STANCE DETECTION IN TURKISH DATASET ON RUSSIA-UKRAINE WAR

ERAY FIRAT

MAY 2023

ÇANKAYA UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

DEPARTMENT OF COMPUTER ENGINEERING

M.Sc. Thesis in

COMPUTER ENGINEERING

STANCE DETECTION IN TURKISH DATASET ON RUSSIA-UKRAINE WAR

ERAY FIRAT

MAY 2023

ABSTRACT

STANCE DETECTION IN TURKISH DATASET ON RUSSIA-UKRAINE WAR

FIRAT, Eray
M.Sc. in Computer Engineering

Supervisor: Asst. Prof. Dr. Serdar ARSLAN
May 2023, 52 pages

Social media has evolved into a crucial informational resource to understand public opinion on various issues in recent years. Therefore, the importance of automatic information extraction from these data has increased. Stance detection, one of the subtasks of natural language processing, is also a crucial issue for automatic information extraction. Stance detection automatically determines the user's side regarding a particular subject, event, or person. In this study, a Turkish-labelled data set focusing on the stance determination task to determine social media users' attitudes towards the Russia-Ukraine War was created, and various machine learning methods were evaluated on this data set.

For this study, 8215 tweets were collected on Twitter and cleaned. The dataset then was tagged with two targets Russia, and Ukraine. Support Vector Machines, Random Forest, k-Nearest Neighbour, XGBoost, Long-Short Term Memory (LSTM), and Gated Recurrent Unit (GRU) models are employed with GloVe and Fastext word embedding. Since the dataset is unbalanced between the targets, undersampling and oversampling methods were also used with these algorithms.

With an F1 score of 0.73 for Russia and 0.81 for Ukraine, the results showed the Support Vector Machines algorithm to produce the best outcomes. In addition to these results, LSTM and GRU also produced outcomes that were highly comparable to those of the Support Vector Machines algorithm.

The newly created Turkish corpus can be regarded as a valuable resource for this research area and in the future, transformer-based approach can be used with this corpus.

Therefore, this study advances the field of stance detection research using Turkish text.

Keywords: Stance Detection, Natural Language Processing, Turkish Dataset.

ÖZET

RUSYA-UKRAYNA SAVAŞI HAKKINDA TÜRKÇE VERİ SETİNDE DURUŞ TESPİTİ

FIRAT, Eray
Bilgisayar Mühendisliği Yüksek Lisans

Danışman: Dr. Öğ. Üyesi Serdar ARSLAN
Mayıs 2023, 52 sayfa

Sosyal medya son yıllarda çeşitli konulardaki kamuoyu görüşlerini anlamak için temel bir bilgi kaynağı haline gelmiştir. Bu nedenle, sosyal medyadan elde edilen verilerden otomatik bilgi çıkarmanın önemi artmıştır. Doğal dil işleme alt görevlerinden biri olan duruş tespiti de, otomatik bilgi çıkarımı için önemli bir konudur. Duruş tespiti, kullanıcının belirli bir konu, olay veya kişiye karşı tutumunu otomatik olarak belirler. Bu çalışmada, Rusya-Ukrayna Savaşı'na ilişkin sosyal medya kullanıcılarının duruşlarını tespit etmeye odaklanan Türkçe etiketlenmiş veri seti oluşturulmuş ve bu veri seti üzerinde çeşitli makine öğrenimi yöntemleri test edilmiştir.

Bu çalışma için Twitter'dan toplanmış Türkçe metinler içinden Rusya ve Ukrayna olmak üzere iki hedefle etiketlenmiş 8215 tane metin-hedef çifti ile yeni bir veri seti oluşturulmuştur. Bu veri setine Destek Vektör Makineleri, Rastgele Orman, k-En Yakın Komşu, XGBoost, Uzun-Kısa Süreli Bellek (LSTM) ve Kapı Özyinelemeli Geçitler (GRU) modelleri GloVe ve Fastext kelime gömme yöntemi ile uygulanmıştır. Veri seti hedefler arasında dengesiz olduğu için, bu algoritmalarla eksik örnekleme ve aşırı örnekleme yöntemleri de kullanılmıştır.

Destek Vektör Makineleri yöntemi ile, Rusya için 0.73 ve Ukrayna için 0.81 F1 puanıyla en iyi sonuçlaın alındığı görülmüştür. Bu sonuçlara ek olarak, LSTM ve

GRU yöntemlerinden elde edilen sonuçlar Destek Vektör Makineleri algoritmasının sonuçlarına oldukça yakındır.

Yeni oluşturulan bu Türkçe veri seti, duruş tespiti araştırma alanı için değerli bir kaynak olarak değerlendirilebilir ve gelecek çalışmalarda bu veri seti ile transformer tabanlı yaklaşımlar kullanılabilir.

Genel olarak, bu çalışma Türkçe metin kullanarak duruş tespiti araştırma alanını katkıda bulunmaktadır.

Anahtar Kelimeler: Duruş Tespiti, Doğal Dil İşleme, Türkçe Veri Seti

ACKNOWLEDGEMENT

I would like to express my sincere thanks to my beloved wife for her support and sacrifice for me, to my beloved son for his great patience and my brother for his support.

Special thanks to my supervisor Asst. Prof. Dr. Serdar ARSLAN for the excellent guidance and providing me with an excellent atmosphere to conduct this research.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ABBREVIATIONS

NLP	: Natural Language Processing
AI	: Artificial Intelligence
ML	: Machine Learning
DL	: Deep Learning
SD	: Stance Detection
FNC	: Fake News Challenge
AGR	: Agree
DSC	: Discuss
DSG	: Disagree
UNR	: Unrelated
SVM	: Support Vector Machines
RF	: Random Forest
EN	: English
KNN	: K-Nearest Neighbor
XGB	: XGBoost
XGBoost	: eXtreme Gradient Boosting
LSTM	: Long Short-Term Memory
GRU	: Gated Recurrent Unit
BERT	: Bidirectional Encoder Representations from Transformers
FP	: False Positive
TP	: True Positive
FN	: False Negative
TN	: True Negative
GloVe	: Global Vectors for Word Representation
RNN	: Recurrent Neural Networks

CHAPTER I
INTRODUCTION

1.1 BACKGROUND

With the information age, connecting and interacting with people and their opinions became easy and very crucial. The Internet makes it easy and some social media platforms make it easier. Recent times social media platforms are also main information source for people. These are also good sources for scientific studies, especially natural language processing (NLP) problems. NLP is a field of Artificial Intelligence, focused on understanding human language. NLP tasks aim to not only comprehend individual words in isolation, but to also comprehend the context and topic of these words. As shown in Figure 1.1 NLP, artificial intelligence (AI), machine learning (ML), and deep learning (DL) are related concepts. NLP specifically pertains to the manipulation and interpretation of natural language by computers, whereas AI encompasses a broader range of technologies and techniques that empower machines to perform tasks that traditionally require human intelligence.ML is a subset of AI that focuses on algorithms and statistical models that allow machines to learn and improve their performance on specific tasks. Finally, DL is a type of machine learning that involves neural networks with many layers, enabling computers to identify complex patterns and relationships in data.

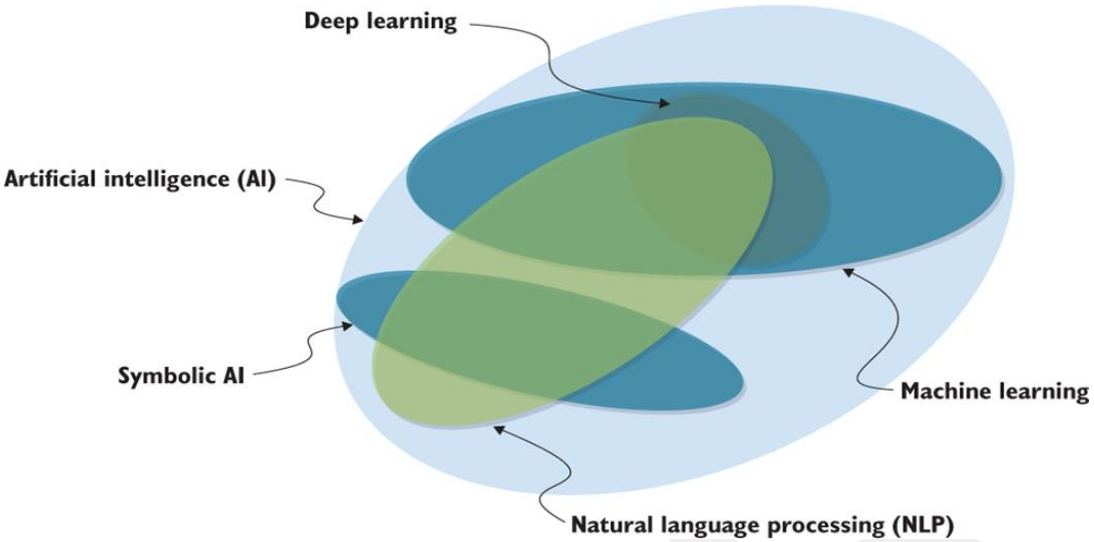

Figure 1.1: Natural language processing Venn diagram [1]

Similar to other branches of AI, early attempts at addressing NLP challenges, like classifying sentences and analyzing sentiment, relied on explicit rules. So, these rules could not be generalized and would easily break. By the advance of hardware and deep learning NLP tasks are popular. The ability to automatically create effective features in neural networks made it easier to apply these methods to different tasks and issues, which shifted human effort towards designing the appropriate neural network structure and configuring various hyperparameters during training [1].

