works in SA; Section 4 presents methodology; Section 5 presents experimental setup; Section 6 presents results and discussions, and in Section 7 we conclude our work.

2. Theoretical Background

2.1. The Sentiment Analysis Task

Sentiment and public opinion are an area of interest among researchers, politicians, and marketers, which traditionally used opinion polling and market surveys, and some sciences like Social sciences, Cognitive sciences, and Psychology. Today, social media has become a very popular tool for everyday communication of Internet users [1]. Furthermore, the recent advances in technology have made it possible to express our emotions through electronic means and in our mother tongue. Because of easy accessibility and free format of messages, Internet users tend to abandon traditional communication tools and rapidly increasing the number of social media users. Many messages appear daily in micro-blogging media such as Facebook⁴, Instagram⁵, Twitter⁶ and all these have

led to the creation of large collections of unstructured textual documents which we can save and potentially study. Those media are a gold mine of opportunities with massive potential to increase coverage of awareness for various purposes. In the current paper we study those texts from the point of view of Text Mining (TM). The issue with this data source is the difficulty of analysis—twitter media messages are extremely noisy and idiosyncratic, but TM offer tools and techniques to face this challenge. The field of Data Analysis has been particularly concerned with methods of extracting information from texts. But how easy is it to discover facts or meanings from a text without human supervision? The use of syntax can provide structure to the expression of written and oral speech, but the rules it applies are not strict and this is because speech has an autonomy that makes it complicated to analyze. More specifically on social media, the texts contain a lot of “noise”. This means that they may contain casual expressions illegible or misspelled. The solution to the study and analysis of such texts is provided by the Sentiment Analysis or otherwise Opinion Mining (OM). Sentiment Analysis, as a research field, has provided systems through which this vast amount of unstructured information has been structured and translated into public opinion about products, services, politics or any other subject on which an opinion can be expressed, thus producing useful knowledge for both the field of science and the field of business and industry.

▪ Definition

There are many definitions of *Sentiment Analysis* (SA) but we prefer those presented by Bing Liu [20]: “SA, also called OM, is the field of study that analyzes people’s opinions, sentiments, evaluations, appraisals, attitudes, and emotions towards entities such as products, services, organizations, individuals, issues, events, topics, and their attributes”. The author states that the field represents a large problem space, and any related names and slightly different tasks, for example, sentiment analysis, opinion mining, opinion analysis, opinion extraction, sentiment mining, subjectivity analysis, affect analysis, emotion analysis, and review mining, are now all under the umbrella of sentiment analysis.

Its primary goal is to develop and implement techniques and systems capable of detecting, extracting, and quantifying subjective information such as opinions and emotions through natural language, whether written or spoken.

Written information, based on its objectivity, can be mainly categorized into two types: opinions and facts. Opinions are subjective expressions that describe a person’s thoughts, assessments, and feelings. On the contrary, the facts are objective and only expressions and descriptions of reality.

Based on this separation, SA, like many other Text Mining and Natural Language Processing problems, can be formulated as a classification problem where two sub-problems must be solved:

- Classify a phrase as *subjective* or *objective*, known as *subjectivity classifica-*

⁴<https://www.facebook.com/>

⁵<https://www.instagram.com/>

⁶<https://twitter.com/>

tion.

- Classification of a phrase as an expression of a *positive*, *negative* or *neutral opinion*, also known as polarity classification or sentiment classification. The sentiment classification part will also be the main research area with which this paper will address.

2.2. Sentiment Analysis Techniques

The Sentiment and Opinion Analysis can be applied at various levels of analysis and classified according to the way we approach the text and the desired detail of the extracted information [21].

1) Document/Text-Level: At this level we think the text expresses a single view of a target, which we are trying to define. The aim at this level is to determine the wider attitude of the author, positive or negative, in a text that includes judgments and opinions. The main challenge is to gather the sentences that will define the sentiment of the whole text [22].

2) Sentence-Level: In this way the sentence is a separate entity that has a positive, negative or neutral emotion [22] [23].

3) Entity and Features Level: Assignment to Entity and feature level or aspect-based attempts a more detailed analysis in relation to the previous two, and focuses on the same terms and not on an analysis of structural elements of the language (text, sentence, phrase). The words of a sentence become its features. The opinion holders can give different opinions for different aspects of the same entity like this sentence [24].

Regardless of the level at which emotion classification is applied, in general, the techniques for its implementation fall into three categories: 1) *machine learning* techniques; 2) *lexicon-based* techniques; 3) and *hybrid*. In the former, all known *machine learning* algorithms are used to classify sentiment, with the main and most popular ones being those based on supervised learning. In *lexicon-based* techniques, sentiment lexicons with emotionally polarized words are employed for the task of classification. These lexicons contain the polarity of each word either they have a positive either a negative emotional impact. Finally, *hybrid* techniques are a combination of the previous two categories.

From the point of view of TM, the problem of SA is a prediction problem, and as the name implies, is used to predict an object feature (target). The two main predictive categories are:

Classification—The most important techniques are: *Decision Trees*, *Bayesian Methods*, *Artificial Neural Networks*, *Support Vector Machines (SVMs)* *Networks*, *Instance Based Methods* etc.

Regression Analysis—The most important techniques are: *Linear Regression* & *Logistic Regression*.

The use of the TM approach in classifying text sentiment is discussed widely in the literature [20] [25] [26] [27] [28]. SA is an essential task to detect the sentiment polarities in the text with many applications. In a recent study of 1800 papers from 2006 to 2019, SA is the 4th topic between 40 most used topics of

twitter-related studies [29]. In those studies, there are three main approaches for SA: *Lexicon-based approach*, *Machine Learning approach* and *Hybrid approach* [30].

Shi *et al.* in [31] propose an integrated concept map of sentiment classification techniques, where, as shown in **Figure 1**, almost all up-to-date SA methods, techniques, and main directions are included

3. Existing Works

There is a variety of related works for SA, but we are only interested in contributions for the Albanian language. Biba & Mane in [32] presents the first approach for Sentiment Analysis in Albanian language. They show through extensive experiments on the Weka⁷ platform, with text data from political news consisting of five different topics, that the proposed approach (based on Machine Learning) is effective in classifying text documents as belonging to a positive or negative opinion regarding the given topic, giving accuracy from 70% to 92.5% and highlighting the higher performance Hyper Pipes algorithm. In their paper, Kadriu & Abazi in [33] crawled Albanian news articles from online sources, used them to train some classifiers, tested them through measuring accuracy for each classifier separately and concluded that accuracy depends on the number of inputs. Perceptron achieved higher average accuracy (0.91) with a lower deviation (0.02). They also analyzed the training and testing time. Kote *et al.* in [34] created

Figure 1. Concept map of sentiment classification techniques [31].

⁷<https://www.cs.waikato.ac.nz/ml/weka/>

a text corpus with five subjects of 500 Albanian written opinions collected from different well-known Albanian newspaper, and evaluated the performance of 53 classifiers for opinion classification on document level in a corpus in Albanian language. They used Weka software to test and train the classification algorithms on their corpus. The top five algorithms achieved a performance from 79% to 94%, while the best performing algorithm was Hyper Pipes with a weighted average of the percentage of instances classified correctly 83.62%. Trandafil *et al.* in [35] collected Albanian text data regarding 20 domains, where each category has 40 documents made up by textual information chosen randomly on the web, respectively from the previous fields and after applying natural language pre-processing steps, they applied several algorithms such as Simple Logistics, Naïve Bayes, k-Nearest Neighbor, Decision Trees, Random Forest, SVM and NN show that Naïve Bayes and Support Vector Machines perform best in classifying Albanian corpuses. Skenduli *et al.* in [17] fetched around 60,000 Facebook posts belonging to 119 Albanian politicians, used a deep learning classifier based on Keras⁸ framework and TensorFlow⁹ as back-end and three classification algorithms, such as NB, IBK, and AMO implemented in Weka showed that DL approach produced better results in terms of classification accuracy. Kadriu *et al.* in [36] focus on the text classification for Albanian news articles using two approaches, the first utilized nine classifiers from the scikit-learn¹⁰ package and the second used the fastText¹¹. They concluded that bag of words model achieves the best accuracy among them. Finally, in [37] they study language challenges in Aspect-based SA and aim to define the biggest challenges that appear in Albanian language in comparison with English, and after analyzing certain amount of data, identified the following issues: inflections, negation, homonyms, dialects, irony, sarcasm and stop-words presence in aspect terms.

