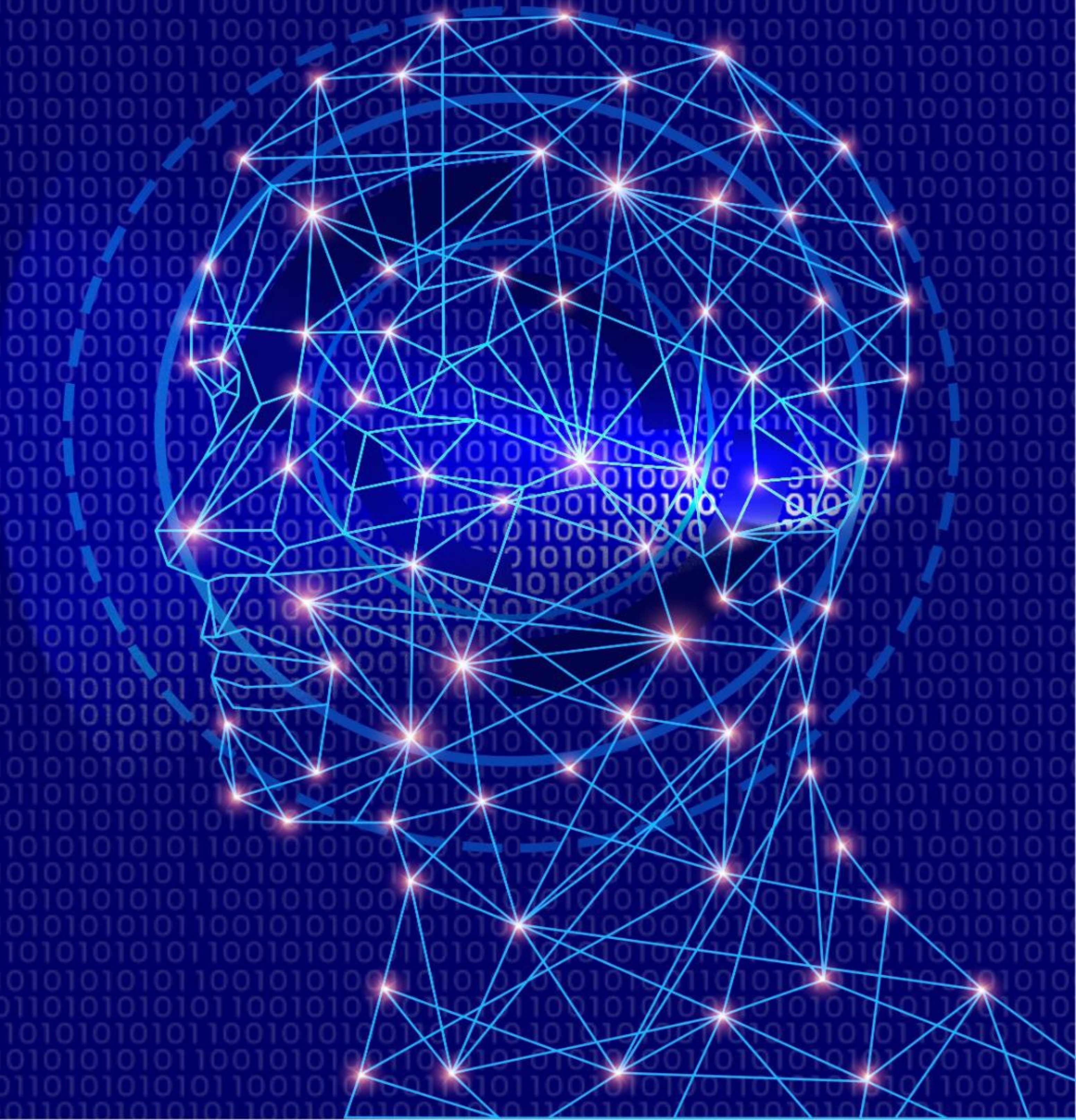
International Journal of

ADVANCES IN ARTIFICIAL INTELLIGENCE RESEARCH

AAIR

Volume 3, Issue 2, 2023
ISSN: 2757-7422



International Journal of
ADVANCES IN ARTIFICIAL INTELLIGENCE RESEARCH
AAIR

Advances in Artificial Intelligence Research

https://dergipark.org.tr/aair

Owner

Osman ÖZKARACA - Muğla Sıtkı Koçman University

Editor-in-Chief

Ali KEÇEBAŞ - Muğla Sıtkı Koçman University
Osman ÖZKARACA - Muğla Sıtkı Koçman University

Editors

Dr. Hüseyin ŞEKER	Staffordshire University, School of Computing and Digital Tech, England
Dr. Tuncay YİĞİT	Süleyman Demirel University, Computer Engineering Department, Turkey
Dr. Uğur Güvenç	Electric-Electronic Engineering, Düzce University, Düzce, Turkey
Dr. Jude HEMANTH	Karunya University, Electronics and Communication Engineering, India
Dr. Yusuf SÖNMEZ	Gazi University, Vocational College of Technical Sciences, Turkey
Dr. Ender ÖZCAN	Nottingham University, Computer Science and Operational Research, England
Dr. Hamdi Tolga KAHRAMAN	Karadeniz Technical University, Software Engineering, Turkey
Dr. Bogdan PATRUT	Alexandru Ioan Cuza University, Faculty of Computer Science, Romania
Dr. Ali Hakan IŞIK	Mehmet Akif Ersoy University, Computer Engineering, Turkey
Dr. İsmail Serkan ÜNCÜ	Isparta Applied Sciences University, Electrical-Electronics Engineering Turkey
Dr. Gürcan Çetin	Information Systems Engineering, Muğla Sıtkı Koçman University, Turkey
Dr. İsmail Yabanova	Mechatronics Engineering, Afyon Kocatepe University, Turkey

Date of Publication	October 2023
Language	English
Frequency	Published twice in a year
Graphic designer	Özden Işıktaş
Correspondence Address	Muğla Sıtkı Koçman University, Faculty of Technology, Information Systems Engineering, 48000 Kötekli/MUĞLA
Phone	0 506 4887818 – 0252 211 5526
Correspondence Mail	osmanozkaraca@mu.edu.tr