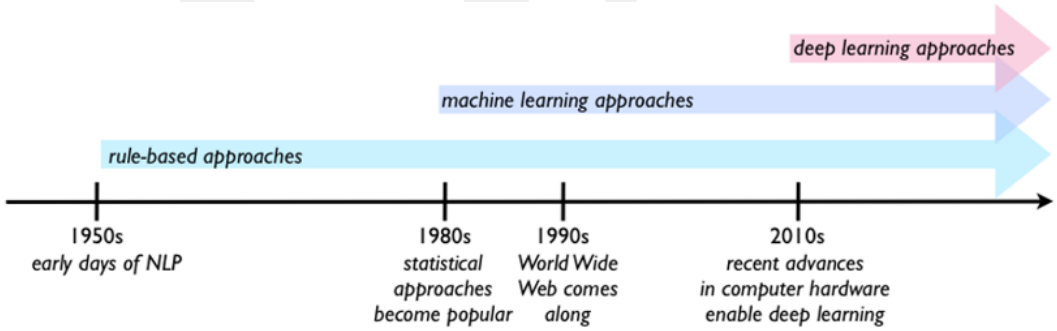

Figure 1.2: NLP approaches timeline visualizing three different types of approaches [2]

Sentiment Analysis is a commonly researched NLP task that aims to identify the emotional polarity of a text, without requiring a specific target to analyze. Its purpose is to detect the sentiment expressed in the text [3]. The sentiment analysis model is typically depicted by Equation 1.1, which takes a text T as input and produces a result that can be categorized as positive, negative, or neutral. However, the specific

output format of the sentiment result can vary, and may include binary polarity, multi-level polarity, or regression (a real-valued score) [3].

$$Sentiment(T) = \{Positive, Negative, Neutral\} \quad (1.1)$$

There are many definitions for Stance. From Longman Dictionary, stance is briefly, person's opinion about something. It is affected by everything according to a person's life, experience, culture, family, friends etc. For example, ethical or political stance.

Stance Detection is commonly acknowledged as a sub-task of sentiment analysis, with the objective of determining the viewpoint or position of a person towards a specific target. This target can be an explicitly mentioned or implied entity, concept, event, idea, opinion, claim, subject, or similar elements found within the text [4]. In contrast to sentiment analysis, stance detection primarily pays attention to determining author's position or opinion toward an object of evaluation, whether that opinion is supportive of (supporting) or opposed to (opposing) the issue. There are many other NLP tasks that related to stance detection and its sub-problems [5].

1.2 RECENT WORKS

1.2.1 In The World

Stance detection in social media is a recently emerged area in NLP, and there is still limited understanding of how language and social interactions influence users' stances. Stance-taking has long been studied in sociolinguistics, with the aim of analyzing an author's perspective through their written language. The primary objective of stance determination is to uncover the underlying viewpoint expressed by the writer in their text. This is accomplished by considering three key elements: linguistic actions, social interactions, and individual identity, and establishing the connection between these factors and the stance adopted by the author. [3].

The first of the important competitions for stance detection is SemEval-2016 Task 6. Mohammad et al. presented the dataset prepared for the competition in [4] Dataset on Twitter: "Atheism", "Climate Change Concern", "Donald Trump", "Feminist Movement", "Hillary Clinton" and "Legalization of Abortion" It consists of 4870 tweet-target pairs collected for their targets. Each tweet-target pair has been

flagged by at least eight people with the stance tags “Favor”, “Against” and “Neither”. In addition to the stance tag, the tweets were also tagged according to their targeting and emotional polarity. Attention was paid to the fact that there were enough tweets for each target in the dataset, that in addition to the tweets that clearly mentioned the target, there were enough tweets that the target was not clearly stated, and that there were enough tweets that expressed an opinion about a target other than the specified target in addition to the tweets that expressed an opinion on the target. The tasks and results related to the competition are explained in [6] by Mohammad et al.

It has been claimed that automating stance evaluation might be a useful first step in aiding human fact checkers in detecting false assertions [7]. As a result, the Fake News Challenge initiative organized a competition (FNC-1) to encourage the development of algorithms for automatically analyzing what a news source says about a certain problem [8]. The NLP community recognized the difficulty, and as a result, 50 teams comprising both academic and industry experts took part in the challenge. FNC-1 required building a system that assesses the stance towards the headline given a news article title and a news article body. One of the following stance labels might be assigned: 'agree,' 'disagree,' 'discuss,' or 'unrelated'. Table 1.1 displays four sample documents that demonstrate these categories. But most of the researchers only use “Favor”, “Against”, “None”. “Unrelated” tag is mostly omitted.

Table 1.1: Headline and text snippets from document bodies with respective stances from the FNC dataset [9]

Headline: Hundreds of Palestinians flee floods in Gaza as Israel opens dams	
Agree (AGR)	GAZA CITY (Ma'an) – Hundreds of Palestinians were evacuated from their homes Sunday morning after Israeli authorities opened a number of dams near the border, flooding the Gaza Valley in the wake of a recent severe winter storm. The Gaza Ministry of Interior said in a statement that civil defense services and teams from the Ministry of Public Works had evacuated more than 80 families from both sides of the Gaza Valley (Wadi Gaza) after their homes flooded as water levels reached more than three meters [..]
Discuss (DSC)	Palestinian officials say hundreds of Gazans were forced to evacuate after Israel opened the gates of several dams on the border with the Gaza Strip, and flooded at least 80 households. Israel has denied the claim as "entirely false". [..]
Disagree (DSG)	Israel has rejected allegations by government officials in the Gaza strip that authorities were responsible for released storm waters flooding parts of the besieged area. "The claim is entirely false, and southern Israel does not have any dams," said a statement from the Coordinator of Government Activities in the Territories (COGAT). "Due to the recent rain, streams were flooded throughout the region with no connection to actions taken by the State of Israel." At least 80 Palestinian families have been evacuated after water levels in the Gaza Valley (Wadi Gaza) rose to almost three meters. [..]
Unrelated (UNR)	Apple is continuing to experience 'Hairgate' problems, but they may just be a publicity stunt [..]

Stance Detection target can be formulated as in equation 1.2 with T text or U user and G stance tag since the purpose of Automatic Stance Detection is to automatically classify the author's stance towards a predetermined target with one of the stance tags [3].

$$Stance(T, U \mid G) = \{Favor, Against, None\} \quad (1.2)$$

There are datasets in the literature mainly in English, but also in Spanish, Italian, Japanese, Arabic, Russian, Chinese, Catalan, English-Hindi and Turkish. These datasets consist of data collected from online forums in the early days of stance detection studies and later from microblogs [5].

1.2.2 In Turkish

Numerous studies in the literature have demonstrated that Turkish presents unique challenges for NLP tasks, owing to its complex inflectional and derivational

structure, which distinguishes it from morphologically simpler languages. As a result, previous research on Turkish NLP has served as a model for similar languages [10]. However, stance detection studies on Turkish are still in their early stages, with only a few investigations conducted on detecting stance in social media and blog posts.

Dilek Küçük in [11] presented a dataset labelled Turkish stance in her study. Galatasaray and Fenerbahçe, which are popular sports clubs in Turkey, were determined as the target for this data set. For these targets, a data set with the “Favor” and “Against” stance tags was created by collecting tweets. Tweets are tagged by only one person, and the dataset contains 700 target-tweet pairs for each target, with 175 tweets with the tag "Favor" and 175 tweets with the hashtag "Against". As a feature, the Support Vector Machine method was applied with 10-fold cross validation using unigram and hashtag information. High performance has been achieved compared to similar studies in literature.

Dilek Küçük and Fazlı Can in [12] defined the data set presented in [11] as version 1 in their study and the data set was labelled by a second person. Version 2, consisting of a total of 686 target-tweet pairs, was created by taking tweets marked with the same tag by two people. To expand the dataset, 400 more tweets were collected, and 379 tweets marked with the same stance tag by two people were added, and version 3, consisting of a total of 1065 target-tweet pairs, was created. Using 3 versions of unigrams, bigrams, hashtags, links, emoticons (such as “<3”, “:(”) and entity names, the Support Vector Machine method was applied with 10-fold cross validation. The use of unigram and hashtag features in 3 versions is similar. Compared to other studies, links and expressions of emotion did not contribute to the performance of the model. The information of entity names, including person, place, and organization names, was added by marking them manually with the NER Tool. Using entity names did not increase the performance of the model.

In [13] Polat et al. targeted to create a dataset for stance detection on Turkish language. Dataset was collected from a famous Turkish blog site “eksisozluk” that has no limit for word usage. Dataset includes various topics. These are “E-Book”, “Working from home”, “Mask”, “E-Cigarette”, “Vaccine”, and “Vegan”. Dataset has 5031 blog posts with an unequal number of topics. Bag of Words, Term Frequency – Reverse Document Frequency and Word embedding techniques were employed to represent textual data. The study presents an analysis of stance detection outcomes

using several machine learning techniques, including Naive Bayes, Support Vector Machine, AdaBoost, XGBoost, Random Forest, and Convolutional Neural Networks, with performance evaluated through the Matthews Correlation Coefficient. The results indicate that XGBoost and Convolutional Neural Network approaches perform the best. Additionally, the study explores the contribution of features used in the Convolutional Neural Network model to prediction performance by applying the integrated gradients method to the extracted features.

In [14] Doğan Küçük and Nursan Arıcı created a Turkish dataset from Twitter about COVID-19 Vaccination. They analyzed this dataset for both sentiment analysis and stance detection. They collected their dataset on two different dates. 300 tweet is collected for Part-1 on 12/18/2020 and 300 tweet is collected for Part-1 on 07/18/2021. A single native Turkish annotator labeled the dataset for both sentiment and stance classes, with the target for the stance detection task being COVID-19 Vaccination. The Part-1 dataset had 122 tweets labeled as Favor, 123 as Against, and 55 as None. The Part-2 dataset had 137 tweets labeled as Favor, 122 as Against, and 41 as None. The training and testing process utilized SVM and Random Forest methods, with 10-fold cross-validation. The limited number of tweets used for training at each fold affected the performance rates. The feature set employed for stance detection included unigrams (single words), the usage of hashtags, and the presence of emoticons. The results indicated that Support Vector Machines (SVM) outperformed Random Forest, with SVM learners demonstrating consistent performance rates across both segments of the dataset.