4. Methodology

In the present work, we addressed the problem of SA in labeled data from Twitter. As it is known, in this platform users interact with each other with messages known as “tweets” [10]. It serves as a means of expressing their opinions or feelings on various topics. Twitter is one of the most widely used microblogging platforms on the internet, which in January 2019 had over 330 million unique monthly active users [38]. Tweets are short, informal, and often unstructured messages that allow for communication, sharing, and interaction. Polarity classification is the process of assigning a positive or negative sentiment to an entire tweet. It is important to note that, unlike other platforms, almost every user’s tweets are public and relevant. On the other hand, this is a tool used by individuals, organizations, even and states for various studies. Moreover, the progress made in recent years in Text Mining, NLP, Machine Learning, Artificial Intelligence, and related fields makes it possible to achieve high accuracy in predicting

⁸<https://keras.io>

⁹<https://www.tensorflow.org/>

¹⁰<https://sklearn.org/>

¹¹<https://fasttext.cc/>

the emotions expressed in these texts.

Twitter allows interaction with its data through an interface (Application programming interface, API). The concrete interface, while providing access to most of its services, does not allow access to the code that performs these services. So as to use this interface you need a programming language like php, ruby or python to create the requests and the data that will be returned will be of JSON form.

We will try to do SA in “tweets” using the techniques referred to in the section 2 of this paper. More specifically, we will try to classify the polarity of “tweets” into two classes: positive or negative. If the tweet has both positive and negative elements, the dominant tag will be. **Figure 2** depicts the procedure for obtaining our models.

4.1. Data Collection

Since there is no publicly labeled and constructed public dataset for the Albanian language, for research purposes in the fields of either linguistics or NLP and creating a database of tweets labeled from scratch with acceptable standards has been difficult to be realized, we used the dataset from Mozetič *et al.* in [18]. Their work includes datasets for 15 languages and is supported by the Slovenian Ministry of Education, Science and Sports. We know in advance that the authors in their assessment of the labels in the Albanian language have expressed “... *Albanian and Spanish, indicate low quality annotators which should be eliminated from further considerations*”, hence the low quality of the annotators, something we noticed when we started doing the first experiments with this dataset.

Twitter is characterized by some features (which is also the list of the information we can get from it). Some of these features we list below:

- **Tweet:** A simple text message posted on Twitter. Its content can be a maximum of 140 characters (until 2017 then 280), it can vary from personal information or personal opinion on the product, event or on links, news, photos, or videos.

Figure 2. Operating diagram of a supervised machine learning models for SA.

- **User/Username:** A user must register on the platform to post tweets.
- **Mention:** Mention in a tweet indicates that the post mentions another user.
- **Replies:** Replies to a tweet are used to indicate that the post is a reply to another tweet and are usually used to create a conversation.
- **Follower:** Followers refer to users who follow tweet posts and user activity.
- **Retweet:** Retweets refer to tweets that have been re-distributed.
- **Hashtag:** Hashtags are used to indicate the connection of a tweet post with a specific topic.
- **Privacy:** Twitter gives the user the option to decide whether his/her tweets will be visible to everyone or only to those who have approved.
- **Lang:** acronym for language (for example “en” for English, “sq” for Albanian).
- **Id:** tweet identifier.
- **Place, coordinates, geo:** available geo-location information.
- **User:** full author profile etc.

Communication with Twitter for extracting tweet posts was done through an API that we analyzed above and is provided by Twitter itself. The platform offers two API interfaces, Stream API and Rest API. The difference between them lies in the fact that Stream API opens a link where data is retrieved in real time, until the user decides to stop, while in Rest API, can be selected concrete communication time interval with a limit of 900 requests/15-minutes. Tweepy, an easy-to-use Python library for accessing Twitter services, was used for this extraction. It is important to note at this point that Twitter requires all applications to use *Oauth* for authentication. For this purpose, we created an application, registered our client application with *Twitter* thus obtaining the necessary *Consumer Key*, *Consumer Secret*, *Access Token* and *Access Token Secret keys*, which also constitute our credentials for access to Twitter. Afterwards, we created a code in Python that follows the rules set by Twitter and downloaded the dataset based on the id of the tweet and saved it to a *csv* file (converting the JSON file that Twitter returns to us).

The original database has 53,005 tweets. From all the above fields we filtered only the tweet field that contains the texts in the Albanian language for each tweet id.

4.2. Text Pre-Processing

This step is considered one of the most important steps for the success of TM algorithms/methods. Most TM approaches are based on the idea that text data can be described by the set of contained words for example Bag-of-Words representation. According to Vasili et al. in [39], steps for preprocessing textual data (see **Figure 3** in next page) are as follows:

- Text Structure Removal.
- Tokenization.
- Stop words Removal.
- Filtering (Removing terms based on their length).

Figure 3. Text Pre-processing Steps.

- Filtering (Removing terms based on their frequency).
- Part Of Speech Tagging (Syntactical and Semantical Analysis)
- Stemming.
- N-grams.
- Term weighting.

4.3. Feature Extraction (Text Representation)

The previous sections addressed the problem of classification in general and SA in particular, as well as the TM techniques that can be used for SA. If we are given several points in an *n-dimensional space* accompanied by a class, the implementation of a mathematical model is required so as to accurately classify the new points in the appropriate class.

So, the above are mathematical tools applied to any classification problem. Although, when we want to classify objects, before running the classification algorithm, they must be represented in some way as vectors of feature space. This process is called *feature extraction*, and the general field of study is “*feature engineering*”.

In such cases, we need to give the classifier raw data without “*feature engineering*” and assign it to find “*higher level*” features that can distinguish classes. In particular, ANN is based on the logic of searching for such features with the help of hidden layers.

We will focus on ML and DL algorithms with unstructured and raw text data and not with “*feature engineering*” techniques. So, in this section, we will present ways to extract features from text data. The problem is to classify a group of tweets into two classes: *positive sentiment* or *negative sentiment*. Therefore, it is necessary to correspond each tweet in a vector representation $\mathbf{x} = (x_1, x_2, \dots, x_n)$ where n is the size of the feature space.

4.3.1. Bag of Words

The simplest way to extract features from text data is the *Bag-of-Words* method. With this, words, phrases, sentences, and texts are represented by sparse vectors (called *One-Hot Encoding*) with high dimension in a high dimensional vector space. As the name implies, the text part is treated as a bag of words. That practically means that it ignores the word order and cares only about the presence or frequency of words in the text. However, it is particularly prevalent for its sim-

plicity and the satisfactory results it gives, especially in the field of SA and Topic Classification [40].

Data sequences are called *n-grams*. In our case, where we process text data, *n-grams* are practically consecutive word sequences. Thus, the sequence of two words is called *bigram*, of three words *trigram*, etc. Also, the simple word namely *n-grams* for $n = 1$ is called *unigram*.

4.3.2. Word Vectors

Given the disadvantages of the Bag-of-Words model, researchers concluded that we need another mathematical model to describe text data. The model needed a way to represent words, phrases, sentences, or complete texts in a small vector space so that close points in space represent semantic similarity. For computational reasons, the vectors must be dense with continuous and non-discrete features. Furthermore, these representations must have a global character for using them in various word processing applications, such as SA, Machine Translation, et cetera.

In the literature, such vector representations are referred to as *embeddings*, *representations* or simply *vectors*, while for word representation we find the terms *word embeddings*, *word representations* or *word vectors*.

According to word embeddings theory, each word corresponds to a vector of a small dimensional space. These vectors are dense, with real number values, and size up to several hundred, while the corresponding becomes such that the relations between the words converted to features of “*word vectors*”. There are two main methods to create vector representations. The first uses dimensionality reduction techniques in co-occurrence matrices, while the second uses neural language models. According to Selva Birunda & Kanniga Devi in [41], the Word Embeddings Techniques are divided into three types:

- 1) **Traditional Word Embedding:** *Count Vector*, *TF-IDF Co-Occurrence Matrix*.
- 2) **Static Word Embedding:** *NPLM*, *Word2Vec*, *GloVe*, and *FastText*.
- 3) **Contextualized Word Embedding:** *CoVe*, *ELMO*, *GPT & GPT 2*, *BERT*, *XLNet*, and *RoBERTa*.

In the present work, we used the traditional techniques (explained in the previous section) and the **GloVe**¹² model discussed below compared to **Word2Vec**¹³.

