Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2020) 28: 3168 – 3182

doi:10.3906/elk-2002-43

Efficient Turkish tweet classification system for crisis response

Saed ALQARALEH*, Merve IŞIK
Computer Engineering Department, Faculty of Engineering, Hasan Kalyoncu University, Gaziantep, Turkey

Received: 10.02.2020 • **Accepted/Published Online:** 11.06.2020 • **Final Version:** 30.11.2020

Abstract: This paper presents a convolutional neural networks Turkish tweet classification system for crisis response. This system has the ability to classify the present information before or during any crisis. In addition, a preprocessing model was also implemented and integrated as a part of the developed system. This paper presents the first ever Turkish tweet dataset for crisis response, which can be widely used and improve similar studies. This dataset has been carefully preprocessed, annotated, and well organized. It is suitable to be used by all the well-known natural language processing tools. Extensive experimental work, using our produced Turkish tweet dataset and the English dataset ("socialmedia-disaster-tweets-relevent"), has been performed to illustrate the performance of the developed approach. In addition, vector space model (VSM) techniques were studied to find out the most suitable technique that can be used for the Turkish language. Overall, the developed approach has achieved a quite good performance, robustness, and stability when processing both Turkish and English languages. Our experiments also compare the performance with some state-of-the-art English language systems, such as "CREES" and "deep multimodal".

Key words: Crises management systems, tweet classification, Turkish language, convolutional neural networks, natural language processing

1. Introduction

Nowadays, accessing social media has become an important activity of people's lives, it refers to online communication among users, and it also plays an important role in information sharing. When an important event occurs, many people share their personal opinions, feelings or acquired information through social media. This shared information can be important in many areas. One of these areas is undoubtedly the crisis management system (CMS).

The analysis of social media data can be done for any natural or human-made disasters such as fire, earthquake, landslide, war, etc. Such a process can help in the assessment of such a situation and leads to take the correct decision on time. In other words, the analysis of social media data leads to gain situational awareness, which may help in preventing or decreasing the effects of disaster by taking the correct responses.

Over the last decade, many researchers have attempted to adopt artificial neural network, particularly the convolutional neural network (CNN), to mimic the human brain and improve the performance of computer based programs. CNN was able to achieve state-of-the-art performance in many tasks such as data mining (ex. translation, text generation), classification, object detection, etc. AlexNet, VGGNet, GoogleNet, and ResNet are examples of some very successful models that are able to achieve performance close to human ability.

The main contribution of the work presented in this paper is of three folds: i) Introducing the first Turkish tweet convolutional neural network system for crisis response. This model was built based on extensive

*Correspondence: saed.alqaraleh@hku.edu.tr

experiments for a large number of CNN models to construct the most suitable and efficient one for the Turkish language. ii) Introducing the first ever well-organized and preprocessed Turkish tweet dataset for crisis response. We aim to make it publicly available for research purposes, which can significantly increase similar researches and lead to improving the field of Turkish language classification system for crisis response. iii) Introducing an efficient Turkish preprocessing model that can help in improving the overall performance of the developed systems.

It is worth mentioning that the Turkish language requires special preprocessing approaches due to its challenging problems. Briefly, one of these problems, it is an agglutinative language: this means that by adding some suffixes to a root word, new and arbitrarily long words can be generated. Another problem is alphabet: Turkish has some letters that do not exist in the English alphabet, i.e. "ğ", "ç", "ı", "ö", "ş", and "ü". In general, people tend to substitute these letters by the closest ASCII characters. Therefore, processing Turkish is more risky to be defeated by erroneous writings. One more problem in Turkish language to be discussed is negations: a small modification can change the whole meaning of a word or statement. In addition, the words can be negated using many ways, in a way that its sentiment polarity will change. For instance, words can be negated with i) the affixes me/ma or ii) siz/sız or iii) using a separate word such as "değil" or "yok".

The remaining of this paper is organized as follows: Section 2 presents the recent developments and studies of CMS. In Section 3, the structure of the proposed system is presented. The experimental evaluation and analysis are presented in Section 4. Finally, conclusions and future works are given in Section 5.

2. Related work

In recent years, social media analytic has become an important research field, which leads to improving many approaches such as CMSs. In addition, impressive attention has been given to mining the publicly available huge amount of data to gain situational awareness, which may help in preventing or decreasing the effect of some disaster by taking the correct responses. In this section, the categories and recent works related to the CMSs have been summarized.

2.1. Categories of CMS

In the following section, we summarize the details and shortcomings of existing CMSs:

1. Calling system, such as 911, is very efficient and has been used for a long time and saved the lives of many people. However, its efficiency significantly decrease during disasters, as a very large number of calls can be received during such cases. In order to process these calls on time, there is a need for a large number of employees. Hence, sending these employees to the location is more important and has the priority.

2. Manual annotation system: such a system is not frequently used compared to the first type, due to the fact that it totally depends on human interpretation. Mainly, users use some mobile apps, websites, and social media to deliver the information, where some employees or volunteers need to read each delivered information and classify it as relevant (disastrous) or irrelevant (nondisastrous, could be gossip, rumor, joke, movie review, etc.). Hence, as this process is time-consuming, it may delay the authority response, which increases the effect and damage of the incident.

3. Automated annotation system: with the impressive improvement in the classification and clustering techniques, where such techniques (ex. CNN) are able to achieve a performance similar to the human level

some automated CMSs have been developed. Briefly, this type was developed to overcome the weakness of the second type and work on detecting only the information, say the tweets, that related to a specific disaster. This process decreases the human offer required to process the data and allow providing the people in charge with the most relevant information only. The main problem of this category is that the majority of the studies and existing systems have been constructed for the English language and very few have been built for other languages such as Chinese. To the best of our knowledge, there is no such a system that can support the Turkish language.

It is worth mentioning that the 'sensors based approaches' can be considered as the fourth category of CMSs, where sensors, such as seismic sensors, are widely used in early detection of the earthquake, tsunami, etc. These sensors provide information about the estimated risks and generate early awareness for people.

