DOKUZ EYLÜL UNIVERSITY

ENGINEERING FACULTY

DEPARTMENT OF COMPUTER ENGINEERING

SENTIMENTAL ANALYSIS ON TWITTER

by

Cem SİNAN

Advisor

Research Assistant. Dr. Yunus DOĞAN

December, 2020

İZMİR

SENTIMENTAL ANALYSIS ON TWITTER

A Thesis Submitted to the

Dokuz Eylül University, Department of Computer Engineering

In Partial Fulfillment of the Requirements for the Degree of B.Sc.

by

Cem SİNAN

Advisor

Research Assistant. Dr. Yunus DOĞAN

December, 2020

İZMİR

SENIOR PROJECT EXAMINATION RESULT FORM

We have read the thesis entitled **“SENTIMENTAL ANALYSIS ON TWITTER”** completed by **Cem SİNAN** under advisor of **Dr. Yunus DOĞAN** and we certify that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of B.Sc.

Advisor

Committee Member

Committee Member

Prof. Dr. Yalçın ÇEBİ

Chair

Department of Computer Engineering

ACKNOWLEDGEMENTS

Thanks to my family, my advisor, friends and to those kind people who put free information on online.

Cem SİNAN

SENTIMENTAL ANALYSIS ON TWITTER

ABSTRACT

As time pass by we, humans, delve into virtual reality more. Regarding the pandemic, past year was a hard time for everybody in the world and it limited face to face conversation due to transmission of virus. This made us use social media more, so we could imitate the socialization part. Consequently, social media platform interactions got more importance. Twitter is one of them, which used by many people ranging from high schoolers to government officials to CEOs of firms. This many usages lead to great possibilities for data mining and processing.

In this project, the aim was to use Turkish Tweets to analyze opinions or sentiments about Coronavirus from the Tweets. To do that, data mining, natural language processing and finally deep learning architectures were decided to do. To mine Tweets, Twitter API, to quantify words, Turkish ANEW corpus, to train neural networks LSTM architecture were used. The experiment is finished as it was planned but the experiment is done for one limited dataset, which gave unconclusive results after training.

TWİTTER ÜZERİNDEN DUYGU ANALİZİ

ÖZET

Zaman ilerledikçe insanlığın kendi yarattığı yapay dünyalarda yaşama süresi arttı. Bu artış pandeminin tüm dünyayı vurmasıyla katlandı ve hepimizi diğerlerinden uzak durmaya itti. Bu zaman boyunca insanların kendi fikirlerini anlatabilmesi ve sosyalleşme hissine kavuşabilmesi için Twitter gibi sosyal medya platformları fazlasıyla kullanıldı. Bunun sonucunda insanların hisleri ve fikirlerini analiz edebilmemiz için uygun bir ortam oluştu. Bu projede de bu boşluğu doldurmak amacıyla yöntemler denenmiştir.

Bu proje yapay zekanın alt dalları olan derin öğrenme, doğal dil işleme, veri madenciliği gibi disiplinleri içermektedir. Tweet'lerin ulaşılmasında Twitter API "korona, korona virüs" gibi filtrelerle daraltılıp kaydedilmiş, bu Tweetlerin doğal dil işlenmesinde kullanılabilmesi için ANEW'in Türkçe sözlüğü ve derin öğrenme modeli olarak ise LSTM mimarisi tercih edilmiştir. Deneyin sonucunda daha fazla verinin ihtiyaç duyulduğu anlaşılmış, yapılan değerlendirmeler düşük bir hata payı vermemiştir, genel trendi takip edebilse de ortalamada kalan bir tahminleme olmuştur.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER ONE

INTRODUCTION

1.1 Background Information

Humanity has always been using communication or information sharing as one of its primary tools for surviving in the wild. From the start of the first human tribal groups to the first civilizations sending messages which are sending the information was crucial. Humans have used information as searching for new food sources, creating alliances, and sharing items. Having said that, pure information has to be supported with non-verbal features such as body language and tone, rhythm even pauses of voice to convey the message. In these times with the creation of the internet, which is a pool of information created by people, and social media, the spread of information got incredibly faster and easier to use. One of the biggest social media platforms for information trade is Twitter.

Twitter is a platform on the internet which lets people share their thoughts and ideas online with others via messages called “tweets”. Tweets are 280 characters long in most of the languages, additionally 140 seconds long videos and pictures can be shared as well. Twitter Inc. was founded in 2006 and now monthly 353 million people actively use Twitter. In Turkey 13.45 million active users use Twitter, which puts Turkey in 6th rank. (IQBAL, 2020)

1.2 Problem Definition

Twitter can be used in many different ways, consumers might comment on a new product, news can be published, interaction or limited debates can happen between political figures, or users can share their ideas on a topic grouped by “hashtags (#)”. One of the problems with pure text-based communication is, it destroys the “natural” way of communicating which is face-to-face. Either sender or receiver, sentiment of the message can be misleading or hard to transmit it online. Sentimental analysis of messages can help both sender and receiver to convey their message without errors which might happen in the transfer.

In the past ten years, studies regarding Turkish language processing and text mining, has been growing. As with more studies are being done, it shows both improvements in the field as well as shortcomings. This leads researchers to have more opportunities to put more research as the field of natural language processing of Turkish language grows. As many as there are studies, there is much to be done to satisfy both business and academia.

1.3 Motivation/Related Works

Companies, political parties, famous figures, news organizations and individuals also use Twitter to get a feedback from their users/readers. These feedbacks can be used to improve the contents, policies, products without questionnaires, by using data mining. More precisely, sentimental analysis by using neural networks, deep learning, and natural language processing.

Natural language processing (NLP) is a subfield of computing and language science, which aims to make use of human languages by teaching or just bare processing by computers. Since the start of the artificial intelligence research, NLP also become an important part of the research, the first studies worked around symbolic NLP, meanwhile after 1990's statistical NLP become more mainstream due to abundant data sources. Neural networks also can be used in this field, because of the feature problems in statistical NLP.

Deep learning is one of artificial intelligence subfield which aims to build abstraction between layers of artificial neural networks. It is also called deep structured learning. Deep learning can be used in fields such as speech recognition, image processing, medical fields (bioinformatics), military and natural language processing. In this project, deep learning will be used in natural language processing.

Sentimental analysis can be defined as getting an emotion from a sentence. Getting an emotion from a sentence can be very subjective, and emotion of a writer cannot be seen by the mimics. This creates communicational problems. Sentimental analysis can be used to reflect the probable reaction of readers, which can help with feedback of mimics on digital world.

NLP is to some extent new for Turkish language. Most definitive studies started in 2010's and it has grown since. Numerous researchers have put conference papers, articles related to Turkish language for NLP and sentimental analysis on Twitter specifically.

Language processing, or text mining studies have done for Turkish language. Many of the studies focused on the machine learning and data mining algorithms for processing data and classification so the processed data can be trained. These studies mostly proved that the success rate of the experiments is between 62% and %79, which is likely successful. However, many studies are also claimed that preprocessing of raw data and classifications can be improved.

1.4 Goal/Contribution

As this paper is being written, the world is having global scale pandemic, which makes us to stay at homes or be insulated from other people, in order to lower the transmission of the virus. Therefore, the need of socialization and public opinion is making through social media outlets such as Twitter. People often express their ideas, frustration or demands to the officials and companies through Twitter, which creates a worthy source of data for sentimental analysis.

The goal is to compare the emotions of users before, after and peak of pandemic via Twitter data. This goal can be done by deep learning and natural language processing algorithms, and emotional scales such as valence, arousal and dominance dictionaries.

1.5 Project Scope

The possibility of text mining is close to limitless, many inductive studies can be done, and it can be materialized with the support of social sciences such as psychology, sociology, and marketing. As it is mentioned in problem definition (1.2) the scope of this project is to compare and deduce the output of sentimental analysis from Twitter in the times of virus pandemic. The different scale of emotions in the norms of valence, arousal and dominance can be a metric.

The scope of the project is also limited by Twitter data. Twitter is a fine source for text mining for sentimental analysis due to character limitation and variance in emotions. Data will be gathered from Twitter by Twitter API. In order to process the data, pre-processing has to be done.

1.6 Methodology/Tools/Libraries

Methodology of this project is scientific theory, which is namely identify the problem, gather data, hypothesis, experiment the hypothesis and check if the experiment and hypothesis matches/coincides. In this project, the problem is to fill the lack of sentimental analysis research by creating a hypothesis which is "What are the most dominant emotion in the times of virus pandemic?". Thus, experiment is this project, which is sentimental analysis from Tweets of real people. Eventually, methodology of this project is scientific theory, but more emphasis is on the experiment.

Tools for this project are mainly Twitter API, programming languages for processing Twitter data such as Python or Java, programming languages for machine learning, deep learning and natural language processing, Python, R, and perhaps JavaScript for in case of API related matters.

There are three frameworks that are published for text mining, Zemberek, İTÜ Doğal Dil İşleme Yazılım Zincir, Lucene and Nlptoolkit framework.

Python has many libraries for jobs mentioned above such as: NLTK, Scikit-learn, NumPy, PyTorch and TensorFlow, for R language MXNetR, darch, deepnet, H2O and deepr.