▪ Word2Vec Model

The *Word2Vec* model is a neural language model used to represent words in small dimensional vector spaces. This model was proposed by Mikolov *et al.* in [42] and attracted academic interest thanks to the quality of the vector representations that it produces. It is capable of capturing complex syntactic and semantic relations between words and transforming them into linear properties of vector space. It is generally a window-based prediction language model, based

¹²<https://nlp.stanford.edu/projects/glove/>

¹³<https://en.wikipedia.org/wiki/Word2vec>

on the architecture of a simple neural network.

Mikolov *et al.* in [43] gave two different implementations of the Word2Vec model where the first is called *Continuous Bag-of-Words (CBOW)* based on the prediction of the target word with given context words, and the second *Skip-Gram (SG)* based on the prediction of context words with a given target word.

▪ GloVe Model

According to Pennington *et al.* in [44], a disadvantage of the Word2Vec algorithm is the non-integration of information from global count statistics in the whole corpus of the text, in vector representations. They show two main ways of extracting vector representations, where the first utilizes general text corpus statistics and usually includes co-occurrence matrix factorization methods, while the second utilizes local information using local context window methods. In the same paper, the authors propose a model that uses information from global count statistics in the dataset and is trained with gradient descent methods. There is even a theoretical analysis that wants this model to be essentially closely related to the Word2Vec model of Mikolov *et al.* in [42] ([45] [46]).

4.4. Sentiment Analysis Models

4.4.1. Semantic Orientation-Based Approach for Sentiment Analysis

These techniques are based on the assumption that the semantic orientation (positive, negative, or neutral) of a text is determined by calculating the orientation of the individual terms (words or phrases) that make it up.

In general, they relate to the process of calculating the semantic orientation of each term of a text. Once the terms orientation has been calculated, then their values are statistically weighted to produce the total orientation of the text (document, sentence, or phrase). Depending on how this calculation is done, they are divided into two subcategories:

- 1) *Corpus-based*.
- 2) *Lexicon/dictionary-based* semantic orientation [47].

1) Corpus-Based Approach

This approach is based on the assumption that when a word co-appears (a relative concept meaning the distance between words e.g. less than 10 words) more often with positively oriented words (e.g. “excellent”), its orientation value tends to be positive, and analogously when co-occurring more often with negatively oriented words (e.g. “bad”), its orientation value tends to be negative. Initially, a set of terms with a known orientation is chosen, belonging to one of the classes (“positive” or “negative” orientation). In the following, statistics are collected for the co-occurrence of each word with words of both classes. Thus, if a word appears more often next to the words “negative”, we consider that word to be “negative” and vice versa.

2) Lexicon/Dictionary-Based Approach

The main dictionary-based strategy starts with assembling a small group of words describing opinion [48] and then searching for corresponding synonyms

and antonyms of these words in a large lexical database such as WordNet [49], to expand this group of words. According to Hu & Liu in [50], the initial set of words does not need to be very large about 30 words are enough. Then, new words are added to the original group and that is repeated until there are no more new words to add.

Sentiment Lexicon approach is a word-level approach to analyzing the sentiment of a text [5], and is one of the most frequently used approaches for classifying tweets [51].

Although much work has been done to develop such lexicons for the English language, the same cannot be said for other languages. Currently, no work has been done on the sentiment lexicons for the Albanian language.

4.4.2. Machine Learning Approach

Machine Learning (ML) deals with the study of computer algorithms that improved automatically through experience [52]. Traditionally, ML approaches are divided into three broad categories: Supervised learning, Unsupervised Learning, and Reinforcement Learning [52]. Usually, a SA task is modeled as a classification problem, by which a classifier is supplied with a text and returns a category (positive, negative, or neutral). Today, many other approaches have been developed which do not fit into this type of classification. Since 2020, Deep Learning has become the predominant approach for most work in progress in the field of Machine Learning [53].

As we said, there is a variety of classifiers, but here is a brief introduction of the ones we will use in our work:

1) Logistic Regression: is a statistical machine learning algorithm that uses a logistic function, also called sigmoid function, to compute the probability for each class and then choosing the class with the maximum probability [54].

2) Naïve Bayes: is a generative classifier that is based on Bayes rules. The algorithm computes the posterior probability of a class, based on the distribution of the words in the document by ignoring the actual position of the words in the document, and working with the “bag of words” assumption [55].

3) Decision Tree: creates a decision tree by using the entropy to determine which attribute of a given instance will optimize the classification of the instances in the dataset and which values of these ranges will provide the best classifying results. Rules can be generated for each path in the tree [56].

4) Multilayer Perceptron (MLP): is a class of feedforward ANN that consists of at least three layers of nodes: an input layer, a hidden layer and an output layer. Except for the input nodes, each node is a neuron that uses a nonlinear activation function [57].

5) Support Vector Machines (SVM) Networks: a non-probabilistic model which uses a representation of text examples as points in a multidimensional space [56]. Examples of different categories (sentiments) are mapped to distinct regions within that space. Then, new texts are assigned a category based on similarities with existing texts and the regions they’re mapped to.

6) Random Forest: creates the forest from a set of decision trees, each created by selecting random subsets of training data. The final class of the new object is assigned to the class with the highest value and is achieved as an outcome of all trees in the forest. Tree ensembles are a divide-and-conquer approach used to improve the performance [58].

4.4.3. Hybrid Approach

1) Deep Learning: a diverse set of algorithms that attempt to mimic the human brain, by employing ANN to process data. Even in this category, there is classifiers plethora, but again here is a brief introduction of the ones we will use in our work:

a) Convolutional Neural Network (CNN): is an ANN that has one or more convolutional layers that are used mainly for image processing, classification, segmentation and also for other auto correlated data. Three types of layers are used: *Convolutional Layer*, *Pooling Layer* and *Fully Connected Layer* (similar to MLP networks). These layers overlap to create a complete CNN architecture [59].

b) Recurrent Neural Networks (RNN): are a kind of ANN that makes it possible to model long-distance dependencies among variables [60] [61]. The main feature of an RNN is the hidden state, which captures sequential dependencies on information.

A special case of RNNs, that can recall information over a long period is **Long Short Term Memory (LSTM)** networks. LSTM introduced by Hochreiter & Schmidhuber in [62], and since then there have been many improvements by many researchers [63].

2) Hybrid Approaches: combine the desirable elements of rule-based and automatic techniques into one system. One huge benefit of these systems is that results are often more accurate. In this paper, we will not deal with these systems but referred to them to complete the coverage map of the techniques used for SA [7] [8] [9].

4.5. Evaluation Measures

While there are many ways to evaluate the performance of an algorithm in a categorization task, the metrics we will use in this task are the following: *Accuracy*, *Precision*, *Recall* and *F1-score* [64], and additionally *Balanced Accuracy* and *MMC* (more suitable for unbalanced datasets) [65]. An easy way to show how the above metrics were calculated, is by using the *confusion matrix*.

▪ Accuracy & Balanced Accuracy

If we consider a two-class dataset (as in SA) e.g. $C = \{+, -\}$, then we can determine the cases of classification of the number of instances of the dataset as following:

- **True Positives—TP:** the number of samples of class “+” that were classified as + by the classifier.
- **True Negatives—TN:** the number of samples of class “-” and the classifier

classified them as “–”.

- **False Positives—FP:** the number of samples of class “–” —that were classified as + by the classifier.
- **False Negatives—FN:** the number of samples of class “+” and the classifier classified them as “–”.

Then *Accuracy* (can be a misleading metric for imbalanced data sets) and *Balanced Accuracy* (don’t suffer from previous problem), can be calculated from the formulas:

$$Accuracy(M) = \frac{TP + TN}{TP + TN + FP + FN}, \quad (1)$$

$$Balanced\ Accuracy(M) = \frac{recall(M) + specif(M)}{2}. \quad (2)$$

where $recall(M)$ and $specif(M)$ will be clarified below, while M refers to the confusion matrix constructed from the model results and looks as in **Table 1**.

▪ Sensitivity & Specificity

Sensitivity, that expresses *True Positive recognition Rate-TPR*, and *Specificity*, that expresses the *True Negative recognition Rate-TNR* are calculated as follows:

$$sensit(M) = \frac{TP}{TP + FP}, \quad (3)$$

$$specif(M) = \frac{TN}{TN + FN}. \quad (4)$$

▪ Precision, Recall & F-measure

According to Wikipedia *precision* is the fraction of relevant instances among the retrieved instances, while *recall* (also known as *sensitivity*) is the fraction of relevant instances that were retrieved. They can be calculated from the formulas:

$$precision(M) = \frac{TP}{TP + FP}, \quad (5)$$

$$recall(M) = \frac{TP}{TP + FN}. \quad (6)$$

However, it can be easily perceived from the above equations, that increasing the size of accuracy leads to reducing the recall and vice versa. To this effect, for problems that require the use of these valuation metrics, the *F-measure* (also known as *F-score*), is very often used, which expresses the harmonic mean of accuracy and retrieval by combining these two quantities, according to the formula:

Table 1. Confusion matrix.

