IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received March 13, 2021, accepted March 30, 2021, date of publication April 6, 2021, date of current version April 19, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3071393

Deep Sentiment Analysis: A Case Study on Stemmed Turkish Twitter Data

HARISU ABDULLAHI SHEHU[1], (Graduate Student Member, IEEE), MD. HAIDAR SHARIF[2], (Member, IEEE), MD. HARIS UDDIN SHARIF[3], (Graduate Student Member, IEEE), RIPON DATTA[3], (Graduate Student Member, IEEE), SEZAI TOKAT[4], SAHIN UYAVER[5], HUSEYIN KUSETOGULLARI[6], (Member, IEEE), AND RABIE A. RAMADAN[7], (Member, IEEE)

[1]School of Engineering and Computer Science, Victoria University of Wellington, Wellington 6012, New Zealand
[2]College of Computer Science and Engineering, University of Hail, Hail 2440, Saudi Arabia
[3]Department of International Graduate Services, University of the Cumberlands, Williamsburg, KY 40769, USA
[4]Department of Computer Engineering, Pamukkale University, 20160 Denizli, Turkey
[5]Department of Energy Science and Technologies, Turkish-German University, 34820 Istanbul, Turkey
[6]Department of Computer Science, Blekinge Institute of Technology, 37141 Karlskrona, Sweden
[7]Computer Engineering Department, College of Engineering, Cairo University, Cairo 12613, Egypt

Corresponding author: Md. Haidar Sharif (md.sharif@uoh.edu.sa)

ABSTRACT Sentiment analysis using stemmed Twitter data from various languages is an emerging research topic. In this paper, we address three data augmentation techniques namely Shift, Shuffle, and Hybrid to increase the size of the training data; and then we use three key types of deep learning (DL) models namely recurrent neural network (RNN), convolution neural network (CNN), and hierarchical attention network (HAN) to classify the stemmed Turkish Twitter data for sentiment analysis. The performance of these DL models has been compared with the existing traditional machine learning (TML) models. The performance of TML models has been affected negatively by the stemmed data, but the performance of DL models has been improved greatly with the utilization of the augmentation techniques. Based on the simulation, experimental, and statistical results analysis deeming identical datasets, it has been concluded that the TML models outperform the DL models with respect to both training-time (*TTM*) and runtime (*RTM*) complexities of the algorithms; but the DL models outperform the TML models with respect to the most important performance factors as well as the average performance rankings.

INDEX TERMS Data augmentation, deep learning, machine learning, neural networks, sentiment analysis, Turkish, Twitter.

PROPOSED ACRONYMS

ACC	= Accuracy
AUC	= Area Under the ROC Curve
CNN	= Convolution Neural Network
DECT	= Decision Tree
DL	= Deep Learning
F1S	= F1 Score
HAN	= Hierarchical Attention Network
MAXE	= Maximum Entropy
RANF	= Random Forests
ROC	= Receiver Operating Characteristic
RNN	= Recurrent Neural Network
RTM	= Runtime
RSVM	= Random Forest + Support Vector Machine
SVMs	= Support Vector Machines
TML	= Traditional Machine Learning
TTM	= Training-time.

The associate editor coordinating the review of this manuscript and approving it for publication was Chun-Wei Tsai.

I. INTRODUCTION

As social media encompasses a wide range of interactive applications for allowing users to create and share content with the public, it plays an important role in modern life [1]. There are numerous social media applications, which can be used for various purposes. For instance, there are dating apps (e.g., Tinder, Bumble, and Zoosk), multi-purpose messaging apps (e.g., WhatsApp, WeChat, and Facebook Messenger), online news apps (e.g., Yahoo News, Google News, and

Flipboard), video chatting apps (e.g., Skype, IMO, and Zoom Meetings), and micro-blogging apps (e.g., Twitter, Tumblr, and FriendFeed) [2]. These apps include both immense advantages and extreme risks associate to the exposure of privacy aspects [3]–[5].

Currently, Twitter is one of the best-known channels in the micro-blogging world allowing tweeple (TWitter pEEPLE) publicly to post their views and opinions on various topics using the hashtag topic ''#topic''. For instances, #football, #security, and #networking hint to talk about football, security, and networking, respectively. Twitter allows tweeple to read and send messages that consist of up to 280 characters. These messages are called tweets. Tweets are widely used for expressing views on certain topics [6]. Twitter includes short posts, graphics interchange formats, article links, and videos. The huge amount of data found on Twitter make it a very compulsive medium for performing data analysis related researches. One of the most common approaches to analyze Twitter data is via sentiment analysis [7]–[19]. Sentiment analysis is the computational study of people's emotions, attitudes, and opinions towards an entity. It could be an event, organization, individual, or a topic [20]. Sentiments have been expressed via social media through text-based messages and images [21]. In sentiment analysis, stemming is a commonly used method applied to textual data to find their roots as part of a pre-processing operation [22]–[27]. The stemming rather reduces the information gained from the data in many languages. In fact, the stemming improves accuracy (*ACC* [28]) achieved by various methods in different languages including not only English [29], [30] but also Arabic [26], [27], [31], [32], Indonesian [23], [33], [34], Japanese [25], [35] French [36]–[38], Portuguese [37], [39], German [37], [40], [41], Hungarian [37], [42], [43], Spanish [44]–[47], and Turkish [48]–[50].

The Turkic language [51] as one of the world's fundamental languages are a language family of at least 35 documented languages [52]–[54], spoken by the Turkic peoples of Eurasia from Southern Europe, Eastern Europe, the Caucasus, Central Asia, Western Asia, North Asia, and East Asia [51]. The total number of the Turkic language speakers is over 200 million [55]. Turkish as one of the Turkic language has the greatest number of speakers, spoken mainly in Anatolia and the Balkans; its native speakers account for about 40% of all Turkic speakers [56]. Based on the estimation of Worldometers [57], in 2019 the world population was approximately 7.7 billion and the native Turkish speakers were estimated as 79.4 million, i.e., 1.08% of the total world population. Roughly, at present Turkic-speaking population is 2.6% of the total world population.

There are many text-based studies found in the literature on sentiment or opinion analysis in English language [29]. However, only a handful number of studies were found to be in Turkish [54], [58]–[62]. This is due to its inherent complexity. The hidden suffix of the Turkish makes a word negative within words or a negative word might have a dissimilar message in a sentence. Negations of the Turkish must be carefully taken into consideration. The key differences between the Turkish and the English [63] have been summarized in TABLEs 1, 2, 3, and 4; where root words can be extended by many suffixes to produce new meanings, an added suffix may change the polarity of a word, words can be negated by suffix hidden within the words, a word that appears to be negative may change its polarity to have a different meaning when used in a sentence, respectively.

TABLE 1. Example of the Turkish root words extended to produce new meaning.

Word	Suffixes	English Meaning
Git		Go
Gitme	Git-me	Go not go
Gittim	Git-ti-m	I went
Gidiyorum	Gid-iyor-um	I am going
Gidebilirim	Gid-ebilir-im	I can go
Gidebilirdim	Gid-ebilir-dim	I could have gone
Gidemeyebilirdim	Gid-emeyebilir-di-m	I might not have been able to go

TABLE 2. Example change in polarity of a word due to an added suffix.

Word	Suffixes	English Meaning	Polarity
Merhametli	Merhamet-**li**	Mercy	Positive Polariy
Merhametsiz	Merhamet-**siz**	Merciless	Negative Polarity

TABLE 3. Example of words negated due to a hidden suffix.

Word	Suffixes	English Meaning
Kırıldı	Kırıl-dı	Broken
Kırılmadı	Kırıl-**ma**-dı	Did not break

Moreover, the existing sentiment analysis methods developed for English rarely possess productive outcomes when it comes to Turkish [64]. For instance, the application of stemming on textual data increases the achieved good accuracy on textual data in English [30] or other languages; but this might not always be the case in Turkish. Besides, sentiment analysis is extremely difficult on Turkish over English texts [63].

We know from our previous work [54], [61], [62], that while the produced words after stemming helps improve the accuracy of the method using polarity lexicon, the achieved accuracy is relatively lower [54] using the traditional machine learning (TML) algorithms such as Naive Bayes (NB), Maximum Entropy (MAXE), Decision Tree (DECT), Random Forests (RANF), and Support Vector Machines (SVMs). Anecdotally, this is because by chopping the end of the tweets, the stemming reduces the amount of information gained from these tweets.

Both DL and TML algorithms can be used to analyze sentiment from Turkish textual data. However, it is unknown if DL or TML algorithms will achieve a better performance on sentiment analysis of stemmed Turkish textual data.

This research aims to use a deep learning algorithms to analyze the sentiment of Turkish Twitter texts. Contrary to the traditional machine learning techniques that trained directly on the reduced data, the research proposed three data augmentation techniques (Shift, Shuffle, and Hybrid) to improve

TABLE 4. Example of a word changing its meaning when used in a sentence.

Sentence	English Meaning
Buradan slayt **yapma** ve video düzenleme programına indirebilirsiniz	You can download slide and video editing program from here
Yürüme yolunu takip ederek ilerleyelim	Let's proceed by following the walking path

TABLE 5. Summary of research efforts on sentiment analysis of Turkish Twitter texts carried out in recent years.

Methods	Contribution	Limitation
Kaya [65]	It analyzes sentiments of Turkish political news collected from different news sites using TML algorithms.	Methods are domain specific.
Kaya [58]	It analyzes the sentiment of Turkish political news using a transfer learning approach.	
Kirelli et al. [11]	It determines the sentiment of Turkish tweets on global warming and climate change.	
Onder [59]	It determines both positivity and negativity of Turkish tweets from a transportation company to analyze customer satisfaction.	It considers only positive and negative polarity of tweets.
Vural [63]	It analyzes both positive and negative polarities of Turkish tweets using polarity lexicon.	
Tocoglu et al. [66]	It introduces a new dataset by gathering data from individuals, analyzing their sentiment, and comparing two stemming methods developed for the Turkish language.	Narrowly baseline results were provided.
Shehu et al. [54], [62]	It analyzes the sentiment of Turkish tweets on three different datasets using miscellaneous TML algorithms.	TML algorithms achieve a lower accuracy on the stemmed data.

the diversity of the data during training in order to improve the accuracy on stemmed data. These techniques improve the number of the training set in a dataset. Subsequently, we used three supreme types of DL models namely recurrent neural network (RNN), convolution neural network (CNN), and hierarchical attention network (HAN) to analyze the sentiment from the stemmed Turkish Twitter data. Moreover, as using accuracy as performance measure might be bias, we used four different types of performance metrics namely runtime (*RTM*), *ACC*, area under curve (*AUC*), and F1 Score (*F1S*) to evaluate the algorithms.

Although the training-time (*TTM*) and *RTM* complexities of TML algorithms are significantly lower than those of the DL algorithms (see Fig. 4), our applied DL algorithms have achieved state-of-the-art performance (see Figs. 5 and 6). This is due to the fact that our proposed augmentation techniques have improved the accuracy on the stemmed data that potential improvement reflects on the performance of the DL algorithms. Consequently, the performance of the DL algorithms yields better than that of the TML algorithms. The obtained results of the DL algorithms have been compared with the existing results of TML algorithms on the identical datasets. On the same ground, the DL algorithms outperformed the TML algorithms with a significant difference. As a matter of fact, stemming minimizes the information picked up from the Turkish data [48] and the TML algorithms are trained directly on the reduced data. Henceforth, the performance of TML algorithms is negatively affected by the stemmed data.

The rest of the paper is organized as follows: Section II highlights the influential work carried out on sentiment analysis of the Turkish Twitter text; Section III explains how tweets are harvested from Twitter, the pre-processing operations is applied to convert the data into a usable format, and the stemming operations applied to find the stems (root word) of the tweets. Section IV introduce the proposed data augmentation techniques along with the DL models (RNN, CNN, and HAN) used in this research. The section also presents the performance evaluation metrics used as well as the time-space complexities of numerous algorithms accompanying their corresponding simulated results; Section V shows experimental results, comparison, and discussion; Section VI presents results from statistical tests and discussion; and finally, Section VII concludes the paper and hints future studies.

II. LITERATURE REVIEW

Much research had been carried out to analyze the sentiment of tweets from English data. However, only a limited number of studies have been carried out to analyze the sentiment of tweets in other languages (e.g., the Turkish). Table 5 presents a summary of recent works carried out on sentiment analysis of Turkish texts in recent years.

A detailed explanation of few of the influential works carried out to analyze the sentiment of Turkish texts are highlighted in this section. The existing works can roughly be categorized into two groups: (i) Sentiment analysis of Turkish texts, and (ii) Sentiment analysis of the stemmed Turkish data.

A. RATING THE TURKISH TEXTS

Kaya *et al.* [65] studied sentiment in the Turkish political news. They used articles from different news sites to construct a dataset consisting of political news. They used a dataset that was constructed with a machine learning-based approach. Besides, that dataset was domain-dependent as it

only consist of data from the political domain. It was found in their studies that the MAXE and N-Grams language model outperformed SVMs and NB. All the approaches used in their study achieved accuracy between the range of 65% to 77%. Nevertheless, their study was rather a domain-specific. As such, it is unclear if the same or similar accuracy will be achieved if the study would be performed on a different domain.

A year following that, the same group [58] performed another research on the same domain, where they determined the sentiment classification of the Turkish sentiment columns. They applied transfer learning from an unlabelled Twitter to labeled political columns to enhance the performance of their methods. Their key aim was to determine whether the whole document was positive or negative regardless of its subject. Different techniques (e.g., SVMs, NB, and N-Grams) were used as machine learning classifiers in their study, which added up to 26% further accuracy. As an extra factor, questions remain as to whether the achieved accuracy will remain the same if each sentence in a document is considered separately. In a different direction, Kirelli *et al.* [11] performed sentiment analysis of shared Turkish tweets on global warming and climate change with data mining methods.

Önder *et al.* [59] performed sentiment analysis to analyze the customer satisfaction of a particular transportation company. The analysis was performed with the tweets of the company's customers found on the Twitter. Their study was performed in binary method to determine whether the tweet was positive or negative. Initially, 20000 data were harvested from the Twitter to perform the analysis. But only 14777 tweets remained after a pre-processing operation was performed to remove the un-useful tweets. Different methods (e.g., SVMs, NB, Multinomial NB, and k-Nearest Neighbor) were used to determine the performance of the analysis, out of which the Multinomial NB algorithm produced the best accuracy result with an *ACC* of 66.06%. In normal circumstances, high precision and high *ACC* are expected from the algorithms [28]. Nonetheless, considering that the analysis was performed to classify the data to be either positive or negative, the achieved accuracy was not very encouraging since even the random guessing has a chance of achieving a 50% *ACC*.

TML methods have been used to analyze sentiment [58], [59], [65] of Turkish Twitter data. However, using the TML algorithms to analyze sentiment from tweets require explicit feature engineering as these algorithms cannot extract features on their own. This is anticipated to increase the workload required to implement these algorithms. In this paper, we aimed to address this problem by using the three different DL models to analyze the sentiment of stemmed Turkish Twitter data.