CHAPTER 2

LITERATURE REVIEW

In this chapter relative literature research will be presented. This project involves in disciplines such as natural language processing, data science and statistical inference, machine learning, deep learning and project's tool related programming fields (Java, Python frameworks and libraries). The research is done via Google Scholar by searching keywords such as "sentimental analysis", "sentimental analysis in twitter", "sentimental analysis using machine learning algorithms" and "sentimental analysis using deep learning algorithms". Since, this project is being done in Turkish language, same searches with Turkish translation has also been made. Most of the research regarding sentimental analysis has done in English language, which has been the playfield of first natural language processing since Dartmouth Conference in 1956. Thus, studies done in English language might provide good insights for this project. However, keeping in the mind that Turkish language and English language is different regarding morphology, new word creation from stem and case dependency.

Turkish language is a member of Ural-Altai language family, Altaic branch and Oghuz group, uses Latin alphabet with 29 letters, 8 letters are being vowels and 21 letters are being consonants. Since Turkish is a highly agglutinative language, inflectional and derivational processes are highly morphologically complex. Moreover, the semantic of the word can be changed via adding suffixes and Turkish language does not have a perfect grammatical order or gender cases. (Mansur Alp Toçoğlu, 2019)

Linguists divided the language into four subjects in order to work with language processing methods. These subjects are phonology, morphology, syntax, and semantics. Turkish language is a rich language, but it creates problems for processing compared to English language due to its own nature. Natural language processing works on subjects such as correcting spelling mistakes, text-to-speech creation, speech-to-text creation, question answering, machine translation, accessibility features and more. Development of these tools altogether helps with the development of the language's processability. Especially languages like Turkish (agglutinative languages) needs different processing from English. Main difference between the languages lies in the suffices and its effect on the meaning of the new word. For spelling correction, some specific rules need to be checked such as sound harmony rules (ses uyum kuralları), syllable structure (hece yapısı), and harmony of attachments (eklerin uyumu). Spell checking is a useful tool for sentimental analysis. Grammatical rules are important aspect of the natural language processing. For Turkish language, fundamentally there are two sections of words; noun bases and verb bases, these sections also have their subsections. (Adalı, 2013)

The importance of harmony rules, syllable structures and harmony of attachments as mentioned above, the tools for natural language processing got higher importance. Operations such as Turkish language translator which changes the letters like “ç”, “ö” and “ğ” to similar English ones then rechanging to Turkish as mentioned in the paper as asciifier/deasciifier, and other tools such as tokenizers/sentence splitter, spell checkers, morphological anaylzer/diasmbiguator, named entity recognizer or dependency parser will be very useful to improve the development of Turkish language processing. (Gökçe Uludoğam, 2019)

Sentimental analysis can also be seen as opinion mining, a subfield of data/text mining discipline. Initial works were done with sentimental polarity which was to determine if a text was positive or negative. S. E. Şeker (2016) in his report paper separates sentimental analysis into three layers called; observation/feature based opinion mining, frequency/relation based opinion mining, and model based opinion mining. Meanwhile the first one delves more into human based analysis, the latter two uses common tools such as Part of Speech (POS)-Tagging, Hidden Markov Models (HMM), Conditional Random Fields (CRF), Artificial Neural Networks (ANN), Bayesian Networks (BN) and Latent Dirichlet Allocation (LDA). Common difficulties in sentimental analysis are sarcastic entries, dependency and knowledge on the domain, subjectivity of the entry, and lastly negation computation in Turkish language. S. E. Şeker shows operations necessary preprocessing of data for sentimental analysis as Term Frequency- Inverse Document Frequency (TF-IDF), word to vector (Word2Vec), standard scaling (StandardScaler), normalization and Principal Component Analysis (PCA). He differentiates the two common ways to work with sentimental analysis; using natural language processing discipline by morphological, syntax and semantic and using statistics on the text. (Şeker, 2016) In the paper he notes that regarding high change of speed on opinions statistics could be a better tool due to higher computational speed compared to natural language processing tools. (Hinrich Schuetze, 1999) (James H. Martin, 2008) Other related works done in English language showed that by using stemming, n-gram, and term frequency techniques the success rate which calculated by F1-score, can reach between 86% and 97%. (Padmini Srinivasan, 2011) For Turkish language a work done on comments for film critiques has showed success rate of 83%. (Alaettin Uçan, 2014)

Text mining works with machine learning and deep learning methods. This section will present machine learning related literatures. In this paper for English language the researchers used three classifications which are positive, negative and neutral. As classification methods, two-step classifiers used which are Naïve-Bayes and Maximum entropy classifiers. The common problem mentioned as limited size, and cryptic style of writing. From previous research, Naïve-Bayes is found the most helpful classifier, alongside with bigrams and negation words such as "no" and "not". Part-of-Speech tagging is not to be found helpful. For dataset, Twitter Sentiment Corpus and Stanford Twitter is used. The researches experimented with preprocessing steps such as punctuations (".","…","!","?"), emotions (emojis such as ":)" or ":("), Twitter specific terms (hashtags "#" and handles "@") although these terms are still helpful with analysis, these changed into regular expressions such as; "#" → "#(\w+)" and to "HASH_\1" and finally stemming. After preprocessing done on these factors, on both datasets 46% of the words have been eliminated. One of the most problematic part for sentimental analysis is labelled datasets for training or testing, preprocessing was needed to use machine learning algorithms. This paper concluded on best accuracy is done with unigrams + bigrams + trigrams + negations trained on Naïve Bayes classifier with the rate of 75.33%, also writers note that Support Vector Machines (SVM) do not have any findings on helping with classification. (Yogesh Gard, 2014)

In a Turkish language sentimental analysis paper, the writers used three target features as positive, negative and neutral, with techniques such as dictionary model and n-gram model. They have gathered three datasets with 500, 1200 and 5100 Tweets and used as a training set by two human supervising. For preprocessing Twitter terms, punctuations were deleted or changed with empty space. Dictionary was chosen as Turkish Language Institution (TDK), with 528 positive, 738 negative words, 100 reinforcement adjectives (pekiştirme sıfatları ve ikilemeler) and 10 jargon words. In the project, sentences were divided in words which provide either negative or positive classification, then counted and result was given as if positive words are more than negative words it has positive label, vice-versa for negative words and if the amount of positive and negative is equal, then the sentence has a neutral label. More specifically, if a label to be positive it needs to be between minus three and positive infinity, [-3, +∞], negative label needs to be between minus five and negative infinity [-∞, -5] and for neutral label it needs to be between minus five and minus three [-5, -3]. Having said that some specific rules such as classifying cursing words, also help with the classification by 5% to 10%. Although languages are similar to human consciousness, could be very hard to interpret but certain rules help with classification. By using 2, 3, and 4-grams with the labelled words, the frequency of these words were calculated and found as the optimal one is 3-grams. However, it has found that frequencies are mostly in the negative side which is the reason of the ranges of positive, negative, and neutral labelling. As a conclusion success rates, which is calculated as F-score, for dictionary and character-based n-grams are found to be 70% and 69% respectively. (E.S Akgül, 2016)

Another work for sentimental analysis in Turkish language is done with different classifiers and different n-grams with the dataset being Twitter tweets. Classifiers are Naïve-Bayes (NB), Support Vector Machines (SVM) and logistic regression (LR), and for text representation 1-gram, 2-gram and 3-gram and combinations of n-grams were used. As a result of this experiment were scored by three different measures which are success rate, F-scoring and AUC scoring. The highest score was 1-gram + 2-gram with NB with 77.78% of success rate and 0.79 of F-score and 0.85 of AUC score, the lowest were 3-gram with SVM with 59.75% of success rate, 3-gram with NB F-score with 0.58, and 3-gram with SVM with AUC with 0.60. The writers noted that lack of POS-tagging tools for Turkish language and if there were, POS-tagging would have also been tried. (Onan, Sentimental Analysis on Twitter Messages Based on Machine Learning Methods, 2017)

Feature selection needs to be differentiable in the dataset so the algorithm and performance of classifications could be better. In their work, for these purposes a correlation based method named CfsSubset algorithm was used, additionally Bag-of-Words (BoW) and N-grams were used for simplifying representations; kNN, SVM, NB and Multinom NB is used for classification. For 10000 samples were positive labelled, 10000 negative labelled 20000 tweet samples were used as a dataset. Labels for positive and negative were decided by emojis such as “ :) , :-) , =) , :D , :(, :-(, =(, ;(“. After preprocessing is done, with eliminating unwanted characters and words, stemming, weighting the terms (TF, Boolean, TF-IDF) feature selection and classification operations were started. One of the trivial findings that I want to mention in this work is to most common positive and negative words in Turkish. Some thirty of them are “tatil”, “iyi”, “ol”, “gel”, “günaydın”, “yok”, “gün”, “et”, “gece”, “ilk”, “güzel”, “okul”, “yap”, “git”, “sev”, “bak”, “başla”, “al”, “bil”, “yaz”, “sabah”, “uyku”, “gör”, “yeni”, “sen”, “iste”, “iş”, “zaman”, “ulan”, “kal”. As a conclusion their findings were N-grams gave better results than BoW, and 3-gram was the best one. The best success rate was Multinom NB with 3-gram with 66.06%, and the worst one is BoW with NB with 59.86% (Ö. Çoban, 2015).