		Predicted Class	
		+	–
Actual Class	+	TP	FN
	–	FP	TN

$$F(M) = \frac{2 * TP}{2 * TP + FN + FP}. \quad (7)$$

▪ **Matthews Correlation Coefficient (MCC) or phi coefficient**

It is another important measure to give an unbiased rating regardless of class sizes. It is more informative than *F1-score* and *Accuracy* in estimating binary classification problems because it takes into account the balance ratios of the four categories of the confusion matrix [65]. Its size is expressed by the formula:

$$MCC(M) = \frac{TP * TN - FP * FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}. \quad (8)$$

5. Experimental Setup

5.1. The Dataset

The data is a “mix” with other data that has no emotion like usernames, #hash-tags, @annotations, Retweets and many of the elements we described above. They should be removed as they will not provide any information during the analysis [66]. Consequently, tweets need to be processed and transformed into a standard form. We also need to extract useful features from the text like uni-grams and bigrams which are a form of representation for tweets.

Dataset by Mozetič *et al.* in [18] comes in a file named “Albanian_Twitter_sentiment.csv”, so in csv (comma-separated values) format with three fields: *TweetID*, *HandLabel* and *AnnotatorID*. The first field is a unique number (18 digits) representing the tweet identifier, the second is the label (Positive, Neutral or Negative) and the third field is a number (3 digits) representing the annotator (the file appears to have used 12 people for labeling, 4 of whom have done most of the work). The database was created from tweet posts in the period June to September 2013 and contains 53,005 posts, of which 8106 Negative, 18,768 Neutral, and 26,131 Positive. At first, we applied a filter to remove all posts that have zero values in the text field, because they are not important, and then we filtered the duplications (6888 posts, which probably makes us think that an oversampling was done to balance the classes, but looking at the filtration results it seems that this is not the case), a total of 36,412 posts remain, of which 5649 Negative, 12,853 Neutral and 17,910 Positive.

In the current work, we will only work with two classes, positive and negative so we will ignore the neutral class.

5.2. Data Pre-Processing & Analysis

As we mentioned above the original data taken from Twitter generally results in a very “noisy” dataset. This happens due to the casual nature of people using social media. Tweets have special features which need to be carefully extracted. Therefore, the data need to be normalized to create a dataset that is easy to learn from the classifiers to be used. Accordingly, we have applied a large number of pre-processing steps to standardize the dataset and reduce its size. The process of filtering tweets that has been performed is as follows:

- 1) Strip Non-ASCII characters.
- 2) Remove references (@), punctuation marks [“”!.,():;] and the symbol # from the hashtag.
- 3) Remove URLs of type (http://) and replace them with the URL shortcut.
- 4) Replace emoticons (Laugh, Love, Smile, Wink) with EMO_POS and (Cry, Sad) with EMO_NEG.
- 5) Remove symbols (=, %, &) and RT shortcut from retweets; non-capital conversion.
- 6) Removing characters with over 2 repetitions leaving them with only 2.
- 7) Tokenization, stop-words removal, stemming—We created a simple stemmer in Python based on work of Sadiku and Biba [67].

As we have pointed out above, the form of data supplied by the classifier is of crucial importance, therefore this section has been given special importance. In the present work, all this was done with the help of the *Regular Expressions (re)* library of Python, where using regular expressions allows the parameterization of alphanumerics. Thus, tweets are ready to be given as an input to the classification models and to be classified into categories. **Table 2** shows some sample tweets from the training dataset, along with their normalized versions.

In **Table 3** in the next page, a preliminary analysis of the contents of the dataset was performed, after pre-processing.

Table 2. Example tweets from the dataset and their normalized versions.

Row	Fule: Mirëpresim zgjedhjet e qeta në Shqipëri http://t.co/rHs1uUtUVf
Normalized	fule mirepresim zgjedhjet qeta shqiperi URL
Norm. Stemmed	fule mirepres zgjedhj qeta shqiper URL
Row	Brazil 2014, perballje interesante ne grupe http://t.co/9yH1hgwEWD
Normalized	brazil perballje interesante grupe URL
Norm. Stemmed	brazil perballj interesant grup URL
Row	Le të fillojnë prishjet... http://t.co/zXTlfAT41m via @ShqiptarjaCom
Normalized	fillojne prishjet URL USER_MENTION
Norm. Stemmed	fillojn prishj URL USER_MENTION

Table 3. Statistics of preprocessed train and test datasets.

	Training Dataset						Testing Dataset					
	Total	Unique	Average	Max	Pos.	Neg.	Total	Unique	Average	Max	Pos.	Neg.
Tweet	19,310	-	-	-	14,742	4568	4829	-	-	-	3686	1143
User Mentions	1704	-	0.08824	7	-	-	425	-	0.088	6	-	-
URLs	17,331	-	0.8975	2	-	-	4345	-	0.8997	3	-	-
Unigrams	128,560	27,303	6.6576	23	-	-	31,994	12,618	6.625	19	-	-
Bigrams	109,290	80,262	5.6597	-	-	-	27,179	23,493	5.628	-	-	-

In the case of using the frequency feature type (number of encounters in a tweet), the vector has a positive value and, for this, the matrix of frequency vectors constructed for the whole training dataset, and then each *term frequency* (**tf**) scaled according to the *inverse-document-frequency* (**idf**) of the term to assign higher values for significant terms. The **idf** of a term **t** defined by the formula:

$$idf(t) = \log_{10} \left(\frac{1 + n_d}{1 + df(d, t)} \right) + 1. \quad (9)$$

where n_d is the number of total documents (in our case total tweets) and $df(d, t)$ is the number of documents (tweets) where encountered the term **t**. To better manage the memory of the computer system, since the rendering of sparse vectors requires a lot of memory, we used the `scipy.sparse.lil_matrix` data structure provided by `Scipy`¹⁴ which is a memory efficient linked list based implementation of sparse matrices. Along with it, we also used Python generator where we could, instead of keeping the complete dataset in memory.

5.3.3. Dense Vector Representation

For dense vector representation we use all vocabulary of *unigrams* of dataset, assign an integer index to each word depending on its rank (starting with 1 for most common word, 2 for the second and so on). Each tweet is then represented by a vector of these indices which is a dense vector.

5.3.4. Word2Vec & Glove

In general, some models such as neural networks process language inputs by replacing the input elements x_i , which are usually words, with the corresponding vectors $v(x_i)$, called *embeddings*. These can either be trained from scratch or can be pre-trained. Pre-trained models are those who learn as one task at a time, and are used to solve another similar problem. These models capture the concept of word semantics and syntax as they are trained in very large corpora and stored for future use. This is a method similar to a new trend in machine learning called “*Transfer Learning*”. For such models, we will use two pre-trained embedding types, *GloVe* and *Word2Vec* which we discussed in section 4.3.2. Since there are pre-trained model embeddings in Albanian only for *fastText* from *Facebook*, which is not the subject of our study, we had to create these models from scratch. We trained the *GloVe* model from the entire *Wikipedia* corpus in Albanian (a file of about 90,670 KB and 47,887 articles after pruning), with the help of the *Gensim*¹⁵ library for word pre-processing and according to the requirements of the model from the work of Pennington et al. in [44] and with the help of instructions on the author’s website. The training was conducted on an *Ubuntu* 20.04 system. For the pre-trained *Word2Vec* model, we mainly used the *Gensim* library which in calling the *Word2Vec* model requires four parameters: **sentences** (prepared and pre-processed database in the required format), **size**

¹⁴<https://scipy.org/>

¹⁵<https://radimrehurek.com/gensim/>

(generated vector dimensions), **windows** (distance between the current word and target one), **min_count** (ignores all lower frequency words) and **workers** (how many threads we want to use, depending on the system processor). The default vector dimension in *Gensim* is 100 features, but we created it with 300, such as the *GloVe* model.