Volume : 3 2023 Issue : 2

International Journal of
ADVANCES IN
ARTIFICIAL
INTELLIGENCE
RESEARCH
AAIR

Table of Contents

Pages	Research Articles
45 – 53	CREATING A NEW DATASET FOR THE CLASSIFICATION OF CYBER BULLYING **Çilem Koçak, Tuncay Yiğit, Mehmet Bilen**
54 - 65	DATA CENTER CONTROL APPLICATION WITH FUZZY LOGIC **Hasan Yılmaz, Adem Alpaslan Altun, Mehmet Bilen**
66 - 75	DEVELOPMENT OF A TRAFFIC SPEED LIMIT SIGN DETECTION SYSTEM BASED ON YOLOV4 NETWORK **Semih Selçuk, Sefa Beker, Ömer Faruk Boyraz**
76 - 84	DEEP LEARNING ENSEMBLE APPROACH TO AGE GROUP CLASSIFICATION BASED ON FINGERPRINT PATTERN **Olufunso Stephen Olorunsola, Olorunshola Oluwaseyi Ezekiel**
85 - 95	A MACHINE LEARNING APPROACH FOR SIMULTANEOUS CLASSIFICATION OF MATERIAL TYPES AND CRACKS **Ömer Mintemur**
96 - 107	ANALYZING THE IMPACT OF AUGMENTATION TECHNIQUES ON DEEP LEARNING MODELS FOR DECEPTIVE REVIEW DETECTION: A COMPARATIVE STUDY **Anusuya Krishnan, Kennedyraj Mariafrancis**

Volume : 3 2023 Issue : 2

Advances in Artificial Intelligence Research (AAIR)
ISSN 2757-7422
www.dergipark.com/aair/

Vol. 3 (No. 2), pp. 45-53, 2023
doi: 10.54569/aair.1206144
Published online: Oct 23, 2023

Creating a New Dataset for the Classification of Cyber Bullying

Çilem Koçak [1,*], Tuncay Yiğit [2], Mehmet Bilen [3]

[1] Isparta University of Applied Sciences, Yalvaç Vocational School of Technical Sciences, Isparta, Türkiye
[2] Süleyman Demirel University, Faculty of Engineering, Department of Computer Engineering, Isparta, Turkey
[3] Mehmet Akif Ersoy University, Golhisar School of Applied Science, Turkey

Abstract

Regardless of young or old, people have quickly stepped into the world of internet with today's communication technologies such as phones, tablets, computers and smart devices. As the place of the Internet in people's lives increases, social media platforms are diversifying and users want to take part in these platforms. With the increase in the number of social media users, some negativities are encountered. The most important problem encountered in social media platforms is cyber bullying. Although cyber bullying seems to be a daily dialogue between social media users or between groups, the situation of encountering is increasing day by day with the diversity of shared information, content and agenda social media environments. With the development of technology, it is necessary to develop a platform that detects bullying with artificial intelligence technologies. One of the biggest difficulties in text classification problems that we encounter during the development of these platforms is the need to train the artificial intelligence algorithm to be used with labeled data. In this study, 21 different people, including journalists, athletes, scientists, doctors, politicians, comedians, social media phenomena, and artists who actively use social media, were selected in order to create the necessary dataset for training the models to be developed to detect cyber bullying situations. The public messages (mentions) of these 21 people sent via Twitter were compiled. After filtering the repetitive and meaningless messages sent by bot accounts out of 10500 tweets compiled, the number of messages in the dataset decreased to 7706. The labeling process, which is necessary for the dataset to be used for training and testing purposes in classification processes, was carried out by three independent people who were given preliminary information about cyberbullying (1=Includes Cyber bullying, 0=Does not include Cyber bullying). The majority of the tags, which were read and assigned by 3 different people, were accepted as the final class of the relevant message. Afterwards, the dataset was preprocessed in accordance with the principles of natural language processing and made suitable for classification algorithms. The findings obtained after the classification processes performed with the basic classification algorithms are shared. When the findings are examined, it is understood that the data set created has the competence to be used in the detection and prevention of cyber bullying. In this context, it is predicted that training specially developed and optimized artificial intelligence algorithms with the relevant dataset for the detection of cyberbullying will greatly increase the success rate.

Keywords: *Cyber bullying; Twitter; artificial intelligence; text classification; data labeling.*

1. Introduction

With the development of telephone and computer technologies and the increase in the number of social media platforms, the likelihood of cyberbullying behaviors is also increasing. Victims of cyberbullying are threatened by users with electronic communication tools, often receive messages containing written insults, and face actions such as making someone look bad with a false identity. In this case, problems of mutual relations between the bully and the victim arise [1]. It is thought that these problems arise from the feeling of revenge caused by the deterioration of friendship and emotional relations between people, and the written disagreement between people who have different views and thoughts [2]. In the case of cyber bullying behavior, regardless of the means and environment in which the cyber bullying is carried out, it is desired to create a destructive result on the victim, to hurt, humiliate, humiliate and leave permanent traces on the victim, but the social relations of the victim are adversely affected. In addition, as a result of this, emotional, social and psychological damage occurs.

Since the platforms where cyberbullying behaviors are most common are social media, the increase in the number of users on these platforms directly increases the victimization [3]. In virtual environments;

- With its feature of hiding identity, it leads to the idea of having the right to say whatever they want to its victims.
- People can easily say things that they cannot say exactly what they want face to face, and they can

*Corresponding author
E-mail address: cilemkocak@isparta.edu.tr

Received: 17/Nov/2022; Received in revised form: 21/Mar/2023; Accepted: 22/Apr/2023.

isolate themselves by giving reactions that show that the other person does not approve or care about them.

- It allows victims to say what they want to say when they want and without censorship.
- Since people do not have any idea about the bully, they cannot express their thoughts and themselves clearly.
- It enables individuals to easily say their gender, social status, race and similar features that they cannot express in face-to-face communication.
- Individuals with an aggressive personality become more aggressive, causing them to express their personal style in an exaggerated way.

Looking at the rate of increase in the use of social media around the world, according to the January 2021 data of Statista [4], 63% of people around the world use Facebook, 61% use YouTube, 48% use WhatsApp, followed by Facebook Messenger, Instagram, Twitter and Snapchat are used.