1.3 RESEARCH AIM

Stance Detection has gained an important place among natural language processing studies. Stance labelled datasets were created in many different languages, especially in English, and stance detection studies were carried out on these datasets. However, the dataset in non-English languages is extremely limited. The increase in the number of social media tools, forums and news sites has led to an increase in scientific publications on stance detection. Today, stance detection studies gain importance as the first step of determining the opinion of the public and detecting fake news. In this respect, existing resources and studies on Turkish are quite limited. Therefore, in this study, it is aimed to create a comprehensive dataset and source for

Turkish stance detection and analyze the success of state-of-the-art natural language processing algorithms.

1.4 HYPOTHESIS

In this study, a comprehensive Turkish stance labelled data was created, and stance detection study was tried to be done on this dataset with various methods. In this direction, data were collected from twitter about trending topic Russia - Ukraine War for two targets. Stance information was added to these collected data. Then, the data was structured by applying the necessary pre-processes. After that machine learning, deep learning and transformers methods were applied with two different text representations on the created dataset. After this stage, the model performances used were discussed.

CHAPTER II
DATASET

In this study, a data set was prepared to detect Turkish stance. The data were collected from tweets sent on the day of the start of the Russia-Ukraine war and the previous day (23 - 24 February 2022). Turkey has both good relations with those countries, there is a notable twitter user in Turkey [15] and this subject is trending topic in those days therefore we preferred this subject for our study.

We scraped tweets that have words “Rusya” or “Ukrayna”. These are also our two separate targets for Turkish stance detection. There were nearly half a million tweets those days with these words. Before labelling data, we have done preprocessing to clean our data for the first stage. In this stage we have removed other information than the tweet, duplicates, non-Turkish tweets, and links that are useless for our study.

After that preprocessing different for each target, each tweet was labelled with one of the tags “Favor”, “Against” by 4 university graduates. We have omitted tweets that have no stance. In Table 2.1, sample target-tweet pairs in the Turkish Stance Labelled Data Set are shown. There is no character limitation in the collected tweets. Table 2.2 shows the word count of the shortest tweet, the average word count of the tweets, and the word count of the longest tweet for each target. At the end of the labelling, we labelled a total of 8215 tweets, 3264 for Ukraine and 4951 for Russia. The generated dataset shows unbalanced class distribution for the targets. The distribution of label numbers for each target is shown in Figure 2.1.

Table 2.1: Sample Target-Tweet-Stance

Target	Tweet	Stance
Russia	İşgalci Rusya hesap vereceksin! (EN) (Invading Russia will give account!)	Against
Russia	Rusya'nın operasyonu işgal değil NATO tehdidine karşı savunmadır. (EN) (Russia's operation is not an invasion, but a defense against the NATO threat.)	Favor
Ukraine	Komedyenden devlet başkanı seçersen, böyle komedi gibi devlet yönetimiyle karşılaşırsın. (EN) (If you choose a comedian for the head of state, you will encounter a state administration like comedy.)	Against
Ukraine	Savaşın her türlüsüne hayır. Ancak dünya burda iki yüzlülüğünü gösterdi, Rusya'ya tepki gösterenler bu yavruların anasını babasını yetim bırakan devletlerdir, Tüm dünya şuan Ukrayna için kenetlendik çok güzel, inşallah birgün bu çocuklar için de bütün dünya kenetleniriz... (EN) (No to any kind of war. However, the world showed their hypocrisy here, those who reacted to Russia are the states that orphaned the parents of these puppies, The whole world is now united for Ukraine, it's very nice, I hope one day we will unite for these children as well...)	Favor

Table 2.2: Word Count Info

Target	Minimum Number of Words	Maximum Number of Words	Average Word Count
Russia	2	46	20.8
Ukraine	2	46	21.3

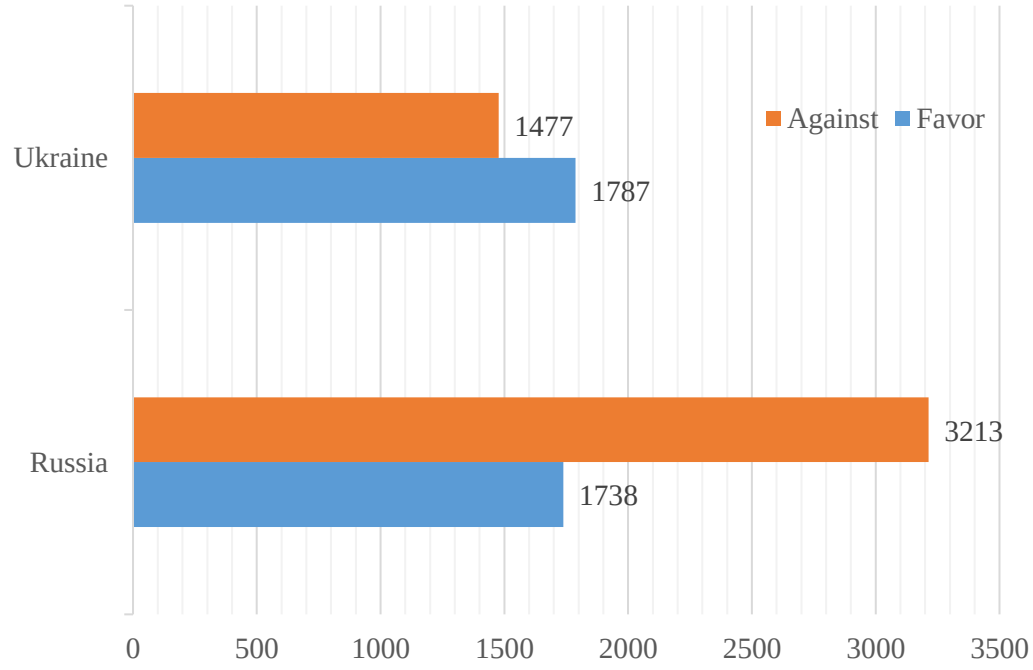

Figure 2.1: Number of Tweets

Before classification dataset have been preprocessed for the second stage. For this stage, punctuation marks, numeric values, emojis, special characters, hashtags, extra spaces, and conjunction words in the text have been removed, all letters have been converted to lowercase.

Stemming is mostly recommended for preprocessing in Turkish NLP tasks [16], but on the other hand stemming is not suggested for embeddings. Word embeddings are compact vector representations of words that encode both semantic and syntactic relationships among words. They are specifically designed to operate with the complete forms of words, rather than their stemmed or truncated versions. By utilizing word embeddings, we can capture and represent the rich associations and contextual meanings of words within a given language [17].

CHAPTER III
METHODOLOGY

We are looking in this study to see the accuracy of various models on the data set we have created. We have experimented with multiple models and evaluated accuracies. The complete process is explained in figure 3.1.

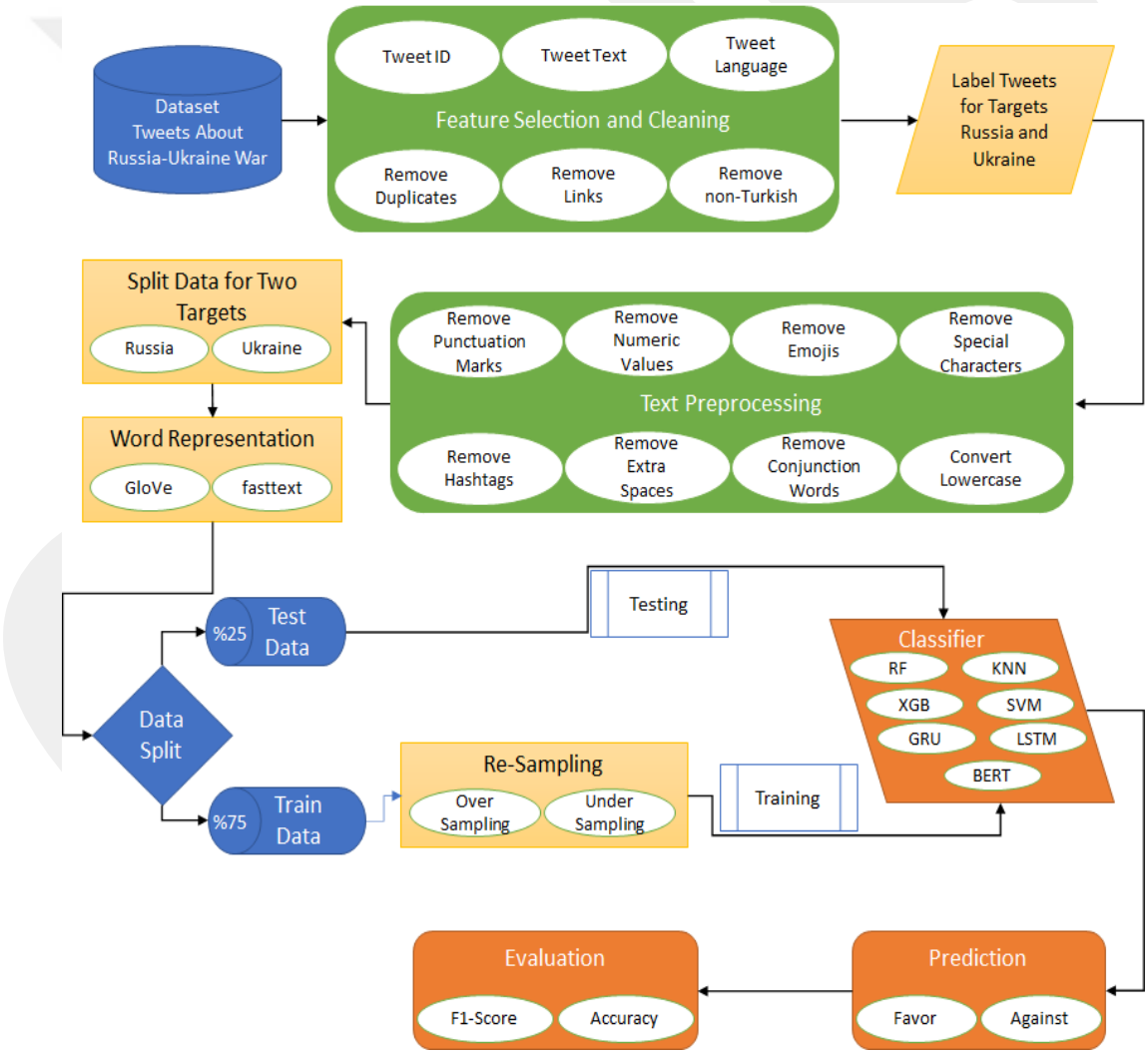

Figure 3.1: Complete Process of Study

3.1 TEXT REPRESENTATIONS

Text representations are a critical component of machine learning models for natural language processing tasks. Text datasets are typical examples of unstructured data. A classifier or a strategy for developing a classifier cannot directly interpret texts. These unstructured texts patterns must be turned into a structured representation before using any mathematical modeling as part of a classifier. These representations encode each word in a text as a vector of numbers, where each element of the vector captures a different aspect of the word's meaning or context [18]. The vectors are then fed into a machine learning model as input features. In this study we have used two popular word representation techniques GloVe and fastText.