5.4. SA Models Implementation

In the present work, we experimented with different models of classifiers. The major part of the experimental work is based on the report of Group No. 29 of the Kaggle¹⁶ competition “*CS5228 Project 2 Twitter Sentiment Analysis*” consisting of A. F. Ansari, A. Seenivasan, A. Anandan and R. Lakshmanan on 12 November 2017. Based on the *Pareto*¹⁷ principle, we divided the dataset into 80% train and 20% test data. Given that we are dealing with an unbalanced dataset, we did not create these two groups with ready-made libraries. Instead, we used Python to maintain the proportionality and randomness of both classes in datasets (approximately 1:3 ratio). Subsequently, in all models, except those otherwise mentioned, we used 10% of the training data as a validation set. In all models, except the hybrid models of section 4.4.3, we have used sparse vector representation of tweets. In hybrid models, we used dense vector representation of tweets.

5.4.1. Lexicon-Based Model

For this model, we used a dictionary created by Yanqing Chen and Steven Skiena in 2014 for 81 languages, including the Albanian language [19]. The vocabulary contains two datasets for each language, the Albanian language a positive with 845 words and a negative with 1231 words (the respective English dictionaries were with 2135 and 4941 words). If the number of negative words in the tweet post is equal to the number of positive words, it is assigned a positive sentiment.

5.4.2. Logistic Regression

For this model, used **Keras** library dense vector representations, where the *unigrams* size was 15,000 and the *bigrams* 10,000. For this, we used the **sequential** model of the **Keras** library (`model.compile(optimizer = “sgd”, loss = “binary_crossentropy”, metrics = [“binary_accuracy”])`).

5.4.3. Naïve Bayes Model

For this model, we used *MultinomialNB* from **sklearn.naive_bayes** of **Sci-kit-learn**. Also, we used the *Laplace* smoothed version of *Naïve Bayes* with the smoothing parameter $\alpha = 1$ and found that presence features outperform frequency features as mentioned also in the above report.

5.4.4. Decision Tree Model

For **Decision Tree** model we used *DecisionTreeClassifier* from **sklearn.tree**

¹⁶<https://www.kaggle.com/c/cs5228-project-2/> & <https://github.com/abdufatir/twitter-sentiment-analysis/>

¹⁷https://en.wikipedia.org/wiki/Pareto_principle/

package provided by *Scikit-learn*. Also, we used *GINI* index to evaluate the split at every node and the best split is chosen always. Experiments using *unigrams* with and without *bigrams* have also been performed with this model.

5.4.5. Multi-Layer Perceptron Model

For the *Multi-Layer Perceptron*, we used *Keras* with *TensorFlow* back-end with a hidden layer consisting of 600 hidden units. Its output is a number that, after passing to a non-linear sigmoid is limited to the range [0, 1]. In order to train the model, we used sparse vector representations of tweets. The output of *NN* gives a probability $P(\text{pozitiv}|\text{tweet})$ for example, the probability for the sentiment of a tweet to be positive. Afterwards, we trained the model using the *binary cross-entropy loss* based on update scheme of Kingma & Ba in [68] which converges to the eighth epoch, as experiments with *stochastic gradient descent* (*SGD*) and *Momentum* required many epochs to converge.

5.4.6. SVM Networks Model

For *SVM Networks* we used *SVM* from *Scikit-learn*. In this model we gave the value of *C* parameter to 0.1. This parameter tells the *SVM optimization* how much you want to avoid misclassifying each training example. We run experiments using *unigrams* with and without *bigrams*.

5.4.7. Random Forest Model

For *Random Forest* model we used *RandomForestClassifier* from *sklearn.ensemble* of *Scikit-learn* library. We conducted various experiments changing the number of used estimators (trees) from 10 to 50.

In the context of the problem channeling with the *Decision Tree* model, we also experimented with the *XGboost* classifier with a maximum tree depth of 20 used for over-training control. Since this model's algorithm utilizes an ensemble of "weak" trees, it is important to optimize the number of estimators to be used, which has been 200.

5.4.8. Convolutional Neural Network (CNN)

For *CNN* model we used *Keras* with *TensorFlow* back-end dense vector representation of the tweets. We used a vocabulary of 27,300 words from the training dataset where we represent each word in our vocabulary with an integer index from 1 to 27,300, where this index represents the rank of the word in the dataset, while index 0 is reserved for the special padding word. Then each of these words is represented by a 300-dimensional vector. The first layer is the *Embedding* layer which is a matrix of shape $(v + 1) \times d$ where v is the vocabulary size ($v = 27,300$) and p is the dimension of each word vector ($p = 300$). We initialize the embedding layer with random weights from $N(0, 0.1)$, where each row in the embedding matrix represents the 300-dimensional word vector in vocabulary. For words in our vocabulary which match *GloVe* pre-trained words explained above we seed the corresponding row of the embedding matrix of from *GloVe* vectors. For example, for each tweet dense vector representation is padded with

0s at the end until its length is equal to *max_length*, which is a parameter we tweak in our experiments. Then, we trained our model using *binary cross entropy loss* with the weight update scheme being the one defined by Kingma & Ba in [68]. Experiments conducted with *stochastic gradient descent (SGD)* and *Momentum* weight updates takes longer to converge compared to validation accuracy equivalent to Kingma & Ba in [68]. Experiments showed that multi-layer convolution architectures had better results. However, we were content with experiments up to 2 layers of convolution because of the time and the computational power. The following is the architecture where we had the best results: *embedding_layer* ($27,301 \times 300$)! *dropout* (0.3)! *conv_1* (400 *filters*)! *relu*! *conv_2* (200 *filters*)! *relu*! *flatten*! *dense* (400)! *relu*! *dropout* (0.2)! *dense* (1)! *sigmoid*.

5.4.9. Recurrent Neural Networks (RNN) Model

For the **RNN** model, we used **ANN** with *LSTM* layers, with vocabulary size 20,000 most frequent words from training dataset. We used the dense vector representation for training our models. We use the *max_length* parameter we explained above to tweak our experiments, cut or fill the dense vector representation to make it equal to *max_length*. The first layer of the network is the *Embedding* layer. We experimented with different LSTM architectures based on pre-trained *Embedding* with *GloVe* and with *Random Embedding*. In order to train our model, we used optimizers from Kingma & Ba in [68] and *Stochastic Gradient Descent* with *Momentum*. **Table 4** & **Table 6** in the next section reflect the results of the following architecture with the best results:

input layer (32)! *embedding_layer* ($27,300 \times 300$)! *dropout* (0.4)! *LSTM* (128)! *relu*! *dense* (64)! *relu*! *dropout* (0.5)! *dense* (1)! *sigmoid*.

Table 4. Comparison of classifiers performance in train & test dataset.

Algorithms	Validation Accuracy	Test Accuracy
Lexicon-based Approach	0.703	0.707
Logistic Regression	0.794	0.782
Naïve Bayes	0.785	0.784
Decision Tree	0.769	0.770
Multilayer Perceptron	0.788	0.778
SVM Networks	0.784	0.781
Random Forest	0.783	0.780
XGBoost	0.771	0.770
Convolutional Neural Network (CNN)	0.775	0.772
Recurrent Neural Network (RNN)	0.791	0.792

6. Results & Discussions

After conducting experiments based on the specifications of the experimental part of section 5, our results, looking at the current literature [69], and the results of existing works on the Albanian language that we referred to in section 3, fluctuate at very good levels. In the experiments, we compared the performance of neural networks with the traditional machine learning methods that have been bypassed in recent years such as the simple *Naive Bayes* classifier and *Support Vector Machines (SVM)* which as we mentioned above, we realized with the help of *scikit-learn* library [70]. It is understood that traditional machine learning algorithms may not respond equally to Sentiment Analysis applications. The large volume of data (the given number of 19,300 tweets is not considered a small number) significantly favors neural networks which have more data available for weight optimization.

There are some parameters, the values of which can be modified to give other results. These are the number of *word vector sizes*, the *maximum length* of each tweet, the *number of neurons*, the *number of filters* and the *kernel size* for *CNN* networks, and so on. After many experiments, we found that most of them, within reasonable limits, do not have any particular impact on the result.

Two important issues that impact the performance of a SA system and classification, in general, are **underfitting** and **overfitting**. The *underfitting* occurs when the model is not sufficiently trained or its architecture is not sufficient to learn good parameters that adequately describe the problem. Performance, in this case, will be higher in the test set (without any guarantee of the ability to generalize the model to new data) and will present lower performance in the training set. While there seems to be no problem as the most important evaluation criterion is performance in the test set, it is important to note that a good training system would have a better performance in the training set from the test set, which would cause a shift for the better in the test set.

On the contrary, we have overfitting when the model is over-adapted to the training data, as a result of which it has learned only that data very well and has *weaknesses* in the generalization ability. Thus, its performance seems to be higher in training data and much lower in testing ones.