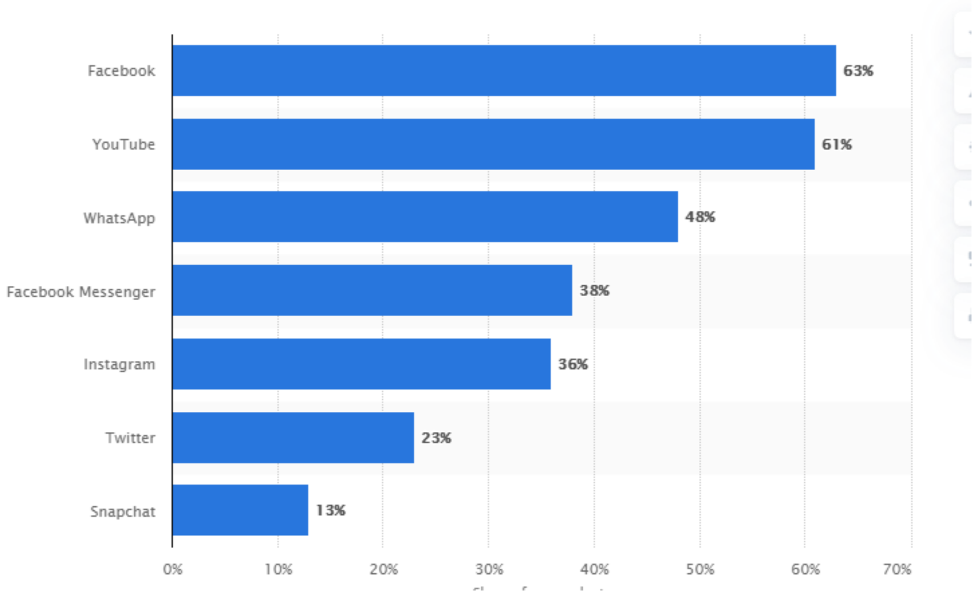


Figure 1. *We Are Social January 2021 Worldwide Social Media Usage Statistics*

According to the We Are Social 2021 report, looking at Turkey's internet, social media and mobile user statistics, 74% of Turkey's population is 62 million internet users, 64% of Turkey's population is 54 million social media users, 92% of Turkey's population is 77 million mobile users' forms. According to the We Are Social 2021 report, worldwide social media usage statistics are given in **Figure 1**.

As of 2021, the number of internet users in Turkey has approached 66 million with an increase of approximately 4 million within a year. This figure corresponds to a 6% increase in 1 year [5]. We see that there are 60 million social media users among the total population approaching 85 million. This means that 70.8% of the population is a social media user. In Turkey, 7 out of 10 people use social media and 9 out of 10 people use mobile devices. According to the results of the research, it is seen that those who are exposed to bullying behaviors experience different psychological disorders. These; mental illnesses such as sleep disorder, attention disorder, feeling of loneliness, depression. In some studies, it has been observed that the tendency to suicide increases in those who are exposed to cyber bullying [2].

Labeled datasets are needed to develop software that can detect cyber bullying behaviors. In this study, a new dataset was created in order to automatically detect cyber bullying behaviors and prevent the individual from being exposed to these behaviors.

In the continuation of the study, the concept of cyber bullying was examined and the studies carried out with machine learning in this field were mentioned in the literature. In the Material and Method heading, the steps followed in the dataset creation stages, the operations performed to make the dataset suitable for machine learning algorithms, and the classification steps are explained in detail. The results obtained in the research findings were interpreted and shared.

2. Cyber Bullying

Cyber bullying and types of bullying in the physical environment are similar to each other, although the environment in which the bullying takes place is different. It is seen that social media tools are used to exhibit cyber bullying behaviors. Bullies engage in cyber bullying in many ways. Common and classified types of cyber bullying;

- Cyber tracking: Keeping a person under constant surveillance in virtual environments,
- Slandering: Making false, harmful and rude statements about a person,
- Presenting Oneself as Someone Else: Impersonating an imaginary person or someone else by hiding their identity on the internet,
- Harassment: Sending offensive or sexually explicit messages to a person,
- Provoking: Encouraging a person for situations that he should not do,
- Wandering and Deception: Spreading embarrassing and private information about a person,
- Separation: Removing or not including a person from a group [1]

Žufic et al. (2017), Ayas and Horzum (2011), Karabatak et al. (2018), Arıcak(2011), Baker and Kavşut(2007), Arıcak, Siyahhan Uzunhasanoğlu, Sarıbeyoğlu, Çubuk, Yılmaz, and Memmedov (2008) tried to determine the information they have on cyber bullying by conducting surveys on the subject of cyber bullying, taking into account the age groups in different fields. carried out their studies. Within the scope of these studies, cyber bullying detection questionnaire and situation assessment questionnaire were used. As a result, it has an important place in the literature in terms of awareness of cyber bullying [6 - 11]. Hussain et al. (2018), Al-Mamun and Akhter (2018) used many artificial intelligence methods and techniques to detect cyber bullying data from different social media platforms [12,13]. The most used of these methods and techniques are; Dvm, Chi2, DVM-RFE, MRMR, C4.5 Decision Tree, k-nn classifiers, Maximum Entropy method, convolutional neural networks (Convolutional Neural Networks: CNN), bidirectional long short-term memories (Long Short-Term Memory: LSTM)) and Gated Recurrent Units (GRU), Naive Baeyes (NB), Random Forest (RF). In addition to these, there are different studies in the literature.

3. Material and Method

In this part of the study, firstly, how the dataset was obtained and the labeling process were explained in detail. Afterwards, the preprocessing performed on the dataset, the methods used to classify the text and the models used in this study for the classification of cyber bullying are presented respectively.

3.1. Dataset

The field of artificial intelligence contains many algorithms that learn from their experiences. A large amount of data is needed to obtain these experiences. In this study, it is aimed to collect data via Twitter social media application and to adapt this data to these algorithms in order to provide resources for artificial intelligence-based studies in the field of cyber bullying.

The tweets sent to the profiles of 21 different people who are famous in many different fields and actively use the Twitter social media application in the last 15 days were compiled using a program written in the Python programming language with the help of the API obtained by creating a Twitter developer account. These tweets must be tagged in order to be used in the training of artificial intelligence. Since the labeling process is done using human power, it is one of the processes that has the greatest impact on the development speed of artificial intelligence. In this study, tagging was carried out by students studying at the graduate level, who were given preliminary information about bullying and artificial intelligence. Each tweet was evaluated independently by three different people (1 = Contains bullying, 0 = No bullying) and the rating with the most votes was assigned to the class information of the relevant tweet. In this evaluation process, tweets that are thought to be sent from bot accounts, repeated and meaningless tweets were removed from the dataset with the initiative of the evaluators. At the end of the evaluation process, which started with 10500 tweets, it decreased to 7706.

3.2. Pretreatment

A large part of artificial intelligence algorithms depends on the number, type, size, etc. of data. is extremely sensitive. For this reason, preprocessing steps are of great importance for both adapting the data to artificial intelligence algorithms and increasing the success of these algorithms. Considering a traditional natural language processing preprocessing process and the characteristics of texts taken from social media, the following processing steps were followed in this study, respectively.