2.2. Recent developments and studies in CMS

In this section, the recent developments and studies related to the CMS and processing the Turkish language are summarized.

Ragini and Anand [1] presented an approach that uses the TF-IDF to represent the text features. Then, the chi-square was used for feature selection, i.e. the chi-square calculated the dependency between the stochastic values to eliminate the irrelevant features. In other words, after representing the text by the TF-IDF vector, the chi-square was calculated between each feature and the class. Then, the N features with the highest chi-square and nonnegative values were selected as the vector of the tweet features. Next, both of the SVM and NB were used as the system classifier.

Alqaraleh and Işik [2] and Alqaraleh [3] investigated the performance of k-nearest neighbor (kNN), naive Bayes (NB), random forest (RF), AdaBoost (AdaBoost), and gradient boosting classifiers (GBC) for enhancing the task of classifying the information available before or even during any crisis. The results of [2] and [3] showed that none of the studied algorithms has stable performance; however, by building an ensemble system, the performance, robust and stability of the classifying process has been improved.

Kumar and Singh [4] proposed a convolutional neural network-based system that predicts the locations of the writer of tweet through its text. In general, the system consists of GloVe embedding, convolutional layers, and a fully connected layer. In this study, English tweets with various location references, such as street name, building name, city, region, and country were used. The results of [4] showed that CNN with the crisis embedding model got better performance compared to the Bi-LSTM with the same embedding model; however, the Bi-LSTM model outperforms CNN when both use the GloVe embedding model.

Burel and Alani [5] developed an approach named crisis event extraction service (CREES) that automatically classifies posts during crisis situations. This approach starts by cleaning and tokenizing the text, i.e. preprocessing stages. Then, Google's pretrained Word2Vec model was used to construct the features of tweet. Finally, the text was classified using the CNN model that was developed as the third component of this system.

A very recent approach named deep multimodal neural network was developed by Kumar et al. [6]. This approach consisted of 1) a long short-term-memory (LSTM) which is developed for processing the text and 2) a pretrained VGG-16 model for processing images. In detail, the text model consisted of an embedding layer followed by two LSTM layers, and finally, softmax was used to classify the tweets informative or noninformative.

Overall, the investigation study of both approaches presented in [5] and [6] showed their ability to achieve the state-of-the-art performance.

To the best of our knowledge, there are no researches and studies that investigated the possibility of building CNN based CMSs for supporting the Turkish language. In the following, we have summarized the existing research related to the existing Turkish language classification systems in general.

Baygın [7] introduced a Turkish text classification system that uses the naive Bayes. This system represents the text features using the n-gram, where n was set to 2, 3 and 4-gram. Overall, the system has archived around 92% accuracy when tested using a dataset that contains only 1150 Turkish documents.

Kılınç [8] investigated the performance of ensemble learning on the Turkish text classification. In more detail, bagging, boosting, and rotation forest ensemble learning methods were used with naive Bayes, decision tree, k-nearest neighbor (kNN), and support vector machine (SVM) classification algorithms. These systems were trained and tested using the TTC-3600 dataset. As expected, ensemble methods were able to improve the performance of all the classifiers other than SVM, where its individual performance was almost the same as the performance of the ensemble methods.

Another study that investigated the performance of ensemble learning for supporting both Turkish and English languages was presented by Kilimci and Akyokus [9]. For this purpose, multinomial naive Bayes (MNB), support vector machine (SVM), multivariate Bernoulli naive Bayes (MVBN), convolutional neural network (CNN), and random forest (RF) were used. In addition, both Word2vec and TF-IDF were used for feature extraction. Results of [9] showed that i) RF and CNN are the best individual classifiers. ii) Both TF-IDF and Word2vec methods achieved almost the same performance.

Kilimci and Akyokuş [10] and Aydoğan and Karci [11] presented an impressive study that fully investigated the effects and the performance of word embedding for Turkish text classification. In more detail, the performance of Word2Vec, FastText, and Glove integrated with deep learning algorithms was presented by Kilimci and Akyokuş [10]. This study showed that FastText combined with an LSTM model has outperformed other studied models. On the other hand, Aydoğan and Karci [11] trained a Word2Vec using an unlabeled large corpus of approximately 11 billion Turkish words. Then the resulted word vectors were used to tune and prepare multiple deep neural network models that use convolutional neural network (CNN), recurrent neural network (RNN), long short-term memory (LSTM), and gated recurrent unit (GRU). These models were tested using another corpus that contains 1.5 million samples belonging to 10 classes. Overall, the results of this study showed that using pretrained word vectors can improve the Turkish text classification at rates of approximately from 5% to 7%.

Unlike all existing studies, the developed approach is the first Turkish tweet convolutional neural network system for crisis response. This model was built based on extensive experiments for a large number of CNN models and layers, to construct the most suitable and efficient one for the Turkish language. This leads to building a simpler system with better performance compared to some state-of-the-art approaches such as [5] and [6]. Also, as compared to these two approaches, a more advanced preprocessing model was also implemented and integrated as a part of the developed system.