B. RATING THE STEMMED TURKISH DATA

Several research to analyze sentiment from Turkish texts have been carried out specifically on stemmed data [63], [66], [54], [62].

Vural *et al.* [63] presented a framework for unsupervised sentiment analysis in the Turkish text documents. The study customized sentiment analysis library called the SentiStrength for the English to the Turkish by translating its polarity lexicon. The SentiStrength [67] is a sentiment analysis library that assigns a positive and a negative score to English text. The polarity was then assigned to each sentence after segmenting the text to sentences by translating the polarity lexicon from English to Turkish. Zemberek [68] library was used for pre-processing to perform an operation including spell checking, negation extraction, and ASCII (American Standard Code for Information Interchange) to the Turkish conversion. The library was also used to convert the data to stemmed data before applying the polarity lexicon method for analyzing the positive and negative polarity of the data with an *ACC* of 76%. They also assigned the polarity of an English dictionary directly to the translated Turkish words. Nonetheless, the polarity of a translated word from one language might not align with the polarity of the word in the original language. As such, questions remain as to whether the result obtained using this dictionary would yield a similar result, assuming the polarity was assigned based on the Turkish language, independent of the original language.

Tocoglu *et al.* [66] gathered data from individuals to form a new dataset. The gathered dataset was divided into two, forming two datasets namely raw dataset and validated dataset. Furthermore, two different stemming methods, the fixed prefix stemming (FPS) [69], which was proven to give better accuracy after the fifth character, and Zemberek or the dictionary-based Turkish stemmer [68] were applied to each dataset to make a total of four different datasets. Several TML algorithms including NB, DECT, RANF, and updated SVMs were used to analyze the sentiment of the gathered datasets. It was concluded that the SVMs classifier yielded a higher accuracy result. It was also found that the model trained with a validated dataset gave a higher result than the model trained with a nonvalidated dataset. This study set a sub-standard for other researchers by comparing the two stemming methods developed for the Turkish language.

In our previous study [54], [62], we analyzed the sentiment of Turkish Twitter data on different datasets. We harvested data from Twitter and applied pre-processing operations (e.g., removal of punctuations and special characters to clean the data). The data were converted to a stemmed data by chopping off the end of the data to produce their root words. Subsequently, four different TML algorithms namely DECT, RANF, MAXE, and SVMs were employed. A dictionary of 6800 was also manually translated from the English to the Turkish to be used as a method of polarity lexicon. While the *ACC* of the method obtained using polarity lexicon increased from 48.2% if the used data were in raw form to 57% after stemming had been applied, the accuracy of the TML algorithms (e.g., RANF, MAXE, and DECT) had all been decreased.

Research to analyze sentiment from Turkish texts has been carried out on stemmed data [54], [62], [63], [66]. While

FIGURE 1. The word cloud of (a), (b), and (c) represents the first dataset, whereas the word cloud of (d), (e), and (f) indicates the second dataset [54].

converting the data to stemmed data yielded a positive result in case of the polarity lexicon method to analyze sentiment. The achieved accuracy on the stemmed data was relatively less as compared to when the data were in their raw or a tokenized form. Anecdotally, this had occurred due to the fewer data available in the tweets after the data had been stemmed. Besides, many classifiers (typically deep models) give a better classification accuracy as more data become available. In this study, we aim to address the issue of having fewer data in tweets by proposing three data augmentation techniques (e.g., Shuffle, Shift, and Hybrid) to increase the number of training data available in tweets. As the augmentation technique increase the diversity of stemmed data, it is anticipated that this will lead to an increase in the accuracy achieved by the DL model.

III. DATA COLLECTION TECHNIQUES

A. HARDWARE SPECIFICATION

An 8GB Graphical Processing Unit (GPU) device GeForce RTX 2080ti with Compute Unified Device Architecture (CUDA) version 10.2 has been employed in this research.

B. DATASET

The Turkish tweets are harvested from the Twitter using the Twitter searched API (Application Program Interface) implemented in *R version 3.4.3*. Below is an example of raw tweets harvested from Twitter.

1) "username: @Twitteruser: SADECE BÜYÜK ACILAR ÇEKENLER #merhamet IN ANLAMINI BILIRLER…VATANA BAYRAGA MILLETE HAINLIK YAPANLARA"
2) "USERNAME: @Twitteruser: Bizim insanimiz merhamet sahibidir, Hayirli Haftalar #anladimki #BuYaz #kafes #Merhamet #ramazan #Canli https://t.co/CGZ…"

Two different datasets[1] were harvested and manually labelled. A dataset that consists of 3000 data with equal distribution from each class (1000 of positive, negative, and neutral tweets), which we refer to as the first dataset and a dataset with 10500 data with equal distribution from each class (3500 of positive, negative, and neutral tweets), that we refer to as the second dataset. To test the generalizability of the proposed method, we performed all analyses on both datasets. The word cloud present in Fig. 1 provides a summary of the harvested tweets [54].

[1]The datasets are available on request for academic use (Email: harisushehu@ecs.vuw.ac.nz). Researchers wanting to use the dataset will have to agree with the terms that they will cite our work.

C. DATASET MODIFICATION

Since certain tweets directly harvested from the Twitter are not in a usable format, various pre-processing methods such as removal of punctuation marks, user identification (Id), and tweet Id, and so on have been applied to clean the tweets. Retweeted tweets and stopwords or the commonly used words have also been removed from the tweets as part of the pre-processing methods. Furthermore, words have been converted to lower case and tokenization has been applied to convert tweets into tokens. The two sentences in the below subsection III-B show an example of how tweets are transformed after the aforementioned operations have been applied.

1) "sadece, büyük, acilar, çekenler, #merhamet, in, anlamini, bilirler, vatana, bayraga, millete, hainlik, yapanlara"
2) "insanimiz, merhamet, sahibidir,, hayirli, haftalar, #anladimki, #buyaz, #kafes, #merhamet, #ramazan, #canli"

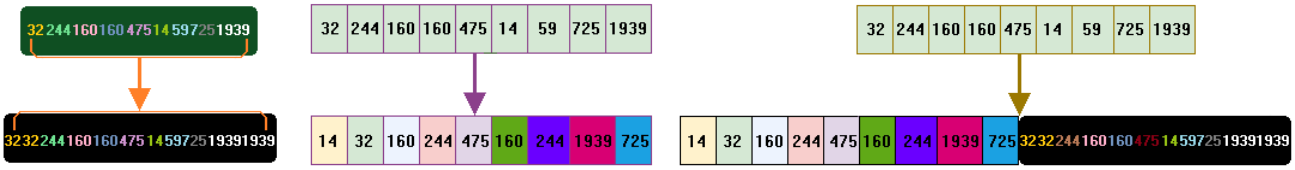
A notable change here is that the words have all been converted to lower cases and tokenized. Another important change is the absence of stopwords like "bizim" in the second part of section III-B which is removed here. The produced information obtained after these operations are called the stopword data. Finally, we find the stem i.e., the root form of the data by chopping the end of words in the tweets. Below is an example of the stemmed data.

1) sadece sade sade ek ek merhamet in in bil vat an millet mil mil hain hain yap
2) insani merhamet sahip sahip hafta hafta hafta haf kafes merhamet ramazan

As was pointed out in the introduction, this study will focus on improving the accuracy of stemmed data. Therefore, having discussed how the data is transformed, the next subsection (III-D) provides more information on the stemming process.

D. STEMMING PROCESS

The stemming is a heuristic process that chops off the end of words. Stemming algorithms have been studied in computer science since the 1960s. The stemming algorithms are typically rule-based. They often include the removal of derivational affixes. For example, a stemming algorithm would reduce the words fishing, fished, and fisher to the stem fish.



(a) Shift augmentation (b) Shuffle augmentation (c) Hybrid augmentation

FIGURE 2. Examples of shift, shuffle, and hybrid augmentation techniques applied to the data.

TABLE 6. Example of stemming on Turkish text.

Word	Meaning	Word After Stem	Meaning
Ucuz	Cheap	Ucuz	Cheap
Eklemek	To add	Ekle	Add
Anlatımak	Be told	Anlat	Tell

TABLE 7. Example of words with more than one stem.

Word	Meaning	Stem	Meaning
Birlik	Union	Birlik Bir	Union One
Yemek	Food	Yemek	Food
İçine	Into	İçine İç	Into Inner

In this paper, the stemming process is performed with the help of Zemberek [68], which is an open-source natural language processing (NLP) library developed for the Turkic languages. TABLE 6 shows an example of the Turkish words and how they are changed after stemming has been applied. However, since certain words might have more than one stem, the stemming operation is performed to include all possible stems of a word. An example of words with more than one stem is presented in TABLE 7. Moreover, the stem of a word might be written more than once based on the plurality of the word and depending on how it is used in a context. For instance, the stem of certain words ending with the suffix ''ler'', which indicates plural in the Turkish are written three times; whereas the stem of words ending with the suffix ''luk'' or ''lik'' are written two times. This is due to the emphasis of the plural in ''ler'' is more as compared to ''luk'' and ''lik''. Few examples of words, which are written more than once have been provided on TABLE 8.

TABLE 8. Example of words that are rewritten more than once.

Word	Meaning	Number of times rewritten * Stem	Meaning
Kötülük	Wickedness	1 * Kötülük; 2 * Kötü	Wickedness; Bad
Güzeller	Beauties	3 * güzel	Beautiful
Güzellik	Beauty	2 * güzel	Beautiful

IV. OUR METHODS

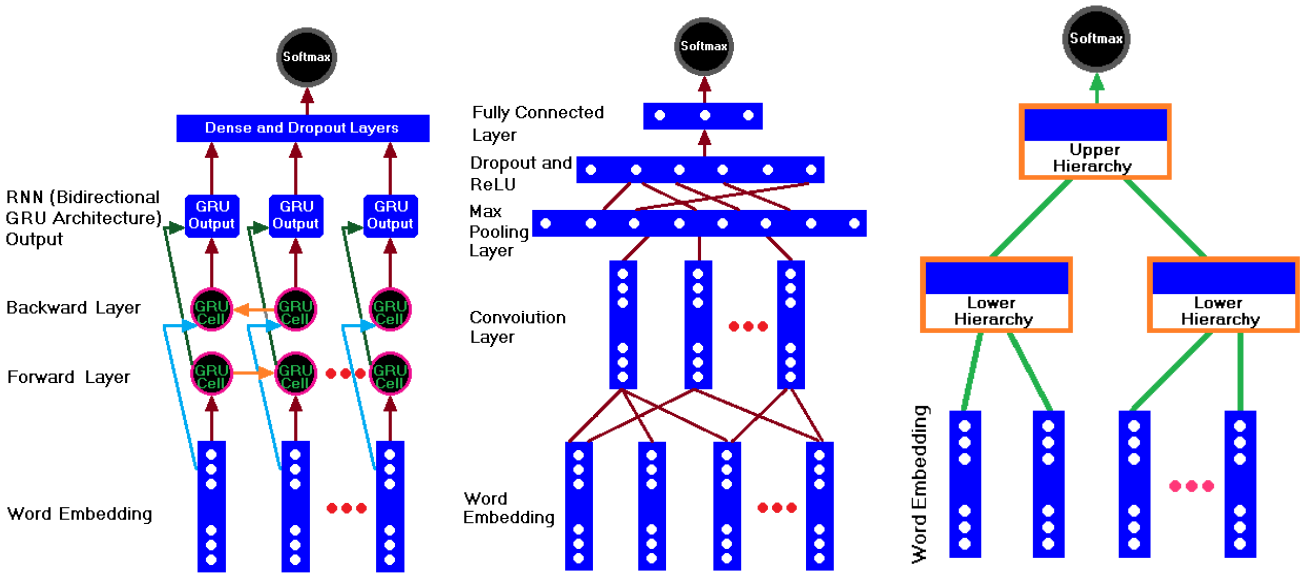
The DL models are computationally intensive and training samples need heavy computations due to their large number of layers. Moreover, training these models requires a lot of training data. Conversely, we also know that stemming minimizes the size of data needed to train/evaluate models, however, augmentation techniques might help overcome the problem by artificially expanding the size of the training data through creating modified versions of texts in the datasets.

A. PROPOSED DATA AUGMENTATION TECHNIQUES

The data augmentation technique is closely related to oversampling in data analysis. It is performed with the aim of increasing the size of the data used for training so as to increase the diversity of the data available for training. It acts as a regularizer. It helps reduce overfitting when training a machine learning model [70]. While the data augmentation technique is a commonly used method when training image data, there are only a limited number of studies carried out on data augmentation on textual data. Therefore, in this paper, we aim to develop a similar method used for augmenting training data in images on textual data to analyze its effect on the accuracy of deep models.

As collecting more data is a tedious and expensive process, we try to make data more diverse by using data augmentation techniques. Each time a sample is processed by the model, it is presented in a slightly different way. This is beneficial as it will make it harder for the model to learn all the parameters of the training samples, which in turn prevents the model from overfitting. Here, we have proposed three different data augmentation techniques to improve the diversity of the data. Fig. 2 illustrates examples of shift, shuffling, and hybrid augmentation techniques.

1) Shift Technique ⇒ The *width_shift* and *height_shift* augmentation method in images is using a threshold value to extend the width and height of a particular image as an augmentation technique. Similar to the *width_shift* and *height_shift* augmentation method in images, this method used a copy of the first and last word of a sentence and add it to the beginning and the end of the same sentence to produce a new sentence in the same class. Fig. 2 (a) shows an example of a sentence generated by the shift augmentation technique.
2) Shuffle Technique ⇒ Similar to crossover [71], the shuffle technique swaps and concatenates words of the same sentence to produce a new sentence of the same class. Fig. 2 (b) exhibits an example of a sentence generated by the shuffle augmentation technique.
3) Hybrid Technique ⇒ The hybrid data augmentation technique combines the two (shift and shuffling)



(a) RNN (b) CNN (c) HAN

FIGURE 3. Architecture of our used DL models.

approaches to produce a new word that is added to the original training data. The aim is to analyze the impact the two proposed methods combine together, will have on the accuracy of the deep models. Fig. 2 (c) demonstrates an example of a sentence generated by the hybrid augmentation technique.

B. OUR PUT IN DL MODELS

Akin to how an infant learns to recognise objects, the DL models needs to be trained with a huge amount of data to be able to generalize on data it has never-seen before. These models are based on neural networks. They take input, which are then processed in hidden layers manipulating weights. The weights are updated during training process. Subsequently, the model expectorates a prediction. The weights are adapted to detect patterns for making better predictions.

In this research, three different types of neural networks that forms the basis for most pre-trained models namely; the RNN, the CNN, and the HAN are used. Fig. 3 depicts the DL models used.