- Emoji cleaning
- Link, hashtag, mention information etc. cleaning up
- Conversion of uppercase letters to lowercase letters by considering Turkish characters
- Rooting each word (stem)
- Words, conjunctions, etc. that do not have a meaning on their own. removal from text (stopwords)

In order to extract the emojis in the sent tweets, the hexadecimal value of each character was obtained and compared with the intervals given in **Table 1**. Values within these ranges were removed from the text and the text was freed from emojis.

Table 1. *Unicode values of emojis*

Unicode Range	**Emoji Type**
0001F600-0001F64F	Feelings
0001F300-0001F5FF	symbols
0001F680-0001F6FF	Map and Logistics Symbols
0001F1E0-0001F1FF	Flags

In the queries written for cleaning links, hashtags and quotes, the characters and strings in **Table 2** were considered distinctive and those matching these characters were cleared from the data set. In addition, the different codes of uppercase and lowercase letters cause words that are actually the same to be perceived as different words by algorithms. For this reason, all capital letters were converted to lowercase letters, paying attention to Turkish characters.

Table 2. *Distinctive character strings*

Character Type	**Type, Variety**
#	hashtag
http://, https://, mailto:, www., .com	Links
@	Mention information
RT	Repost information

Since Turkish is an agglutinative language, natural language processing processes are more difficult than conventional (English, German, French, Spanish, etc.) languages. Unlike these languages, each word in Turkish can have more than one suffix. Each suffix causes the related word to be evaluated as another word. For this reason, it is necessary to purify the words from their suffixes. Although it is thought that the performance of artificial intelligence is negatively affected due to the elimination of attachments that contribute to the meaning in this process, this step should be carried out in order to minimize the operating cost of algorithms and to eliminate the incompatibility problems caused by excessive data complexity and data size. The "TurkishStemmer" library was used in the study developed with Python in order to separate the words into their roots [14]. The "turkishStopwords" list in NLTK [15], which is a frequently used natural language processing library, was used to extract words that are mentioned as stopwords in the literature and that do not have a meaning on their own. The words in the list are:
'acaba', 'ama', 'aslında', 'az', 'bazı', 'belki', 'biri', 'birkaç', 'birşey', 'biz', 'bu', 'çok', 'çünkü', 'da', 'daha', de', 'defa', 'diye', 'eğer', 'en', 'gibi', 'hem', 'hep', 'hepsi', 'her', 'hiç', 'için', 'ile', 'ise', 'kez', 'ki', kim', mı', 'mu', 'mü', 'nasıl', 'ne', 'neden', 'nerde', 'nerede', 'nereye', 'niçin', 'niye', 'o', 'sanki', 'şey', 'siz', şu', 'tüm', 've', 'veya', 'ya', 'yani'

3.3. Text Classification

The text classification problem can be summarized as determining or predicting which class the texts belong to according to the meaning they contain. It has become a critical work area with digital transformation. The fact that the amount of text that appears daily around the world has reached extremely large sizes has made it necessary to perform text classification automatically [16]. In order for the text to be analyzed with certain algorithms, it must be digitized. In this way, the data can be structured and the relationships between the data can be revealed [17]. When the studies in the literature in the field of text classification are examined, we come across three different methods for solving the problem;

- Rule Based Systems
- Machine Learning Based Systems
- Hybrid Systems

The rule-based approach is seen in the literature as a classification method in which the cause-effect relationship can be examined. With this method, the details of the classification process can be observed and additions can be made easily to improve the result [18]. Rule-based approaches consist of a set of if-not (IF-ELSE) rules [19]. Thanks to these rules, language-specific patterns are determined and it consists of structures that decide which class the relevant text belongs to. Jrip (Rajput, 2011), OneR (Buddhinath, 2006), ZeroR (Sayad, 2022) can be given as examples of these systems [20 - 22].

The biggest disadvantage of rule-based approaches is that rules must be created by linguistics experts. Contrary to rule-based approaches, machine learning algorithms used for text classification can learn from labeled data and create the relationships and rules between texts. In **Figure 2**, the flow diagram of traditional models that perform text classification with machine learning is given.

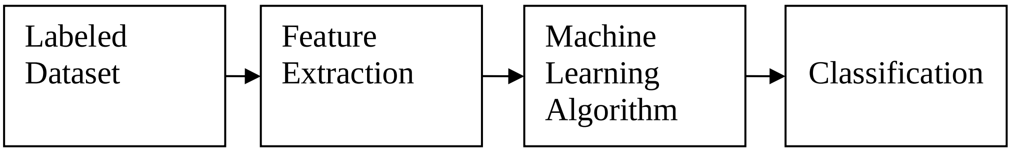


Figure 2. *Text classification model with machine learning*

In machine learning-based approaches, firstly, the labeled data is pre-processed to make it suitable for these algorithms and digitize it by feature extraction. Then, it is aimed to reveal the relationships between the data by using the new dataset obtained for the training of the algorithm. Finally, with the help of learned relations, a model emerges that can decide which class the new data belong to. Many machine learning algorithms such as artificial neural networks, support vector machine, Naive Bayes and Deep learning can be used by training for text classification purposes. However, due to the complex mathematical structures of these algorithms, the training phases take quite a long time. K Nearest Neighbors algorithm, which is an effective machine learning algorithm, was preferred in this study because of its simple structure and fast operation in order to determine whether the data set is classifiable or not.

Hybrid approaches, on the other hand, consist of a combination of a trained machine learning algorithm and the advantageous aspects of a rule-based approach to increase classification success.

3.4. Approach

The method followed to perform the training and classification processes with the machine learning algorithm after the compilation of the dataset is given in **Figure 3** as a flow chart.