3. Proposed system

This work aims to build a system that can classify and identify social media data related to the crisis efficiently, in order to inform the authority asap (almost a real-time response) to gain situational awareness, which may help in preventing or decreasing the effect of some disaster by taking the correct responses. Figure 1a shows the structure of training the proposed approach and Figure 1b shows the mechanism of testing and/or using

the approach in real-time. Overall, the developed approach consists of the following stages:

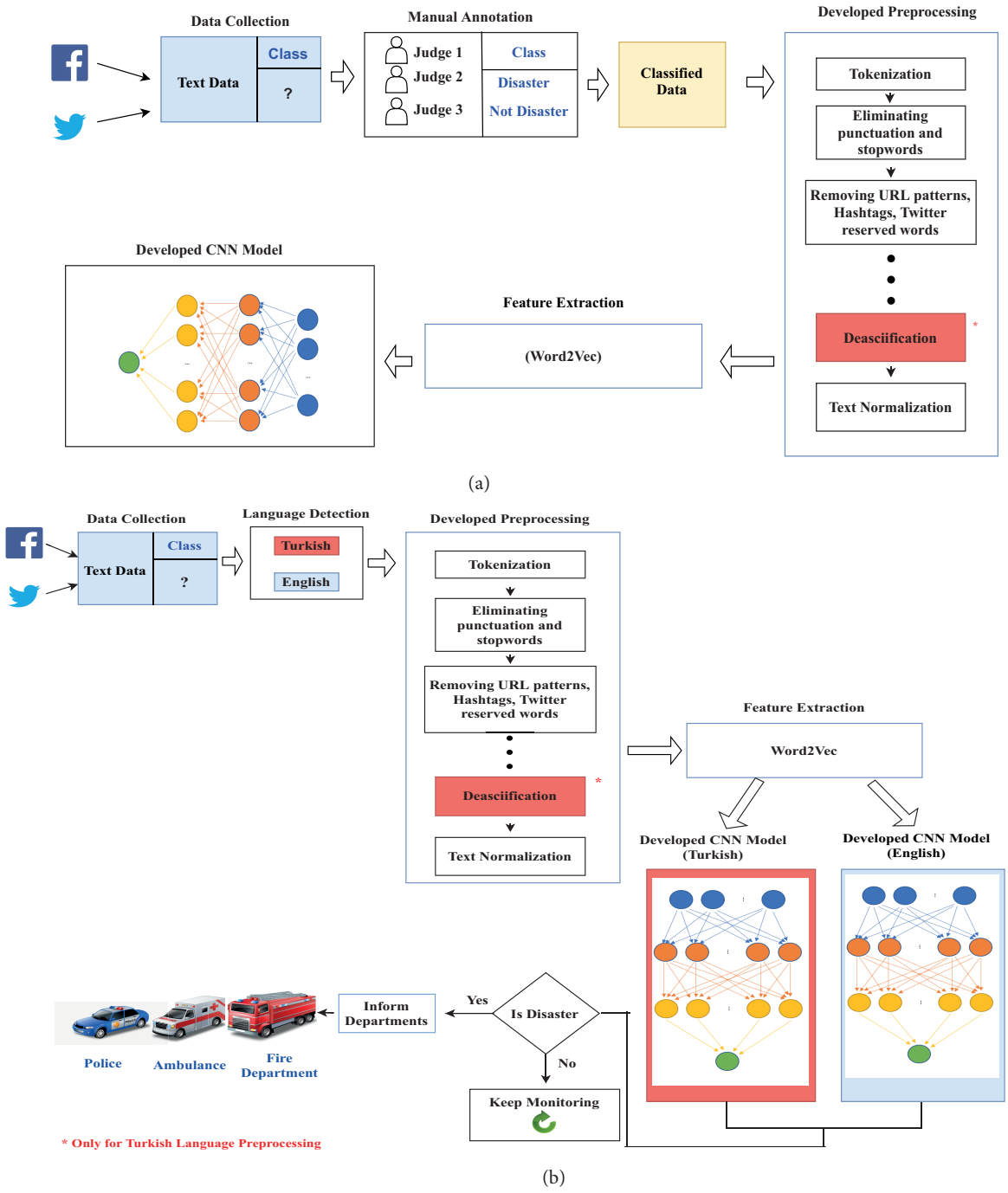


Figure 1. (A) The structure of training the proposed approach, where the annotated tweets are preprocessed, and its features are extracted using Word2Vec to be used to train the developed system. (B) The mechanism of testing and/or using the approach in real-time. This process starts by detecting the language of the recently collected sample(s), and similar processes of the training are applied, i.e. preprocessed, features extraction, then it will be classified by the developed approach. Hence, if the class of the current sample is not disaster the system will keep monitoring and whenever a sample(s) are classified as disasters the system will inform the authority.

Data collection

To this moment, there is no available Turkish dataset about disasters that can be used for building such a system. To overcome this problem, we have constructed a well-organized and preprocessed dataset that consisted of a totally 20,000 tweets, where around 9500 is truly relevant to some disasters that occurred in Turkey, while the remaining tweets were published during the disasters. However, they are irrelevant (not disaster, most of these were jokes, gossip, rumor, movie review, etc). Table 1 shows a sample of the collected tweets, and the main steps of constructing this dataset are explained below.

Table 1. Sample of the collected Turkish tweets.

Topic	Tweet	Meaning	Class
Deprem (Earthquake)	İZMİR'DE DEPREM! İzmir merkezde hissedilen şiddetli bir deprem meydana geldi. Ayrıntılar birazdan...	EARTHQUAKE IN İZMİR! A severe earthquake occurred in the center of İzmir. The details are soon ...	Disaster
	Galatasaray maçı öncesi yine deprem heyecan dalgası ♯deprem	The wave of earthquake excitement before Galatasaray match again ♯earthquake	Not Disaster
Trafik kazası (Traffic accident)	Karabük'te zincirleme trafik kazası: 1 ölü, 6 yaralı	A traffic accident in Karabük: 1 dead, 6 injured	Disaster
İş kazası (Work accident)	Aydın'da tren istasyonunda işçi olarak çalışan babası bir kaza sonucu vefat etti...	His father, who worked as a worker at the train station in Aydın, died as a result of an accident...	Disaster
Sel (Flood)	Haziran ve Ağustos aylarında yaşanan sel felaketinin ardından ilçemiz de ciddi zararlar meydana gelmiştir.	After the flood disaster in June and August, serious damages occurred in our district.	Disaster
Yangın (Fire)	Bu gece dokunmayın bana yüreğim yangın yeri.	Don't touch me tonight, my heart is a fireplace.	Not Disaster
	Andırın'da yıldırım düştü ormanlık alanda yangın çıktı	A fire broke out in the forest because lightning fell in Andirin	Disaster

1. Data collection procedure: This step was performed through the Twitter streaming API. Data is collected with some Turkish keywords related to disasters, such as ♯deprem which means earthquake and ♯yangın, which means fire, ♯trafik kazası which means traffic accident, ♯iş kazası which means work accident, ♯sel which means flood, etc.