3.1.1 Pre-Trained Word Vectors

3.1.1.1 GloVe

GloVe [19], short for "Global Vectors for Word Representation," is a word representation technique widely employed in natural language processing. Following Word2Vec [20] in natural language processing. It was developed by Pennington et al at Stanford University. GloVe is a method that trains on global word-to-word counts, thus allowing statistics to be used more effectively.

GloVe represents each word as a vector, with the dimensions of the vector representing different semantic relationships between the words. For example, one dimension may represent the semantic relationship between the words "king" and "queen".

GloVe has demonstrated its effectiveness in various NLP tasks and has versions for different NLP tasks. In this study GloVe Twitter 27B pre-trained word vector version with 200 dimensions was used for word representation.

3.1.1.2 FastText

FastText [21], developed by Facebook in 2016, an extension of Word2Vec that incorporates sub word information into the vector representation of a word. Instead of providing individual words as input to the neural network, the approach involves dividing words into "n-grams," which are formed by grouping together several letters. For example, for the word “fast”, trigram is “fas” and “ast”. In the N-gram expression, n represents the degree of repetition. In other words, the n expression here provides

that we will divide by how many times. It allows us to understand how much of a word or letter. Fast's word vector is the sum of all these n-gram vectors. After the training is complete, we will have the word vectors for all the n-grams given in the training set.

FastText offers pre-trained word vectors in many languages, including Turkish. In the study, pre-trained word vectors provided by FastText were used as text representations. These representations consist of 300-dimensional word vectors trained on Wikipedia [22].

3.2 RESAMPLING

An imbalanced dataset and classifying it is another problem area for statisticians and machine learning community. Classifying imbalanced dataset mostly overfits for. The classifier typically ignores samples in the minority class, as they are often deemed mislabeled [23]. Several methods attempt to address the imbalanced class distribution in the training dataset, such as increasing the number of instances in the minority class through oversampling or reducing the instances in the majority class through undersampling [24].

We have implemented oversampling using the RandomOverSampler and undersampling using the RandomUnderSampler [25].

3.3 CLASSIFIERS

3.3.1 K-Nearest Neighbor

K-Nearest Neighbor (KNN) is a supervised machine learning algorithm that can be used for text classification. The algorithm is a non-parametric method that relies on the similarity between the features of the training data and the new inputs to make a classification decision. In KNN, the user chooses the number of neighbors (k) to examine when making a prediction for a new observation [26]. The algorithm then calculates the distance between the new observation and all training observations and selects the k nearest neighbors based on this distance measure. The most common distance measure used in KNN is the Euclidean distances, shown in equation 3.1, but other measures such as Manhattan distance and cosine distance can also be used.

$$\sqrt{\sum_{i=1}^{k}(x_i - y_i)^2} \tag{3.1}$$

3.3.2 Random Forest

The Random Forest is a type of machine learning model that improves prediction accuracy by using a combination of decision trees. The approach of using decision trees in parallel was first introduced by [27] in 1995. As shown in Figure 3.2 Random Forest model is a combination of decision trees that are created using a dataset that has been randomly split and relies on the divide-and-conquer approach. Each decision tree is created using a random subset of the data and a feature selection method, such as information gain or the Gini index. In a classification problem, each tree casts a vote, and the most popular class is selected as the final result. In a regression case, the mean of all tree outputs is assumed to be the result. Compared to other non-linear classification techniques, Random Forest is easier to understand and more powerful.

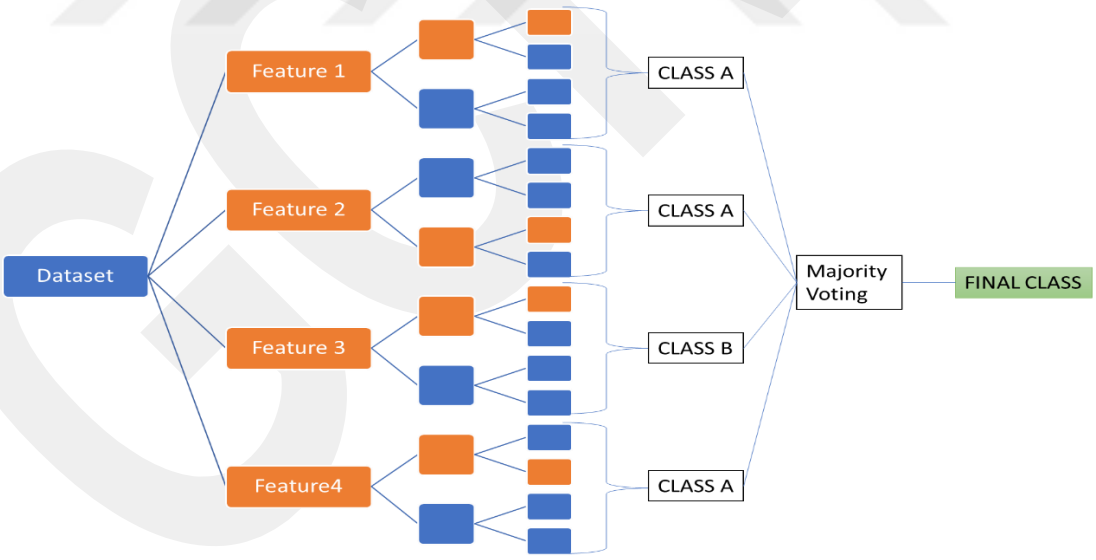

Figure 3.2: Random Forest Architecture

3.3.3 Support Vector Machines

Support Vector Machines (SVM) was originally proposed by [28]. Although SVM was designed for binary classification it was later adapted for regression and multi classification. In a problem where the data cannot be separated linearly, the best

decision boundary between the samples is tried to be determined by moving the samples in the data set to a higher dimensional space where they can be linearly separated with the help of the kernel function.

SVM is frequently used in various studies because it works fast, gives good results in high-dimensional data, does not overfit, is not affected much by outliers, and gives successful results in general. One of these studies is text classification. Successful results are obtained with the SVM method because text classification problems are generally linearly separable, consist of high-dimensional input space, contain few unrelated features, and document vectors usually consist of zeros [29].

To find the most effective hyperplane for separating data, it is essential to maximize the margins between the support points in the training dataset. This has several advantages, including better empirical performance, reduced misclassification risk even if the boundary is slightly off, improved classification accuracy, and avoiding local minima[30]. SVM uses a hyperplane to separate data, but it can also use a kernel approach to handle non-linear boundaries. The kernel maps the input data to a space with a high number of dimensions, where it can be separated linearly. The kernel does this by transforming the data into a feature space that enables the construction of a similarity metric using the dot product. Figure 3.3 shows the non-linear and linear classifier which is used for 2 − dimension datasets.

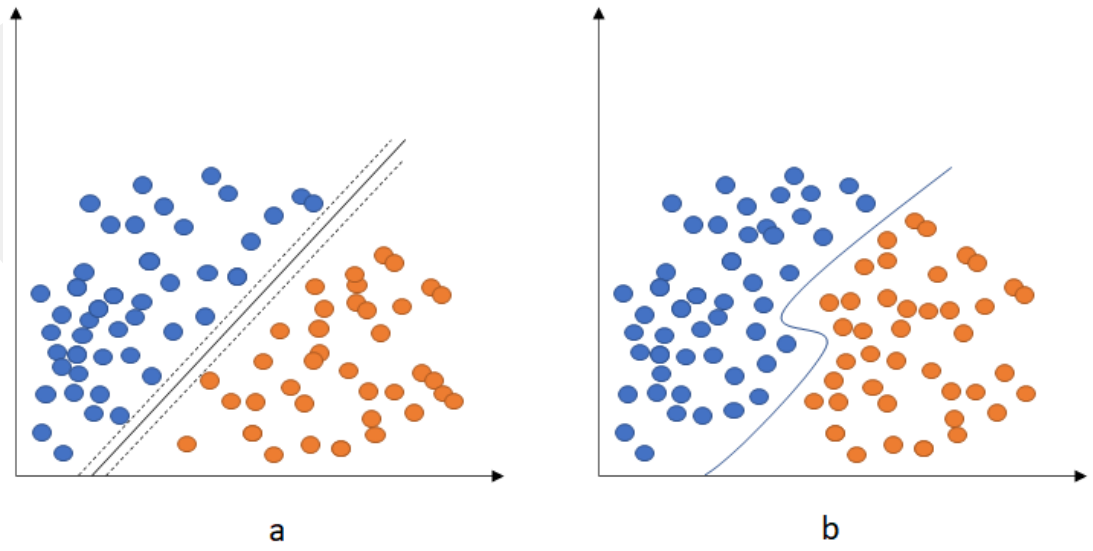

Figure 3.3: The illustration of (a) SVM with linear decision hyper-plane, and (b) nonlinear decision hyper-plane that uses kernel function [31]

3.3.4 XGBoost

XGBoost is a popular machine learning algorithm that uses a gradient boosting framework to make predictions [32]. The algorithm iteratively trains a series of decision trees to improve the accuracy of the predictions. Each tree is trained to correct the errors of the previous tree, and the final prediction is a weighted sum of the predictions from all the trees.