Overfitting is more common than *underfitting*, due to the nature of *Machine Learning* and it is much more difficult to deal with. Even *CNN* and *RNN*, due to the complexity of their architecture, tend to overfit very often. **Table 4** shows a comparison of the performance of two datasets, and the approximate results show that fairly good training of the models has been done, which is confirmed by the final results of **Table 6**.

Table 5 in next page shows the results of model comparison with the optimal values of the evaluation criteria that emerged for each model with and without *bigrams*, where it is clear that *bigrams* positively affect the performance of the models. In addition to the *unigram* and *bigram* features, the same table also shows the role of *stemming* that slightly increases (about 1%) the *Accuracy* and

F-measure criteria, and significantly the other two measure 4.2% *Balanced Accuracy* and 9.9% *MMC coefficient*".

Finally, **Table 6** shows the optimal performance values of the models according to measures that we explained in section 4.5 above. It should be noted that since the *cross-validation* method has a high computational cost for a large volume of data, we used the *dropout method* which gives very good results [71].

Table 5. Comparison of classifiers that use sparse vector representation with & without bigrams.

Algorithms	Unigrams				Unigrams + Bigrams			
	Accuracy	F-measure (F-score)	Balanced Accuracy	MCC	Accuracy	F-measure (F-score)	Balanced Accuracy	MCC
Naïve Bayes with stemming	0.776	0.870	0.825	0.501	0.784	0.874	0.863	0.589
Naïve Bayes without stemming	0.776	0.870	0.819	0.490	0.782	0.872	0.824	0.512
Logistic Regression	0.772	0.869	0.834	0.512	0.782	0.874	0.874	0.608
Decision Tree	0.767	0.864	0.755	0.354	0.770	0.866	0.775	0.573
Multilayer Perceptron	0.782	0.869	0.773	0.434	0.788	0.875	0.823	0.521
SVM Networks	0.780	0.872	0.824	0.508	0.781	0.872	0.832	0.524
Random Forest	0.780	0.865	0.741	0.401	0.780	0.865	0.744	0.401
XGBoost	0.768	0.856	0.711	0.346	0.770	0.857	0.714	0.353

Table 6. Evaluation of models according to measures of section 4.5.

Algorithms	Accuracy	Sensitivity Recall-TPR	Precision TPR-PPV	Specificity TNR	F-measure (F-score)	Balanced Accuracy	MCC
Lexicon-based Approach	0.707	0.829	0.796	0.362	0.812	0.596	0.161
Logistic Regression	0.782	0.989	0.783	0.759	0.874	0.874	0.608
Naïve Bayes	0.784	0.986	0.786	0.740	0.874	0.863	0.589
Decision Tree	0.770	0.976	0.779	0.573	0.866	0.775	0.394
Multilayer Perceptron	0.778	0.970	0.796	0.676	0.875	0.823	0.521
SVM Networks	0.781	0.980	0.786	0.684	0.872	0.832	0.524
Random Forest	0.780	0.925	0.865	0.563	0.865	0.744	0.401
XGBoost	0.770	0.907	0.812	0.521	0.857	0.714	0.353
Convolutional Neural Network (CNN)	0.772	0.921	0.808	0.535	0.861	0.728	0.366
Recurrent Neural Network (RNN)	0.792	0.983	0.793	0.760	0.878	0.872	0.617

7. Conclusions

In the current work, we try to use the large volume of data (like Twitter data) to reveal useful information, and address the problem of *Sentiment Analysis* in text data and more specifically from tweets in Albanian. In the beginning, we made a review of the latest technologies in this field, as well as similar works with contributions for the Albanian language. Based on *Text Mining techniques*, we have described simple methods for pre-processing text, created *dictionary-based* and *Machine Learning* models, and trained them based on *Twitter* manually annotated data. With the help of these models, we can classify the data into *positive* or *negative*. To be more concrete:

1) We extracted and analyzed features with *Traditional Word Embedding* and *Static Word Embedding* (training from zero *Word2Vec* and *GloVe* models for the Albanian language in *Wikipedia*) and found that the latter gives slightly better results (based on the performance of the models where we used). For the consideration of the *bigram* feature, the results show that they slightly increase the *Accuracy* and *F-measure* criteria (0.3% - 1%), up to 5% the *Balanced Accuracy* criterion, and up to 20% the *MMC coefficient*, which significantly affects the *performance increase* of models.

2) With the help of the Python language and its rich libraries in the field of data science, we have built and experimented with all three categories of “sentiment” classification techniques according to Shi *et al.* in [30], building and analyzing first, *traditional Machine Learning* techniques such as *Logistic Regression*, *Naïve Bayes*, *Decision Tree*, *Multilayer Perceptron*, *SVM Networks*, and *Random Forest*, secondly *Sentiment Lexicon-based* techniques and thirdly *hybrid approach* where we focused more in Deep Learning (DL) methods such as *Convolutional Neural Network (CNN)* and *Recurrent Neural Network (RNN)*.

3) For model evaluation, since they were trained in unbalanced data, we used not only classical evaluation criteria such as *Accuracy*, *Specificity*, *Precision*, and *Recall* but more appropriate criteria such as *F-measure*, *Balanced Accuracy*, and *Matthews Correlation Coefficient (MCC)*.

The results show that *Neural Network-based* models perform slightly better than others, while *Lexicon-based* techniques fall short to others by a margin of 7% - 9% *Accuracy*, 10% - 27% *Balanced Accuracy*, and have a coefficient $MCC = 0.161$, which is so low that it approximates to random models (considered models with $MCC \approx 0.14$). This is probably a consequence because *firstly* in the Albanian language the words of the sentiment lexicon are as much as 1/3 of the English lexicon and *secondly* in the created model, we have not realized the modeling of *negation*.

The best model according to all criteria was *LSTM*-based *RNN* with ***Accuracy* = 79.2%**, ***F-score* = 87.8%**, ***Balanced Accuracy* = 87.2%** and the highest value ***MCC* = 0.617**, followed by *Logistic Regression*, which is no surprise because this model specializes in binary forecasting and this is shown by the fact that this model performs first in ***Balanced Accuracy* = 0.874**, second in *MCC*, which is

very important for our case and also has high results in the other criteria.

▪ Future directions

Proposals for future extensions are summarized as following:

- 1) Unification of works (pre-processing of data, forecasting model) in a unique working tool. Input will be given as a Tweet in raw form, the characteristics will be processed and extracted which will be given as an introduction to the forecast model and the sentiment classification (positive, negative, neutral) will be produced.
- 2) Attempts to utilize more feature extraction techniques and text pre-processing.
- 3) Attempts to design and implement more complex architectures for forecasting models (combination of state-of-the-art techniques, use of hybrid techniques).
- 4) Collaboration with other scientific fields, such as psychology, for further deepening in the way a person expressing, which brings improvement in the quality of the extracted features.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

- [1] Chandler, J. D., Salvador, R. and Kim, Y. (2018) Language, Brand and Speech Acts on Twitter. *Journal of Product and Brand Management*, **27**, 375-384. <https://doi.org/10.1108/JPBM-06-2017-1493>
- [2] Liu, B. and Zhang, L. (2012) A Survey of Opinion Mining and Sentiment Analysis. In: *Mining Text Data*, Springer, Boston, 415-463. https://doi.org/10.1007/978-1-4614-3223-4_13
- [3] Arora, M. and Kansal, V. (2019) Character Level Embedding with Deep Convolutional Neural Network for Text Normalization of Unstructured Data for Twitter Sentiment Analysis. *Social Network Analysis and Mining*, **9**, Article No. 12. <https://doi.org/10.1007/s13278-019-0557-y>
- [4] Goularas, D. and Kamis, S. (2019) Evaluation of Deep Learning Techniques in Sentiment Analysis from Twitter Data. *2019 International Conference on Deep Learning and Machine Learning in Emerging Applications (Deep-ML)*, Istanbul, 26-28 August 2019, 12-17. <https://doi.org/10.1109/Deep-ML.2019.00011>
- [5] Jose, R. and Chooralil, V.S. (2016) Prediction of Election Result by Enhanced Sentiment Analysis on Twitter Data Using Classifier Ensemble Approach. *2016 International Conference on Data Mining and Advanced Computing (SAPIENCE)*, Ernakulam, 16-18 March 2016, 64-67. <https://doi.org/10.1109/SAPIENCE.2016.7684133>
- [6] Kolchyna, O., Souza, T.T., Treleaven, P. and Aste, T. (2015) Twitter Sentiment Analysis: Lexicon Method, Machine Learning Method and Their Combination. In: Mitra, G. and Yu, X., Eds., *Handbook of Sentiment Analysis in Finance*, OptiRisk Systems Ltd, Uxbridge, arXiv: 1507.00955.
- [7] Gupta, I. and Joshi, N. (2020) Enhanced Twitter Sentiment Analysis Using Hybrid Approach and by Accounting Local Contextual Semantic. *Journal of Intelligent*