2. Annotation procedure: Three judges (annotators) were responsible for reading the tweets and vote whether each tweet is relevant or not, i.e. "disaster" or "not disaster". Then, the class of each tweet is defined using the majority voting. Next, the inter-annotator agreement, which can be used to measure how well the three annotators made the same annotation decisions, has been observed. In other words, the inter-annotator agreement refers to the degree of trust worth of the annotation process.

3. Assessment of inter-annotator agreement: Basically, the inter-annotator agreement can be calculated using: the percentage of overlapping choices between the annotators. In this work, for each of the proposed subdatasets, we have found the average of the agreements of ($Annotator_1$, $Annotator_2$), ($Annotator_1$,

$Annotator_3$) and ($Annotator_2$, $Annotator_3$). Overall, the percentage was above 0.997. However, it is known that this method is not suitable and the obtained results are expected when the number of categories is small, two in our case. The second method is the kappa, which takes into account such a priori overlaps. In this manner, we have used equation (1), i.e. Cohen's kappa to estimate the inter-annotator agreements between each pair of the three annotators.

$$Kappa_{(Annotator_i, Annotator_j)} = \frac{P_a - P_\varepsilon}{1 - P_\varepsilon} \tag{1}$$

where P_a = the proportion of observations in agreement and P_ε = the proportion in agreement due to chance. Then, we calculated the average of the kappa for the three annotators. Overall, the average kappa of fire, traffic accident, and flood datasets was 0.995 and 0.922 for the earthquake dataset.

Data preprocessing

We proposed a new preprocessing model that is suitable for Turkish and can solve its challenging problems, mentioned in Section 1. This model consisted of the following steps. However, it is important to note that the first two steps are the same for almost all languages, and the remaining ones are specific for Turkish.

1. Tokenization: Each text data (tweet) can be tokenized into sentences and/or words. Tokenization simply refers to dividing the text into smaller parts.

2. Eliminating the useless information: In general, some part of the tweet is useless (does not give a specific meaning and may degrade the performance of the system). This step contains the elimination of the following parts: stop words, to eliminate stop words, the list of Turkish stop words available in [12] was used. Punctuation, URL patterns, and hashtags are also removed from the text content. Twitter reserved words and has some abbreviations, i.e. special words such as RT, FAV, VIA. Removing such text or tag helps in getting the actual informative content. In addition to the above, extra space, single-character word, and emoji are eliminated.

3. Deasciification: As mentioned before, there are six Turkish language-specific characters: 'ğ', 'ç','ş','ö','ü','ı'. People tend to substitute these Turkish letters by the closest ASCII characters and even some devices allow the user to use only standard ASCII characters. In this case, the data should be converted into its correct form. This operation is called deasciification.

4. Text normalization: This step consisted of multiple substeps such as converting all text to the same letter size (upper or lower), converting numbers to word equivalents, handling the misspellings (spatially the ones have done intentionally such as "geliyoruuumm" should be normalized as "geliyorum"), etc.

Feature extraction

Words in a text are usually discrete and categorical features. Therefore, we need to represent it using vector space [convert the text to the vector space model (VSM)]. This process mainly consisted of two steps: First: Create a dictionary of terms of dataset (in our case, it is the dataset of tweets). In other words, each unique term in the dataset is defined in the vector space with a unique identity. The second step is to obtain and add the numerical representation of terms into the vector space. To do so, i.e. representing each term in our vector

space, some methods such as term frequency (TF), term frequency-inverse document frequency (TF-IDF), and word embedding can be used. For more details about TF and TF-IDF, the reader is referred to [13].

Related to the word embedding [14], it can be considered as the state of the art method that can represent words in low-dimensional vector space while effectively preserving contextual similarity. In addition, it has the ability to obtain almost the same representation for similar meaning words. On the other hand, in order to achieve a quite good performance, embeddings must be trained using a huge amount of text data. Some of the most popular embedding approaches are Word2Vec, GloVe, FastText, BERT, ElMo, XLNet [15–17].

Based on our preliminary investigation, and the results of [18], related to CNN models, the Word2Vec with a pretrained word vector outperforms other embedding approaches when processing the Turkish language.

To sum up, the preprocessed tweets are passed to the selected VSM method, which will produce an output in the shape of a $2D$ vector. Each row in this vector represent a word in the processed tweet, i.e. number of row in this vector is equal to the number of words of the tweet (N). In addition, the row of each word has a feature vector that can be represented as shown in Equation (2).

$$VSM_{(word_i)} = [F_1, F_2, F_3, \ldots, F_k] \tag{2}$$

where $i = [1, N]$, f is a float number, and k is the count of float numbers representing the $(word_i)$. Hence, $k = 300$ means that each word is represented by a vector of 300 float numbers.

Classification

In this paper, a new system that can efficiently process the Turkish language, as well as the English language, has been introduced. This system was developed after an intensive investigation to find the most suitable layers such as embedding, global max pool, dropout, dense, CONV, LSTM, GRU etc. The visualization of the proposed CNN architecture is shown in Figure 2, and the details of the main layers of the developed CNN model are summarized below.

1. Embedding layer: This layer was mainly developed for natural language processing and can efficiently preserve the word's contextual similarity, i.e. the representation of similar meanings of words is almost the same [19]. In the developed approach, the embedding layer is the first used layer.

2. The second used layer is the pooling: This layer mainly works on down-sampling the feature maps in order to reduce the size of the activation maps.