XGBoost has been one of the most used methods because it has high performance, works very fast, contains solutions to prevent excessive learning, and can be used for regression and classification problems. XGBoost uses gradient boosting to make stronger predictions using many weak learners. Gradient boosting is a collective learning method often used in regression problems. The basic idea of gradient boosting is to optimize the differentiable loss functions of weak classifiers to obtain a stronger estimation [33]. It works similarly to the XGBoost gradient boost method. There are minor differences between them. In the XGBoost algorithm, calculations are made by creating trees in parallel, so it works much faster and there are fixes that control over-learning. XGBoost tries to find the best parameters for the training samples.

3.3.5 Long Short-Term Memory

The Long Short-Term Memory (LSTM) neural network model was proposed by Hochreiter and Schmidhuber in their study [34] in 1997 as a kind of recursive neural network model together with a gradient-based learning algorithm. Today, it is frequently used to solve many different problems. It has been successful in producing positive outcomes in the area of natural language processing. LSTM is designed to solve the problem of long-term addiction. With a model created to predict the next word, the last word of a sentence can be predicted without any further context. But real-life problems are not that simple. There are situations where other contexts are needed. For example, in the sentence “I was born in Turkey … my mother tongue is Turkish”, the last word may be guessed to be a language, but more information is needed to guess which language it is. LSTM is successful in solving this problem. Gradients may be lost during backpropagation when a long sentence is used as input, so learning does not occur. LSTM avoids this problem thanks to its design. The reason why the LSTM model is valuable is the cell state used in its design and the various

structures called gates. Thanks to the cell state, information that may be meaningful in model prediction can be carried throughout the model. We can call this the memory of the model. The gate, on the other hand, is the structure that decides what information should be forgotten and what information should be carried forward with its activation functions. Basically, an LSTM cell consists of 4 structures. These are the forget gate, the input gate, the output gate, and the cell state. Figure 3.4 shows the structure of the LSTM cell.

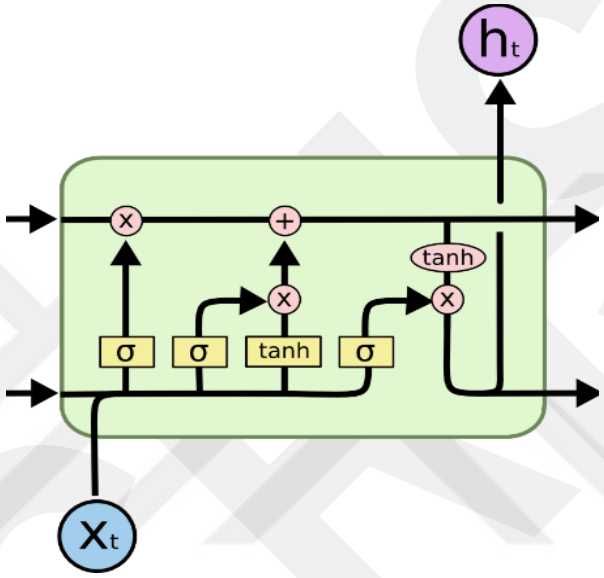

Figure 3.4: LSTM Cell Architecture [35]

In the LSTM cell, there is the forget gate first. As the name suggests, it is the place where it is decided which information will be removed from the cell. For this, it decides with the sigmoid activation function by looking at the previous hidden state and the current input.

After the forget gate, it is decided which information to keep for the cell state. Initially, the input gate decides which values to update with sigmoid. After that, vectors are created for values that can be added to the cell state by using the tanh function. Then these two pieces of information are combined.

Finally, output is chosen. This output, though filtered, will be based on the state of the cell. A sigmoid layer is run first to determine which parts of the cell state will be output. It is then decided to multiply by the output of the sigmoid gate after passing the cell state through tanh, to output only the partitions (to push the values to be between -1 and 1).

3.3.6 Gated Recurrent Unit

GRU is a type of recurrent neural network (RNN) and simpler alternative to the LSTM architecture [36]. Like LSTM, GRU is designed to overcome the vanishing gradient problem in traditional RNNs by allowing the network to selectively forget or remember information from previous time steps.

As shown in figure 3.5 input and forget gates are combined to form an "update gate." Along with making other adjustments, it merges the hidden state and cell state. The resulting model, which is relatively simpler than the traditional LSTM model, has been gaining popularity.

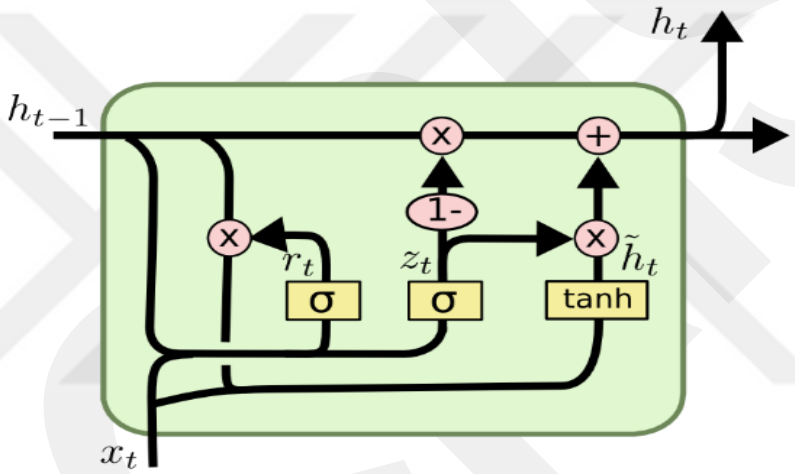

Figure 3.5: GRU Cell Architecture [35]

3.3.7 BERT

Transformers are a type of neural network architecture for natural language processing tasks. They are based on the self-attention mechanism [37], which allows the network to selectively attend to different parts of the input sequence during processing. The key idea behind transformers is to compute a matrix of attention weights that represents the importance of each input token with respect to all the other tokens. This matrix is then used to weight the input embeddings and compute the output sequence.

Bidirectional Encoder Representations from Transformers (BERT)[38] is a pre-trained transformer-based language for various tasks related to natural language processing. It is bidirectional, meaning that it considers both the left and right context

of each token during training, that is called Masked Language Model (MLM). Next Sentence Prediction (NSP) method is designed to predict whether a given sentence follows the previous sentence or not, where the goal is to understand the relationships and dependencies between sentences in a text. It has achieved state-of-the-art performance on many benchmarks and has become a popular choice for natural language processing tasks.

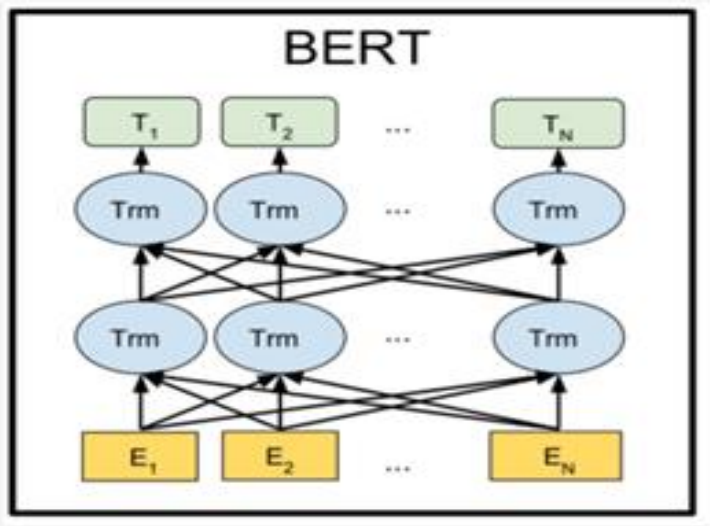

Figure 3.6: BERT Architecture [38]

The BERT model employs an attention mechanism to comprehend the contextual relationships between words in a text, along with its transformative structure, to facilitate NLP tasks. The transformer structure comprises an encoder that reads text inputs and a decoder that produces task predictions. The BERT model processes a text string with a maximum length of 512 words and produces a data representation. Tokenization is performed in two steps, namely pretext normalization and punctuation separation, using the WordPiece token [39]. The tokenized sequence is marked with a [CLS] token at the start of each sentence and a [SEP] token at the end of each sentence. Text classification is performed by using the last hidden h state of the first token ([CLS]) as a representation of the resulting token sequences [40].

The BERT base model has 12 layers, each with 768 hidden units, and 12 self-attention heads. The BERT-large model, on the other hand, has 24 layers, each with 1024 hidden units, and 16 self-attention heads.

After BERT model was proposed, similar models trained with many different variations were introduced to the literature [41,42]. There are limited number of variations for Turkish language. The Turkish BERT model (BerTurk) [43], which was brought to the literature to eliminate this deficiency, is a model whose pre-training was done on the Turkish corpus. BerTurk pre-trained on Oscar Corpus, Opus Corpora, and Wikipedia dump. BerTurk models differ in 32K and 128K vocabulary size, both have cased and uncased versions.

3.4 EVALUATION

In literature, to compare results recall shown in equation 3.10, precision shown in equation 3.11, and accuracy shown in equation 3.12 obtained from confusion matrix (Table 3.1), is mostly used. However, the class imbalance has a large effect on measurements and dominant class has negatively affects accuracy. Therefore, F1 score shown in equation 3.13, which is the harmonic mean of precision and recall is commonly used to compare results in studies on unbalanced dataset.