Since many machine learning based sentimental analysis experiments have been done, some of previously done experiments in Turkish language for sentimental analysis using Twitter data are:

- Nizam and Akın (2014) created two biased and balanced datasets including tweets, and used Naïve-Bayes, Random Forests (RF), SVM and C4.5 decision tree classifiers.
- Meral and Diri (2014) experimented twitter messages on emotional analysis with word based 2 and 3-grams, using Naïve-Bayes, RF, SVM classifiers and found successful classification rate of 89.5%.
- Çoban et al. (2015) used Bag-of-Words (BoW) and n-grams for word representation, Naïve-Bayes and K-Nearest Neighbor (kNN) algorithms for classification into two categories; positive and negative sentiment. Accuracy rate was 66.06% for 14,777 tweets.
- Türkmen and Cemgil (2015) used Turkish tweets related to Gezi Park Protests

and tried to get an analysis on the political tendencies, statistical method of chi-square and RF algorithm was used.

- Akgül et al. (2016) developed an application to analyze Twitter data, in this application dictionary based, 2, 3, 4-grams used as a word representation.

Deep learning algorithms and architectures are commonly used in natural language processing. It has found out that these algorithms are highly configurable depending on the demands of the task. Some of deep learning models and its explanations are given below:

- Deep Neural Networks (DNN): can be used in different domains, downside is time consumption for learning regarding the type of domain. For NLP tasks, DNN can help with information extraction.
- Deep Auto-Encoders: Deep learning model that does not require supervision. For NLP tasks, Deep Autoencoders can help with sentimental analysis.
- Deep Belief Networks (DBN): Supervised and unsupervised learning model, can be very slow to learn.
- Deep Boltzmann Machine (DBM): It is a deeper version of Deep Belief Networks, which has multi-directional connections between layers other than top two layers.
- Recurrent Neural Networks (RNN): It works well with sequential patterns and natural language processing, has variations like Long Short-Term Memory (LSTM). For NLP tasks, RNNs and LSTMs work well with text classification, word tagging and entity name recognition.
- Convolutional Neural Networks (CNN): It works well with computer vision, image processing and natural language processing. Downside of this model is it requires high amount of data to process. For NLP tasks, CNNs work well with text classification, text extraction, entity name recognition and sentimental analysis operations (Doğan Küçük, 2018).

Natural language processing problems are very highly dimensional and feature engineering takes a lot of time for shallow learning models to be efficient. Alternatively, deep learning models have been shown to be outperforming than the machine learning algorithms (Ronan Collobert, 2011). In this review paper, deep learning models and methods such as CNN, RNN and a statistical method of processing natural language were reviewed. Since statistical method of natural language processing suffers from curse of dimensionality, researchers came up with idea of low dimension space (Y. Bengio, 2003) operations such as word embeddings, word2vec, character embeddings. Authors also gave frameworks for word embeddings namely; S-Space, Semanticvectors for Java programming language; Gensim, Pydsm, Dissect, Fasttext for Python programming language. After this statistical based method, CNNs are being reviewed. CNNs are used sentence modelling, multi-task learning, providing good predictions for NLP tasks such as POS tags, named-entity tags, semantic roles, as well as creating informative latent semantic representations of the sentence using n-grams. An example CNN framework for word class prediction has the layers from input to output as; input sentence, lookup table which holds features, convolution layer, max-pool overtime, fully connected layer and lastly softmax classification (R. Collobert, 2008). CNNs work well with n-grams but word order and long-term dependencies typically ignored. Recurrent Neural Networks (RNN) has an inherent advantage of sequentially pattern processing, and with natural languages can be used in tasks with context requirements, application to long texts are better than CNNs. However, RNNs most important part being hidden layers has the disadvantage of vanishing gradient problem. In order to combat that two networks were developed called long short-term memory (LSTM), gated recurrent units (GRUs) and residual networks (ResNets) which the latter is not mostly used in NLP tasks. RNNs work great with word-level and sentence-level classifications. As a conclusion to this paper, the authors mentioned successful experiments regarding named-entity recognition, semantic role labelling and sentiment classification on two different datasets. The highest scored ones are Lue et al. (G. Luo, 2015) with semi-CRF jointly trained with linking, F1 score of 91.20%, He et al. (L. He, 2017)bidirectional LSTM with highway connections, F1 score of 83.2 % first dataset, 83.4% second dataset, and lastly Yu et al. (L. Yu, 2017) three-LSTM with refined word embeddings with accuracy rate of 54% of first dataset and 90.4 of second dataset which is binary version of first one. (Tom Young, 2018)

This experiment paper involves in emotional analysis rather than sentimental analysis. The writers used six basic emotions named; joy, sadness, anger, fear, disgust, and surprise

(Ekman, 1992). After writers explained why deep neural networks could be better for data from both twitter and generally downside of feature engineering, they gathered 205,278 tweet data from Twitter using a library named Tweepy, and Twitter API. In order to use with dictionary named TEL (M. A. Toçoğlu, 2019), words are lemmatized by TurkLemma (Civriz, 2011) and annotated tweets to emotions mentioned above. For preprocessing necessary eliminations are done such as “http”, emojis, punctuations after the eliminations stemming is done via fixed length (F5) (F. Can, 2008) and Snowball Stemmer (SS), with the usage of stop-words in Turkish. The neural networks in this experiment consisted of CNNs, RNN with LSTM, and ANN with 90% of samples were trained and rest were tested without use of cross-validation due to high computational requirement. Additionally, experiments were done with different stemming methods and with and without stop-words in Turkish. Conclusion of this experiment showed us that the best model for this CNN outperformed other deep learning models and all deep learning models outperformed classic machine learning models such as SVMs, RFs, NBs, kNNs and DTs. After CNN the second best model was LSTM, and experiment showed that stemming was not necessary. For the future, writers suggest that bigger datasets and word embeddings schemes such as word2vec, fastText, gloVe and LDA2Vec could be used and performance could be improved. (Mansur Alp Toçoğlu, 2019)

In this project, dictionary method will be used. This method is similar to methods used by (Mansur Alp Toçoğlu, 2019), by using these words in different aspects such as valence (the pleasantness of a stimulus) these words go from unhappy to happy, arousal (the intensity of emotion proved by a stimulus) and dominance (the degree of control exerted by a stimulus) these words go from weak/submissive to strong/dominant. These word norms have been collected since 1999 called Affective Norms for English Words (ANEW) with new words being added regarding gender, age, educational differences in emotion norms and categorical norms such as types of dieses, occupations and taboo words. The variance of these norms are helpful for different aspect sentiment expressions expressed from texts. Moreover, new words can also be added and compared to older ones to fit into their place. (Amy Warriner, 2013)

CHAPTER 3

REQUIREMENTS/REQUIREMENTS ANALYSIS

In this chapter necessary requirements and requirement analysis will be investigated. As the nature of the experiment-like projects, the stakeholders of this project will be academia only and as such results will be shared as well as the codes. Thus, it is expected to not be used by generic end-users.

3.1 Functional Requirements

As it was mentioned in the introduction of this chapter, the nature of the project limits the requirements analysis compared to object-oriented based projects. The end-users will only be able to analyze the results of the project. Thus, functional requirements will be the functions that have been used in the project. So, chronologically the necessary functions:

- Connection to Twitter: Data source is the Twitter, more precisely Twitter databases. In order to use these data, connection to Twitter databases has to be done successfully so twits from different times can be reached. For this purpose, Twitter has developed an API (Application Programming Interface) which allows certain users to connect and request different kind of twits, hashtags, users and more.
- Containing the data: Twitter data come in JSON files, and these files need to be grouped in an organized way and be ready for pre-processing operations.
- Pre-processing the datasets: Before using deep neural networks algorithms, data need to be prepared. From research, some significant operations have been done for high accuracy solutions. These can be removing repetitive letters or fixing the grammar issues, to deciding on whether emojis should be removed or what.
- Feeding through deep neural networks: This is an abstracted function, which is basically the most processer consuming stage of the project. Convolutional and recurrent neural networks are possible architectures for this requirement. More information will be given in the implementation chapter.

- Getting the results: At the end of the experiment project, the results will be given regarding the performance of deep learning architectures as well as findings about the changes on the psyche of Twitter users in the times of pandemic.

3.2. Non-Functional Requirements

Important non-functional requirements (NFRs) are performance, security (in this context data abstraction), and scalability.

Non-functional requirements of this project can be counted as security and performance or scalability. Since this project has no stakeholders that will use in real-world problems, the non-functional requirements of the project may be seen as redundant, and it actually may. However, there are still NFRs that might fit into this project. The project includes data from Twitter users and the privacy of writers needs to be secured. Data should be taken diligently and used as such. This can be done in two ways, either everything needs to be done in a local machine, or the twit handles (user tags) need to be deleted as soon as gathered.

Second important non-functional requirement is performance. Training deep neural networks requires high performance machines. Size of the datasets change how long the learning will take. This can be very problematic if the performance of the algorithms got lower, and experiments and findings would get lagged. In this project an experiment will be done which means different deep learning models will be tested, and consequently performance of this experiment is very important. Unnecessary architectural layers need to be analyzed and used from the start. As a summary, speed of the deep learning algorithms is important, as well as the correctness of the model.

CHAPTER 4

DESIGN

4.1. Architectural View

The design of this project overall is shown below. The source of data for the project comes from Twitter databases and have been brought up to be processed and used in the local machine.