3. Dropout: In general, dropout refers to ignoring some randomly chosen neurons along with all its incoming and outgoing connections during the training phase [20]. In other words, it is temporarily dropping out some neurons for every training sample. The main aim of this process is to prevent the model overfitting.

4. Nonlinearity layer: Mainly, artificial neural networks are designed as universal functions that have the ability to calculate and learn any function. Nonlinear functions help networks to learn more strongly [21,22] sigmoid function and unipolar sigmoid are two popular activation functions.

4. Experiments

In this section, the performance of the proposed system for supporting both Turkish and English languages was investigated through multiple experiments. To ensure the robustness of the developed system, multiple databases, in which its details are shown in the following subsection, have been used in this study.

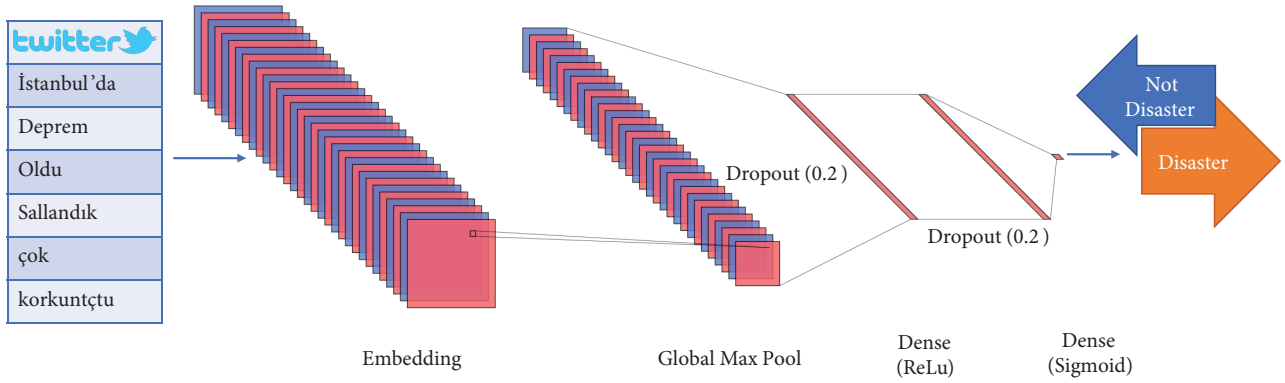


Figure 2. A visualization of the developed CNN architecture. The network consists of an embedding layer, a pooling layer followed by two dense layers.

4.1. Datasets and evaluation

In this work, three subdatasets were constructed from the newly developed Turkish dataset. These subdatasets were balanced, i.e. the number of disaster and nondisaster tweets is equal, where, the first one is related to the deprem, i.e. "earthquake", the second one is related to the yangın, i.e. "fire", and the third one is related to the trafik kazası "traffic accident". These datasets contain 4300, 2000, and 5700 tweets respectively. Related to the English language, another three subdatasets that contains 3000, 6000, and 10,860 tweets respectively were constructed from the ("socialmedia-disaster-tweets-relevent") dataset that includes 10,860 tweets collected using some search keywords such as "ablaze", "quarantine", and "pandemonium", then manually grouped into disaster and nondisastrous.

In the following experiments, accuracy, precision, recall, and $F1$ score which are standard metrics for evaluated classification systems have been used.

4.2. Implementation and setups of experiment

The proposed approach was implemented using Python, and libraries such as Keras, Pandas, NumPy. In more detail, related to the Turkish language, we used the library Turkish-deasciifier presented in [23] for the deasciification process, Turkishnlp[1] for text normalization, and Stemming was done using the python implementation of the TurkishStemmer presented in [24]. In addition, the preprocessing and feature extraction functions were imported from both Keras and Sklearn.

Related to the developed approach, it is important to note that for such model, i.e. CNN based, it is very critical to set the value of the following parameters, i) The best window-length, i.e. size of the used filters, and ii) The number of filters. The values 1, 3, 5, and 7 were tested for window length. While, 32, 64, 128, and 256 values were used for the number of the filter. In addition, selecting the most efficient technique for the VSM is a must step. Furthermore, ensuring the quality of both training and testing data is an essential request for building a system that can work efficiently in real-time. Table 2 shows the experimental setups, i.e. values of the main hyperparameters of experiment. It is important to note that the optimal values for the mentioned hyperparameters, i.e. the process of performing hyperparameter tuning in order to determine the best value was performed using the sklearn's GridSearchCV method [25].

[1]Çetinkaya M (2019). Turkishnlp [online]. Website https://pypi.org/project/turkishnlp/ [accessed 1 March 2020].

Table 2. Main hyperparameters of experiment.

Experiment #	Window size	Number of filters	Feature extraction techniques	Preprocessing
Experiment 1	1, 3, 5, and 7	64	TF-IDF	Not applied
Experiment 2	3	32, 64, 128, and 256	TF-IDF	Not applied
Experiment 3	3	64	TF, TF-IDF, FastText, BERT and Word2Vec	Not applied
Experiment 4	3	64	Word2Vec	Applied
Experiment 5	3	64	Word2Vec	Applied
Experiment 6				

Also, the variables "vocabulary size" was set to (the number of unique tokens in the used dataset + 1), the "embedding dim, i.e. dimension of the dense embedding, was set to 300, and the "maxlen, which refers to the length of the vector representing each tweet was set to the length of the longest tweet in the used dataset and whenever a tweet has fewer words, zeros will be appended to its vector. Last but not least, the cross-validation was used where each dataset is divided into 5 parts, one part for each of the test and validation, and the three remaining parts are used for training.

4.3. Experiments and results analysis

Section ♯1: Text preprocessing and hyperparameters specification

In this section, the abovementioned parameters were investigated in detail.

Experiment 1: Investigating the effect of changing the size of filters of CNN.