In all the experiments conducted with these three DL models, the dataset was split in to two such that 90% of the data was used for training and 10% was used for testing. The training set was further divided into two using 90-10 split and the 10% was used as the validation set to evaluate the performance of the models. Due to the stochastic nature of processes, all experiments were run 30 times and the results provided are an average of the 30 runs.

1) RNN ARCHITECTURE

The RNN architecture is a type of DL algorithm that processes variable sequence of inputs using their internal states [72]. It allows a dynamic behavior derived from a feed-forward neural network, which allows them to be applicable in miscellaneous tasks including speech [73], handwriting recognition [74], tumour detection with classification [75], network traffic analysis [76]–[78], text classification [79]–[82], and sentiment analysis [83]–[89].

In this paper, we aim to use the RNN model because its outputs are not only influenced by the weights but also by a hidden state vector representing the context on prior inputs. This is beneficial as it helps the network remember things learned from prior input, which might increase the accuracy of the model. Besides, its learning of high prevalent content [90], [91] and its proven performance [92], [93] made us more inclined to its use for our current sentiment analysis problem.

Fig. 3(a) demonstrates the RNN architecture from the cell package used in this paper. The model is set up to run with bidirectional gated recurrent units (Bidirectional GRU) as the type of the RNN architecture, number of hidden GRU cells (an RNN unit) of 200, an attention context or the size of hidden layer in the attention mechanism is set to 300, and a dropout rate of 0.5. The model uses Adam [94] optimizer with an initial learning rate of 0.0002 and the exponential decay rate for the first and second momentum estimates were set to 0.900 and 0.999 respectively. Finally, the softmax function is used at the last layer to perform the classification task.

2) CNN ARCHITECTURE

The CNN architecture is a type of DL network that takes an input and assigns an importance learnable weights to various aspects of the input. Conventionally, these inputs are the stemmed tweets. The CNN model has frequently been used

to perform text classification [95]–[100], as well as sentiment analysis task [89], [101]–[104].

In this research, we aim to use the CNN model because it requires less pre-processing operation as compared to other classification algorithms. Besides, it has the capacity to perform end-to-end learning.

The CNN architecture used in this paper is shown in Fig. 3(b). The CNN architecture is designed to have three layers of 100 channels (with window sizes of 3, 4, and 5 words) and a stride of one word. All words in a tweets are first embedded before they are fed to the CNN, where important features are extracted. Extracted features are passed to the activation layer followed by a dropout rate of 0.10. The resulting output is passed as an input to the fully connected layer which outputs logits that are finally classified by the softmax function.

3) HAN ARCHITECTURE

The HAN architecture is a type of DL model that considers the hierarchical structure of sentences or words. It scrutinizes the hierarchical structure of documents (e.g., document, sentences, and words) for text classification [105]–[107] or sentiment analysis [108]–[114]. It includes an attention mechanism that is able to find the key words and sentences in a document.

The HAN architecture used in this paper is shown in Fig. 3(c). It comprises of two hierarchies - a lower hierarchy and an upper hierarchy. The lower hierarchy takes a single sentence and then it breaks down into words embedding. Finally, it outputs weighted sentence embedding relevant to the classification task. Conversely, the upper hierarchy takes one document (tweet) and then breaks it down into sentence embedding. Ultimately, it outputs document embedding relevant to the classification task. A dropout rate of 0.10 is applied to the final output from the upper hierarchy before passing the output to the softmax function to perform the classification task.

The HAN model has been chosen to be used in this research because it includes an attention mechanism that finds the most important words in a sentence while taking a particular context into consideration. It returns the predominant weights resulting from previous words.

C. PERFORMANCE EVALUATION METRICS

Performance evaluation of any machine learning algorithm is an essential part. An algorithm may give a satisfying results when evaluated using a metric (e.g., *ACC*), but it may give poor results when evaluated against other metrics (e.g., *F1S*). Usually, the classification accuracy is used to measure the performance of machine learning algorithms. However, using only the classification accuracy is not enough to evaluate the performance of the model.

To truly judge any machine learning algorithm, different types of evaluation metrics such as *ACC*, *AUC*, *F1S*, and *RTM* can be used.

The *ACC* can be calculated using Eq. 1 as:

$$ACC = \frac{t_n + t_p}{t_p + t_n + f_p + f_n} \tag{1}$$

where (t_n) represents true negative, (t_p) represents true positive,(f_p) represents false positive, and (f_n) represents false negative.

Sometimes, the word accuracy (ACC) is used interchangeably with percent correct classification (PCC).

The *AUC* is one of the most widely used metrics for evaluation [28]. The *AUC* of a classifier equals to the probability that the classifier ranks a randomly chosen positive sample higher than a randomly chosen negative sample. The *AUC* has a ranges of 0 to 1. If the predictions of a model are 100% wrong, then its $AUC = 0.00$; conversely, if the predictions are 100% correct then its $AUC = 1.00$.

The *F1S* is the harmonic mean between precision and recall. It is also called the F-score or F-measure. It is used in machine learning [115]. It conveys the balance between precision and recall. It also tells us how many instances are classified correctly. The highest possible value i.e. 1 indicates perfect precision and recall. However, the lowest possible value i.e. 0 implies that the precision or the recall is zero.

The F1S can be calculated using the following formula:

$$F1S = \frac{2}{\frac{1}{precision} + \frac{1}{recall}} = \frac{t_p}{t_p + \frac{f_p + f_n}{2}} \tag{2}$$

where *precision* is the number of correct positive results divided by the number of positive results predicted with the classifier, and *recall* is the number of correct positive results divided by the number of all relevant samples.

Estimating the *RTM* complexity of algorithms is mandatory for many applications (e.g., embedded real-time systems [116]). The optimization of the *RTM* complexity of an algorithm in an application is highly expected [117]–[119]. The total *RTM* can prove to be one of the most important determinative performance factors in many software-intensive systems.

D. TIME-SPACE COMPLEXITIES OF ALGORITHMS

The time complexity describes the amount of computer time it takes to run an algorithm. It is not equal to the actual time required to execute an algorithm. The space complexity, like the time complexity, is often expressed as a function of the input size. It specifies the amount of memory needed during the execution of an algorithm. TABLE 9 compares time and space complexities required by the numerous models to predict their outputs. The complexity of the RANF [54] algorithm increases with the number of DECTs. If there exist a huge number of data with many features, multi-core processing can be used for parallelizing the RANF [54] to train different DECTs. During training, each stand learner can be trained on the dissimilar core of the computer. The theoretical complexities suggest that when we have large data with low dimensionality, the DECT [54] can be used. The MAXE [54] model suits the best for applications (e.g., [120]),

TABLE 9. Theoretical time and space complexity of various algorithms considering *d* dimensional training data having *n* points; The symbols *h*, *m*, *r*, and *v* indicate the maximum depth of tree, the number of decision trees, the number of nodes in tree, and the number of support vectors, respectively.

Type	Algorithms	TTM Complexity	RTM Complexity	Space Complexity
TML	RANF [54]	$O(dmn\,log(n))$	$O(hm)$	$O(rm)$
	DECT [54]	$O(dn\,log(n))$	$O(h)$	$O(r)$
	MAXE [54]	$O(dn)$	$O(d)$	$O(d)$
	SVMs [54]	$O(n^2)$	$O(dv)$	$O(v)$
	RSVM [61]	$O(n^2)$	$O(hm+dv)$	$O(rm+v)$
DL	RNN [Ours]	$O(n^5)$	$O(n^4)$	$O(1)$
	CNN [Ours]	$O(n^5)$	$O(n^4)$	$O(1)$
	HAN [Ours]	$O(n^5)$	$O(n^4)$	$O(1)$

where the dimension of the data is small. It is like a logistic regression, which is very suitable for low latency applications. The runtime and space complexities of SVMs [54] are linear with respect to v.

Each layer of the neural networks, a matrix multiplication and an activation (element-wise) function are computed. If a matrix multiplication has an asymptotic runtime of $O(n^3)$, an element-wise function has a runtime of $O(n)$, the number of performed multiplications is counted as n, and the element-wise function are applied n times; then the total runtime becomes $O(n(n^3 + n))$, i.e., we can estimate the approximate runtime complexity of $O(n^4)$ for either RNN or CNN or HAN. If there are n layers each with n neurons and n number of iterations (epochs), we would estimate the approximate TTM complexity of $O(n^5)$ for either RNN or CNN or HAN. But these theoretical complexities do not have significant effect on real world applications, if parallel processing (e.g., a GPU) is used for running the matrix multiplication. Merrill *et al.* [121] described a useful range between narrow upper and lower bounds of the space complexities for various models of neural networks. The space complexity of RNN, CNN, and HAN is $O(1)$ [121]. The DL algorithms (e.g., RNN) can use hidden layer as memory store to learn sequences. This also helps the DL algorithms to capture semantics of text better than TML algorithms. Normally, if any TML algorithm loads too much data into the working memory of a computer, the TML code cannot run successfully.

E. SIMULATED COMPUTATIONAL COMPLEXITIES

In statistics, dimensionality refers to the number of attributes in a dataset. One column may indicate each dimension in a real world data representation (e.g., spreadsheet). A minimum of two support vectors are required for each decision hyperplane in the model. Henceforth, the lowest $v = 2$, irrespective of the number of dimensions or size of a dataset. To make a good balance between AUC and processing time, any RANF should have a number of trees between $2^6 = 64$ and $2^7 = 128$ trees [122]. The DECT [54] considers all features (or variables) of an entire dataset, whereas the RANF [54] randomly considers observations (or rows) along with defined features (or variables) to make multiple decision trees and ends up with the averages results. In brief, the RANF [54] combines the output of multiple randomly created DECTs to make the final output. As a result, computational complexity of the RANF [54] is higher than that of the DECT [54]. The computational complexity of the SVMs [54] is much higher than that of the RANF [54]. This is due to the fact that to train any SVM takes longer than to train any RANF if the size of the training data goes higher. Fig. 4 depicts the simulated computational complexities of several TML and DL algorithms. These simulated results support our initial assumption related to the computational costs of DL models.

V. EXPERIMENTAL RESULTS AND DISCUSSION

A. IMPROVEMENT BY AUGMENTATION TECHNIQUES

TABLE 10 demonstrates the result obtained by the RNN on the first and second datasets before and after the different augmentation techniques were applied to the data. In TABLE 10, *Original* represents accuracy obtained from the originally stemmed data; *Shift* indicates accuracy obtained from the stemmed data after the width and height shift was applied as data augmentation methods; *Shuffled* shows accuracy got after shuffling was applied as data augmentation method; and finally, *hybrid* acts for accuracy obtained after the width and height shift, as well as, shuffling augmentation method was applied to the data. Due to the stochastic nature of processes and non-deterministic nature of the RNN, all experiments were run 30 times. The results in TABLE 10 are the average of the 30 runs with upper and lower bounds of a 95% confidence interval.

TABLE 10. Accuracy obtained by RNN on the first and second datasets. The * indicates the accuracy obtained from the best model after 30 runs.

Dataset	Original	Different Augmentation Techniques		
		Shift	**Shuffle**	**Hybrid**
First	69.2±2.0(*73.7)	**74.5±0.5(*77.0)**	71.5±0.9(*73.7)	71.8±1.8(*76.7)
Second	87.5±1.1(*89.5)	86.1±2.2(*90.2)	88.1±1.2(*90.0)	**88.8±1.7(*92.3)**

Upon looking at the achieved accuracy on the first dataset, the data augmentation method improved the achieved accuracy by the RNN model in all three cases, when *shift*, *shuffle*, and *hybrid* augmentation techniques had been applied. In statistics, a one-way ANalysis Of VAriance (abbreviated as one-way ANOVA) is a technique that can be used to compare means of two or more samples. The one-way ANOVA was conducted to compare the effect of the different methods used on the achieved accuracy. It was found that the used methodologies have a significant effect on the classification accuracy for the four conditions $[F(3, 116)] = 8.731$, $p < 0.001$. Post-hoc comparison of two-sample unpaired t-test with Bonferroni [123] correction between three ($t(58) = 14.0813$, $p < 0.001$ (Shift), $t(58) = 5.7440$, $p < 0.001$ (Shuffle), and $t(58) = 5.2925$, $p < 0.001$ (Hybrid)) different groups at the level of significance $\alpha = 0.017$ comparing accuracy obtained by each augmentation method to the accuracy achieved on the original data (with no augmentation) on the first dataset all showed that there was a significant difference (Original M = 69.2, Shift M = 74.5, Shuffle M = 71.5, Hybrid M = 71.8).

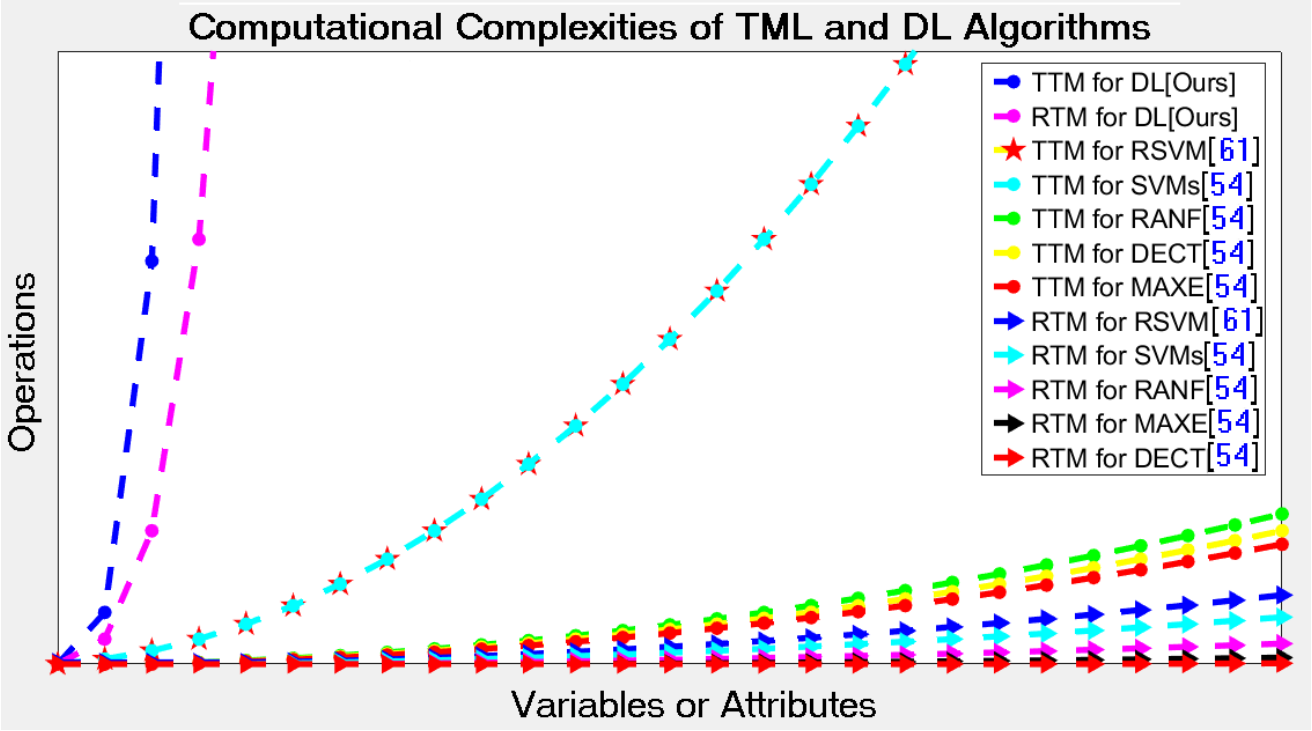


FIGURE 4. Training-time (*TTM*) and runtime (*RTM*) complexities of various TML and DL algorithms.