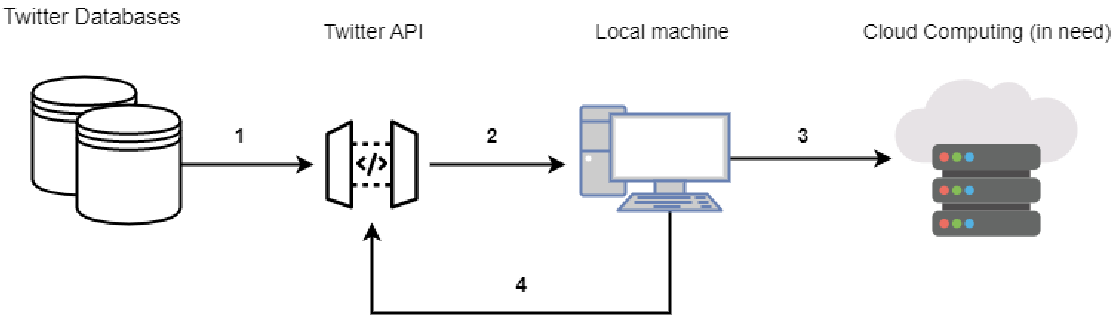

Figure 4.1: High level representation of project.

As it was numbered in the general design, explanations of these numbers are:

1. Data comes from Twitter databases. Twitter has opened up their data for developers. In order to use that, Twitter has developed endpoints that can be reached from Twitter API. By using Twitter API, some specific data about whether in a timeline or specific event or person’s twits can be gathered. This section can be done via specific web developments methods as well as libraries such as “Tweepy” in Python.
2. After necessary data has been collected on the local machine, project’s experiment part could be started. This includes pre-processing data, and using to feed into deep neural networks.
3. Since deep neural networks are very computation heavy, if in necessary computing on cloud by using some services can be done.
4. After the experiments are done and analyzes have been made, if it is needed more data can be fetched via Twitter API until satisfactory evaluations have been made.

4.2. Sequence Diagram

This sequence diagram shows as the initial start-point as the local machine which is the computer is used for this project. Initially, by using Python libraries as mentioned above requests for specific data has been done. Afterwards the Twitter database returns this request to the local machine through Twitter API. These data are held in the local machine to be used in deep neural networks. This sequence finishes as the evaluation has been done.

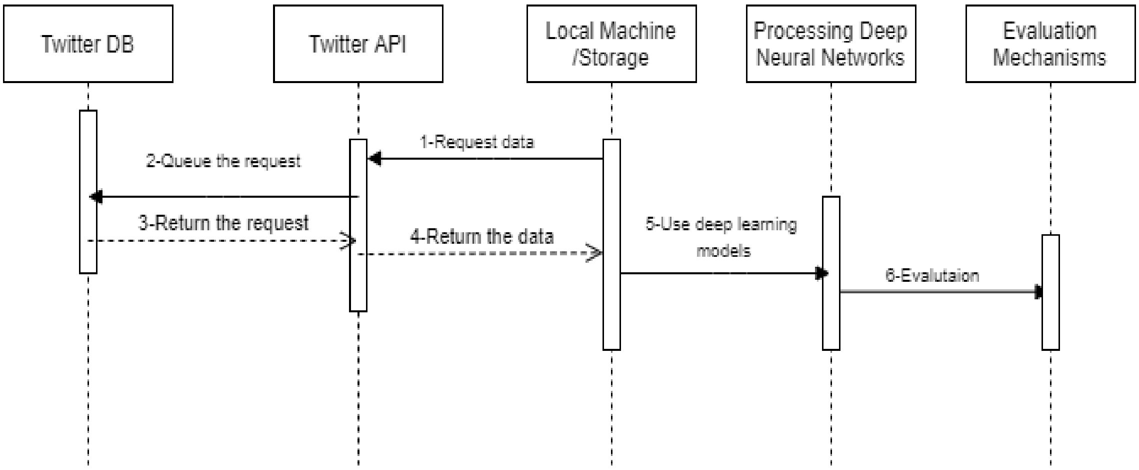

Figure 4.2: Sequence diagram of this project.

4.3. Activity Diagram

The activity diagram is very similar to sequence diagram. In this diagram the only difference is shown in loop to the pre-processing activity from evaluations. As it was shown in architectural view, the loop does not only go for fetching new data, it can also go to pre-processing activity so novel experimental processing can be done.

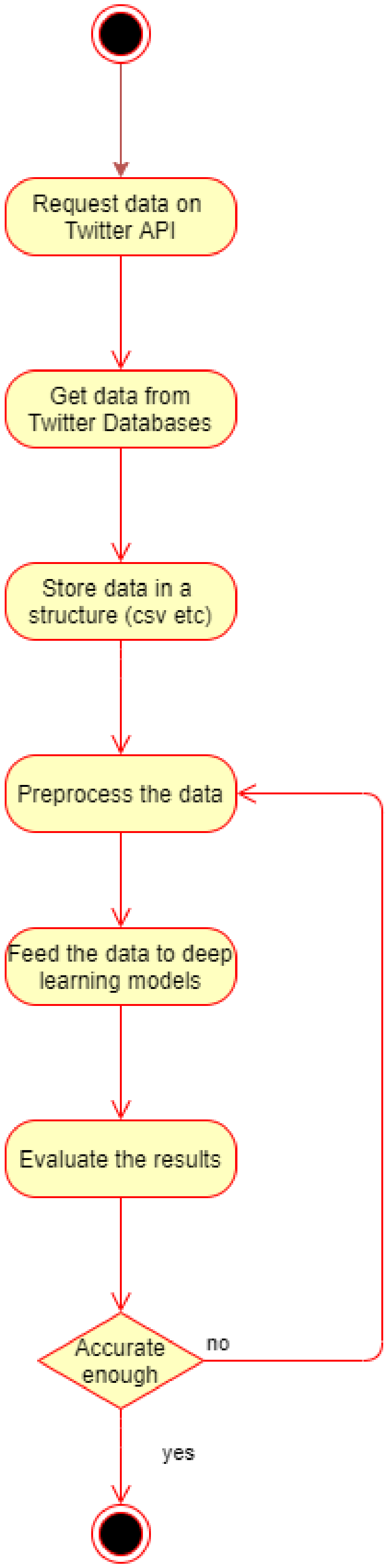

Figure 4.3: Activity diagram of this project.

CHAPTER 5

IMPLEMENTATION

5.1. Technologies

Getting supported with finished products make projects more convenient. In this topic some of the finished products or implementation tools will be presented as an addition to introduction part. Main programming language is Python 3, which is matured and popular programming language for deep/machine learning applications. “PyTorch” framework will be used as a main framework for deep learning models, testing and evaluation. For faster calculations on local graphics card Nvidia’s “CUDA” software, which is a parallel computing platform and programming model for GPUs has been used alongside with PyTorch. Moreover, famous libraries for deep/machine learning such as “pandas”, “NumPy”, “MatPlotLib”, and “sklearn” libraries have been used for data preparation, graphical interpretation and visualization. For data mining from Twitter, a library called “TweePy”, for Turkish NLP operations “Zeyrek” libraries have been used. As the development environment “Anaconda” development environment has been used, as text editors and IDEs mostly “Jupyter’s Notebook” and “Microsoft’s VSCode” have been used.

5.2. Data Gathering

Deep learning methods work with gradient descent algorithm alongside with various types of hidden layers. These layers use data to adjust the weights and biases of the model. Therefore, some amount of data is required for training and for evaluation. A general rule of thumb is the more data is the better models become.

This project use Twitter data, Tweets, which are mined through Twitter’s API. The API lets endpoint users query and mine Tweets. This can be done via HTTP commands but in this project Tweepy library has been used. Tweepy library lets Python call HTTP commands via built-in functions. As much as Twitter provides free API with registration, standard version usage of this API has limitations on backward search such as limiting the number of Tweets that can be get, and other filters such as geolocation, URLs or language of tweet. For example, comparing with standard access, and academic research access, in standard access 100 Tweets can be fetched every request and only 50 requests can be fetched in a month with queries that 128 characters long without any specific filters. In academic research access 500 Tweets can be fetched per request and up to 100 requests per month with additional filters such as geolocation, polls, language and more.

An example from official documentations is shown below in Figure 5.1:

```
$ curl --request GET
 --url 'https://api.twitter.com/1.1/search/tweets.json?q=nasa&result_t
ype=popular'
 --header 'authorization: OAuth oauth_consumer_key="consumer-key-for-a
pp",
 oauth_nonce="generated-nonce", oauth_signature="generated-signature",
 oauth_signature_method="HMAC-SHA1", oauth_timestamp="generated-timest
amp",
 oauth_token="access-token-for-authed-user", oauth_version="1.0"'
$ twurl /1.1/search/tweets.json?q=nasa&result_type=popular
```

Figure 5.1: Tweet mining example in Twitter API documentation.

Similar tweet search can be done via Tweepy as shown below in Figure 5.2:

```
import tweepy
api_key = "x"
api_key_secret= "x"
bearer_token = "x"
access_token = "x"
access_token_secret = "x"

auth = tweepy.OAuthHandler(api_key, api_key_secret)
auth.set_access_token(access_token,access_token_secret)
api = tweepy.API(auth)

search_query = tweepy.Cursor(api.search, q = "covid-
19", geocode= "TR", lang = "tr")
```

Figure 5.2: Authentication code with “TweePy”.

Tweets are mined through Tweepy library as a JSON object. These JSON objects have many fields such as creation date and time, ids, hashtags, URLs, geographic locations and user related fields and many more unrelated fields.

Meanwhile making search queries and using regex will benefit the mining operation as in better samples for the model, not making good enough filters might end up with spams or self-promotion and Tweets that promotes advertisements. Regarding, full archive search, Twitter API limits the amount of filters that can be done on the query. The sampling is planned to be mined through fifteen plus months that have passed since the pandemic has started. These samples will be compared by sentiments they depict throughout the months.