In this experiment, the effect of changing the size of filters of CNN, i.e. the window-length was investigated. The size was set to 1, 3, 5, and 7. In more detail, the introduced CNN model was used in this experiment to investigate the performance of all predefined filter sizes. In addition, As shown in Table 2, the number of filters was set to 64, TF-IDF is used for the feature extraction and finally, the row data was used, i.e. preprocessing was not applied.

As depicted in Table 3, almost all the tested values achieved the same performance. On the other hand, we have observed that increasing the window-length can significantly decrease the processing time.

Experiment 2: Effect of changing the number of CNN's filters.

One of the main factors that affect the overall performance of any CNN model is the number of used filters. In this experiment, the effect of changing the number of filters when processing both Turkish and English text was studied. The number of filters was set to 32, 64, 128 and 256. Similar to the previous experiment, the introduced CNN model was used to investigate the performance of all tested values. The results are shown in Table 4. It has been found that in most cases 64 was the best number of filters for the Turkish language. However, in the case of the English language, none of the numbers defeated the others significantly. Related to the processing (execution) time, it increases by increasing the number of filters, wherein our case, 32 filter requires the least execution time, and 256 filter requires the longest time.

Experiment 3: Comparing the performance of feature extraction techniques.

In this experiment, the performance of TF, TF-IDF, FastText, BERT, and Word2Vec were investigated. In more detail, each of these methods was integrated into the developed system and their performance was

Table 3. The effect of changing the size of filters of CNN.

Dataset	Accuracy			
	Window size			
	1	3	5	7
1st Turkish dataset	92.37	92.09	93.77	92.19
2nd Turkish dataset	90.60	90.60	88.00	86.80
3rd Turkish dataset	88.23	88.18	88.86	87.90
1st English dataset	74.83	76.56	75.50	75.77
2nd English dataset	79.41	77.55	78.28	79.75
3rd English dataset	80.77	80.52	80.37	80.00

Table 4. The effect of changing the number the used of filters.

Dataset	Accuracy			
	Number of filters			
	32	64	128	256
1st Turkish dataset	92.00	92.65	91.72	92.65
2nd Turkish dataset	90.40	90.00	89.60	89.00
3rd Turkish dataset	88.51	88.23	88.44	87.92
1st English dataset	77.23	75.23	77.23	73.37
2nd English dataset	77.81	77.28	76.82	78.08
3rd English dataset	80.92	79.45	81.29	80.59

investigated to find out which one is more suitable for the Turkish language. In addition, both bigrams and trigrams methods were used while implementing the FastText. It is important to note that this experiment has been started by training both Word2Vec and FastText using the "Turkish Wikipedia dump" dataset. In addition, we have also used both of the following Turkish pretrained word vectors: The first one that was trained on Common Crawl and Wikipedia using FastText [26], and the second one, which was originally trained using the "Turkish CoNLL17 corpus" that consisted of 3633786 vocabulary size. Overall, we decided to use the later as it has achieved better performance. On the other hand, related to BERT, we used i) the BERT tokenizer imported from the library "bert-for-tf2"(the BERT library for TensorFlow) and ii) the "bert base multilingual uncased", which supports 102 languages, including Turkish.

As shown in Table 5, with the improvement that was introduced by word embedding approaches, all FastText, BERT, and Word2Vec have significantly outperformed the TF and TF-IDF. In addition, we have proven that Word2Vec is the most appropriate compared to the other approaches for building an effacing system for processing both Turkish and English languages.

Experiment 4: The effects of the developed preprocessing model.

In this experiment, we have investigated the effect of the developed preprocessing model on the performance of processing both Turkish and English tweets. The main aims of this experiment are to find the affection degree of the performance by either using the row data or the preprocessed data. In addition, it is expected that some languages can be more sensitive to the nature of the data (preprocessed vs. row data). In other

words, we will find the percentage of improvement, if existed, for each of the studied languages. The results are shown in Table 6, and it is clear that preprocessing has improved the performance when processing the Turkish and English languages. In addition, the effect of the preprocessing increases and has become more important whenever the size of the used dataset increases. Furthermore, the developed CNN model achieved 6.26% as an improvement when processing the Turkish datasets. Hence, the preprocessing is a must step for building an efficient system that supports the Turkish language.

Table 5. Accuracy of TF, TF-IDF, FastText, BERT and Word2Vec when processing the Turkish and English languages.

Dataset	Accuracy					
	TF	TF-IDF	FastText		BERT	Word2Vec
			Bigrams	Trigrams		
1st Turkish dataset	57.12	51.53	90.63	91.15	91.96	93.21
2nd Turkish dataset	61.60	55.80	89.4	89.13	88.4	90.00
3rd Turkish dataset	57.32	49.46	87.24	87.39	85.47	88.44
1st English dataset	52.06	48.34	73.5	69.69	71.91	76.03
2nd English dataset	50.30	49.70	75.31	75.88	76.02	77.35
3rd English dataset	59.01	55.47	78.24	78.05	78.92	80.33

Table 6. The accuracy of classification with and without the preprocessing operation.

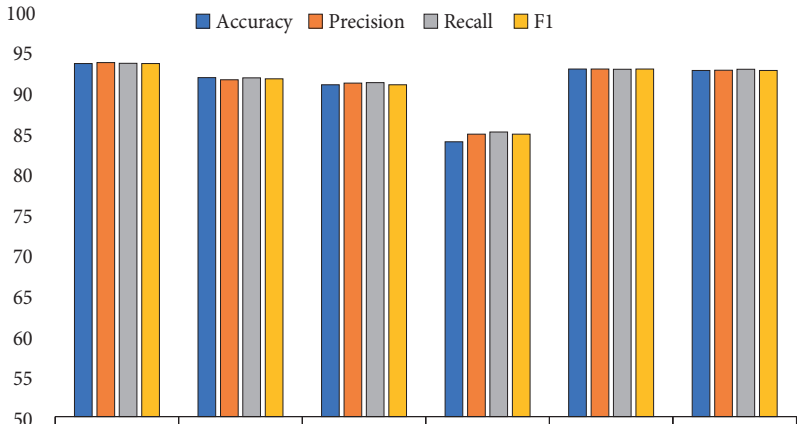
Dataset	Accuracy		Improvement (%)
	Without preprocessing	With preprocessing	
1st Turkish dataset	86.4	90.5	4.75
2nd Turkish dataset	83.36	87.29	4.71
3rd Turkish dataset	86.13	91.52	6.26
1st English dataset	76	79.67	4.83
2nd English dataset	75.6	78.27	3.53
3rd English dataset	79.56	83.7	5.20

Section ♯2: Overall performance of the proposed system

Experiment 5: Robustness and scalability of the proposed system.