Table 3.1: Confusion Matrix

	Actual		
Predicted		Positive	Negative
	Positive	True Positive (TP)	False Positive (FP)
	Negative	False Negative (FN)	True Negative (TN)

$$Recall = \frac{TP}{TP + FN} \tag{3.10}$$

$$Precision = \frac{TP}{TP + FP} \tag{3.11}$$

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{3.12}$$

$$F1\ Score = \frac{2 * Precision * Recall}{Precision + Recall} \tag{3.13}$$

F1 score for each class was calculated by model but there are three type of F1 score for calculating the model success. These are Macro-Averaged, Micro-Averaged, Weighted-Averaged F1 Score.

Macro-Averaged F1 Score: The macro-averaged F1 score calculates the F1 score for each class independently and then takes the average across all classes. Each class is given equal weight in the calculation, regardless of its frequency or sample size. This approach treats each class equally and is useful when you want to assess the model's performance across all classes.

Micro-Averaged F1 Score: The micro-averaged F1 score calculates the F1 score by considering the total number of true positives, false negatives, and false positives across all classes. It aggregates the counts across all classes and then computes the F1 score. This approach gives more weight to classes with a larger number of instances, making it suitable for imbalanced datasets or situations where the focus is on overall performance across all classes.

Weighted-Averaged F1 Score: The weighted-average F1 score is similar to the macro-averaged F1 score but takes into account the class imbalance by incorporating the class weights. Each class's F1 score is weighted by the proportion of samples in that class. It provides a balanced evaluation that considers both the performance of each class and their relative importance based on their distribution in the dataset.

The Weighted-Averaged F1 Score is particularly useful when dealing with imbalanced datasets. In imbalanced datasets, the number of samples in different classes may vary significantly, leading to challenges in evaluating the performance of a classification model. The Weighted-Averaged F1 Score considers the class imbalance by incorporating class weights. According to the dataset we have the weighted average is more suitable to evaluate the model.

CHAPTER IV
RESULTS AND DISCUSSION

On the Turkish stance-labeled dataset, binary classification was performed separately for each target using several machine learning algorithms: k-Nearest Neighbor, Random Forest, Support Vector Machines, XGBoost, Long Short-Term Memory, Gated Recurrent Unit, and BERT methods. To represent words in the dataset, two different word embeddings, namely GloVe and fastText, were applied.

During the application of fastText embeddings, no out-of-vocabulary words were encountered. However, while applying GloVe embeddings, any out-of-vocabulary words were assigned a value of zero.

For the k-Nearest Neighbor classifier, the parameter "n_neighbour" was set to 4, and the "metric" parameter was set to Euclidean distance. Random Forest and Support Vector Machines classifiers were applied to the dataset with their default parameters.

The XGBoost classifier was trained with the following parameters as specified in Table 4.1.

Table 4.1: XGBoost Parameters

Parameter	**Value**
learning_rate	0.1
max_depth	7
n_estimators	80
use_label_encoder	False
eval_metric	auc

LSTM model architecture was established by Keras Sequential model which allows for creation of a linear stack of layers. The first layer added to the model was an Embedding layer, established with Glove or fastText and responsible for mapping the input tokens to dense vectors of fixed size. It takes three parameters: "vocab_len" (the size of the vocabulary), "emb_dim" (the dimension of the embedding space), and trainable (specifying whether the embedding weights should be updated during

training). The weights parameter is set to “embedding_matrix”, which represents the pre-trained word embeddings used for initialization.

Next, an LSTM layer with 128 units is added to the model. The “return_sequences” parameter is set to “False”, indicating that only the final output of the LSTM sequence is returned.

A Dropout layer with a rate of 0.5 is added after the LSTM layer. Dropout helps prevent overfitting by randomly dropping out a fraction of the connections between neurons during training.

Finally, a Dense layer with a single unit and a sigmoid activation function is added to produce the binary classification output.

The model is compiled using the binary cross-entropy loss function (binary_crossentropy), which is commonly used for binary classification tasks. The Adam optimizer is used to optimize the model's weights. The accuracy metric is specified to be tracked during training.

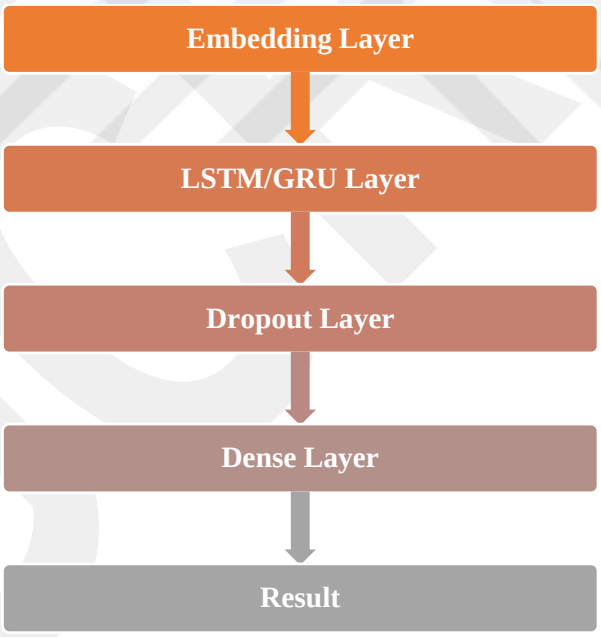

Figure 4.1: LSTM and GRU Model Architecture

GRU Model is established as same as LSTM Model with same parameters.

Finally, the BERTurk Model was applied to the dataset on Google Colab with GPU. In this model the optimizer is initialized using the Adam algorithm to update the parameters of the model during training with learning rate 5e-5 and epsilon 1e-8. The Adam optimizer is a powerful tool that can be used to train deep learning models. The

learning rate is a hyperparameter that controls how quickly the model learns. The epsilon value is a hyperparameter that controls the stability of the Adam optimizer. By feeding the model with 32 inputs, the training was carried out in 4 epochs.

As shown in Table 4.1 the accuracy and F1 scores for various classifiers that were trained on two datasets - one for Russia and one for Ukraine - using two different word embedding methods: fastText and GloVe. The classifiers used were K-Nearest Neighbors, Random Forest, Support Vector Machine, XGBoost, Gated Recurrent Unit, Long Short-Term Memory, and BERT. As a first observation, the results show that the classifiers performed better on the Ukraine dataset compared to the Russia dataset. This could be because the Ukraine dataset has more balanced data for training compared to the Russian dataset.

Regarding the classification models, SVM showed the best performance among the models for both targets. This result is consistent with previous studies on stance detection, where SVM has been shown to perform well in various languages and domains. This may be due to the fact that SVM is a simple yet effective model that can handle high-dimensional data and is less prone to overfitting.

The performance of the other models, such as K-Nearest Neighbor (KNN), Random Forest (RF), and XGBoost, varied depending on the target and the embedding used. For example, RF with fastText embeddings achieved competitive results for the Ukraine target but not for the Russia target. This may be due to the fact that RF is a non-linear model that can capture complex patterns in the data, but it may not be as effective in capturing the subtle nuances of stance expressions in tweets.

The LSTM and GRU models, which are variants of Recurrent Neural Networks (RNNs), performed relatively well for the Ukraine target, achieving competitive results compared to SVM with fastText embeddings. This result may be due to the fact that RNNs are designed to handle sequential data and are able to capture the temporal dependencies in the tweets.

Finally, the BERT model, which is a state-of-the-art language model based on transformers, outperformed all other models, achieving the highest accuracy and F1-score for both targets. This result is consistent with recent studies that have shown the effectiveness of BERT in various natural language processing tasks [38]. BERT is a pre-trained language model that can be fine-tuned on specific tasks, such as stance detection, and it is able to capture the contextual information of the tweets.

Table 4.2: Results

		Russia		Ukraine	
Classifier	Embeddings	Accuracy	F1-score	Accuracy	F1-score
KNN	fastText	0.632	0.639	0.593	0.564
KNN	GloVe	0.588	0.597	0.640	0.610
RF	fastText	0.696	0.639	0.760	0.759
RF	GloVe	0.672	0.585	0.749	0.746
SVM	fastText	0.721	0.686	0.809	0.809
SVM	GloVe	0.700	0.645	0.767	0.767
XGBoost	fastText	0.717	0.693	0.782	0.782
XGBoost	GloVe	0.691	0.651	0.771	0.769
LSTM	fastText	0.713	0.712	0.752	0.752
LSTM	GloVe	0.680	0.687	0.807	0.807
GRU	fastText	0.683	0.684	0.755	0.755
GRU	GloVe	0.687	0.685	0.807	0.807
BERT		**0.784**	**0.787**	**0.872**	**0.870**

The oversampling technique aims to balance the number of samples in each class by generating synthetic samples of the minority class to match the number of samples in the majority class. On the other hand, the undersampling technique reduces the number of samples in the majority class to match the number of samples in the minority class.

We applied the same models as in the initial experiments to the oversampled and undersampled datasets and evaluated their performance using the same metrics. As shown in Table 4.2 the results showed that the oversampling technique improved the performance of most models for both targets, while the undersampling technique did not result in significant improvements.

For the Russia target, the best performing model with oversampling was the BERT, which achieved an accuracy of 0.774 and an F1-score of 0.776. For the Ukraine target, the best performing model with undersampling was again the BERT model, which achieved an accuracy of 0.835 and an F1-score of 0.834.