The sampled Tweets are stored as comma-separated values (CSV) files. Tweet times, authors, and other fields such as geolocation and number of favorites will not be used, thus these fields will not be included in the CSV file. By using "search_full_archive" method of Twitter API, and Tweepy function, a search query and date limitation parts of filters have been used. Twitter API gives queries more power by including filters and logical operations such as AND, OR. On the screenshot below Figure 5.3 there are few queries that have been tested to gather the maximum relevance to gather tweets. Finally, query5 is chosen due to restrictions between basic "search" and "full_archive_search" methods.

```
query1 = "korona OR covid-19 OR covid AND -is:retweet"
query2 = "(#korona OR #covid-19 OR covid-19 OR korona) AND (is:verified)"
query3 = "korona covid covid-19"
query4 = "-RT -https lang:tr (korona OR covid OR covid-
19) (#korona OR #covid OR #covid-19)"
query5 = "lang:tr (korona OR covid-19)"
test1 = ["korona -@ -https", "covid -@ -https","covid-19-@ -https"]
```

Figure 5.3: Some queries that have been tested.

After these queries are prepared, search queries are saved as "ResultSet" object, then this object is cast into "Status" object which allows JSON type results to easily harvest the text part. On below example, the search-through saving in a CSV file is shown. As it can be seen in Figure 5.4, dates and number of Tweets are specified in the function which is saved in a list structure to be saved as a CSV file.

```
tweets2 = tweepy.Cursor(api.search_full_archive,
      environment_name="tweetMining",
      query=query5,
      fromDate="201912010001",
      toDate="201912310001").items(100)

tweetS_list2 = [tweet.text for tweet in tweets2]

with open('dec19.csv', mode='a', encoding='utf-8') as file1:
    writer1 = csv.writer(file1, delimiter = ';', quotechar='"')
    writer1.writerow(tweetS_list2)
```

Figure 5.4: Searching and storing Tweets.

5.3. Natural Language Processing

Tweets have been acquired as batches via Twitter API. These tweets are saved as months in a CSV file. In order to process the datasets regarding NLP opeartions, Python's base libraries and an NLP library called "Zeyrek" library has been used. Another NLP tools that have been used is dictionaries, in this project this dictionary is called "Affective Norms for English Words" (ANEW) dictionary. As a corpus for Turkish translation of will be used. ANEW dictionary characterizes words in dimensions of valence, arousal, and dominance. These dimensions are called dimensional norms, that affects stimuli. Positive valence represents appetite and negative valence represents defensive systems. Arousal represents intensity of activation. This kind of approach is parallel to word2vec approaches which makes it easier for computable functions. Since the paper of ANEW dictionary published, different numerical approaches on different languages have been done, and word size of the dictionaries have grown.

In case of cultural differences, Spanish, Greek, and Portuguese translations are found and can be used for statistically correct values. For example, in Figure 5.5, Figure 5.6, Figure 5.7 and Figure 5.8:

Index	Greek word	Index of Anew woı	Valence	Arousal	Dominance
1	αβγό	288	0.25	-0.214286	0.190476
2	αβοήθητος	421	-0.7	0.166667	-0.440476
3	άγαλμα	881	0.223684	-0.190476	0.154762
4	αγάπη	555	0.763158	0.428571	0.190476
5	αγαπημένος	556	0.625	0	0.309524

Figure 5.5: Greek translation of corpus.

A	B	C	D	E	F
		V	A	D	
1000000	million	4.28	3.19	3.41	NN
1000000000	billion	3.9	3.22	3.75	NN
1000000000000	trillion	3.43	3.1	3.13	NN
ab	aqua	3.75	2.3	2.91	NN
abacı	drone	2.25	2.65	2.95	NN
abanoz	ebony	3.57	2.66	2.86	NN
abartı	exaggerat	2.6	3.4	2.92	NN
abartılmış	overrated	2.33	3.2	2.91	ADJ

Figure 5.6: Turkish translation of corpus.

The bottom two are European-Portuguese and Spanish. The difference than Greek, and Turkish is all three features have their statistical representation as medians and standard deviations, as well as other statistical values.

Number	E-Word	EP-Word	Val-M	Val-SD	Arou-M	Arou-SD	Dom-M	Dom-SD	Freq	Nlett	Nsyll	GClass	Neigh
1	abuse	maus-tratos	1.51	1.24	7.37	2.28	4.41	2.83	0.19	11	2	N	UnAv
2	accident	acidente	2.25	1.28	6.95	1.69	3.33	2.22	124.36	8	4	N	1
3	achievement	realização	7.31	1.35	4.67	2.73	6.96	1.97	88.65	10	5	N	0

Figure 5.7: European-Portuguese translation of corpus.

ID	Word	English Trans.	POS	VAL_M	VAL_SD	VAL_N	ARO_M	ARO_SD	ARO_N	CON_M	CON_SD	CON_N	IMA_M	IMA_SD	IMA_N	AVA_M	AVA_SD	AVA_N	FAM_M	FAM_SD	FAM_N
1	abandono	*abandonment*	*N V*	1.09	0.29	23	7.85	1.42	20	4.52	1.73	23	3.70	1.77	23	3.95	1.76	22	5.65	1.27	20
2	abanico	*fan*	*N V*	5.64	0.99	28	3.57	2.02	23	5.64	1.56	22	6.86	0.36	21	4.67	1.93	21	5.15	2.06	20
3	abeja	*bee*	*N*	2.90	1.62	20	7.14	1.61	22	6.86	0.35	22	6.77	0.53	22	5.78	1.88	23	5.48	2.02	21

Figure 5.8: Spanish translation of corpus.

The main point is in this part is to have Tweets have representative columns that describes the values of emotion in valence, arousal and dominance columns.

The ANEW dictionary’s valence, arousal and dominance columns have similar statistics. Minimum and maximum values are squeezed between 1.1 and 4.5 which makes model to correlate harder. In the screenshot below in Figure 5.9, ANEW dictionary’s statistics is given.

	valence	arousal	dominance
count	15218.000000	15218.000000	15218.000000
mean	3.053836	2.603538	3.113214
std	0.652980	0.445302	0.482177
min	1.130000	1.340000	1.340000
25%	2.630000	2.290000	2.810000
50%	3.130000	2.550000	3.160000
75%	3.500000	2.870000	3.450000
max	4.770000	4.400000	4.430000

Figure 5.9: ANEW Turkish dictionary statistics.

The unprocessed Tweets are held as CSV files that are separated as months such as “dec20”, “aug20” starting from “dec19” until “march21”. In order to process these tweets, all words have to be strip from punctuations, webpage links to other pages, and unnecessary whitespaces. Afterwards, a function is put in use to a single twit that lowers all the letters, tokenize the sentence into words which is put into a list. A lemmatization list can be seen in Figure 5.10.

```
[[('rt', ['Rt'])], [('dünya', ['Dünya', 'dünya'])], [('sağlık', ['Sağlık', 'sağlık', 'sağ'])], [('örgütü', ['örgüt'])], [('coro
na', ['Corona'])], [('virüsü', ['virüs'])], [('nedeniyle', ['neden', 'nedeniyle'])], [('uluslararası', ['uluslararası'])], [('a
cil', ['acil'])], [('durum', ['duru', 'durum'])], [('ilan', ['ilân'])], [('ederken', ['etmek', 'eder'])], [('birleşik', ['birle
şik'])], [('krallık', ['kral', 'Krallık'])], [('ülkede', ['ülke', 'Ülke'])], [('corona', ['Corona'])], [('virüsün', ['virü
s'])]]
```

Figure 5.10: Lemmatization via Zeyrek.

Words are then lemmatized by using Turkish NLP tool “Zeyrek” library. As much as this library successfully lemmatizes words into different stems, either this stem is not available in ANEW dictionary, or many close related stems are available. To combat this, every possible stem is taken into consideration by getting mean value of every three columns (valence, arousal, and dominance). After every tokenized word that has a representation in the dictionary had assigned their values, the labels of valence, arousal, and dominance values are calculated by getting mean of these tokenized words. In Figure 5.11 an example of assigned Tweet values are given.

	Twits	Valence	Arousal	Dominance
0	7 bin kişilik gemide Korona virüsü şüphesi	2.981667	2.550833	2.886667
1	7 bin kişilik gemide Korona virüsü şüphesi	2.981667	2.550833	2.886667
2	RT _ Dünya Sağlık Örgütü korona virüsü için acil durum ilan etti Korona Virüsü için alınan tedbirler araştırılsın önergesi AKP	3.037692	2.669615	3.092692
3	RT _HelpDesk ℹ Korona Virüsü kaynaklı durumu ve son gelişmeleri ulusal ve uluslararası otoritelerle değerlendirmek için Pekin Guangzh	3.081111	2.698333	3.202778
4	Çinde korona virüs vakası nedeniyle ölü sayısı 170e yükseldi	2.696667	2.570833	2.556667

Figure 5.11: Assigning the dimensions to Tweets.