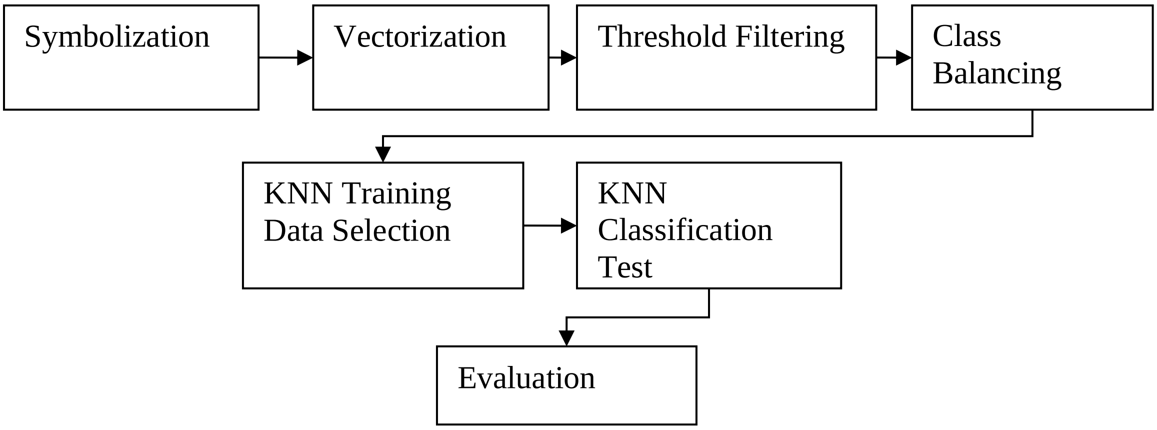


Figure 3. *Method followed for text classification*

Symbolization is simply the process of assigning a number to each word. In this way, a code transformation that algorithms can process is applied to the data. However, in this mathematical transformation, the numerical difference between a code of a text and another code has no meaning. For this reason, each sample in the dataset must be coded separately by representing which text it contains or does not contain with 0s and 1s. In this case, each sample has as many attributes as the number of unique texts and the size of the dataset grows and becomes more complex. In order to prevent this, words with a low number of repetitions were removed from the data set by selecting a certain threshold value. Just before the training and testing phase, a new data subset was created by randomly selecting the same number of samples from both classes in order to eliminate the uneven distribution of the classes in the dataset. 80% of the data obtained was used for training purposes to create the experience of the KNN algorithm. The remaining 20% was used for classification testing and the success of the model was tested. The results obtained were evaluated by sharing them under the title of research findings.

4. Results and Discussion

The tweets collected from the Twitter social media application are presented in **Figure 4**. When the figure is examined, a dataset with an unbalanced class distribution is seen. This imbalance is a situation that negatively affects the success of machine learning algorithms. Most of the machine learning algorithms are not sensitive to this imbalance and therefore they tend to predict in favor of the class with the large number of samples. To solve this problem, the samples in the 0 (No Cyberbullying) class were randomly eliminated and the number of samples in the two classes was equalized.

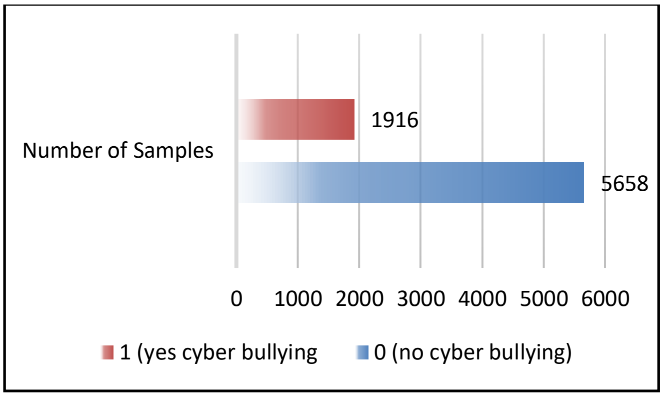


Figure 4. *Distribution of Sample Numbers by Classes*

In order to examine the characteristic features of the compiled tweets, the histogram of the number of words they contain is given in **Figure 5**. When the histogram is examined, it is seen that most of the tweets sent are under 10 words. It is expected that the number of words will be low due to the character limit set by the relevant social media application in the sent messages.

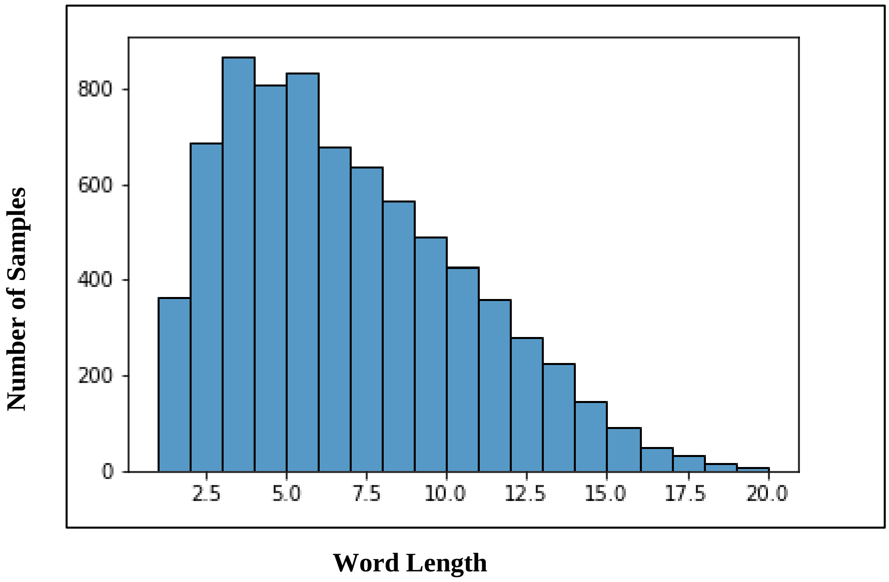


Figure 5. *Word Length Histogram*

The bi-grams in the dataset are sized depending on the repetition frequency and presented in Figure 6.

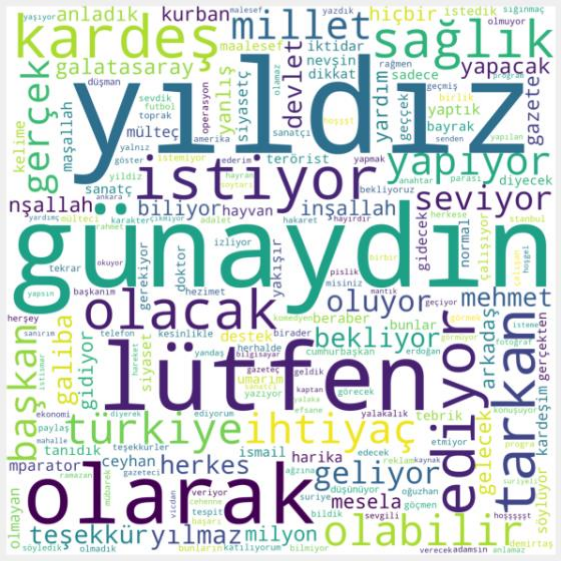


Figure 6. *Word Cloud (Frequency / Dimension)*

Classification analyzes after the preprocessing steps performed on the dataset and the selection processes performed to eliminate the imbalance were carried out by writing Python code on the Colaboratory Jupiter Notebook service provided by Google and on the virtual computer offered by Google.