TABLE 11. Runtime (*RTM*) in seconds and other performance factors obtained by various algorithms on the same datasets.

Type	Algorithms	**RTM**	AUC	**1-AUC**	ACC	**1-ACC**	F1S	**1-F1S**
TML	RANF [54]	**0003**	71.2%	**28.8%**	0.7429	**0.2571**	0.8318	**0.1682**
	DECT [54]	**0001**	41.9%	**58.1%**	0.3395	**0.6605**	0.5909	**0.4091**
	MAXE [54]	**0002**	67.8%	**32.2%**	0.7163	**0.2837**	0.8083	**0.1917**
	SVMs [54]	**0004**	67.6%	**32.4%**	0.6793	**0.3207**	0.8051	**0.1949**
	RSVM [61]	**0006**	82.8%	**17.2%**	0.8338	**0.1662**	0.9061	**0.0939**
DL	RNN [Ours]	**8074**	88.8%	**11.2%**	0.8913	**0.1087**	0.9409	**0.0591**
	CNN [Ours]	**6330**	89.8%	**10.2%**	0.9034	**0.0966**	0.9462	**0.0538**
	HAN [Ours]	**9644**	89.9%	**10.1%**	0.9041	**0.0959**	0.9467	**0.0533**

In contrast to the accuracy achieved on the first dataset in which the augmentation method increased the accuracy achieved in the three cases, the augmentation method increased the achieved accuracy on two of the three cases on the second dataset. A two-sample unpaired t-test [124] with Bonferroni [123] correction was conducted to test the significance of the achieved accuracy on the two cases (Shuffle and Hybrid) that outperformed the accuracy achieved from the original data on the second dataset. However, only $t(58) = 3.5165$, $p < 0.0009$ (Hybrid) was found to be significant (Original M = 87.5, Hybrid M = 88.8) whereas ($t(58) = 2.0188$, $p < 0.0481$ (Shuffle) showed that there was no significant difference (Original M = 87.5, Shuffle M = 88.1).

B. MISCELLANEOUS METHODS

TABLE 11 and its associated Fig. 5 demonstrate the performance factors of *RTM* in seconds, *AUC*, *1-AUC*, *ACC*, *1-ACC*, *F1S* and *1-F1S* for the algorithms of RANF [54], DECT [54], MAXE [54], SVMs [54], RSVM [61], RNN [Ours], CNN [Ours], and HAN [Ours] using identical datasets.

The experimental results demonstrated in TABLE 11 will doubtless be much scrutinized, but there are some immediately dependable conclusions for the achieved results. It can be seen from TABLE 11, the values of *RTM* obtained by the TML algorithms of RANF [54], DECT [54], MAXE [54], SVMs [54], and RSVM [61] are extremely lesser than those of RNN [Ours], CNN [Ours], and HAN [Ours]. Conversely, the achieved values of the performance factors for *AUC*, *ACC*, and *F1S* obtained by the DL algorithms of RNN [Ours], CNN [Ours], and HAN [Ours] is much higher than those of the TML algorithms of RANF [54], DECT [54], MAXE [54], SVMs [54], and RSVM [61]. The TML algorithms required on the average 3.20 seconds, whereas the DL algorithms needed on the average 8016 seconds. This implies that the TML algorithms are 8016/3.20= 2504 times faster than the DL algorithms. Like the simulation results in Fig. 4, the practical results of *RTM* in Fig. 5 also support our initial

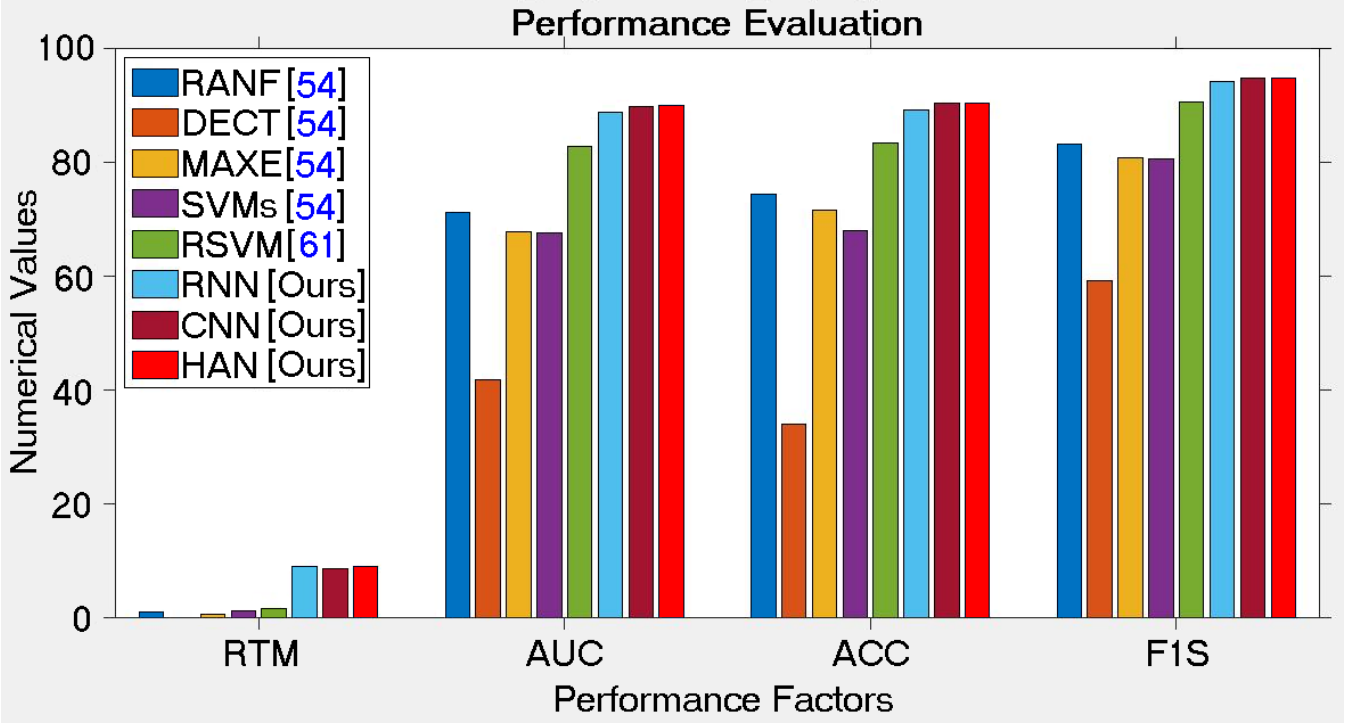


FIGURE 5. Plotting of *RTM, AUC, ACC,* and *F1S* data from TABLE 11; where *RTM* exhibits in logarithmic scale but others are in 100 scale.

computational costs assumption of the DL models. The TML algorithms showed the average performance of $AUC = 66.26\%$, $ACC = 0.6624$, and $F1S = 0.7954$, whereas the DL algorithms showed the average performance of $AUC = 89.50\%$, $ACC = 0.8996$, and $F1S = 0.9446$. This implies that on the average the DL algorithms can obtain (89.50/66.26)-1=35.07% for *AUC*, (0.8996/0.6624)-1=35.81% for *ACC*, and (0.9446/0.7954)-1=18.76% for *F1S* better performance as compared to the TML algorithms. In brief, the DL algorithms are highly recommended to use in applications where accuracy is more important than the *RTM* of the algorithm. Otherwise, the TML algorithms will provide quick results for analyzing sentiments in an online manner.

VI. RESULTS FROM STATISTICAL TESTS

Normally, multiple comparisons with a control algorithm are applied to statistically present that the performance of one algorithm is better than that of its alternatives in areas related to computer science [71], [125]. The main reason of applying the non-parametric tests [126] is that they do not make any assumption regarding the underlying distribution of the data.

A. MULTIPLE COMPARISON WITH STATISTICAL TESTS

We have considered data of *RTM* in second, *1-AUC*, *1-ACC*, and *1-F1S* from TABLE 11 as input parameters for conducting tests for multiple comparisons along with a set of post-hoc procedures to compare a control algorithm with others (i.e., $1 \times N$ comparisons) and to perform all possible pairwise comparisons (i.e., $N \times N$ comparisons). For these purposes, we have used the open source statistical software applications from University of Granada [127].

1) MISCELLANEOUS NONPARAMETRIC TESTS

In the case of $1 \times N$ comparisons, the post-hoc procedures consist of Bonferroni-Dunn's [128], Holm's [129], Hochberg's [130], Hommel's [131], [132], Holland's [133], Rom's [134], Finner's [135], and Li's [136], procedures; whereas in the case of $N \times N$ comparisons, they make up of Nemenyi's [137], Shaffer's [138], and Bergmann-Hommel's [139] procedures. In the case of Bonferroni-Dunn's procedure [128], the performance of two algorithms is considerably divergent if the corresponding mean of rankings is at least as large as its discriminating divergence. A better one is Holm's procedure [129], which examines in a consecutive manner all hypotheses ordered based on their p-values from inferior to superior. All hypotheses for which p-value is less than α divided by the number of algorithms minus the number of a successive step are rejected. All hypotheses having larger p-values are upheld. Holm's procedure [129] adjusts α in a step-down manner. In the same way, both Holland's [133] and Finner's [135] procedures adjust α in a step-down method. Nevertheless, the Hochberg's procedure [130] works in the contrasting direction to the Holland's procedure [133]. It compares the largest p-value with α, the next largest with $\alpha/2$, and so on, until it encounters a hypothesis it can reject. The Rom [134] proposed

TABLE 12. Average rankings using the nonparametric statistical procedures, statistics, and *p*-values.

Ranking	Algorithms	Multiple Comparison Tests		
		Friedman [140]	Friedman's aligned rank [141]	Quade [142]
1	**HAN [Ours]**	**2.50**	**9.00**	**3.10**
2	CNN [Ours]	3.00	15.50	3.90
3	RNN [Ours]	4.25	17.50	5.00
4	RSVM [61]	4.25	14.00	4.40
5	RANF [54]	4.50	16.00	4.20
6	MAXE [54]	5.00	17.75	4.40
7	SVMs [54]	6.25	20.25	5.80
8	DECT [54]	6.25	22.00	5.20
Various Statistics		8.500000	3.446236	0.472669
p-value		29.057%	84.087%	84.340%

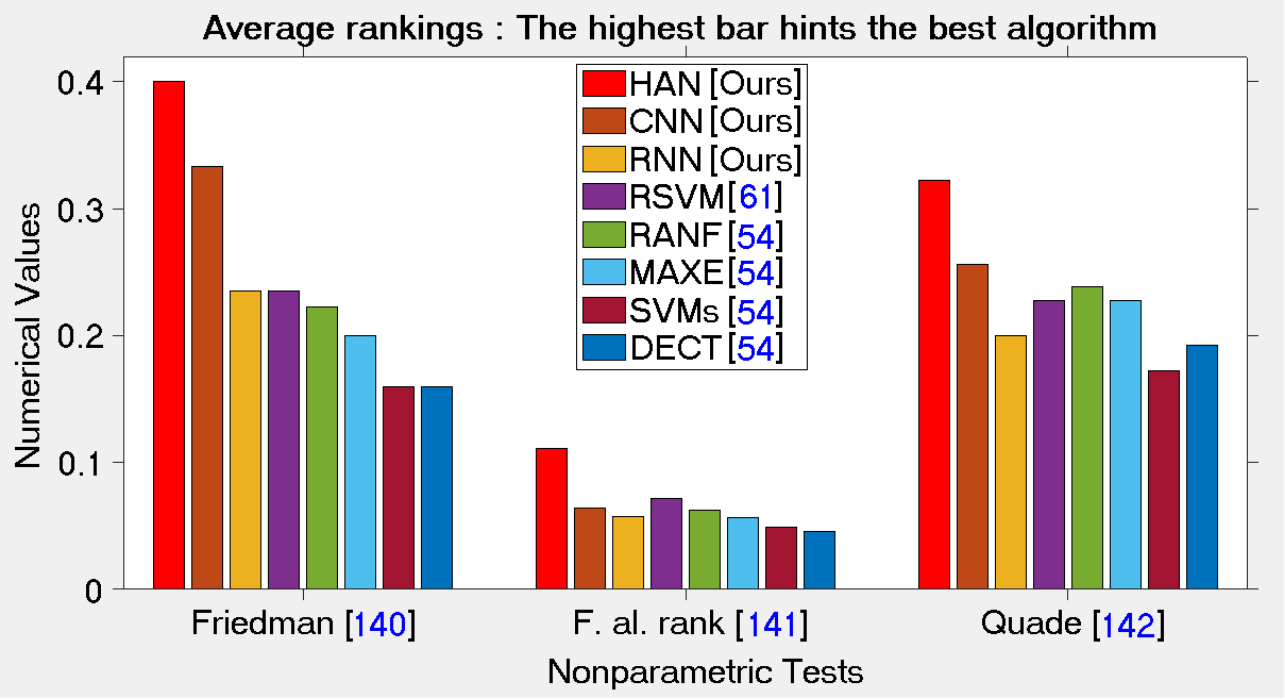


FIGURE 6. Plotting of average rankings data from TABLE 12; where each value x is plotted as $1/x$ to visualize the highest ranking with the tallest bar.

a modification to Hochberg's step-up procedure [130] to enhance its power. In turn, Li [136] suggested a two-step rejection procedure.

2) MULTIPLE COMPARISON NONPARAMETRIC TESTS

TABLE 12 exhibits the average ranking computed by using Friedman [140], Friedman's aligned rank test [141], and Quade [142] nonparametric tests. To achieve the test results Friedman [140], Friedman's aligned rank test [141], and Quade [142] non-parametric tests are applied to the data of *RTM* in seconds, *1-AUC*, *1-ACC*, and *1-F1S* from TABLE 11. The sight of applying Friedman [140], Friedman's aligned rank test [141], and Quade [142] non-parametric tests is to realize whether there are significant differences among various algorithms considered over a given sets of data [142], [143]. These tests give ranking of the algorithms for each individual dataset, i.e., the best performing algorithm receives the highest rank of 1, the second best algorithm gets the rank of 2, and so on. The mathematical equations and further explanation of the non-parametric procedures of Friedman [140], Friedman's aligned rank test [141], and Quade [142] can be found in Quade [142] and Westfall *et al.* [143].