- Systems*, **29**, 1611-1625. <https://doi.org/10.1515/jisys-2019-0106>
- [8] Hassonah, M.A., Al-Sayyed, R., Rodan, A., Al-Zoubi, A.M., Aljarah, I. and Faris, H. (2020) An Efficient Hybrid Filter and Evolutionary Wrapper Approach for Sentiment Analysis of Various Topics on Twitter. *Knowledge-Based Systems*, **192**, Article ID: 105353. <https://doi.org/10.1016/j.knosys.2019.105353>
 - [9] Vaitheeswaran, G. and Arockiam, L. (2016) Hybrid Based Approach to Enhance the Accuracy of Sentiment Analysis on Tweets. *International Journal of Computer Science & Engineering Technology*, **6**, 185-190.
 - [10] Pak, A. and Paroubek, P. (2010) Twitter as a Corpus for Sentiment Analysis and Opinion Mining. *Proceedings of The International Conference on Language Resources and Evaluation Conference*, Malta, 17-23 May 2010, 1320-1326.
 - [11] Alahmary, R.M., Al-Dossari, H. and Emam, A.Z. (2019) Sentiment Analysis of Saudi Dialect Using Deep Learning Techniques. 2019 *International Conference on Electronics, Information, and Communication (ICEIC)*, Auckland, 22-25 January 2019, 1-6. <https://doi.org/10.23919/ELINFOCOM.2019.8706408>
 - [12] Brum, H., Araújo, F. and Kepler, F. (2016) Sentiment Analysis for Brazilian Portuguese over a Skewed Class Corpora. *International Conference on Computational Processing of the Portuguese Language (PROPOR 2016)*, Tomar, 13-15 July, 134-138. https://doi.org/10.1007/978-3-319-41552-9_14
 - [13] Duwairi, R., Ahmed, N.A. and Al-Rifai, S.Y. (2015) Detecting Sentiment Embedded in Arabic Social Media—A Lexicon-Based Approach. *Journal of Intelligent & Fuzzy Systems*, **29**, 107-117. <https://doi.org/10.3233/IFS-151574>
 - [14] Madan, A. and Ghose, U. (2021) Sentiment Analysis for Twitter Data in the Hindi Language. 2021 *11th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*, Noida, 28-29 January 2021, 784-789. <https://doi.org/10.1109/Confluence51648.2021.9377142>
 - [15] Ochoa-Luna, J. and Ari, D. (2019) Word Embeddings and Deep Learning for Spanish Twitter Sentiment Analysis. *Annual International Symposium on Information Management and Big Data*, Lima, 3-5 September 2018, 19-31. https://doi.org/10.1007/978-3-030-11680-4_4
 - [16] Soumya, S. and Pramod, K.V. (2020) Sentiment Analysis of Malayalam Tweets Using Machine Learning Techniques. *ICT Express*, **6**, 300-305. <https://doi.org/10.1016/j.icte.2020.04.003>
 - [17] Skenduli, M.P., Biba, M., Loglisci, C., Ceci, M. and Malerba, D. (2018) User-Emotion Detection Through Sentence-Based Classification Using Deep Learning: A Case-Study with Microblogs in Albanian. *International Symposium on Methodologies for Intelligent Systems*, Limassol, 29-31 October 2018, 258-267. https://doi.org/10.1007/978-3-030-01851-1_25
 - [18] Mozetič, I., Grčar, M. and Smailović, J. (2016) Multilingual Twitter Sentiment Classification: The Role of Human Annotators. *PLoS ONE*, **11**, e0155036. <https://doi.org/10.1371/journal.pone.0155036>
 - [19] Chen, Y. and Skiena, S. (2014) Building Sentiment Lexicons for All Major Languages. *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, Baltimore, 23-25 June 2014, 383-389. <https://doi.org/10.3115/v1/P14-2063>
 - [20] Liu, B. (2015) Sentiment Analysis: Mining Opinions, Sentiments, and Emotions. Cambridge University Press, New York. <https://doi.org/10.1017/CBO9781139084789>
 - [21] Ravi, K. and Ravi, V. (2015) A Survey on Opinion Mining and Sentiment Analysis:

- Tasks, Approaches and Applications. *Knowledge-Based Systems*, **89**, 14-46. <https://doi.org/10.1016/j.knosys.2015.06.015>
- [22] Jagtap, V.S. and Pawar, K. (2013) Analysis of Different Approaches to Sentence-Level Sentiment Classification. *International Journal of Scientific Engineering and Technology*, **2**, 164-170.
- [23] Wang, H., Liu, B., Li, C., Yang, Y. and Li, T. (2019) Learning with Noisy Labels for Sentence-Level Sentiment Classification. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language & Proceedings of the 9th International Joint Conference on Natural Language Processing*, Hong Kong, November 2019, 6286-6292. <https://doi.org/10.18653/v1/D19-1655>
- [24] Pang, B. and Lee, L. (2008) 4.1.2 Subjectivity Detection and Opinion Identification. In: de Rijke, M., Liu, Y. and Kelly, D., Eds., *Opinion Mining and Sentiment Analysis*, Now Publishers Inc., Delft, 1-135.
- [25] Balbi, S., Misuraca, M. and Scepi, G. (2018) Combining Different Evaluation Systems on Social Media for Measuring User Satisfaction. *Information Processing & Management*, **54**, 674-685. <https://doi.org/10.1016/j.ipm.2018.04.009>
- [26] Fronzetti Colladon, A. (2018) The Semantic Brand Score. *Journal of Business Research*, **88**, 150-160. <https://doi.org/10.1016/j.jbusres.2018.03.026>
- [27] Gloor, P.A. (2017) Sociometrics and Human Relationships: Analyzing Social Networks to Manage Brands, Predict Trends, and Improve Organizational Performance. Emerald Publishing Limited, London. <https://doi.org/10.1108/9781787141124>
- [28] Jeong, B., Yoon, J. and Lee, J.M. (2019) Social Media Mining for Product Planning: A Product Opportunity Mining Approach Based on Topic Modeling and Sentiment Analysis. *International Journal of Information Management*, **48**, 280-290. <https://doi.org/10.1016/j.ijinfomgt.2017.09.009>
- [29] Karami, A., Lundy, M., Webb, F. and Dwivedi, Y.K. (2020) Twitter and Research: A Systematic Literature Review through Text Mining. *IEEE Access*, **8**, 67698-67717. <https://doi.org/10.1109/ACCESS.2020.2983656>
- [30] Medhat, W., Hassan, A. and Korashy, H. (2014) Sentiment Analysis Algorithms and Applications: A Survey. *Ain Shams Engineering Journal*, **5**, 1093-1113. <https://doi.org/10.1016/j.asej.2014.04.011>
- [31] Shi, Y., Zhu, L., Li, W., Guo, K. and Zheng, Y. (2019) Survey on Classic and Latest Textual Sentiment Analysis Articles and Techniques. *International Journal of Information Technology & Decision Making*, **18**, 1243-1287. <https://doi.org/10.1142/S0219622019300015>
- [32] Biba, M. and Mane, M. (2014) Sentiment Analysis through Machine Learning: An Experimental Evaluation for Albanian. In: Thampi, S., Abraham, A., Pal, S. and Rodriguez, J., Eds., *Recent Advances in Intelligent Informatics*, Springer International Publishing, Cham, 195-203. https://doi.org/10.1007/978-3-319-01778-5_20
- [33] Kadriu, A. and Abazi, L. (2017) A Comparison of Algorithms for Text Classification of Albanian News Articles. *Entrenova*, **3**, 62-68.
- [34] Kote, N., Biba, M. and Trandafil, E. (2018) An Experimental Evaluation of Algorithms for Opinion Mining in Multi-Domain Corpus in Albanian. *International Symposium on Methodologies for Intelligent Systems*, Limassol, 29-31 October 2018, 439-447. https://doi.org/10.1007/978-3-030-01851-1_42
- [35] Trandafil, E., Kote, N. and Biba, M. (2018) Performance Evaluation of Text Categorization Algorithms Using an Albanian Corpus. *International Conference on Emerging Internetworking, Data & Web Technologies*, Tirana, 15-17 March 2018,