In this experiment, the robustness and scalability of the proposed system was investigated using all the constricted Turkish and English datasets. Also, the accuracy, precision, recall, and the F1 score have been calculated. Figure 3 shows the results of this experiment, and it can be summarized as follows. a) For processing the Turkish language, the developed system was able to achieve above 90% accuracy and was able to process all the Turkish datasets efficiently. b) For processing the English language, the developed system was able to achieve accuracy in the range between 83%–92%. However, further improvement for the performance of the developed approach regarding English language is kept for future work, as the main aim of this paper is to support the Turkish language. c) Regarding the robustness and scalability while increasing the number of processed tweets, as shown in Figure 3, the developed system was able to keep achieving good results while increasing the number of samples(tweets). Overall, this system has stability and can handle all the datasets

efficiently.



	Turkish dataset			English dataset		
	1st	2nd	3rd	1st	2nd	3rd
Accuracy	93.52	91.78	90.89	83.88	92.82	92.65
Precision	93.63	91.52	91.09	84.82	92.83	92.69
Recall	93.53	91.75	91.16	85.06	92.81	92.79
F1	93.52	91.63	90.89	84.8	92.82	92.65

Figure 3. The accuracy, precision, recall, and F1 score for the developed system. Overall, the developed system was able to achieve accuracy above 90% for all the Turkish datasets and between 83%–92% for the English ones.

Experiment 6: Comparing our approach with the automated text classification, CREES, and deep multimodal.

In this experiment, the performance of the proposed approach was compared with the approaches presented in [1], [5] and [6]. In detail, we have reimplemented the aforementioned approaches, and their hyperparameters were set as stated in the respected references. In addition, the value of N, for the approach of [1], which refers to the number of used features that have the highest chi-square and nonnegative values, was set to 100 in this experiment. Furthermore, as this paper interested in processing the text, in the case of [6], only the text model was considered and implemented.

Figure 4 shows the experiment's results, and it is clear that the developed approach has significantly outperformed all approaches when processing the Turkish datasets. On the other hand, related to the English datasets, our approach and the state-of-the-art approaches presented in [5] and [6] were able to achieve a comparable performance for the first datasets (the smallest datasets), but for the other largest two datasets, our approach was also able to outperform all other ones. In addition, surprisingly, the approach of [1] was able to achieve good results compared to the CNN models. Based on our observations, this due to the fact that the "chi-square" method was able to select the most representative features.

5. Conclusion and Future works

Nowadays, social media analytic is an important research field and has improved the internet-based CMSs. In this work, a CNN tweet classification system for crisis response that fully supports the Turkish Language was developed. In addition, the system has the ability to handle the English language. The system was built after a deep study that aims to find the most suitable layers and components to build an efficient CNN. In addition,

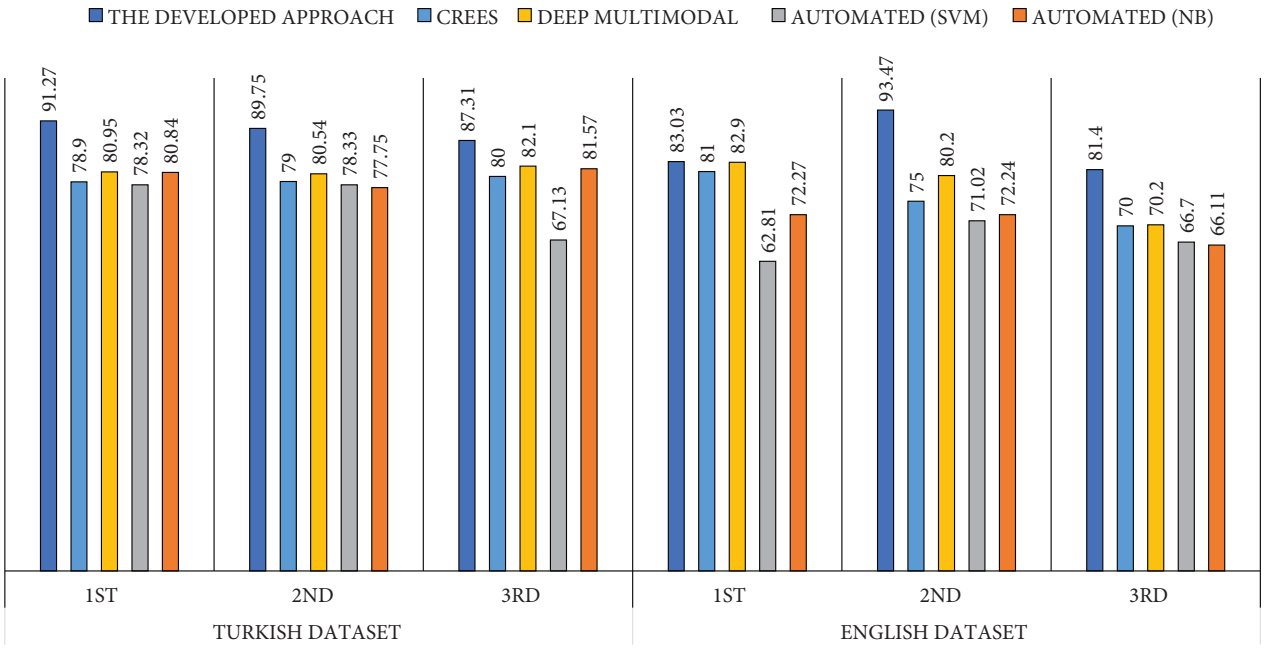


Figure 4. The accuracy of the developed system, automated text classification [1], CREES [5] and deep multimodal [6]. Overall, the proposed approach has clearly outperformed the others especially for the Turkish datasets.

a Turkish tweet dataset for crisis response, which can be widely used for similar studies, was contracted. This dataset has been carefully preprocessed, annotated, and well organized. It is compatible with NLP well-known tools.