Based on the obtained results in the TABLE 12, HAN [Ours] is the best performing algorithm of the comparison, with average ranking of 2.50, 9.00, and 3.10 for Friedman test [140], Friedman's aligned rank test [141], and Quade test [142], respectively. This indicates that HAN [Ours] provides the greatest performance for sentiment analysis from the stemmed Turkish Twitter data. Friedman statistic considered reduction performance (distributed according to chi-square with 6 degrees of freedom) of 8.500000. Aligned Friedman statistic considered reduction performance (distributed according to chi-square with 6 degrees of freedom) of 3.446236. Quade statistic considered reduction performance (distributed according to F-distribution with 6 and

TABLE 13. Results achieved on post-hoc comparisons for adjusted *p*-values, $\alpha = 0.05$, and $\alpha = 0.10$.

Index	Algorithms	*p*-values	$\alpha = 0.05$		$\alpha = 0.10$	
			Holm [129]	Shaffer [138]	Holm [129]	Shaffer [138]
1	DECT [54] vs. CNN [Ours]	04.331%	0.001786	0.001786	0.003571	0.003571
2	SVMs [54] vs. CNN [Ours]	04.331%	0.001852	0.001852	0.003704	0.003704
3	DECT [54] vs. HAN [Ours]	06.060%	0.001923	0.001923	0.003846	0.003846
4	SVMs [54] vs. HAN [Ours]	06.060%	0.002000	0.002000	0.004000	0.004000
5	DECT [54] vs. RNN [Ours]	19.393%	0.002083	0.002083	0.004167	0.004167
6	MAXE [54] vs. CNN [Ours]	19.393%	0.002174	0.002174	0.004348	0.004348
7	SVMs [54] vs. RNN [Ours]	19.393%	0.002273	0.002273	0.004545	0.004545
8	DECT [54] vs. RSVM [61]	24.821%	0.002381	0.002381	0.004762	0.004762
9	MAXE [54] vs. HAN [Ours]	24.821%	0.002500	0.002500	0.005000	0.005000
10	SVMs [54] vs. RSVM [61]	24.821%	0.002632	0.002632	0.005263	0.005263
11	RANF [54] vs. DECT [54]	31.232%	0.002778	0.002778	0.005556	0.005556
12	RANF [54] vs. SVMs [54]	31.232%	0.002941	0.002941	0.005882	0.005882
13	RANF [54] vs. CNN [Ours]	31.232%	0.003125	0.003125	0.006250	0.006250
14	RANF [54] vs. HAN [Ours]	38.648%	0.003333	0.003333	0.006667	0.006667
15	RSVM [61] vs. CNN [Ours]	38.648%	0.003571	0.003571	0.007143	0.007143
16	DECT [54] vs. MAXE [54]	47.049%	0.003846	0.003846	0.007692	0.007692
17	MAXE [54] vs. SVMs [54]	47.049%	0.004167	0.004167	0.008333	0.008333
18	RSVM [61] vs. HAN [Ours]	47.049%	0.004545	0.004545	0.009091	0.009091
19	RNN [Ours] vs. CNN [Ours]	47.049%	0.005000	0.005000	0.010000	0.010000
20	MAXE [54] vs. RNN [Ours]	56.370%	0.005556	0.005556	0.011111	0.011111
21	RNN [Ours] vs. HAN [Ours]	56.370%	0.006250	0.006250	0.012500	0.012500
22	MAXE [54] vs. RSVM [61]	66.501%	0.007143	0.007143	0.014286	0.014286
23	RANF [54] vs. MAXE [54]	77.283%	0.008333	0.008333	0.016667	0.016667
24	RANF [54] vs. RNN [Ours]	77.283%	0.010000	0.010000	0.020000	0.020000
25	RANF [54] vs. RSVM [61]	88.523%	0.012500	0.012500	0.025000	0.025000
26	RSVM [61] vs. RNN [Ours]	88.523%	0.016667	0.016667	0.033333	0.033333
27	CNN [Ours] vs. HAN [Ours]	88.523%	0.025000	0.025000	0.050000	0.050000
28	DECT [54] vs. SVMs [54]	99.999%	0.050000	0.050000	0.100000	0.100000

42 degrees of freedom) of 0.472669. The *p*-values computed through Friedman statistic, aligned Friedman statistic, and Quade statistic are 29.057%, 84.087%, and 84.340%, respectively. Iman and Davenport [144] statistic considering reduction performance with distributed according to F-distribution with 7 and 21 degrees of freedom is 1.200000. The *p*-value computed by Iman and Daveport [144] test is 34.541%. TABLE 13 demonstrates the results obtained on post-hoc comparisons of adjusted *p*-values, $\alpha = 0.05$, and $\alpha = 0.10$.

3) POST-HOC PROCEDURES FOR $1 \times N$ COMPARISONS

In the case of $1 \times N$ comparisons, the post-hoc procedures consist of Bonferroni-Dunn's [128], Holm's [129], Hochberg's [130], Hommel's [131], [132], Holland's [133], Rom's [134], Finner's [135], and Li's [136] procedures. In these statistical analysis tests, multiple comparison post-hoc procedures considered for comparing the control algorithm HAN [Ours] with the other algorithms. The results are shown by computing *p*-values for each comparison. TABLE 14 depicts obtained *p*-values using the ranks computed by the Friedman [140], Friedman's aligned rank test [141], and Quade [142] non-parametric tests, respectively. Based on the computed results, all tests show significant improvements of HAN [Ours] over CNN [Ours], RNN [Ours], RSVM [61], RANF [54], MAXE [54], SVMs [54], and DECT [54] for all the post-hoc procedures considered. Besides this, the Li's [136] procedure does the greatest performance, reaching the lowest *p*-values in the comparisons.

4) POST-HOC PROCEDURES FOR $N \times N$ COMPARISONS

In the case of $N \times N$ comparisons, the post-hoc procedures consist of Nemenyi's [137], Shaffer's [138], as well as Bergmann-Hommel's [139] procedures. TABLE 15 presents 28 hypotheses of equality among the 6 different algorithms and the *p*-values achieved. Using level of significance $\alpha = 0.05$, Nemenyi's [137] procedure rejects those hypotheses that have an unadjusted $p\text{-}value \leq 0.179\%$. Holm's [129] procedure rejects those hypotheses that have an unadjusted $p\text{-}value \leq 0.179\%$ for $\alpha = 0.05$. Bergmann's [139] procedure does not reject any hypotheses for $\alpha = 0.05$. Using level of significance $\alpha = 0.10$, Nemenyi's [137] procedure rejects those hypotheses that have an unadjusted $p\text{-}value \leq 0.357\%$. Holm's [129] procedure rejects those hypotheses that have an unadjusted $p\text{-}value \leq 0.357\%$ for $\alpha = 0.10$. Bergmann's [139] procedure does not reject any hypotheses for $\alpha = 0.10$. During the post-hoc methods over the results of Quade [142] procedure, Bonferroni-Dunn's [128] procedure rejects those hypotheses that have an unadjusted $p\text{-}value \leq 0.714\%$; Holm's [129] procedure rejects those hypotheses that have an unadjusted $p\text{-}value \leq 0.714\%$; Hommel's [131], [132] procedure rejects those hypotheses that have an unadjusted $p\text{-}value \leq 0.714\%$; Holland's [133] procedure rejects those hypotheses that have an unadjusted $p\text{-}value \leq 0.730\%$; Finner's [135] procedure rejects those hypotheses that have an unadjusted $p\text{-}value \leq 0.730\%$; and Li's [136] procedure rejects those hypotheses that have an unadjusted $p\text{-}value \leq 0.939\%$.

TABLE 14. Adjusted *p*-values for various tests considering HAN [Ours] as control method.

Tests	Algorithms	Unadjusted p-values	$1 \times N$ post-hoc procedures							
			One/Two step procedures		Step-down procedures			Step-up procedures		
			p_{Bonf} [128]	p_{Li} [136]	p_{Holm} [129]	p_{Hol} [133]	p_{Finn} [135]	p_{Hoch} [130]	p_{Hom} [131]	p_{Rom} [134]
Friedman [140]	DECT [54]	03.038%	0.212680	0.117967	0.212679	0.728494	0.194247	0.182297	0.182297	0.173336
	SVMs [54]	03.038%	0.212680	0.117967	0.212679	0.520857	0.194247	0.182297	0.182297	0.173336
	MAXE [54]	14.892%	1.042403	0.395960	0.744573	0.520851	0.313558	0.624643	0.446744	0.624643
	RANF [54]	24.821%	1.737492	0.522133	0.992852	0.520850	0.393032	0.624643	0.496426	0.624643
	RSVM [61]	31.232%	2.186250	0.578918	0.992852	0.217355	0.407975	0.624643	0.624643	0.624643
	RNN [Ours]	31.232%	2.186250	0.578918	0.992852	0.002566	0.407975	0.624643	0.624643	0.624643
	CNN [Ours]	77.283%	5.409810	0.578918	0.992852	0.000079	0.772829	0.772829	0.772829	0.772829
F. al. rank [141]	DECT [54]	05.002%	0.350115	0.083495	0.350115	0.301747	0.301747	0.350115	0.314599	0.332873
	SVMs [54]	08.989%	0.629199	0.140687	0.539314	0.431702	0.301747	0.450982	0.408912	0.450982
	MAXE [54]	18.713%	1.309921	0.254203	0.935658	0.645104	0.383338	0.450982	0.450982	0.450982
	RANF [54]	20.005%	2.039050	0.267060	0.935658	0.645104	0.383338	0.450982	0.035345	0.450982
	RSVM [61]	29.129%	3.156876	0.346649	0.935658	0.645104	0.450982	0.450982	0.450982	0.450982
	RNN [Ours]	32.713%	1.400315	0.373373	0.935658	0.645104	0.383338	0.450982	0.450982	0.450982
	CNN [Ours]	45.098%	2.289907	0.450982	0.935658	0.645104	0.383338	0.450982	0.450982	0.450982
Quade [142]	DECT [54]	55.411%	3.878792	0.714785	3.324679	0.992141	0.984159	0.821688	0.821688	0.821688
	SVMs [54]	44.687%	3.128110	0.756546	3.128110	0.984159	0.984159	0.821688	0.821688	0.821688
	MAXE [54]	71.419%	4.999336	0.768660	3.324679	0.993327	0.984159	0.821688	0.821688	0.821688
	RANF [54]	75.665%	5.296513	0.800211	3.324679	0.993327	0.984159	0.821688	0.821688	0.821688
	RSVM [61]	71.419%	4.999336	0.800211	3.324679	0.993327	0.984159	0.821688	0.821688	0.821688
	RNN [Ours]	59.247%	4.147269	0.809283	3.324679	0.992141	0.984159	0.821688	0.821688	0.821688
	CNN [Ours]	82.169%	5.751816	0.821688	3.324679	0.993327	0.984159	0.821688	0.821688	0.821688

TABLE 15. Adjusted *p*-values for tests for multiple comparisons among all methods.

Index	Hypothesis	Unadjusted p-values	$N \times N$ post-hoc procedures			
			Nemenyi [137]	Holm [129]	Shaffer [138]	Bergmann [139]
1	DECT [54] vs .CNN [Ours]	04.331%	1.212628	1.212628	1.212628	1.212628
2	SVMs [54] vs .CNN [Ours]	04.331%	1.212628	1.212628	1.212628	1.212628
3	DECT [54] vs .HAN [Ours]	06.060%	1.696855	1.575651	1.272641	1.272641
4	SVMs [54] vs .HAN [Ours]	06.060%	1.696855	1.575651	1.272641	1.272641
5	DECT [54] vs .RNN [Ours]	19.393%	5.430064	4.654339	4.072548	3.102894
6	MAXE [54] vs .CNN [Ours]	19.393%	5.430064	4.654339	4.072548	3.102894
7	SVMs [54] vs .RNN [Ours]	19.393%	5.430064	4.654339	4.072548	3.102894
8	DECT [54] vs .RSVM [61]	24.821%	6.949966	5.212475	5.212475	3.226769
9	MAXE [54] vs .HAN [Ours]	24.821%	6.949966	5.212475	5.212475	3.226769
10	SVMs [54] vs .RSVM [61]	24.821%	6.949966	5.212475	5.212475	3.226769
11	RANF [54] vs .DECT [54]	31.232%	8.744999	5.621786	5.212475	3.747857
12	RANF [54] vs .SVMs [54]	31.232%	8.744999	5.621786	5.212475	3.747857
13	RANF [54] vs .CNN [Ours]	31.232%	8.744999	5.621786	5.212475	4.060178
14	RANF [54] vs .HAN [Ours]	38.648%	10.82134	5.797143	5.797143	4.060178
15	RSVM [61] vs .CNN [Ours]	38.648%	10.82134	5.797143	5.797143	4.060178
16	DECT [54] vs .MAXE [54]	47.049%	13.17362	6.116323	6.116323	4.060178
17	MAXE [54] vs .SVMs [54]	47.049%	13.17362	6.116323	6.116323	4.060178
18	RSVM [61] vs .HAN [Ours]	47.049%	13.17362	6.116323	6.116323	4.060178
19	RNN [Ours] vs .CNN [Ours]	47.049%	13.17362	6.116323	6.116323	4.060178
20	MAXE [54] vs .RNN [Ours]	56.370%	15.78368	6.116323	6.116323	4.509623
21	RNN [Ours] vs .HAN [Ours]	56.370%	15.78368	6.116323	6.116323	4.509623
22	MAXE [54] vs .RSVM [61]	66.501%	18.62016	6.116323	6.116323	4.509623
23	RANF [54] vs .MAXE [54]	77.283%	21.63924	6.116323	6.116323	4.509623
24	RANF [54] vs .RNN [Ours]	77.283%	21.63924	6.116323	6.116323	4.509623
25	RANF [54] vs .RSVM [61]	88.523%	24.78655	6.116323	6.116323	4.509623
26	RSVM [61] vs .RNN [Ours]	88.523%	24.78655	6.116323	6.116323	4.509623
27	CNN [Ours] vs .HAN [Ours]	88.523%	24.78655	6.116323	6.116323	4.509623
28	DECT [54] vs .SVMs [54]	99.999%	28.00000	6.116323	6.116323	4.509623

In sum and substance, based on the aforementioned experimental and statistical test results, it would be easy to make an explicit conclusion that the HAN [Ours] outperforms over CNN [Ours], RNN [Ours], RSVM [61], RANF [54], MAXE [54], SVMs [54], and DECT [54]. Intuitively speaking, it is observed that the performance of HAN [Ours] surpasses those of other alternative algorithms for solving deep sentiment analysis problems especially on the stemmed Turkish Twitter data.