For conveniency reasons, an automated value assigning code has been implemented. As an input, it gets the “month+year.csv” formula and returns “month_p+year.csv” file. In Figure 5.12 and Figure 5.13 input and output operations of dataset operations.

```
#Getting input as the name of the file.
month_name = input("Please enter filename (e.g:jan20.csv): ")

twit_jan20_DF = pd.read_csv(month_name, sep=";", header=None)
```

Figure 5.12: Reading CSV files as datasets.

```
#Output file name operations
x_ = month_name.split(".")
month_p_name = x_[0]+"_p."+x_[1]
df1.to_csv(month_p_name, header=True, sep=';', decimal='.')
print(month_p_name, "is created.")
```

Figure 5.13: Writing CSV datasets and adds “_p”.

As the operations that are presented in Figure 5.12 and Figure 5.13 input and output sections of the automation algorithms. One of the important functions for these operations is called “nlp_operations_vad” which takes a Tweet string as a parameter and calculates the means of each columns by checking the dictionary. The function gets a string which comes from dataset, lowers the letters, splits the words on whitespaces and append into a list. These words are now can be lemmatized and versions of lemmatizations can be appended into another list, in this piece of code in Figure 5.14, this list is named “t1”.

```
def nlp_operations_VAD(a_twit):

    #Getting input twit
    string1 = str(a_twit)
    #Lowercase it
    string1 = string1.lower()
    #Split it (or call it tokenizing)
    split1 = [i for i in string1.split(' ')]

    t1 = []
    #This lemmatizes the words in the string
    for i in range(len(split1)):
        t1.append(analyzer.lemmatize(split1[i]))

    #creating lists for valence, arousal and dominance values' mean, so that row of twit can have these values
    t1_V, t1_A, t1_D = [], [], []
```

Figure 5.14: First part of “nlp_operations_VAD” function.

After lemmatized words are created, these words are put into another lists for getting mean values. As it can be seen in Figure 5.15, “word_sum_X” lists represent the mean value of different stems from lemmatization, “t1_X” lists represent overall mean values of the Tweet.

In this part, more appropriate mathematical functions could be used rather than getting the mean value, however, in order to move forward basic mean calculation has been chosen.

```
#creating lists for valence, arousal and dominance values' mean, so that row of twit can have these values
t1_V, t1_A, t1_D = [], [], []


#iterate over list that holds the lemmatized words
for i in range(len(t1)):
    #iterate over lemmatization tuples
    for words in t1[i][0]:
        #Lemmas can be more than one so getting mean of interpretations could be useful.
        word_sum_V, word_sum_A, word_sum_D = [], [], []

        #Lemmtization products are in a list, take it by itself
        if type(words) == list:
            for i in words:
                if (anew_turkish["isimler"] == i).any():
                    word_sum_V.append(anew_turkish.loc[anew_turkish["isimler"] == i]["valence"].iloc[0])
                    word_sum_A.append(anew_turkish.loc[anew_turkish["isimler"] == i]["arousal"].iloc[0])
                    word_sum_D.append(anew_turkish.loc[anew_turkish["isimler"] == i]["dominance"].iloc[0])

        #Just string, which is the word without lemmazitaion
        elif type(words) == str:

            if (anew_turkish["isimler"] == i).any():
                word_sum_V.append(anew_turkish.loc[anew_turkish["isimler"] == i]["valence"].iloc[0])
                word_sum_A.append(anew_turkish.loc[anew_turkish["isimler"] == i]["arousal"].iloc[0])
                word_sum_D.append(anew_turkish.loc[anew_turkish["isimler"] == i]["dominance"].iloc[0])

        #Bunch up the words in mean to twit part.
        t1_V.append(get_mean(word_sum_V))
        t1_A.append(get_mean(word_sum_A))
        t1_D.append(get_mean(word_sum_D))


return [round(get_mean_twit(t1_V), 2), round(get_mean_twit(t1_A), 2), round(get_mean_twit(t1_D), 2)]
```

Figure 5.15: Second part of the “nlp_operations_VAD” function.

5.4. Deep Learning Model

Theoretically, text related models are trained with recurrent deep learning models. More precisely, long-short term memory (LSTM) is favored for word to vector applications. In this project time-series version of long-short term memory have been used. In figure 5.16, the stereotypical LSTM implementation is shown. In constructor, inheriting superclass’ method and assigning necessary parameters as it can be seen. After “forward” function is called, weight layers and biases are assigned zeros due to LSTM’s architecture.

```
class LSTM(nn.Module):
    def __init__(self, num_classes, input_size, hidden_size, num_layers):
        super(LSTM, self).__init__()

        self.num_classes = num_classes
        self.num_layers = num_layers
        self.input_size = input_size
        self.hidden_size = hidden_size
        self.seq_length = seq_length
        self.lstm = nn.LSTM(input_size=input_size, hidden_size=hidden_size, num_layers=num_layers, batch_first=True)

        self.fc = nn.Linear(hidden_size, num_classes)

    def forward(self, x):

        h_0 = Variable(torch.zeros(self.num_layers, x.size(0), self.hidden_size))
        c_0 = Variable(torch.zeros(self.num_layers, x.size(0), self.hidden_size))

        ula, (h_out, _) = self.lstm(x, (h_0, c_0))
        h_out = h_out.view(-1, self.hidden_size)
        out = self.fc(h_out)
        return out
```

Figure 5.16: Implementation of LSTM network by using PyTorch framework.

Long-short term memories are used in time-series applications due to having memory and forget cell that can realize the underlying relationships between parameters. In this project, two similar LSTM architectures have been created to experiment on. In figure 5.17 an example of visualization of neural network architecture is given, for conveniency reasons 100 neurons are not shown.

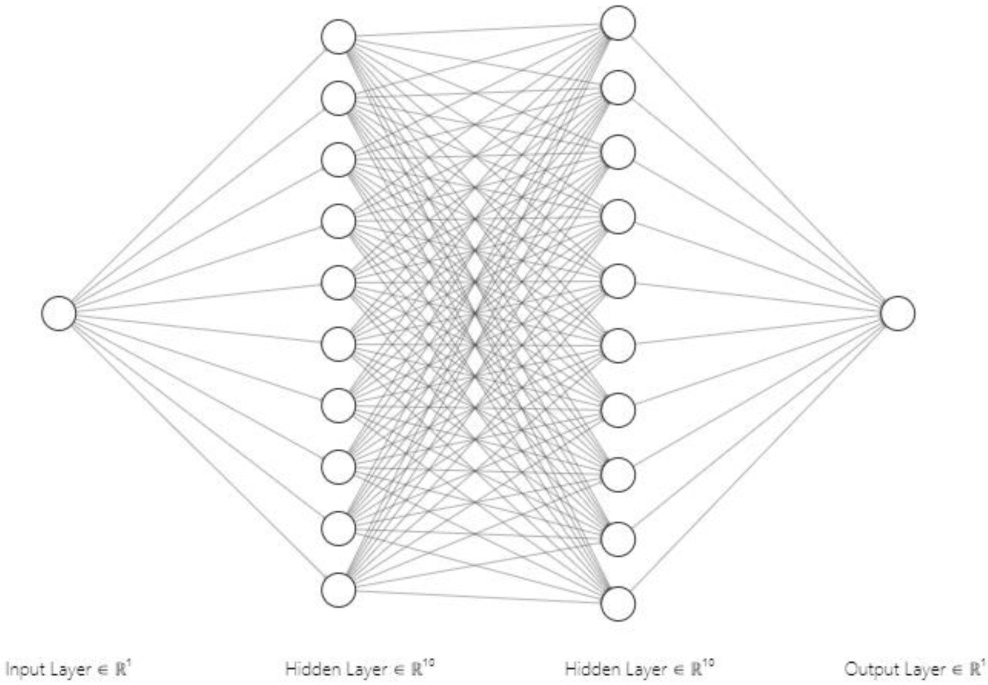

Figure 5.17: A representation of neural network.

Additionally, normalizing data and partitioning data into batches for sequence-to-sequence processing have been implemented. Tensors which is a main tool for deep learning work with specific dimensions. To fit that dimension, matrices must be shaped. In our project, data is Tweet's feature column(s) which is "Valence" column, "seq_length" is the sequence length which is taken as 100 because every month has 100 Tweets, hence, data. Additionally, normalization between 0 and 1 is useful for gradient step operation. In Figure 5.18, the implementation of these steps are shown.

```
def batching(data, seq_length):
    x, y = [], []

    for i in range(len(data)-seq_length-1):
        _x = data[i:(i+seq_length)]
        _y = data[i+seq_length]
        x.append(_x)
        y.append(_y)

    return np.array(x), np.array(y)

normalizer = MinMaxScaler()
training_data = normalizer.fit_transform(dA)
seq_length = 100
x, y = batching(training_data, seq_length)
```

Figure 5.18: Implementation of "batching" function for Tensors.

For loss function "Mean Squared Error" loss function, for optimizer Adam optimizer have been used. Adam optimizer is an "first-order gradient-based optimization of stochastic objective functions". Learning rate is fixed on 0.001. For both architectures number of epochs has been assigned as 200, as it can be seen, after 100th epoch, loss function starts to converge. Input and output size were assigned as one neuron. Only difference between architectures were neurons in hidden layers. As different number of neurons assigned, changes on outcome has not been observed.