The results show oversampling and undersampling had mixed effects on the performance of the classifiers. Some models showed improvement in accuracy and F1-score with oversampling or undersampling, while others showed a decrease in performance. For example, the KNN classifier with undersampling using GloVe embeddings had a higher accuracy and F1-score for Ukraine, while the KNN classifier

with undersampling using GloVe embeddings had a lower accuracy and F1-score for Russia. On the other hand, the Random Forest classifier with undersampling using either GloVe or fastText embeddings consistently showed detoriation in accuracy and F1-score for both countries. On the contrary, the Support Vector Machine classifier with oversampling using either GloVe or fastText embeddings consistently showed improvement in accuracy and F1-score for target Russia.

It is important to note that the effectiveness of oversampling and undersampling can depend on the dataset and the specific classifier being used. Therefore, it is important to experiment with both techniques to determine which one provides the best performance for a particular task.

Table 4.3: Results with Resampling Target Russia

		Russia	
Classifier	**Embeddings**	**Accuracy**	**F1-score**
KNN oversampling	fasttext	0.573	0.576
KNN oversampling	GloVe	0.557	0.561
KNN undersampling	fasttext	0.552	0.547
KNN undersampling	GloVe	0.531	0.520
RF oversampling	fasttext	0.712	0.681
RF oversampling	GloVe	0.699	0.657
RF undersampling	fasttext	0.662	0.668
RF undersampling	GloVe	0.655	0.662
SVM oversampling	fasttext	0.728	0.732
SVM oversampling	GloVe	0.715	0.720
SVM undersampling	fasttext	0.732	0.737
SVM undersampling	GloVe	0.694	0.700
XGBoost oversampling	fasttext	0.712	0.699
XGBoost oversampling	GloVe	0.705	0.690
XGBoost undersampling	fasttext	0.679	0.685
XGBoost undersampling	GloVe	0.661	0.668
LSTM oversampling	fasttext	0.697	0.699
LSTM oversampling	GloVe	0.680	0.687
LSTM undersampling	fasttext	0.667	0.671
LSTM undersampling	GloVe	0.668	0.673
GRU oversampling	fasttext	0.680	0.681
GRU oversampling	GloVe	0.674	0.680
GRU undersampling	fasttext	0.655	0.659
GRU undersampling	GloVe	0.686	0.690
BERT oversampling		**0.774**	**0.776**
BERT undersampling		0.740	0.746

Table 4.4: Results with Resampling Target Ukraine

		Ukraine	
Classifier	**Embeddings**	**Accuracy**	**F1-score**
KNN oversampling	fasttext	0.599	0.581
KNN oversampling	GloVe	0.625	0.606
KNN undersampling	fasttext	0.602	0.583
KNN undersampling	GloVe	0.641	0.626
RF oversampling	fasttext	0.754	0.754
RF oversampling	GloVe	0.754	0.751
RF undersampling	fasttext	0.745	0.746
RF undersampling	GloVe	0.744	0.744
SVM oversampling	fasttext	0.805	0.805
SVM oversampling	GloVe	0.775	0.775
SVM undersampling	fasttext	0.803	0.803
SVM undersampling	GloVe	0.763	0.764
XGBoost oversampling	fasttext	0.773	0.773
XGBoost oversampling	GloVe	0.762	0.762
XGBoost undersampling	fasttext	0.761	0.762
XGBoost undersampling	GloVe	0.768	0.769
LSTM oversampling	fasttext	0.781	0.781
LSTM oversampling	GloVe	0.795	0.796
LSTM undersampling	fasttext	0.766	0.765
LSTM undersampling	GloVe	0.790	0.790
GRU oversampling	fasttext	0.796	0.797
GRU oversampling	GloVe	0.819	0.820
GRU undersampling	fasttext	0.761	0.761
GRU undersampling	GloVe	0.786	0.787
BERT oversampling		0.813	0.808
BERT undersampling		**0.835**	**0.834**

CHAPTER V
CONCLUSION AND FUTURE WORK

5.1 CONCLUSION

In this study, we aimed to develop an effective stance detection model for newly created Turkish tweets related to the Russia-Ukraine conflict. Detecting the stance of social media users towards this conflict is of significant interest for researchers, policymakers, and journalists. However, this task is challenging due to the unbalanced nature of the dataset, where one stance is dominant compared to the other. To address this issue, we explored several machine learning algorithms and techniques to develop an effective and efficient model.

We experimented with six different machine learning algorithms, including support vector machines, random forest, k-nearest neighbor, XGBoost, LSTM, and GRU with word embeddings fastText and glove. To balance the unbalanced dataset, we applied both undersampling and oversampling techniques. Our experimental results showed that support vector machines with fastText and undersampling data achieved the best performance for detecting the stance of tweets related to Russia, with an F1 score of 0.738. For detecting the stance of tweets related to Ukraine, support vector machines with fastText achieved the best performance, with an F1 score of 0.809. We also observed that LSTM and GRU are very close to support vector machines in performance for detecting the stance of tweets related to Ukraine.

Furthermore, we evaluated the performance of the recently introduced state-of-the-art BERT model on our dataset. We fine-tuned the pre-trained BERT model on our dataset and obtained a significant improvement in performance compared to other machine learning algorithms. BERT achieved the best performance for both targets, with an F1 score of 0.787 for Russia and 0.870 for Ukraine, outperforming other models significantly. The results suggest that fine-tuned BERT is a promising approach for stance detection on social media platforms.

In conclusion, our study demonstrates that stance detection on newly created Turkish tweets related to the Russia-Ukraine conflict is feasible and effective, even on

unbalanced datasets. Our experiments highlight the effectiveness of support vector machines with fastText and BERT in detecting the stance of tweets related to this conflict. Our results can be useful for researchers, policymakers, and journalists interested in monitoring and analyzing social media activities related to the Russia-Ukraine conflict. Moreover, our study provides insights into the effectiveness of different machine learning algorithms and techniques for stance detection tasks, particularly for non-English languages.

5.2 FUTURE WORK

While our study has provided valuable insights into developing an effective stance detection model for Turkish tweets related to the Russia-Ukraine conflict, there are several areas that warrant further investigation and improvement. The following future work suggestions aim to enhance the performance and applicability of the model.

Although fine-tuning the Turkish pretrained BERT model (BERTurk) yielded significant improvements, future work could involve exploring other advanced pretrained language models specifically designed for Turkish text, such as ALBERTurk, DistilBERTurk, or ConvBERTurk. Evaluating these models and comparing their performance on stance detection tasks in Turkish tweets related to the Russia-Ukraine conflict could provide insights into the effectiveness of different architectures for this specific domain.

To address the challenge of limited labeled data, future work could explore transfer learning techniques to leverage labeled data from related domains or tasks. Fine-tuning models pretrained on larger datasets, such as a general Turkish sentiment analysis corpus or other political discourse datasets, could help the stance detection model generalize better to the Russia-Ukraine conflict domain.

Ensemble methods, such as model averaging, stacking, or boosting, have shown success in improving performance and robustness in various natural language processing tasks. Future work could investigate ensemble techniques for stance detection, combining predictions from multiple machine learning algorithms or pretrained models. This could potentially lead to better generalization and more accurate predictions by leveraging the complementary strengths of different models.

Stance detection in politically sensitive topics like the Russia-Ukraine conflict can be influenced by socio-political biases and contextual factors. Future work could delve deeper into understanding and mitigating biases in the dataset and model predictions. This may involve analyzing the impact of different user demographics, social networks, or contextual features on stance expression.

Expanding the model's applicability, future work could focus on developing a real-time stance detection system that continuously monitors Turkish tweets related to the Russia-Ukraine conflict and provides timely and accurate stance predictions. Such a system could incorporate user feedback and iterative model updates, allowing for user interaction and engagement, and providing a valuable tool for researchers, policymakers, and journalists to analyze public opinion dynamics in real-time.

By addressing these future work areas, researchers can advance the field of stance detection for newly created Turkish tweets related to the Russia-Ukraine conflict, enabling a better understanding of public sentiment and supporting evidence-based decision-making by researchers, policymakers, and journalists.

REFERENCES

[1] AZUNRE Paul (2021), *Transfer Learning for Natural Language Processing*, pp. 7, Manning Publications, New York.

[2] KOCHMAR Ekaterina (2019), *Getting Started with Natural Language Processing*, pp. 4, Manning Publications, New York.

[3] ALDAYEL Abeer and MAGDY Walid (2021), "Stance detection on social media: State of the art and trends", *Information Processing and Management*, Vol. 58, No. 4, pp. 102597, https://www.sciencedirect.com/science/article/pii/S0306457321000960, DoA.12.03.2023.

[4] MOHAMMAD Saif M., KIRITCHENKO Svetlana, SOBHANI Parinaz, ZHU Xiaodan and CHERRY Colin (2016), "A Dataset for Detecting Stance in Tweets", http://alt.qcri.org/semeval2016/task6/, DoA.05.04.2023.

[5] KÜÇÜK Dilek and CAN Fazlı (2020), "Stance detection: A survey", *ACM Computing Surveys*. Vol. 53, No. 1, pp. 1-37.

[6] MOHAMMAD Saif, KIRITCHENKO Svetlana, SOBHANI Parinaz, ZHU Xiaodan and CHERRY Colin (2016), "SemEval-2016 Task 6: Detecting Stance in Tweets", *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pp. 31-41, USA.

[7] RIEDEL Benjamin, AUGENSTEIN Isabelle, SPITHOURAKIS Georgios and RIEDEL Sebastian (2017), "A simple but tough-to-beat baseline for the Fake News Challenge stance detection task", *arXiv preprint arXiv:1707.03264*, http://arxiv.org/abs/1707.03264, DoA. 27.05.2022.

[8] POMERLEAU Dean and RAO Delip (2017), "Fake News Challenge", http://www.fakenewschallenge.org/, DoA.05.04.2023.

[9] HANSELOWSKI Andreas, PVS Avinesh, SCHILLER Benjamin, CASPELHERR Felix, CHAUDHURI Debanjan, MEYER Christian M. and GUREVYCH Iryna (2018), "A Retrospective Analysis of the Fake News Challenge Stance Detection Task", *Proceedings of the 27th International Conference on Computational Linguistics*, pp. 1859–1874, USA.