5) OUR FINDINGS

Ahead of this study, the evidence that the DL algorithms will perform better than the TML algorithms those used in our previous study was purely anecdotal. However, after a comprehensive investigation that was made on this study, we found that the mean performance of our used DL algorithms (e.g., RNN, CNN, and HAN) outperformed than that of the TML algorithms. One reason behind this fact includes that the DL algorithms are powerful feature extractors and

learning tool as they extract and learn features that are increasingly complicated and detailed. Another reason could be due to their ability to find patterns input data and their nonlinear combination of the extracted features to predict the output. The TML algorithms solely perform feature learning during training, whereas the DL algorithms take a longer time to train usually because of their large number of layers. Although the *RTM* of TML algorithms is almost zero as compared to the DL algorithms, the performance of the former algorithms is significantly lower than that of the later algorithms. In effect, the performance of TML models has been degraded by the stemmed data, whereas a higher performance of DL models has been dignified by the augmentation techniques. The optimized *RTM* is a desirable factor for any algorithm. Nevertheless, the effectiveness is a great factor than the *RTM* of an algorithm in many real world applications. The HAN [Ours] became the best performative algorithm among our underlaid both TML and DL algorithms. In sentiment analysis, generally, not all words are equally important as some words characterize a sentence more than others. One possible reason why the HAN [Ours] performs better than other networks could be hinted the fact that its utilization of the sentence vector such that more attention is given to ''important'' words. In contrast to the other neural network models (e.g., CNN [Ours] and RNN [Ours]), the HAN [Ours] does not only performs end-to-end learning, but also it learns the meaning behind the sequence of words as well as it returns vector corresponding to each word. In other words, it calculates the weighted sum of each vector.

VII. CONCLUSION

We proposed three data augmentation techniques to increase the diversity of the training data, and then used three DL algorithms (e.g., RNN, CNN, and HAN) for sentiment analysis of the stemmed Turkish textual data obtained from the Twitter. The obtained results of these algorithms had been compared with the TML algorithms (e.g., RSVM [61], RANF [54], MAXE [54], SVMs [54], and DECT [54]). Deeming simulation (e.g., Fig. 4), experimental (e.g., Fig. 5), and statistical (e.g., Fig. 6) results on the identical stemmed Turkish Twitter datasets, it had been supported that: (i) In case of both *TTM* and *RTM* complexities of the algorithms, the TML algorithms outperformed the DL algorithms (see Fig. 4); (ii) In case of cardinal performance factors (e.g., *AUC*, *ACC*, and *F1S*), the DL algorithms outperformed the TML algorithms (see Fig. 5); and (iii) On the average performance rankings, the DL algorithms empowered by the augmentation techniques work as powerful feature extractors, and henceforth, they took the topmost rankings as compared to the TML algorithms (see Fig. 6).

The DL algorithms possess high computational cost, but they capture semantics of text better than the TML algorithms. Prior to this study, the evidence of the accuracy of the TML algorithms is reduced due to inadequate information available in the data was purely anecdotal. But our simulation, experimental, and statistical detailed study in this paper has given us the idea that the application of the augmentation method on the stemmed Turkish textual data might lead to a significant increase in the achieved performance by DL model. To the best of our knowledge, this is the first research to apply the data augmentation technique to the stemmed Turkish textual data. Although the DL algorithms used have resulted a significantly better performance as compared to our previously proposed TML algorithms on the stemmed data, the generalisability of the obtained results is subject to certain limitations. For instance, it is not known whether the proposed algorithms will achieve a higher or at least an equivalent result on the raw or the stopwords data. Therefore, further investigation is important to know the effectiveness of these algorithms on the raw and stopword data.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their appreciative and constructive comments on the draft of this article.

REFERENCES

[1] Y. Liu, X. Yu, B. Liu, and Z. Chen, ''Sentence-level sentiment analysis in the presence of modalities,'' in *Proc. 15th Int. Conf. (CICLing)*, in Lecture Notes in Computer Science, vol. 8404, A. F. Gelbukh, Ed., Kathmandu, Nepal, 2014, pp. 1–16.

[2] H. A. Shehu, ''Kutupsallık sözlüğü ve yapay zeka yardımı ile Türkçe Twitter verileri üzerinde duygu analizi,'' M.S. thesis, Bilgisayar Mühendisliği Bölümü, Fen Bilimleri Enstitusu, Pamukkale Universitesi, Denizli, Turkey, 2019.

[3] A. P. Lenton-Brym, V. A. Santiago, B. K. Fredborg, and M. M. Antony, ''Associations between social anxiety, depression, and use of mobile dating applications,'' *Cyberpsychol., Behav., Social Netw.*, vol. 24, no. 2, pp. 86–93, Feb. 2021.

[4] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescapé, ''Multi-classification approaches for classifying mobile app traffic,'' *J. Netw. Comput. Appl.*, vol. 103, pp. 131–145, Feb. 2018.

[5] A. Razaghpanah, N. Vallina-Rodriguez, S. Sundaresan, C. Kreibich, P. Gill, M. Allman, and V. Paxson, ''Haystack: *In situ* mobile traffic analysis in user space,'' *CoRR*, vol. abs/1510.01419, 2015. [Online]. Available: http://arxiv.org/abs/1510.01419

[6] A. P. Jain and V. D. Katkar, ''Sentiments analysis of Twitter data using data mining,'' in *Proc. Int. Conf. Inf. Process. (ICIP)*, Dec. 2015, pp. 807–810.

[7] K. Sailunaz and R. Alhajj, ''Emotion and sentiment analysis from Twitter text,'' *J. Comput. Sci.*, vol. 36, Sep. 2019, Art. no. 101003.

[8] S. E. Saad and J. Yang, ''Twitter sentiment analysis based on ordinal regression,'' *IEEE Access*, vol. 7, pp. 163677–163685, 2019.

[9] A. Feizollah, S. Ainin, N. B. Anuar, N. A. B. Abdullah, and M. Hazim, ''Halal products on Twitter: Data extraction and sentiment analysis using stack of deep learning algorithms,'' *IEEE Access*, vol. 7, pp. 83354–83362, 2019.

[10] M. Bibi, W. Aziz, M. Almaraashi, I. H. Khan, M. S. A. Nadeem, and N. Habib, ''A cooperative binary-clustering framework based on majority voting for Twitter sentiment analysis,'' *IEEE Access*, vol. 8, pp. 68580–68592, 2020.

[11] Y. Kirelli and S. Arslankaya, ''Sentiment analysis of shared tweets on global warming on Twitter with data mining methods: A case study on Turkish language,'' *Comput. Intell. Neurosci.*, vol. 2020, pp. 1904172:1–1904172:9, Sep. 2020.

[12] A. Kumar and A. Jaiswal, ''Systematic literature review of sentiment analysis on Twitter using soft computing techniques,'' *Concurrency Comput., Pract. Exp.*, vol. 32, no. 1, Jan. 2020, Art. no. e5107.

[13] F. Z. Kermani, F. Sadeghi, and E. Eslami, ''Solving the Twitter sentiment analysis problem based on a machine learning-based approach,'' *Evol. Intell.*, vol. 13, no. 3, pp. 381–398, 2020.

[14] U. Naseem, I. Razzak, K. Musial, and M. Imran, ''Transformer based deep intelligent contextual embedding for Twitter sentiment analysis,'' *Future Gener. Comput. Syst.*, vol. 113, pp. 58–69, Dec. 2020.

[15] S. Qaiser, N. Yusoff, F. K. Ahmad, and R. Ali, "Sentiment analysis of impact of technology on employment from text on Twitter," *Int. J. Interact. Mobile Technol.*, vol. 14, no. 7, pp. 88–103, Jul. 2020.
[16] J. R. Alharbi and W. S. Alhalabi, "Hybrid approach for sentiment analysis of Twitter posts using a dictionary-based approach and fuzzy logic methods: Study case on cloud service providers," *Int. J. Semantic Web Inf. Syst.*, vol. 16, no. 1, pp. 116–145, Jan. 2020.
[17] M. Emadi and M. Rahgozar, "Twitter sentiment analysis using fuzzy integral classifier fusion," *J. Inf. Sci.*, vol. 46, no. 2, pp. 226–242, Apr. 2020.
[18] H. Rehioui and A. Idrissi, "New clustering algorithms for Twitter sentiment analysis," *IEEE Syst. J.*, vol. 14, no. 1, pp. 530–537, Mar. 2020.
[19] D. Antonakaki, P. Fragopoulou, and S. Ioannidis, "A survey of Twitter research: Data model, graph structure, sentiment analysis and attacks," *Expert Syst. Appl.*, vol. 164, Feb. 2021, Art. no. 114006.
[20] K. W. Kiprono and E. Abade, "Comparative Twitter sentiment analysis based on linear and probabilistic models," *Int. J. Data Sci. Technol.*, vol. 2, no. 4, pp. 41–45, 2016.
[21] M. Anjaria and R. M. R. Guddeti, "Influence factor based opinion mining of Twitter data using supervised learning," in *Proc. 6th Int. Conf. Commun. Syst. Netw. (COMSNETS)*, Jan. 2014, pp. 1–8.
[22] S. Räbiger, M. Kazmi, Y. Saygın, P. Schüller, and M. Spiliopoulou, "SteM at SemEval-2016 task 4: Applying active learning to improve sentiment classification," in *Proc. 10th Int. Workshop Semantic Eval. (SemEval)*. San Diego, CA, USA: NAACL-HLT, Jun. 2016, pp. 64–70.
[23] R. B. S. Putra and E. Utami, "Non-formal affixed word stemming in Indonesian language," in *Proc. Int. Conf. Inf. Commun. Technol. (ICOIACT)*, Mar. 2018, pp. 531–536.
[24] S. Al-Saqqa, A. Awajan, and S. Ghoul, "Stemming effects on sentiment analysis using large Arabic multi-domain resources," in *Proc. 6th Int. Conf. Social Netw. Anal., Manage. Secur. (SNAMS)*, M. A. Alsmirat and Y. Jararweh, Eds., Granada, Spain, Oct. 2019, pp. 211–216.
[25] S. Bao and N. Togawa, "Document-level sentiment classification in japanese by stem-based segmentation with category and data-source information," in *Proc. IEEE 14th Int. Conf. Semantic Comput. (ICSC)*, San Diego, CA, USA, Feb. 2020, pp. 311–314.
[26] H. A. Almuzaini and A. M. Azmi, "Impact of stemming and word embedding on deep learning-based arabic text categorization," *IEEE Access*, vol. 8, pp. 127913–127928, 2020.
[27] T. Ma, R. Al-Sabri, L. Zhang, B. Marah, and N. Al-Nabhan, "The impact of weighting schemes and stemming process on topic modeling of Arabic long and short texts," *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, vol. 19, no. 6, pp. 81:1–81:23, 2020.
[28] M. H. Sharif, "An eigenvalue approach to detect flows and events in crowd videos," *J. Circuits, Syst. Comput.*, vol. 26, no. 7, Jul. 2017, Art. no. 1750110.
[29] B. Pang and L. Lee, *Opinion Mining and Sentiment Analysis*. Norwell, MA, USA: Now Foundations and Trends, 2008.
[30] J. R. Ragini, P. M. R. Anand, and V. Bhaskar, "Big data analytics for disaster response and recovery through sentiment analysis," *Int. J. Inf. Manage.*, vol. 42, pp. 13–24, Oct. 2018.
[31] A. M. Alayba, V. Palade, M. England, and R. Iqbal, "Arabic language sentiment analysis on health services," in *Proc. 1st Int. Workshop Arabic Script Anal. Recognit. (ASAR)*, Nancy, France, Apr. 2017, pp. 114–118.
[32] Y. A. Alhaj, J. Xiang, D. Zhao, M. A. A. Al-Qaness, M. A. Elaziz, and A. Dahou, "A study of the effects of stemming strategies on arabic document classification," *IEEE Access*, vol. 7, pp. 32664–32671, 2019.
[33] L. S. Indradjaja and S. Bressan, "Automatic learning of stemming rules for the Indonesian language," in *Proc. 17th Pacific Asia Conf. Lang., Inf. Comput. (PACLIC)*, Singapore, D. Ji and K. Teng, Eds., 2003, pp. 62–68.
[34] J. Asian, H. E. Williams, and S. M. M. Tahaghoghi, "Stemming Indonesian," in *Proc. 28th Australas. Comput. Sci. Conf. (ACSC)*, vol. 38, Newcastle, NSW, Australia, 2005, pp. 307–314.
[35] M. Tateno, H. Masuichi, and H. Umemoto, "The Japanese lexical transducer based on stem-suffix style forms," *Natural Lang. Eng.*, vol. 2, no. 4, pp. 329–330, Dec. 1996.
[36] J. Savoy, "Stemming of French words based on grammatical categories," *J. Amer. Soc. Inf. Sci.*, vol. 44, no. 1, pp. 1–9, Jan. 1993.
[37] J. Savoy, "Light stemming approaches for the French, Portuguese, German and Hungarian languages," in *Proc. ACM Symp. Appl. Comput. (SAC)*, H. Haddad, Ed., Dijon, France, 2006, pp. 1031–1035.
[38] P. Majumder, M. Mitra, and K. Datta, "Statistical vs. rule-based stemming for monolingual French retrieval," in *Proc. 7th Workshop Cross-Lang. Eval. Forum Eur. Lang. (CLEF)*, in Lecture Notes in Computer Science, vol. 4730, Alicante, Spain, 2006, pp. 107–110.
[39] W. G. Ferreira, W. A. dos Santos, B. M. P. de Souza, T. M. M. Zaidan, and W. C. Brandao, "Assessing the efficiency of suffix stripping approaches for Portuguese stemming," in *Proc. 22nd Int. Symp. (SPIRE)*, in Lecture Notes in Computer Science, vol. 9309, London, U.K., 2015, pp. 210–221.
[40] M. Braschler and B. Ripplinger, "Stemming and decompounding for German text retrieval," in *Proc. 25th Eur. Conf. Inf. Retr. (ECIR)*, in Lecture Notes in Computer Science, vol. 2633, Pisa, Italy, 2003, pp. 177–192.
[41] M. Braschler and B. Ripplinger, "How effective is stemming and decompounding for German text retrieval?" *Inf. Retr.*, vol. 7, nos. 3–4, pp. 291–316, Sep. 2004.
[42] A. Tordai and M. de Rijke, "Four stemmers and a funeral: Stemming in hungarian at CLEF 2005," in *Proc. 6th Workshop Cross-Lang. Eval. Forum Eur. Lang. (CLEF)*, in Lecture Notes in Computer Science, vol. 4022, Vienna, Austria, 2005, pp. 179–186.
[43] P. Halacsy and V. Tron, "Benefits of resource-based stemming in Hungarian information retrieval," in *Proc. 7th Workshop Cross-Lang. Eval. Forum Eur. Lang. (CLEF)*, in Lecture Notes in Computer Science, vol. 4730, Alicante, Spain, 2006, pp. 99–106.
[44] C. G. Figuerola, R. G. Diaz, and E. L. de San Roman, "Stemming and n-grams in Spanish: An evaluation of their impact on information retrieval," *J. Inf. Sci.*, vol. 26, no. 6, pp. 461–467, 2000.
[45] M. R. Luna, "Stemming process in Spanish words with the successor variety method. methodology and result," in *Proc. 4th Int. Conf. Enterprise Inf. Syst.*, Ciudad Real, Spain, 2002, pp. 838–842.
[46] A. Medina-Urrea, "Towards the automatic lemmatization of 16th century Mexican Spanish: A stemming scheme for the CHEM," in *Proc. Int. Conf. Comput. Linguistics Intell. Text Process. (CICLing)*, in Lecture Notes in Computer Science, vol. 3878, Mexico City, Mexico, 2006, pp. 101–104.
[47] M. A. Paredes-Valverde, R. Colomo-Palacios, M. del Pilar Salas-Zárate, and R. Valencia-García, "Sentiment analysis in Spanish for improvement of products and services: A deep learning approach," *Sci. Program.*, vol. 2017, pp. 1329281:1–1329281:6, Oct. 2017.
[48] H. Sever and Y. Bitirim, "Findstem: Analysis and evaluation of a Turkish stemming algorithm," in *Proc. 10th Int. Symp. String Process. Inf. Retr. (SPIRE)*, in Lecture Notes in Computer Science, vol. 2857, Manaus, Brazil, 2003, pp. 238–251.
[49] M. Y. Nuzumlalı and A. Özgür, "Analyzing stemming approaches for Turkish multi-document summarization," in *Proc. Conf. Empirical Methods Natural Lang. Process., Meeting SIGDAT (EMNLP)*. Doha, Qatar: ACL, 2014, pp. 702–706.
[50] M. Çağatayli and E. Çelebi, "The effect of stemming and stop-word-removal on automatic text classification in Turkish language," in *Proc. 22nd Int. Conf. ICONIP*, in Lecture Notes in Computer Science, vol. 9489, İstanbul, Turkey, 2015, Nov. 2015, pp. 168–176.
[51] Wikipedia. (2021). *Turkic Languages*. [Online]. Available: https://en.wikipedia.org/wiki/Turkic_languages
[52] A. V. Dybo. (2007). *Chronology of Turkic languages and Linguistic Contacts of Early Turks*. Moscow, Russia. [Online]. Available: https://web.archive.org/web/20050311224856/http://altaica.narod.ru/LIBRARY/xronol_tu.pdf
[53] Wikipedia. (2021). *List of Turkic Languages*. [Online]. Available: https://en.wikipedia.org/wiki/List_of_Turkic_languages
[54] H. A. Shehu, M. H. Sharif, S. Uyaver, S. Tokat, and R. A. Ramadan, "Sentiment analysis of Turkish Twitter data using polarity lexicon and artificial intelligence," in *Proc. Int. Conf. Emerg. Technol. Comput.*, 2020, pp. 113–125.
[55] B. Moser and M. W. Weithmann, *Landeskunde Türkei: Geschichte, Gesellschaft und Kultur*. Hamburg, Germany: Helmut Buske Verlag, 2008, p. 173.
[56] K. Katzner, *Languages of the World*, 3rd ed. Abingdon, U.K.: Routledge, 2002.
[57] Worldometers. (2019). *World Population Clock: 7.7 Billion People (2019)—Worldometers*. [Online]. Available: https://www.worldometers.info
[58] M. Kaya, G. Fidan, and I. H. Toroslu, "Transfer learning using Twitter data for improving sentiment classification of Turkish political news," in *Information Sciences and Systems*. Cham, Switzerland: Springer, 2013, pp. 139–148.
[59] O. Coban, B. Ozyer, and G. T. Ozyer, "Sentiment analysis for Turkish Twitter feeds," in *Proc. 23rd Signal Process. Commun. Appl. Conf. (SIU)*, May 2015, pp. 2388–2391.
[60] B. B. Ogul and G. Ercan, "Sentiment classification on Turkish hotel reviews," in *Proc. 24th Signal Process. Commun. Appl. Conf. (SIU)*, May 2016, pp. 497–500.