As shown in the experiments section, the developed approach has achieved a quite good performance, robustness, and stability when processing both Turkish and English languages.

One of the directions that can be done as future work is to integrate multiple CNN models in parallel for improving such systems. Another direction is to investigate the performance of the recurrent neural network (RNN) for building such systems.

References

[1] Ragini JR, Anand PR. An empirical analysis and classification of crisis related tweets. In: 2016 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC); New York, NY, USA; 2016. pp. 1-4.

[2] Alqaraleh S, Işik M. Comparative analysis of classification techniques for crisis management systems. In: The International Conference of Materials and Engineering Technologies (TICMET19); Gaziantep, Turkey; 2019. pp. 584-592.

[3] Alqaraleh S. Classification of Turkish text using machine learning: a case study using disasters tweets. International Journal of Scientific & Technology Research 2020; 9 (3): 4953-4956.

[4] Kumar A, Singh JP. Location reference identification from tweets during emergencies: a deep learning approach. International Journal of Disaster Risk Reduction 2019; 33: 365-375.

[5] Burel G, Alani H. Crisis event extraction service (CREES) - automatic detection and classification of crisis-related content on social media. In: ISCRAM - 15th International Conference on Information Systems for Crisis Response and Management; New York, NY, USA; 2018. pp. 597-608.

[6] Kumar A, Singh JP, Dwivedi YK, Rana NP. A deep multi-modal neural network for informative Twitter content classification during emergencies. Annals of Operations Research 2020; 1: 1-32.

[7] Baygın M. Classification of text documents based on naive Bayes using N-Gram features. In: 2018 International Conference on Artificial Intelligence and Data Processing (IDAP); Malatya, Turkey; 2018. pp. 1-5.

[8] Kılınç D. The effect of ensemble learning models on Turkish text classification. Celal Bayar Üniversitesi Fen Bilimleri Dergisi 2016; 12 (2): 14-25.

[9] Kilimci ZH, Akyokus S. Deep learning-and word embedding-based heterogeneous classifier ensembles for text classification. Complexity 2018; 2018: 1-20.

[10] Kilimci ZH, Akyokuş S. The evaluation of word embedding models and deep learning algorithms for Turkish text classification. In: 2019 4th International Conference on Computer Science and Engineering (UBMK); Samsun, Turkey; 2019. pp. 548-553.

[11] Aydoğan M, Karci A. Improving the accuracy using pre-trained word embeddings on deep neural networks for Turkish text classification. Physica A: Statistical Mechanics and Its Applications 2020; 1: 541.

[12] Perkins J. Python 3 Text Processing with NLTK 3 Cookbook. USA: Packt Publishing Ltd, 2014.

[13] Christian H, Agus MP, Suhartono D. Single document automatic text summarization using term frequency-inverse document frequency (TF-IDF). ComTech: Computer, Mathematics and Engineering Applications 2016; 7 (4): 285-294.

[14] Li H, Caragea D, Li X, Caragea C. Comparison of word embeddings and sentence encodings as generalized representations for crisis Tweet classification tasks. In: ISCRAM Asian Pacific Conference; Wellington, New Zealand; 2018. pp. 1-13.

[15] Liu Q, Kusner MJ, Blunsom P. A survey on contextual embeddings. ArXiv 2020; arXiv:2003.07278 [cs.CL].

[16] Pennington J, Socher R, Manning CD. Glove: Global vectors for word representation. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP); Doha, Qatar; 2014. pp. 1532-1543.

[17] Joulin A, Grave E, Bojanowski P, Douze M, Jégou H et al. FastText.zip: Compressing text classification models. ArXiv 2016; arXiv:1612.03651 [cs.CL].

[18] Şahin G. Turkish document classification based on Word2Vec and SVM classifier. In: 25th Signal Processing and Communications Applications Conference (SIU); Antalya, Turkey; 2017. pp. 1-4.

[19] O'Keefe SE, Alrashdi RM. Deep learning and word embeddings for tweet classification for crisis response. In: The 3rd National Computing Colleges Conference; Abha, Saudi Arabia; 2018. pp.1-20.

[20] Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R. Dropout: a simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research 2014; 15 (1): 1929-1958.

[21] Khan SH, Hayat M, Porikli F. Regularization of deep neural networks with spectral dropout. Neural Networks 2019; 110: 82-90.

[22] Ramachandran P, Zoph B, Le QV. Searching for activation functions. In: Sixth International Conference on Learning Representations; Vancouver, Canada; 2018. pp.1-20.

[23] Adali K, Eryiğit G. Vowel and diacritic restoration for social media texts. In: Proceedings of the 5th Workshop on Language Analysis for Social Media (LASM); Gothenburg, Sweden; 2014. pp. 53-61.

[24] Eryiğit G, Adalı E. An affix stripping morphological analyzer for Turkish. In: Proceedings of the IASTED international conference artificial intelligence and applications; Innsbruck, Austria; 2004. pp. 299-304.

[25] Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B et al. Scikit-learn: machine learning in Python. Journal of Machine Learning Research 2011; 12: 2825-2830.

[26] Grave E, Bojanowski P, Gupta P, Joulin A, Mikolov T. Learning word vectors for 157 languages. In: Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018); Miyazaki, Japan; 2018. pp. 3483-3487.