CHAPTER 6

TEST/EXPERIMENTS

6.1. Data Mining and NLP

Tweet mining through the codes that given in previous chapter, gave out tweets with handles “@”, emojis, and Twitter’s quick links “https://t.co/...”, and separated with semicolon “;”. In Figure 6.1 an example of raw Tweet data is shown. This is how it stored after requested from Twitter API.

```
7 bin kişilik gemide Korona virüsü şüphesi https://t.co/whdJ45XrOW;7 bin kişilik gemide Korona virüsü şüphesi https://t.co/...
https://t.co/...;"RT @(x): Dünya Sağlık Örgütü korona virüsü için acil durum ilan etti.

Korona Virüsü için alınan tedbirler araştırılsın önergesi AKP…";"RT @(x): ℹ
Korona Virüsü kaynaklı durumu ve son gelişmeleri ulusal ve uluslararası otoritelerle değerlendirmek için; Pekin, Guangzh…";Çin’de korona
virüs vakası nedeniyle ölü sayısı 170’e yükseldi.. https://t.co/...;"RT @(x): Dünya Sağlık Örgütü  Corona virüsü nedeniyle uluslararası acil
durum ilan ederken .
```

Figure 6.1: Raw data of Tweets.

As an example of comparison, a tweet from December 2019:

“@(a handle) korona maçı hakkında yorumun ne olur”

An example from August 2020:

“Ankara Dışkapı Eğitim Araştırma Hastanesinde bir hastaya plazma tedavisi için daha…”.

As it can be seen, back in December, the Tweets were mostly about either the football team or the beverage corona (commonly written as korona in Turkish), not about the virus. The first cases and general precautions were taken in between March 2020 and February 2020. After six months, every Tweet that was filtered was about corona virus.

The Tweets were processed through string manipulations and using regex operations. After processing has been done, Tweets were prepared to make use in neural network. These operations changed the Tweets which can be seen in Figure 6.2.

In this project around 1200 Tweets have been mined through Twitter API. In these kind of experiments or projects more data is needed.

```
;Twits;Valence;Arousal;Dominance
0;7 bin kişilik gemide Korona virüsü şüphesi;2.98;2.55;2.89
1;7 bin kişilik gemide Korona virüsü şüphesi;2.98;2.55;2.89
2;"RT _ Dünya Sağlık Örgütü korona virüsü için acil durum ilan etti

Korona Virüsü için alınan tedbirler araştırılsın önergesi AKP";3.04;2.67;3.09
3;"RT _HelpDesk i
Korona Virüsü kaynaklı durumu ve son gelişmeleri ulusal ve uluslararası otoritelerle değerlendirmek için Pekin Guangzh";3.08;2.7;3.2
4;Çinde korona virüs vakası nedeniyle ölü sayısı 170e yükseldi;2.7;2.57;2.56
5;"RT Dünya Sağlık Örgütü Corona virüsü nedeniyle uluslararası acil durum ilan ederken

Birleşik Krallık ülkede Corona virüsün";3.06;2.63;3.05
```

Figure 6.2: Same Tweets preprocessed, and dimensional norms values added.

6.2. Deep Learning

Before training the model, as an example, a graph from January 2020 values are given below in Figure 6.3. As it can be seen, the x dimension represents the individual Tweets.

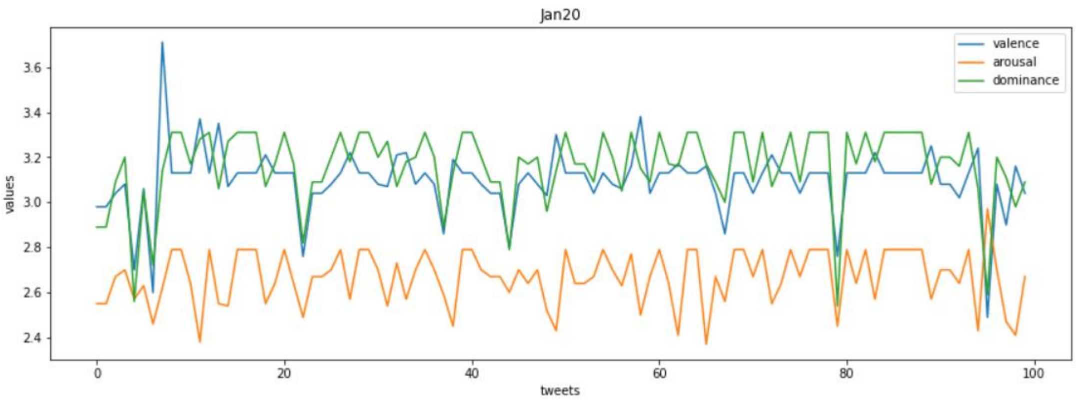

Figure 6.3: Tweets and dimensional norm values in January 2020.

The purpose of this project is to compare this month’s values to next month until future analysis can be done. In below Figure 6.4, a graph of January 2021 is given.

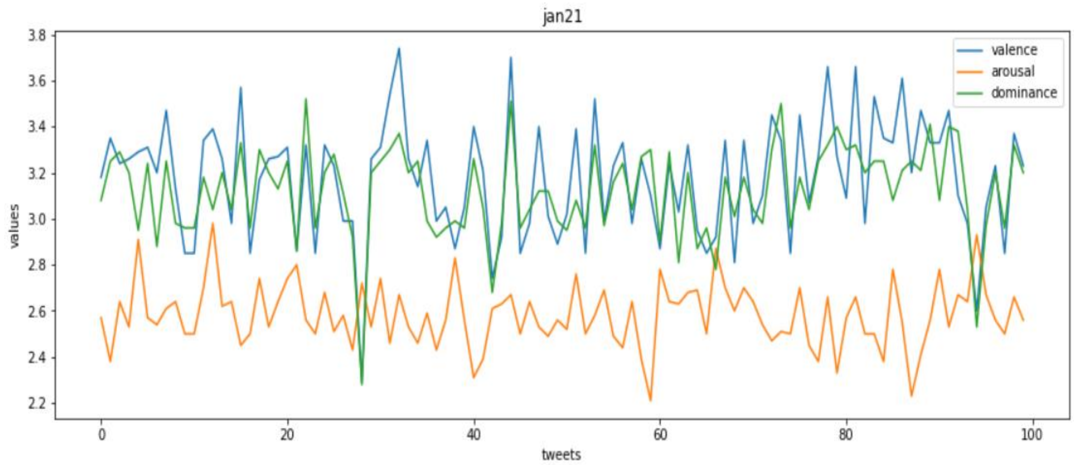

Figure 6.4: Tweets from January 2021.

After concatenating the datasets as timely manner, the graph below is given in Figure 6.5. The numbers in x dimension shows the months since it started to end. As it can be seen many fluctuations can be observed.

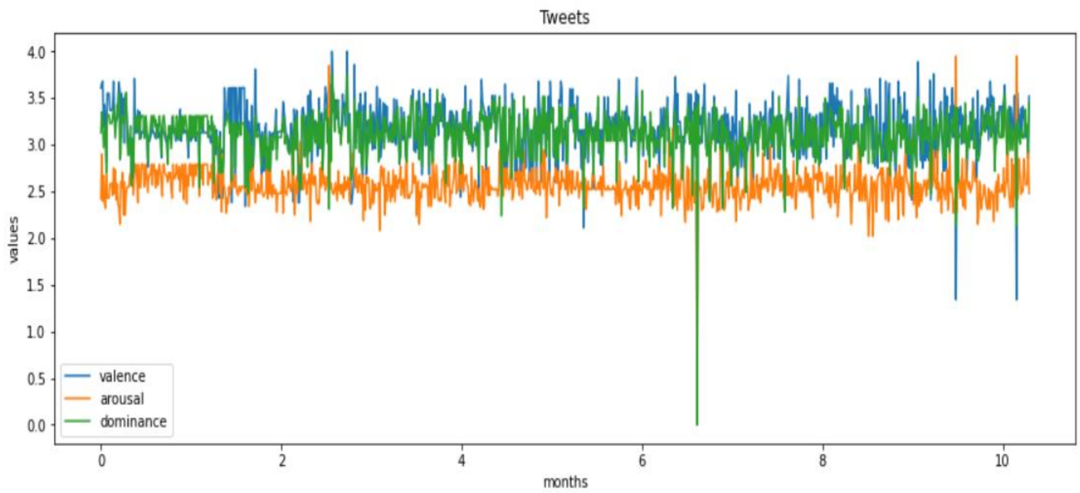

Figure 6.5: Graph of all datasets.

For training valence label has been selected. First a normalization is done in order to get lower loss function and data is sampled as training and testing. In Figure 6.6 it can be seen that after 75th epoch, loss function values have not changed. Overall, the training batches it has been observed that 100th epoch was the convergent value for loss function.

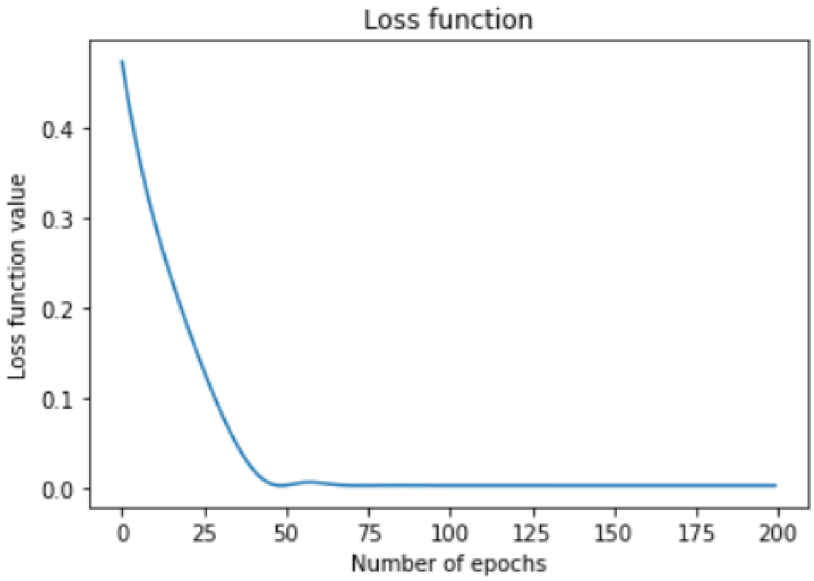

Figure 6.6: Loss function through 200 epochs.