[10] YILDIRIM Ezgi, ÇETİN Fatih, ERYİĞİT Gülşen and TEMEL Tanel (2014), "The Impact of NLP on Turkish Sentiment Analysis", *Türkiye Bilişim Vakfi Bilgisayar Bilimleri ve Mühendisliği Dergisi*, Vol. 7. No. 1, pp. 43-51.

[11] KÜÇÜK Dilek (2017), "Stance Detection in Turkish Tweets", *arXiv preprint arXiv:1706.06894*, http://arxiv.org/abs/1706.06894, DoA. 27.05.2022.

[12] KÜÇÜK Dilek and CAN Fazlı (2018), "Stance Detection on Tweets: An SVM-based Approach", *arXiv preprint arXiv:1803.08910*, http://arxiv.org/abs/1803.08910, DoA. 27.05.2022.

[13] POLAT Kaan Kemal, BAYAZIT Nilgün Güler and YILDIZ Olcay Taner (2021), "Türkçe Duruş Tespit Analizi", *European Journal of Science and Technology*, No. 23, pp. 99-107, DOI:10.31590/ejosat.851584.

[14] KÜÇÜK Doğan and ARICI Nursal (2022), "Sentiment Analysis and Stance Detection in Turkish Tweets About COVID-19 Vaccination", *In, Handbook of Research on Opinion Mining and Text Analytics on Literary Works and Social Media*, Eds. Pantea Keikhosrokiani, Moussa Pourya Asl, pp. 371–387, IGI Global, Malaysia.

[15] DIXON Stacy Jo (2022), “Countries with most Twitter users 2022”, https://www.statista.com/statistics/242606/number-of-active-twitter-users-in-selected-countries/, DoA.01.05.2023.

[16] TUNALI Volkan and BİLGİN Turgay Tugay (2012), "Examining the impact of stemming on clustering Turkish texts", *INISTA 2012 - International Symposium on Innovations in Intelligent Systems and Applications*, pp. 1-4, Trabzon.

[17] JURAFSKY Daniel and MARTIN James H. (2008), *Speech and Language Processing*, pp. 23, 2nd Edition, Prentice Hall, New Jersey, USA.

[18] BENGIO Yoshua, DUCHARME Réjean, VINCENT Pascal and JAUVIN Christian (2003), "A Neural Probabilistic Language Model", *Journal of Machine Learning Research*, Vol. 3, No. 1, pp. 1137-1155.

[19] PENNINGTON Jeffrey, SOCHER Richard and MANNING Christopher D. (2014), "GloVe: Global vectors for word representation", *EMNLP 2014 - Conference on Empirical Methods in Natural Language Processing*, pp. 1532–1543, Doha, Qatar.

[20] MIKOLOV Tomas, CHEN Kai, CORRADO Greg and DEAN Jeffrey (2013), "Efficient Estimation of Word Representations in Vector Space", *arXiv preprint arXiv:1301.3781,* http://arxiv.org/abs/1301.3781, DoA. 04.03.2023.

[21] BOJANOWSKI Piotr, GRAVE Edouard, JOULIN Armand and MIKOLOV Tomas (2016), "Enriching Word Vectors with Subword Information", *Transactions of the Association for Computational Linguistics*, Vol. 5, No. 1, pp. 135-146.

[22] GRAVE Edouard, BOJANOWSKI Piotr, GUPTA Prakhar, JOULIN Armand and MIKOLOV Tomas (2018), "Learning Word Vectors for 157 Languages", *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*. Ed. Nicoletta Calzolari, Khalid Choukri, Christopher Cieri, Thierry Declerck, Sara Goggi, Koiti Hasida, Hitoshi Isahara, Bente Maegaard, Joseph Mariani, Hélène Mazo, Asuncion Moreno, Jan Odijk, Stelios Piperidis, Takenobu Tokunaga, pp. 3483-3487, Miyazaki, Japan.

[23] LE Tuong, HOANG Son Le, VO Minh Thanh, LEE Mi Young and BAIK Sung Wook (2018), "A cluster-based boosting algorithm for bankruptcy prediction in a highly imbalanced dataset", *Symmetry*, Vol. 10, No. 7, pp. 250-261.

[24] LIU Yang, CHAWLA Nitesh V., HARPER Mary P., SHRIBERG Elizabeth and STOLCKE Andreas (2006), "A study in machine learning from imbalanced data for sentence boundary detection in speech", *Computer Speech & Language*, Vol. 20, pp. 468–94. DOI: 10.1016/J.CSL.2005.06.002, DoA. 08.01.2022.

[25] CHAWLA Nitesh V, BOWYER Kevin W., HALL Lawrence O. and KEGELMEYER W. Philip (2002), "SMOTE: Synthetic Minority Over-sampling Technique", *Journal of Artificial Intelligence Research*, Vol. 16, pp. 321-357.

[26] COVER T.M. and HART P.E. (1967), "Nearest Neighbor Pattern Classification", *IEEE Transactions on Information Theory*, Vol. 13, pp. 21–7. DOI:10.1109/TIT.1967.1053964.

[27] HO Tin Kam (1995), "Random decision forests", *Proceedings of the International Conference on Document Analysis and Recognition*, Vol. 1, pp. 278–282, DOI:10.1109/ICDAR.1995.598994.

[28] CORTES Corinna and VAPNIK Vladimir (1995), "Support-Vector Networks", *Machine Learning*, Vol. 20, pp. 273-297.

[29] JOACHIMS Thorsten (1998), "Text Categorization with Support Vector Machines", *ECML-98: 10th European Conference on Machine Learning*, pp. 137-142, Germany.

[30] NOBLE William S. (2006), "What is a support vector machine?", *Nature Biotechnology*, Vol. 24, pp. 1565–7, DOI:10.1038/nbt1206-1565.

[31] GU Rentao, YANG Zeyuan and JI Yuefeng (2020), "Machine Learning for Intelligent Optical Networks: A Comprehensive Survey", *Journal of Network and Computer Applications*, Vol. 157, pp. 102576, DOI:10.1016/j.jnca.2020.102576.

[32] CHEN Tianqi and GUESTRIN Carlos (2016), "XGBoost: A Scalable Tree Boosting System", *Proceedings of the 22nd ACM SIGKDD International Conference On Knowledge Discovery And Data Mining*, pp. 785-794, New York, USA, DOI:10.1145/2939672.2939785.

[33] NATEKIN Alexey, KNOLL Alois and MICHEL Olivier (2013), "Gradient boosting machines, a tutorial", *Frontiers in Neurorobotics*, Vol. 7 https://www.frontiersin.org/articles/10.3389/fnbot.2013.00021, DOI:10.3389/fnbot.2013.00021 DoA. 02.05.2023.

[34] HOCHREITER Sepp and SCHMIDHUBER Jürgen (1997), "Long Short-Term Memory", *Neural Computation*, Vol. 9, No. 8, pp. 1735–1780.

[35] OLAH Christopher (2015), *Understanding LSTM Networks*, https://colah.github.io/posts/2015-08-Understanding-LSTMs/, DoA.05.04.2023.

[36] CHO Kyunghyun, MERRIENBOER Bart van, GULCEHRE Caglar, BAHDANAU Dzmitry, BOUGARES Fethi, SCHWENK Holger and BENGIO Yoshua (2014), "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation", *arXiv preprint arXiv:1406.1078*, https://arxiv.org/abs/1406.1078, DoA. 08.04.2023.

[37] VASWANI Ashish, SHAZEER Noam, PARMAR Niki, USZKOREIT Jakob, JONES Llion, GOMEZ Aidan N., KAISER Lukasz and POLOSUKHIN Illia (2017), "Attention Is All You Need", *arXiv preprint arXiv:1706.03762*, http://arxiv.org/abs/1706.03762, DoA. 08.04.2023.

[38] DEVLIN Jacob, CHANG Ming-Wei, LEE Kenton and KRISTINA Toutanova (2019), *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*, https://github.com/tensorflow/tensor2tensor, DoA.05.04.2023.

[39] JOHNSON Melvin, SCHUSTER Mike, LE Quoc V, KRIKUN Maxim, WU Yonghui, CHEN Zhifeng, THORAT Nikhil, VIEGAS Fernanda, WATTENBERG Martin, CORRADO Greg, HUGHES Macduff and JEFFREY Dean (2016), "Google's Multilingual Neural Machine Translation System: Enabling Zero-Shot Translation", *Transactions of the Association for Computational Linguistics*, Vol. 5, pp. 339-351, DOI:10.1162/tacl_a_00065/1567476/tacl_a_00065.pdf.

[40] SMETANIN Sergey and KOMAROV Mikhail (2021), "Deep transfer learning baselines for sentiment analysis in Russian", *Information Processing & Management*, Vol. 58, pp. 102484, DOI:10.1016/J.IPM.2020.102484.

[41] LIU Yinhan, OTT Myle, GOYAL Naman, DU Jingfei, JOSHI Mandar, CHEN Danqi, LEVY Omer, LEWIS Mike, ZETTLEMOYER Luke and STOYANOV Veselin (2019), "RoBERTa: A Robustly Optimized BERT Pretraining Approach", *arXiv preprint arXiv: 1907.11692*, http://arxiv.org/abs/1907.11692, DoA. 05.04.2023.

[42] LAN Zhenzhong, CHEN Mingda, GOODMAN Sebastian, GIMPEL Kevin, SHARMA Piyush and SORICUT Radu (2019), "Albert: A Lite Bert For Self-Supervised Learning Of Language Representations", *International Conference on Learning Representations*, New Orleans, USA.

[43] SCHWETER Stefan (2020), *BERTurk - BERT models for Turkish*, https://github.com/stefan-it/turkish-bert, DOI:10.5281/zenodo.3770924, DoA. 04.05.2023.