As mentioned in the method section, the threshold value filtering method was used to reduce the complexity of the data size. Words with repetitions below a certain threshold were excluded from the dataset. All values between 1 and 250 were repeated to determine this threshold value (**Figure 7**). It is seen that the highest success in the classification processes performed by selecting different threshold values is obtained around 100 repetitions. When the operating time of the algorithm is examined (**Figure 8**), it is seen that the same threshold value has a positive contribution to the reduction of the time taken for analysis. Since the threshold value of 100 was confirmed in both graphs, this value was determined as the threshold value for the analyzes performed.

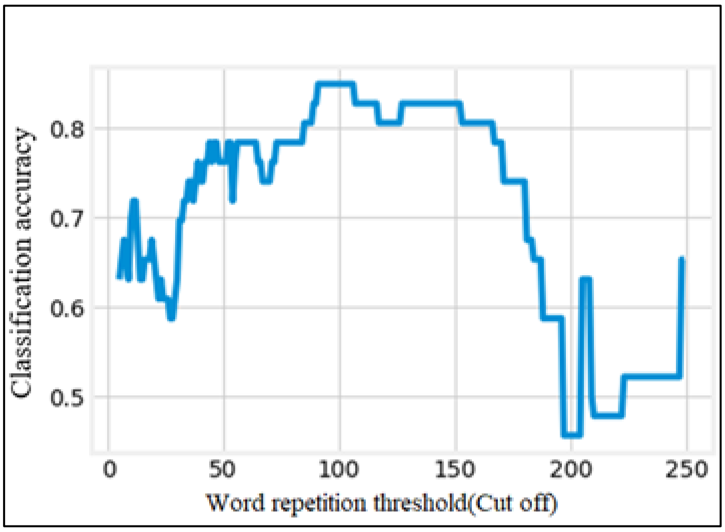


Figure 7. *Effect of Threshold Filtering on Classification Success*

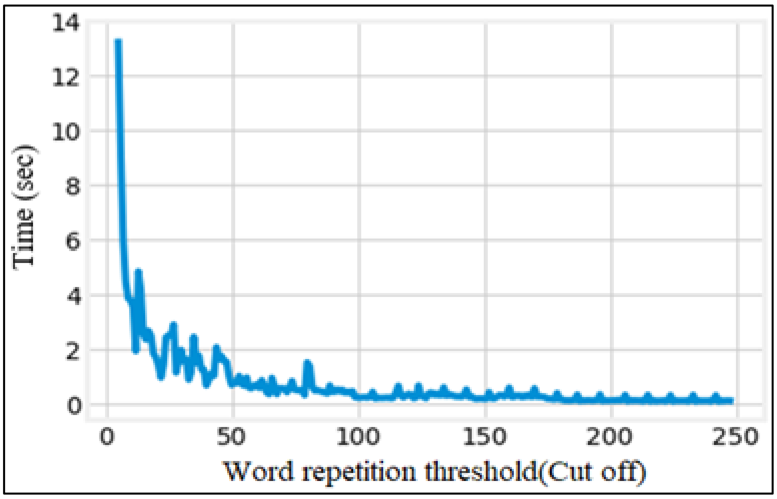


Figure 8. *Effect of Threshold Filtering on Algorithm Training and Classification Time*

KNN is a simple and fast but effective algorithm compared to other machine learning algorithms. The K value in the algorithm is one of the most important parameters that determine the success performance of this algorithm. Since there is no linear method to determine this parameter, which differs according to each data set, it is aimed to determine the most appropriate K parameter by iterating with different K values, as in the threshold value determination method. The graph obtained as a result of iteration is presented in **Figure 9**. When the graph is examined, it is seen that the value of 3 is the most appropriate K parameter in the classification processes to be performed with KNN on the data set prepared in this study.

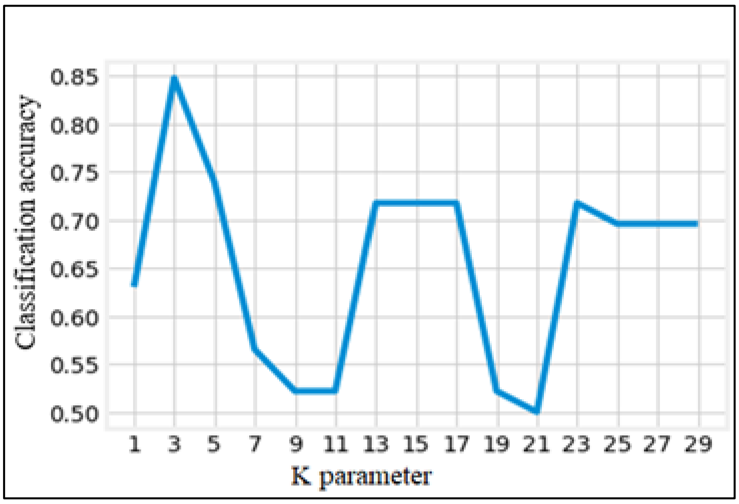


Figure 9. *Effect of K Parameter on Classification Result*

The findings obtained from the classification test processes performed after the determination of the most appropriate word repeat threshold value and the K parameter are presented in **Table 3**. The table shows that the prepared dataset can be successfully classified even with an algorithm that requires little training, such as KNN. It is seen that the prepared dataset is distinguishable and can identify the tweets containing bullying with a success rate of 85%.

Table 3. *Classification Performance Values*

	Precision	Recall	F1-Puanı
0 – No Bullying	0.79	0.9	0.87
1 – There Is Bullying	0.94	0.73	0.82
Percentage of Success			85
Macro Average	0.87	0.84	0.84
Micro Average	0.86	0.85	0.85

5. Conclusion and Suggestions

In this study, a new dataset was created to be used in machine learning algorithms to be developed to detect cyber bullying behaviors. As a result of the improvement processes and analyzes performed on the data set, it is seen that the data set can be classified at a rate of 85% with the KNN algorithm. It is thought that this rate will increase greatly with the evaluation of the dataset by algorithms with long training time such as deep learning.

Declaration of interest

The authors declare that there is no conflict of interest. It was presented as a summary at the ICAIAME 2022 conference.

References

[1] Gezgin, D. M., & Çuhadar, C. "Bilgisayar ve öğretim teknolojileri eğitimi bölümü öğrencilerinin siber zorbalığa ilişkin duyarlılık düzeylerinin incelenmesi", *Eğitim Bilimleri Araştırmaları Dergisi*, *2*(2) (2012), 93-104.