Finally, the model is put into testing. In the Figure 6.7 below it can be seen that the model is limited itself into between the mean of the values but can predict outliers as being outside of mean values. The orange curve is predictions, blue curve is actual data. X dimension is accumulated number of Tweets, with low error margin it represents 15 months of data. The prediction data changes three decimals after the point which makes it hard to analyze, eventually converges on 2.56.

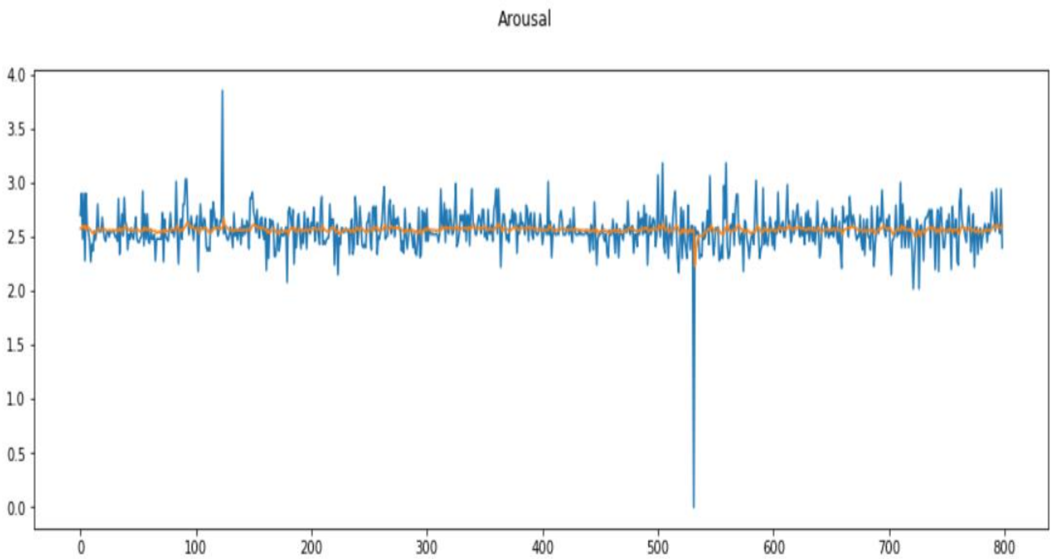

Figure 6.7: Orange predictions, blue real data.

CHAPTER 7

CONCLUSIONS

This project was an experiment to see if we can compare the sentiments of Turkish people throughout the isolation of global pandemic. The Tweets from Twitter has been used as a main data source and ANEW corpus has been used for the long-short term memory deep learning architecture to analyze the sentiment.

For improvements, more data is required this could be made with Twitter's Academic application, as well as better quantitative methods could be used for assigning Tweet values of dimensional norms. Better architectures with LSTM and NLP methods could be done, however this is limited by knowledge and background (mathematical) knowledge of writer.

At the end, this experiment has been finished, and it is shown that taking mean value of values, will limit the predictor's ability to predict. Better models could be implemented on both NLP and Neural Networks parts. Sentimental or opinion analysis is popular subject in artificial intelligence community and better models with better predictions can always be made. Quantifying can open more analyze possibilities that can help us understand about our nature too.

REFERENCES

A.C. Türköem, A. C. (2014). Political Interes and Tendency Prediction from Microblog Data. *Signal Processing and Communications Applications (SIU) Conference* (pp. 1327-1330). IEEE.

Adalı, E. (2013). *Doğal Dil İşleme.* İstanbul.

Alaettin Uçan, E. A. (2014). Assessment of Feature Selection Metrics for Sentiment Analysis: "Turkish Movie Reviews". *European Conference Data Mining 2014 and International Conference Intelligent Systems* (pp. 180-184). Theory and Practice in Modern Computing.

Amy Warriner, V. K. (2013). Norms of valence, arousal and dominance for 13,915 English Lemmas. *Psychonomic Society,* 1191-1207.

Civriz, M. (2011). *Dictionary-based effective and efficient Turkish lemmatizer.* İzmir.

Doğan Küçük, N. A. (2018). *Doğal Dil İşlemede Derin Öğrenme Uygulamaları Üzerine Bir Literatür Çalışması.* 76-86: Uluslararası Yönetim Bilişim Sistemleri ve Bilgisayar Bilimleri Dergisi.

E.S Akgül, C. E. (2016). Twitter Verileri ile Duygu Analizi. *Pamukkale Üniversitesi Mühendislik Bilimleri Dergisi,* 106-110.

Ekman, P. (1992). Are there basic emotions? *Psychology Review, Vol: 99, No: 3,* 550-553.

Eyüp Sercan Akgül, C. E. (2016). Twitter Verileri ile Duygu Analizi. *Pamukkale Üniversitesi Mühendislik Bilimleri dergisi*, 106-110.

F. Can, S. K. (2008). Information retrieval on Turkish texts. *Journal of American Society of Information Science and Technology, Vol: 52, No: 3*, 407-421.

G. Luo, X. H. (2015). Joint Named Entity Recognition and Disambiguation. *Conference Empirical Methods Natural Language Processing*, (pp. 876-880).

Gökçe Uludoğam, R. Ö. (2019). Türkçe Doğal Dil İşleme için Arayüzler. *ResearchGate*.

H. Nizam, S. A. (2014). Sosyal Medyada Makine Öğrenmesi il Duygu Analizinde Dengeli ve Dengesiz Veri Setlerinin Performanslarının Karşılaştırılması. *XIX Türkiye'de İnternet Konferansı.*

Hinrich Schuetze, C. M. (1999). *Foundations of Statistical Natural Language Processing.* MA: MIT Press.

IQBAL, M. (2020, November 29). *Twitter Revenue and Usage Statistics (2020)*. Retrieved from BusinessofApps: https://www.businessofapps.com/data/twitter-statistics/#:~:text=We%20saw%20a%20recovery%20to,187%20million%20in%20Q3%202020.&text=It%20was%20rated%204th%20in,a%20year%20rich%20with%20news.

James H. Martin, D. J. (2008). *Speech and Language Processing.* Prentice Hall.

L. He, K. L. (2017). Deep Semantic Role Labeling: What works and what's next. *Proceeding Annual Meeting Association for Computational Linguistics*, (pp. 473-483).

L. Yu, J. W. (2017). Refining Word Embeddings for Sentimental Analysis. *Proceeding Conference Empirical Methods Natural Language Proceedings*, (pp. 545-550).

M. A. Toçoğlu, A. A. (2019). Lexicon-based emotion analysis in Turkish. *Turkish Journal of Electrical Engineering and Computer Sciences*, 1213-1227.

M. Meral, B. D. (2014). Twitter Üzerinde Duygu Analizi. *IEEE 22. Sinyal İşleme ve İletişim Uygulamaları Kurultayı*, (pp. 23-25).

Mansur Alp Toçoğlu, O. Ö. (2019). Emotion Analysis from Turkish Tweets Using Deep Neural Networks. *Special Section on Advanced Data Mining for Social Computing IEEE Access Volume 7*.

Onan, A. (2017). Sentimental Analysis on Twitter Messages Based on Machine Learning Methods. *Yönetim Bilişim Sistemleri*, 1-14.

Onan, A. (2017). Sentimental Analysis on Twitter Messages Based on Machine Learning aMethods. *Yönetim Bilişim Sistemleri Dergisi, Cilt: 3, Sayı: 2*, 1-17.

Ö. Çoban, B. Ö. (2015). Sentiment Analysis for Turkish Twitter Feeds. *Signal Processing and Communications Applications Conference (SUI)* (pp. 2388-2391). IEEE.

Önder Çoban, B. Ö. (2015). *Türkçe Twitter Mesajlarının Duygu Analizi.* Erzurum: IEEE.

Padmini Srinivasan, Y. M. (2011). Exploring Feature Definition and Selection for Sentiment Classifiers. *Procedings of the Fifth International AAAI Conference on Weblogs and Social Media*, (pp. 546-549).

R. Collobert, J. W. (2008). A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning. *Proceedings of the 25th International Conference on Machine Learning*, 160-167.

Ronan Collobert, J. W. (2011). Natural Language Processing (Almost) from Scratch.

Journal of Machine Learning Research, 2493-2537.

Şeker, S. E. (2016). Duygu Analizi (Sentimental Analysis). *YBS Ansiklopedi*, *Cilt: 3, Sayı: 3*, 19-34.

Tom Young, D. H. (2018). Recent Trends in Deep Learning Based Natural Language Processing. *IEEE Computational Intelligence Magazine*, 55-75.

Y. Bengio, R. D. (2003). A Neural Probabalistic Language Model. *Journal of Machine Learning Research*, 1137-1155.

Yogesh Gard, N. C. (2014). Sentimental Analysis of Twitter Feeds. In S. M. S. Srinivasa, *Book Big Data Analysis* (pp. 32-52). New Delhi: Springer International Publishing Switzerland.