[61] H. A. Shehu and S. Tokat, "A hybrid approach for the sentiment analysis of Turkish Twitter data," in *Artificial Intelligence and Applied Mathematics in Engineering Problems*. Cham, Switzerland: Springer, 2020, pp. 182–190.
[62] H. A. Shehu, S. Tokat, M. H. Sharif, and S. Uyaver, "Sentiment analysis of Turkish Twitter data," in *Proc. AIP Conf.* New York, NY, USA: AIP, 2019, Art. no. 080004.
[63] A. G. Vural, B. B. Cambazoglu, P. Senkul, and Z. O. Tokgoz, "A framework for sentiment analysis in Turkish: Application to polarity detection of movie reviews in Turkish," in *Proc. 27th Int. Symp. Comput. Inf. Sci.*, Paris, France, Oct. 2012, pp. 437–445.
[64] F. Saglam, H. Sever, and B. Genc, "Developing Turkish sentiment lexicon for sentiment analysis using online news media," in *Proc. IEEE/ACS 13th Int. Conf. Comput. Syst. Appl. (AICCSA)*, Agadir, Morocco, Nov. 2016, pp. 1–5.
[65] M. Kaya, G. Fidan, and I. H. Toroslu, "Sentiment analysis of Turkish political news," in *Proc. IEEE/WIC/ACM Int. Conf. Web Intell. Intell. Agent Technol.*, vol. 1, Dec. 2012, pp. 174–180.
[66] M. A. Tocoglu and A. Alpkocak, "TREMO: A dataset for emotion analysis in Turkish," *J. Inf. Sci.*, vol. 44, no. 6, pp. 848–860, Dec. 2018.
[67] M. Thelwall, K. Buckley, and G. Paltoglou, "Sentiment strength detection for the social Web," *J. Amer. Soc. Inf. Sci. Technol.*, vol. 63, no. 1, pp. 163–173, Jan. 2012.
[68] A. Akin and M. Akin. (2007). *Zemberek, An Open Source NLP Framework for Turkic Languages*. [Online]. Available: https://github.com/ahmetaa/zemberek-nlp
[69] F. Can, S. Kocberber, E. Balcik, C. Kaynak, H. C. Ocalan, and O. M. Vursavas, "Information retrieval on Turkish texts," *J. Assoc. Inf. Sci. Technol.*, vol. 59, no. 3, pp. 407–421, 2008.
[70] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *J. Big Data*, vol. 6, no. 1, p. 60, Dec. 2019.
[71] H. Kusetogullari, M. H. Sharif, M. S. Leeson, and T. Celik, "A reduced uncertainty-based hybrid evolutionary algorithm for solving dynamic shortest-path routing problem," *J. Circuits, Syst. Comput.*, vol. 24, no. 5, Jun. 2015, Art. no. 1550067.
[72] A. Sherstinsky, "Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network," *Phys. D, Nonlinear Phenomena*, vol. 404, 2020, Art. no. 132306. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167278919305974, doi: 10.1016/j.physd.2019.132306.
[73] A. Graves, A.-R. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, Vancouver, BC, Canada, May 2013, pp. 6645–6649.
[74] A. Graves and J. Schmidhuber, "Offline handwriting recognition with multidimensional recurrent neural networks," in *Proc. 22nd Annu. Conf. Neural Inf. Process. Syst.*, Vancouver, BC, Canada: Curran Associates, 2008, pp. 545–552.
[75] S. S. Begum and D. R. Lakshmi, "Combining optimal wavelet statistical texture and recurrent neural network for tumour detection and classification over MRI," *Multimedia Tools Appl.*, vol. 79, nos. 19–20, pp. 14009–14030, May 2020.
[76] M. Lopez-Martin, B. Carro, A. Sanchez-Esguevillas, and J. Lloret, "Network traffic classifier with convolutional and recurrent neural networks for Internet of Things," *IEEE Access*, vol. 5, pp. 18042–18050, 2017.
[77] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescapè, "MIMETIC: Mobile encrypted traffic classification using multimodal deep learning," *Comput. Netw.*, vol. 165, Dec. 2019, Art. no. 106944.
[78] V. F. Taylor, R. Spolaor, M. Conti, and I. Martinovic, "AppScanner: Automatic fingerprinting of smartphone apps from encrypted network traffic," in *Proc. IEEE Eur. Symp. Secur. Privacy (EuroS&P)*, Mar. 2016, pp. 439–454.
[79] D. Tang, B. Qin, and T. Liu, "Document modeling with gated recurrent neural network for sentiment classification," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, Lisbon, Portugal, 2015, pp. 1422–1432.
[80] Y. Lan, Y. Hao, K. Xia, B. Qian, and C. Li, "Stacked residual recurrent neural networks with cross-layer attention for text classification," *IEEE Access*, vol. 8, pp. 70401–70410, 2020.
[81] C. Liu and X. Wang, "Quality-related english text classification based on recurrent neural network," *J. Vis. Commun. Image Represent.*, vol. 71, Aug. 2020, Art. no. 102724.
[82] C. Du and L. Huang, "Text classification research with attention-based recurrent neural networks," *Int. J. Comput. Commun. Control*, vol. 13, no. 1, pp. 50–61, 2018.
[83] J. Nowak, A. Taspinar, and R. Scherer, "LSTM recurrent neural networks for short text and sentiment classification," in *Proc. Int. Conf. Artif. Intell. Soft Comput. (ICAISC)*, in Lecture Notes in Computer Science, vol. 10246. Zakopane, Poland: Springer, 2017, pp. 553–562.
[84] C. Chen, R. Zhuo, and J. Ren, "Gated recurrent neural network with sentimental relations for sentiment classification," *Inf. Sci.*, vol. 502, pp. 268–278, Oct. 2019.
[85] Z. Mahmood, I. Safder, R. M. A. Nawab, F. Bukhari, R. Nawaz, A. S. Alfakeeh, N. R. Aljohani, and S.-U. Hassan, "Deep sentiments in roman urdu text using recurrent convolutional neural network model," *Inf. Process. Manage.*, vol. 57, no. 4, Jul. 2020, Art. no. 102233.
[86] Y. Ma, H. Fan, and C. Zhao, "Feature-based fusion adversarial recurrent neural networks for text sentiment classification," *IEEE Access*, vol. 7, pp. 132542–132551, 2019.
[87] C. R. Aydin and T. Güngör, "Combination of recursive and recurrent neural networks for aspect-based sentiment analysis using inter-aspect relations," *IEEE Access*, vol. 8, pp. 77820–77832, 2020.
[88] H. Jelodar, Y. Wang, R. Orji, and S. Huang, "Deep sentiment classification and topic discovery on novel coronavirus or COVID-19 online discussions: NLP using LSTM recurrent neural network approach," *IEEE J. Biomed. Health Informat.*, vol. 24, no. 10, pp. 2733–2742, Oct. 2020.
[89] F. Abid, C. Li, and M. Alam, "Multi-source social media data sentiment analysis using bidirectional recurrent convolutional neural networks," *Comput. Commun.*, vol. 157, pp. 102–115, May 2020.
[90] R. Dey and F. M. Salem, "Gate-variants of gated recurrent unit (GRU) neural networks," in *Proc. IEEE 60th Int. Midwest Symp. Circuits Syst. (MWSCAS)*, Boston, MA, USA, Aug. 2017, pp. 1597–1600.
[91] N. Gruber and A. Jockisch, "Are GRU cells more specific and LSTM cells more sensitive in motive classification of text?" *Frontiers Artif. Intell.*, vol. 3, p. 40, Jun. 2020.
[92] S. Gao, M. T. Young, J. X. Qiu, H.-J. Yoon, J. B. Christian, P. A. Fearn, G. D. Tourassi, and A. Ramanthan, "Hierarchical attention networks for information extraction from cancer pathology reports," *J. Amer. Med. Inform. Assoc.*, vol. 25, no. 3, pp. 321–330, Mar. 2018.
[93] M. Abdollahi, X. Gao, Y. Mei, S. Ghosh, and J. Li, "Ontology-guided data augmentation for medical document classification," in *Proc. 18th Int. Conf. Artif. Intell. Med. (AIME)*, in Lecture Notes in Computer Science, vol. 12299, M. Michalowski and R. Moskovitch, Eds., Minneapolis, MN, USA, 2020, pp. 78–88.
[94] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. 3rd Int. Conf. Learn. Represent. (ICLR)*, Y. Bengio and Y. LeCun, Eds., San Diego, CA, USA, 2015, pp. 1–15.
[95] J. Wang, Y. Li, J. Shan, J. Bao, C. Zong, and L. Zhao, "Large-scale text classification using scope-based convolutional neural network: A deep learning approach," *IEEE Access*, vol. 7, pp. 171548–171558, 2019.
[96] Q. Li, P. Li, K. Mao, and E. Y.-M. Lo, "Improving convolutional neural network for text classification by recursive data pruning," *Neurocomputing*, vol. 414, pp. 143–152, Nov. 2020.
[97] B. Liu, Y. Zhou, and W. Sun, "Character-level text classification via convolutional neural network and gated recurrent unit," *Int. J. Mach. Learn. Cybern.*, vol. 11, no. 8, pp. 1939–1949, Aug. 2020.
[98] X. Wu, Y. Cai, Q. Li, J. Xu, and H.-F. Leung, "Combining weighted category-aware contextual information in convolutional neural networks for text classification," *World Wide Web*, vol. 23, no. 5, pp. 2815–2834, Sep. 2020.
[99] J. Xu and Q. Du, "On the interpretation of convolutional neural networks for text classification," in *Proc. Eur. Conf. Artif. Intell. (ECAI)*, vol. 325. Santiago de Compostela, Spain: IOS Press, Aug./Sep. 2020, pp. 2252–2259.
[100] Y. Liang, H. Li, B. Guo, Z. Yu, X. Zheng, S. Samtani, and D. D. Zeng, "Fusion of heterogeneous attention mechanisms in multi-view convolutional neural network for text classification," *Inf. Sci.*, vol. 548, pp. 295–312, Feb. 2021.
[101] Y. Xing, C. Xiao, Y. Wu, and Z. Ding, "A convolutional neural network for aspect-level sentiment classification," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 33, no. 14, pp. 1959046:1–1959046:13, Dec. 2019.
[102] A. Dahou, M. A. Elaziz, J. Zhou, and S. Xiong, "Arabic sentiment classification using convolutional neural network and differential evolution algorithm," *Comput. Intell. Neurosci.*, vol. 2019, pp. 2537689:1–2537689:16, Feb. 2019.
[103] M. Dong, Y. Li, X. Tang, J. Xu, S. Bi, and Y. Cai, "Variable convolution and pooling convolutional neural network for text sentiment classification," *IEEE Access*, vol. 8, pp. 16174–16186, 2020.