[2] Özdemir, M., & Akar, F. "Lise Öğrencilerinin Siber-Zorbalığa İlişkin Görüşlerinin Bazı Değişkenler Bakımından İncelenmesi", Kuram ve Uygulamada Eğitim Yönetimi, 4(4) (2011), 605-626.

[3] Eroğlu, Y., Güler, N. "Koşullu Öz-Değer, Riskli İnternet Davranışları ve Siber Zorbalık/Mağduriyet Arasındaki İlişkinin İncelenmesi", Sakarya University Journal Of Education, 5(3) (2015), 118-129.

[4] Global social media usage report 2021, https://www.statista.com/ (accessed: Apr 10, 2022).

[5] Turkey Internet, social media and Mobile User Statistics According to We Are Social 2020-2021 Report Https://Wearesocial.Com/ (accessed: Jun 15 2022).

[6] Žufić, T. Žajgar, S. Prkić, "Children Online Safety", 2017 40th International Convention On Information And Communication Technology, Electronics And Microelectronics (MIPRO), 22-26 May 2017, Opatija, Croatia

[7] Ayas, T., & Horzum, M. B. (2011). Exploring The Teachers' Cyber Bullying Perception In Terms Of Various Variables. International Online Journal of Educational Sciences, 3(2).

[8] S. Karabatak, A. Namlı, M. Karabatak, "Perceptions of High School Students Regarding Cyberbullying and Precautions on Coping With Cyberbullying", 2018 6th International Symposium On Digital Forensic And Security (ISDFS), 22-25 March 2018, Antalya, Turkey.

[9] Arıcak, O. T. "Siber Zorbalık: Gençlerimizi Bekleyen Yeni Tehlike", Kariyer Penceresi, 2(6) (2011), 10-12

[10] Erdur-Baker, Ö. and Kavşut, F. "Akran Zorbalığının Yeni Yüzü: Siber Zorbalık", Eurasian Journal of Educational Research (EJER), 27(2007), pp, 31-42.

[11] Aricak, T., Siyahhan, S., Uzunhasanoglu, A., Saribeyoglu, S., Ciplak, S., Yilmaz, N., & Memmedov, C. Cyberbullying Among Turkish Adolescents. Cyberpsychology & Behavior, 11(3) (2008), 253-261.

[12] M. G. Hussain, T. Al Mahmud, W. Akthar, "An Approach To Detect Abusive Bangla Text", International Conference On Innovation İn Engineering And Technology (ICIET), 27-29 December 2018.

[13] Al-Mamun, S. Akhter, "Social Media Bullying Detection Using Machine Learning On Bangla Text", 10th International Conference On Electrical And Computer Engineering, 20-22 December 2018, Dhaka, Bangladesh

[14] Turkishstemmer. Https://Github.Com/Otuncelli/Turkish-Stemmer-Python (accessed: Jun 13, 2022).

[15] NLTK: Https://Www.Nltk.Org/ (accessed: Jun 10, 2022).

[16] Ikonomakis, M., Kotsiantis, S., & Tampakas, V. (2005). Text Classification Using Machine Learning Techniques. WSEAS Transactions On Computers, 4(8), 966-974.

[17] Wilkinson, A. W. Literature Review on Advance Directives. U.S. Department of Health and Human Services. Washington: RAND Corporation, 2007.

[18] Abuaid, A. M., & Mishra, A. (2010). A Rule-Based Approach to Embedding Techniques for Text Document Classification. Applied Science, 10(11), 4009.

[19] Ross, T. J. (2005). Fuzzy Logic with Engineering Applications. West Sussex, United Kingdom: John Wiley & Sons.

[20] Rajput, A. A. (2011). J48 And JRIP Rules For E-Governance Data. International Journal of Computer Science and Security, 5(2), 201.

[21] Buddhinath, G. D. (2006). A Simple Enhancement To One Rule Classification. Melbourne, Australia: Department of Computer Science & Software Engineering University Of Melbourne.

[22] Sayad, S. (2022). Zeror, Saedsayad: Https://Www.Saedsayad.Com/Zeror.Htm (accessed: Jun 10, 2022).

Advances in Artificial Intelligence Research (AAIR)
ISSN 2757-7422
www.dergipark.com/aair/

Vol. 3 (No. 2), pp. 54-65, 2023
doi: 10.54569/aair.1203155
Published online: Oct 23, 2023

Data Center Control Application with Fuzzy Logic

Hasan Yılmaz [1, *], Adem Alpaslan Altun [2], Mehmet Bilen [3]

[1] Konya Selçuk University, Beyşehir Ali Akkanat Faculty of Management, Konya, Turkey
[2] Konya Selcuk University, Faculty of Technology, Department of Computer Engineering, Konya, Turkey
[3] Burdur Mehmet Akif Ersoy University, Golhisar School of Applied Science, Turkey

Abstract

Data centers are systems that host devices utilizing recording and communication technologies, which are expected to operate securely and accurately. Consequently, transforming data centers into smart environments for control purposes has become a significant area of focus. In this study, we monitor the cabinet environment within data centers and ensure that the control system reaches the predetermined optimal state values in the event of undesirable situations. Threshold control was implemented for humidity and flame data, while fuzzy logic theory was applied to temperature data. Fuzzy clusters can be adjusted according to the data center's location at the user's request. This approach allows users to input desired optimal and threshold values into the system, which are then evaluated based on the situation. The designed system ensures data center security with minimal personnel involvement. Additionally, all problematic events are recorded in the system, enabling them to be viewed on a webpage and communicated to designated personnel via email. In the conducted study, the fuzzy-controlled temperature value outputs are reported as heating (40%), cooling (53%), and instances where the system does not perform heating or cooling.

Keywords: *Fuzzy Logic; Embedded Systems; Environment Monitoring; Data Center.*

1. Introduction

Today, the use of computers and the internet is among the indispensables of people. As people's computer and internet usage rates increase, the development accelerates even more. The development of technology has shown its effect in many areas and the machine age has left its place to the age of informatics and artificial intelligence. Hardware, which we define as physical devices, has not remained unfamiliar to this age. As it is seen today, it is not only people who use the element of communication [1]. While the increase in the use of computers and the development of the internet of things brought some needs and wishes of people in their daily lives, the development of computer systems and technology brought these needs together with smart systems and environments.