- 537-547. https://doi.org/10.1007/978-3-319-75928-9_48
- [36] Kadriu, A., Abazi, L. and Abazi, H. (2019) Albanian Text Classification: Bag of Words Model and Word Analogies. *Business Systems Research Journal*, **10**, 74-87. <https://doi.org/10.2478/bsrj-2019-0006>
 - [37] Anxhiu, M. (2019) Language Challenges in Aspect-Based Sentiment Analysis: A Review of Albanian Language. *Knowledge—International Journal*, **31**, 1709-1712. <https://ikm.mk/ojs/index.php/KIJ/article/view/1336>
 - [38] Tankovska, H. (2021) Twitter: Number of Monthly Active Users 2010-2019. <https://www.statista.com/statistics/282087/number-of-monthly-active-twitter-users/>
 - [39] Vasili, R., Xhina, E., Ninka, I. and Souliotis, T. (2018) A Comparative Review of Text Mining & Related Technologies. *RTA-CSIT*, Tirana, November 23-24, 2018, 1-10.
 - [40] Wang, S. and Manning, C. D. (2012) Baselines and Bigrams: Simple, Good Sentiment and Topic Classification. *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, Jeju, 8-14 July 2012, 90-94.
 - [41] Selva Birunda, S. and Kanniga Devi, R. (2021) A Review on Word Embedding Techniques for Text Classification. In: Raj, J.S., Ilyasu, A.M., Bestak, R. and Baig, Z.A., Eds., *Innovative Data Communication Technologies and Application*, Springer, Singapore, 267-281. https://doi.org/10.1007/978-981-15-9651-3_23
 - [42] Mikolov, T., Corrado, G., Chen, K. and Dean, J. (2013) Efficient Estimation of Word Representations in Vector Space. *Proceedings of the Workshop at ICLR*, Scottsdale, 2-4 May 2013, 1-12.
 - [43] Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S. and Dean, J. (2013) Distributed Representations of Words and Phrases and their Compositionality. In: Burges, C., Bottou, L., Welling, M., Ghahramani, Z. and Weinberger, K., Eds., *Advances in Neural Information Processing Systems*, Vol. 26, Curran Associates, Inc., Red Hook, 3111-3119.
 - [44] Pennington, J., Socher, R. and Manning, C. (2014) GloVe: Global Vectors for Word Representation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, 25-29 October 2014, 1532-1543. <https://doi.org/10.3115/v1/D14-1162>
 - [45] Shi, T. and Liu, Z. (2014) Linking GloVe with word2vec. <https://arxiv.org/abs/1411.5595v2>
 - [46] Rong, X. (2014) Word2Vec Parameter Learning Explained. arxiv: 1411.2738.
 - [47] Agarwal, B. and Mittal, N. (2015) Semantic Orientation-Based Approach for Sentiment Analysis. In: Agarwal, B. and Mittal, N., Eds., *Prominent Feature Extraction for Sentiment Analysis*, Springer International Publishing, Cham, 77-88. https://doi.org/10.1007/978-3-319-25343-5_6
 - [48] Hailong, Z., Wenyan, G. and Bo, J. (2014) Machine Learning and Lexicon-Based Methods for Sentiment Classification: A Survey. 2014 11th *Web Information System and Application Conference*, Tianjin, 12-14 September 2014, 262-265. <https://doi.org/10.1109/WISA.2014.55>
 - [49] Miller, G.A. (1995) WordNet: A Lexical Database for English. *Communications of the ACM*, **38**, 39-41. <https://doi.org/10.1145/219717.219748>
 - [50] Hu, M. and Liu, B. (2004) Mining and Summarizing Customer Reviews. *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, Seattle, 22-25 August 2004, 168-177. <https://doi.org/10.1145/1014052.1014073>
 - [51] Park, S. and Kim, Y. (2016) Building Thesaurus Lexicon Using Dictionary-Based

- Approach for Sentiment Classification. 2016 *IEEE 14th International Conference on Software Engineering Research, Management and Applications (SERA)*, Towson, 8-10 June 2016, 39-44. <https://doi.org/10.1109/SERA.2016.7516126>
- [52] Mitchell, T. (1997) Machine Learning. McGraw Hill, New York.
- [53] Alpaydin, E. (2020) Introduction to Machine Learning. 4th Edition, MIT Press Academic, Cambridge.
- [54] Jurafsky, D. and Martin, J.H. (2019) Logistic Regression. In: *Speech and Language Processing*, 3rd Edition (Draft), 75-93. https://web.stanford.edu/~jurafsky/slp3/ed3book_dec302020.pdf
- [55] Jurafsky, D. and Martin, J.H. (2019) Naive Bayes and Sentiment Classification. In *Speech and Language Processing*, 3rd Edition (Draft), 56-74.
- [56] Sun, L., Fu, S. and Wang, F. (2019) Decision Tree SVM Model with Fisher Feature Selection for Speech Emotion Recognition. *EURASIP Journal on Audio, Speech, and Music Processing*, **2019**, Article No. 2. <https://doi.org/10.1186/s13636-018-0145-5>
- [57] Jurafsky, D. and Martin, J.H. (2019) Neural Networks and Neural Language Models. In: *Speech and Language Processing*, 3rd Edition (Draft), 123-142.
- [58] Farzi, R. and Bolandi, V. (2016) Estimation of Organic Facies Using Ensemble Methods in Comparison with Conventional Intelligent Approaches: A Case Study of the South Pars Gas Field, Persian Gulf, Iran. *Modeling Earth Systems and Environment*, **2**, Article No. 105. <https://doi.org/10.1007/s40808-016-0165-z>
- [59] Vo, Q.-H., Nguyen, H.-T., Le, B. and Nguyen, M.-L. (2017) Multi-channel LSTM-CNN Model for Vietnamese Sentiment Analysis. 2017 *9th International Conference on Knowledge and Systems Engineering (KSE)*, Hue, 19-21 October 2017, 24-29. <https://doi.org/10.1109/KSE.2017.8119429>
- [60] Jurafsky, D. and Martin, J.H. (2019) Sequence Processing with Recurrent Networks. In: *Speech and Language Processing*, 3rd Edition (Draft), 169-190.
- [61] Sutskever, I., Martens, J. and Hinton, G.E. (2011) Generating Text with Recurrent Neural Networks. *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, Bellevue, 28 June-2 July 2011, 1017-1024.
- [62] Hochreiter, S. and Schmidhuber, J. (1997) Long Short-Term Memory. *Neural Computing*, **9**, 1735-1780. <https://doi.org/10.1162/neco.1997.9.8.1735>
- [63] Ko, C.-R. and Chang, H.-T. (2021) LSTM-Based Sentiment Analysis for Stock Price Forecast. *PeerJ Computer Science*, **7**, e408. <https://doi.org/10.7717/peerj-cs.408>
- [64] Zhang, H., Gan, W. and Jiang, B. (2014) Machine Learning and Lexicon-Based Methods for Sentiment Classification: A Survey. *11th Web Information System and Application Conference*, Tianjin, 12-14 September 2014, 262-265. <https://doi.org/10.1109/WISA.2014.55>
- [65] Chicco, D. and Jurman, G. (2020) The Advantages of the Matthews Correlation Coefficient (MCC) over F1 Score and Accuracy in Binary Classification Evaluation. *BMC Genomics*, **21**, Article No. 6. <https://doi.org/10.1186/s12864-019-6413-7>
- [66] Vaghela, V.B. and Jadav, B.M. (2016) Analysis of Various Sentiment Classification Techniques. *International Journal of Computer Applications*, **140**, 22-27. <https://doi.org/10.5120/ijca2016909259>
- [67] Sadiku, J. and Biba, M. (2012) Automatic Stemming of Albanian through a Rule-Based Approach. *Journal of International Scientific Publications: Language, Individual Society*, **6**, 173-190.
- [68] Kingma, D.P. and Ba, J. (2014) Adam: A Method for Stochastic Optimization.

<http://arxiv.org/abs/1412.6980>

- [69] Ma, S., Sun, X., Lin, J. and Ren, X. (2018, July) A Hierarchical End-to-End Model for Jointly Improving Text Summarization and Sentiment Classification. *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI-18)*, Stockholm, 9-19 July 2018, 4251-4257. <https://doi.org/10.24963/ijcai.2018/591>
- [70] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., et al. (2011) Scikit-Learn: Machine Learning in Python. *Journal of Machine Learning Research*, **12**, 2825-2830.
- [71] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. and Salakhutdinov, R. (2014) Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *The Journal of Machine Learning Research*, **15**, 1929-1958.