[104] M. Alam, F. Abid, C. Guangpei, and L. V. Yunrong, "Social media sentiment analysis through parallel dilated convolutional neural network for smart city applications," *Comput. Commun.*, vol. 154, pp. 129–137, Mar. 2020.
[105] W. Huang, E. Chen, Q. Liu, Y. Chen, Z. Huang, Y. Liu, Z. Zhao, D. Zhang, and S. Wang, "Hierarchical multi-label text classification: An attention-based recurrent network approach," in *Proc. Int. Conf. Inf. Knowl. Manage. (CIKM)*, Beijing, China, 2019, pp. 1051–1060.
[106] R. You, Z. Zhang, S. Dai, and S. Zhu, "HAXMLNet: Hierarchical attention network for extreme multi-label text classification," *CoRR*, vol. abs/1904.12578, 2019. [Online]. Available: http://arxiv.org/abs/1904.12578
[107] S. Gao, A. Ramanathan, and G. Tourassi, "Hierarchical convolutional attention networks for text classification," in *Proc. 3rd Workshop Represent. Learn. NLP*, Melbourne, VIC, Australia, 2018, pp. 11–23.
[108] J. Cheng, S. Zhao, J. Zhang, I. King, X. Zhang, and H. Wang, "Aspect-level sentiment classification with HEAT (HiErarchical ATtention) network," in *Proc. ACM Conf. Inf. Knowl. Manage. (CIKM)*, Singapore, Nov. 2017, pp. 97–106.
[109] L. Li, Y. Liu, and A. Zhou, "Hierarchical attention based position-aware network for aspect-level sentiment analysis," in *Proc. 22nd Conf. Comput. Natural Lang. Learn. (CoNLL)*, Brussels, Belgium, 2018, pp. 181–189.
[110] H. Du and J. Qian, "Hierarchical gated convolutional networks with multi-head attention for text classification," in *Proc. Int. Conf. Syst. Informat. (ICSAI)*, Nanjing, China, Nov. 2018, pp. 1170–1175.
[111] Y. Gao, J. Liu, P. Li, and D. Zhou, "CE-HEAT: An aspect-level sentiment classification approach with collaborative extraction hierarchical attention network," *IEEE Access*, vol. 7, pp. 168548–168556, 2019.
[112] T. Manshu and W. Bing, "Adding prior knowledge in hierarchical attention neural network for cross domain sentiment classification," *IEEE Access*, vol. 7, pp. 32578–32588, 2019.
[113] Y. Zhang, D. Miao, and J. Wang, "Hierarchical attention generative adversarial networks for cross-domain sentiment classification," *CoRR*, vol. abs/1903.11334, 2019. [Online]. Available: http://arxiv.org/abs/1903.11334
[114] C. Gan, L. Wang, and Z. Zhang, "Multi-entity sentiment analysis using self-attention based hierarchical dilated convolutional neural network," *Future Gener. Comput. Syst.*, vol. 112, pp. 116–125, Nov. 2020.
[115] E. F. T. K. Sang and F. De Meulder, "Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition," in *Proc. Conf. Natural Lang. Learn. (CoNLL)*, W. Daelemans and M. Osborne, Eds. Edmonton, AB, Canada: HLT-NAACL, May/Jun. 2003, pp. 142–147.
[116] P. Jungklass and M. Berekovic, "Static allocation of basic blocks based on runtime and memory requirements in embedded real-time systems with hierarchical memory layout," in *Proc. 2nd Workshop Next Gener. Real-Time Embedded Syst. (NG-RES)*, vol. 87. Budapest, Hungary: HiPEAC, 2021, pp. 3:1–3:14.
[117] M. H. Sharif, "A numerical approach for tracking unknown number of individual targets in videos," *Digit. Signal Process.*, vol. 57, pp. 106–127, Oct. 2016.
[118] M. H. Sharif, A. Basermann, C. Seidel, and A. Hunger, "High-performance computing of $1/\sqrt{x_i}$ and $exp(\pm x_i)$ for a vector of inputs x_i on Alpha and IA-64 CPUs," *J. Syst. Archit., Embedded Syst. Design*, vol. 54, no. 7, pp. 638–650, 2008.
[119] M. H. Sharif, "High-performance mathematical functions for single-core architectures," *J. Circuits, Syst. Comput.*, vol. 23, no. 4, Apr. 2014, Art. no. 1450051.
[120] M. H. Sharif and C. Djeraba, "An entropy approach for abnormal activities detection in video streams," *Pattern Recognit.*, vol. 45, no. 7, pp. 2543–2561, Jul. 2012.
[121] W. Merrill, G. Weiss, Y. Goldberg, R. Schwartz, N. A. Smith, and E. Yahav, "A formal hierarchy of RNN architectures," in *Proc. 58th Annu. Meeting Assoc. Comput. Linguistics (ACL)*, Jul. 2020, pp. 443–459.
[122] T. M. Oshiro, P. S. Perez, and J. A. Baranauskas, "How many trees in a random forest?" in *Proc. 8th Int. Conf. MLDM*, in Lecture Notes in Computer Science, vol. 7376, Berlin, Germany, 2012, pp. 154–168.
[123] C. Bonferroni, "Teoria statistica delle classi e calcolo delle probabilita," *Pubblicazioni del R Istituto Superiore di Scienze Economiche e Commerciali di Firenze*, vol. 8, no. 8, pp. 3–62, 1936.
[124] Student, "The probable error of a mean," *Biometrika*, vol. 6, no. 1, pp. 1–25, Mar. 1908.
[125] H. A. Shehu, M. H. Sharif, and R. A. Ramadan, "Distributed mutual exclusion algorithms for intersection traffic problems," *IEEE Access*, vol. 8, pp. 138277–138296, 2020.
[126] B. Trawiński, M. Smętek, Z. Telec, and T. Lasota, "Nonparametric statistical analysis for multiple comparison of machine learning regression algorithms," *Int. J. Appl. Math. Comput. Sci.*, vol. 22, no. 4, pp. 867–881, Dec. 2012.
[127] University of Granada. (2020). *Soft Computing and Intelligent Information Systems*. [Online]. Available: https://sci2s.ugr.es/sicidm
[128] O. J. Dunn, "Multiple comparisons among means," *J. Amer. Stat. Assoc.*, vol. 56, no. 293, pp. 52–64, Mar. 1961.
[129] S. Holm, "A simple sequentially rejective multiple test procedure," *Scand. J. Statist.*, vol. 6, no. 2, pp. 65–70, 1979.
[130] Y. Hochberg, "A sharper Bonferroni procedure for multiple tests of significance," *Biometrika*, vol. 75, pp. 800–803, Dec. 1988.
[131] G. Hommel, "A stagewise rejective multiple test procedure based on a modified Bonferroni test," *Biometrika*, vol. 75, no. 2, pp. 383–386, 1988.
[132] G. Hommel and G. Bernhard, "A rapid algorithm and a computer program for multiple test procedures using logical structures of hypotheses," *Comput. Methods Programs Biomed.*, vol. 43, nos. 3–4, pp. 213–216, Jun. 1994.
[133] M. Holland and M. D. Copenhaver, "An improved sequentially rejective Bonferroni test procedure," *Biometrics*, vol. 43, pp. 417–423, Jun. 1987.
[134] D. M. Rom, "A sequentially rejective test procedure based on a modified Bonferroni inequality," *Biometrika*, vol. 77, no. 3, pp. 663–665, 1990.
[135] H. Finner, "On a monotonicity problem in step-down multiple test procedures," *J. Amer. Stat. Assoc.*, vol. 88, no. 423, pp. 920–923, Sep. 1993.
[136] J. D. Li, "A two-step rejection procedure for testing multiple hypotheses," *J. Stat. Planning Inference*, vol. 138, no. 6, pp. 1521–1527, Jul. 2008.
[137] P. Nemenyi, "Distribution-free multiple comparisons," Ph.D. dissertation, Dept. Math., Princeton, NJ, USA: Princeton Univ. Press, 1963.
[138] J. P. Shaffer, "Modified sequentially rejective multiple test procedures," *J. Amer. Stat. Assoc.*, vol. 81, no. 395, pp. 826–831, Sep. 1986.
[139] G. Bergmann and G. Hommel, "Improvements of general multiple test procedures for redundant systems of hypotheses," in *Multiple Hypotheses Testing*, E. S. P. Bauer and G. Hommel, Ed. Berlin, Germany: Springer, 1988, pp. 100–115.
[140] M. Friedman, "The use of ranks to avoid the assumption of normality implicit in the analysis of variance," *J. Amer. Stat. Assoc.*, vol. 32, no. 200, pp. 675–701, Dec. 1937.
[141] J. L. Hodges and E. L. Lehmann, "Ranks methods for combination of independent experiments in analysis of variance," *Ann. Math. Statist.*, vol. 33, no. 2, pp. 482–497, 1962.
[142] D. Quade, "Using weighted rankings in the analysis of complete blocks with additive block effects," *J. Amer. Stat. Assoc.*, vol. 74, no. 367, pp. 680–683, Sep. 1979.
[143] P. Westfall and S. Young, *Resampling-Based Multiple Testing: Examples and Methods for P-Value Adjustment*. Hoboken, NJ, USA: Wiley, 2004.
[144] R. L. Iman and J. M. Davenport, "Approximations of the critical region of the Friedman statistic," *Commun. Statist., Theory Methods*, vol. 9, no. 6, pp. 571–595, Jan. 1980.

HARISU ABDULLAHI SHEHU (Graduate Student Member, IEEE) received the B.Sc. degree (Hons.) in computer engineering from Gediz University, Turkey, and the M.Sc. degree in computer engineering from Pamukkale University, Turkey. His M.Sc. thesis focused on sentiment analysis of Turkish Twitter texts. He is currently a Ph.D. Researcher with the Department of Engineering and Computer Science, Victoria University of Wellington, New Zealand. His research interests include sentiment analysis and emotion detection from patterns of facial movements and physiological changes. His paper presented at the iCETiC'20 Conference was nominated as the Best Paper. He was one of the recipients of the Highly Commended Award from the New Zealand Ministry of Education as part of the Wellington International Student Excellence Award.

MD. HAIDAR SHARIF (Member, IEEE) was born in Jessore, Bangladesh, in 1977. He received the B.Sc. degree in electronics and computer science from Jahangirnagor University, Bangladesh, in 2001, the M.Sc. degree in computer engineering from Duisburg-Essen University, Germany, in 2006, and the Ph.D. degree in computer science from the University of Lille, Lille, France, in 2010. From January 2011 to January 2016, he had been working with Izmir University Bakırçay (old Gediz), Turkey, as an Assistant Professor. From April 2016 to June 2017, he had been working with the International University of Sarajevo, Bosnia and Herzegovina, as an Assistant Professor. From October 2017 to September 2018, he had been working with International Balkan University, North Macedonia, as an Associate Professor. He has been working as an Associate Professor with the University of Hail, Saudi Arabia, since November 2018. He is the author of the book entitled *Sundry Applications and Computations of sin() & cos()* with ISBN 978-1-63802-003-5. His research interests include applications of computer vision, brain–computer interface (BCI) technologies, computer architecture, and computational intelligence algorithms.

MD. HARIS UDDIN SHARIF (Graduate Student Member, IEEE) is currently pursuing the Ph.D. degree in information technology with the University of the Cumberlands, Williamsburg, KY, USA. Besides, he has solid hands-on programming experience in developing enterprise software applications. Since last few years, he has been leading a team of developers towards successful project accomplishments, establishment of good practices, and production of high-quality software applications. He possesses exceptional project and resource management skills. His research interests include cyber security, artificial intelligence, computer vision, advanced software engineering, blockchain, and cryptography.

RIPON DATTA (Graduate Student Member, IEEE) received the B.Sc. degree in computer science and engineering from the University of Science and Technology Chittagong, in 2010, and the M.Sc. degree in computer science from MUM, USA, in 2018. He is currently pursuing the Ph.D. degree in information technology with the University of the Cumberlands, Williamsburg, KY, USA. He is currently working as a Senior Software Engineer at Financial Corporation, where he is an instrumental contributor to advanced software engineering, complex systems development, and machine learning. His research interests include neural networks, deep learning, artificial intelligence, blockchain, advanced software engineering, and cloud computing.

SEZAI TOKAT was born in Turkey, in 1972. He received the B.Sc. degree in computer and control engineering from Istanbul Technical University (ITU), in 1994, the M.Sc. degree in systems and control engineering from Boğaziçi University, in 1997, and the Ph.D. degree in computer and control engineering from ITU, in 2003. He was with ITU as a Research Assistant. Since 2017, he has been a Professor with the Department of Computer Engineering, Pamukkale University, Denizli, Turkey. His research interests include intelligent and robust control, artificial intelligence in operations research, and logistics.

SAHIN UYAVER received the B.Sc. and M.Sc. degrees in physics from Yıldız Teknik Universitesi, in 1992 and 1996, respectively, and the Ph.D. degree from Universitat Potsdam (Max Planck Institute of Colloids and Interfaces), in 2004. He was worked as an Assistant Professor with Maltepe Universitesi, from 2005 to 2009, Istanbul Ticaret Universitesi, from 2010 to 2014, and The University of Oklahoma, from 2014 to 2015. He has been working as an Associate Prof with Turk-Alman Universitesi, since 2015. His research interests include polymers in solution and at interfaces, brain–computer interface (BCI) technologies, applications of computer vision and pattern recognition, and artificial intelligence.

HUSEYIN KUSETOGULLARI (Member, IEEE) received the Ph.D. degree from the University of Warwick, U.K., in 2012. After completing his Ph.D. degree, he worked as a Postdoctoral Researcher with the Image Processing and Expert Systems Laboratory, School of Engineering, Warwick University, U.K. After that, he worked on multiobjectivisation problems with the Department of Computer Science, Aberystwyth University, as a Research Associate. He is currently working as a Senior Lecturer with the Department of Computer Science, Blekinge Institute of Technology, and the School of Informatics, University of Skovde. His research interests include image and video processing, artificial intelligence, evolutionary methods, remote sensing, and optimization.

RABIE A. RAMADAN (Member, IEEE) received the Ph.D. degree in computer engineering from Cairo University, Cairo, Egypt, and Southern Methodist University (SMU), Dallas, TX, USA, in 2007. He is currently the Chair of the Department of Computer Engineering, Hail University, Hail, Saudi Arabia. He is also an Associate Professor with Cairo University and Hail University. He is an author of more than 125 articles in the field of big data, cloud computing, IoT, and computational intelligence. He served as the General Chair, the Program Committee Chair, and a TPC Member for many of the conferences and journals, including *Web of Science* and IEEE TRANSACTIONS journals. He also served as a Co-Chair for the International Conference on Recent Advances in Computer Systems (RACS-2015) and the 2nd National Computing Colleges Conference (NC3 2017) held at Hail University, the General Chair for the International Conference on New Computer Science and Engineering Trends (NCSET2020), and the Chair for the ACM Programming Competition in Saudi Arabia associated with NC3 2017 conference. He is a Co-Founder of IEEE Computational Intelligence, Egypt Chapter, the Director of industrial partnership program (CISCO, Oracle, and Microsoft), College of Computer Science and Engineering, Hail University, and the Founder of the FabLab and the Center of Programming and Applications, Hail University, sponsored by the Ministry of Education, Saudi Arabia. Moreover, he is also the Founder of TripleTech expert house sponsored by Hail University.