In order to design smart systems, it is necessary to examine the environmental characteristics. Environments have their own unique standards. Various systems have been implemented to maintain and control these standards. Monitoring the environment data controlled by the sensing devices is called environment monitoring, which helps to monitor the requested data with a software and to run the necessary control and process status on the system [2].

Today, with the developing technology, there is a great increase in the amount of data collected and the issue of data security gains more importance. Since the time we met computers, computer and system rooms have always been seen as a part of the system. Monitoring in data centers; It is a service for security and to detect bad scenarios that may occur beforehand. Apart from monitoring with the help of cameras, it should be monitored that physical conditions may also be included in the system [3].

Microchips or microcontrollers such as Arduino, Raspberry Pi, PIC are widely used for environment monitoring and control in smart environments. In this study, it is aimed to carry out the necessary control process by using Raspberry Pi microcontroller and fuzzy logic structure in order to keep the system at the level of a dark data center without the need for personnel by monitoring the events and situations that may occur in the data centers.

In the second part of the study, similar studies in the literature are presented. The third, fourth and fifth sections constitute the material and method sections. Information about data centers and their features are given in the third section, fuzzy logic method and control in the fourth section, and the software and hardware used in the study in the fifth section. In the sixth chapter, the research findings are given. In the seventh chapter, the results of the study are evaluated and suggestions for future studies are made.

2. Related Works

In the literature, there are many studies on fuzzy logic and environment control. In fact, most of these

*Corresponding author
E-mail address: yilmazhasan@selcuk.edu.tr

Received: 12/Nov/2022; Received in revised form: 27/Mar/2023; Accepted: 22/Apr/2023.

studies are about the management of systems. It has been observed that the fuzzy logic theory is used in different areas, especially in industrial applications. The general lines of our study can be considered as a synthesis of the source research examined.

Most of the studies [4-6] have handled the control of smart environments using Arduino. From similar studies using Raspberry Pi; [7] While making the necessary notifications via WhatsApp messages by reading the temperature and humidity values of the server rooms, [8] the people defined in the system could be informed via e-mail when predetermined alarm situations occurred by monitoring the environment. [8] indicates that the system is not controlled and the optimum situations will be regained only with the help of personnel. It is thought that it will be superior to other studies due to the selection of Raspberry Pi, an advanced microcontroller, in the study.

Studies with fuzzy logic control as a method [5, 9, 10] generally use MATLAB simulation. The use of Python and C# programming languages in system simulation and programming shows that it has a different structure from other products. Since the control factor of temperature and humidity is taken into consideration in the selection of the researched resources, it is seen that control is realized in many industrial areas.

Contrary to the solutions used today, a more economical solution has been presented with the study. As a result of the instant evaluation of the environmental conditions, the monitoring of the environment was carried out. In this way, it is seen that the personnel entrances to the data center will be less, that is, it will become a dark data center. In addition, the reduction of inputs to the environment will indirectly reduce energy consumption to a minimum. The situations and needs that will arise with the designed system were determined in advance, and it was ensured that the personnel took necessary precautions and were informed. It is also obvious that the devices, which have a high cost to install, will be protected and will not harm the institution or organization financially.

3. Data Center

Data center; It is an environment in which data processing devices such as servers, telecommunications and network devices, data storage devices such as data warehouse systems [11, 12], as well as power and air conditioning devices, fire protection and ambient lighting systems are located [13]. Data centers, also called server or system room [14], computer room or data farm [11], show their presence in every field from technology to education, from government institutions to social media [11]. In addition, the systems used in the data center vary according to the processing, transmission and storage of the data [14].

Considering that the data center installation costs are quite high, it is seen that not every business has a data center of its own and data centers are configured in modules [15].

3.1. Data Center Standards

When data centers are examined, design elements such as electrical, mechanical, security, features of the building and rooms are configured differently for each data center [11]. The needs of the design environment realized in the data center have created standards in different ways with the influence of communication design, setting up the environment and quality management [13].

TIA Standart 942

TIA-942, which is determined as the telecommunication infrastructure standard; has a structure that includes the establishment, design and management of the data center [13]. One of the reasons for the existence of data centers is redundancy [3]. Thanks to redundancy, data centers provide greater reliability. Businesses have different levels of available layers for the reliability and redundancy of their data centers [11].

The four-tier data center classification (Tiers), developed by the Uptime Institute in 1993, is a globally recognized and accepted standard in the areas of redundancy, reliability, overall performance and fault tolerance [11, 13, 16]. Tier levels follow each other in the I-IV range. Although the levels are not superior to each other, each level provides the implementation of different types of operations [17].

Tier standards are performance-based, flexible and have a certification that can realize the life cycle in stages. With Uptime certificates, businesses offer an infrastructure service that reduces cost and risk while making them more efficient [16].

Tier standards are given in three different ways as “Design”, “Facility” and “Operations” (**Figure 1**). Design; It is the certificate given to the design documents in which the mechanical and electrical systems, the facility and its architecture are evaluated. Facility; examines that the facility has been established in accordance with the standards. Operations; It examines and reports the operational sustainability and the compliance of the transactions to be performed in the data center with the standards [18].

Figure 1. *Tier standard and certificate examples [19]*

ASHRAE Thermal Guidelines

The American Society of Heating, Refrigeration and Air Conditioning Engineers (ASHRAE), whose foundations date back to 1894, publishes a series of standards and guides on HVAC&R (heating, ventilation, air conditioning and refrigeration) systems and their problems. These contents are periodically updated and republished [20]. ASHRAE determined the necessary values for data centers to work with the first version of the guides published in 2004 and the second version in 2008 for data processing environments [21]. With the third edition published in 2011, new classes were added that expanded the temperature range, while the fourth edition, the 2015 guide, expanded the relative humidity values in addition to those published in the 2011 guide. As seen in **Table 1**, temperature and humidity limit values were determined for data centers recommended in the guidelines published in 2004, 2008, 2011 and 2015 [22].

Table 1. *Comparison of environment limits in ASHRAE 2004, 2008, 2011 and 2015 guidelines*

	ASHRAE	2004	2008	2011	2015
Temperature	Lower Limit	20°C (68°F)	18°C (64.4°F)	15°C (59°F)	5°C (41°F)
	Upper Limit	25°C (77°F)	27°C (80.6°F)	32°C (89.6°F)	45°C (113°F)
Humudity	Lower Limit	40% RH	5.5°C DP (41.9°F)	20% RH	8% RH
	Upper Limit	55% RH	60% RH & 15°C DP (59°F)	80% RH	80% RH

3.2. Air Conditioning

Air conditioning is the whole process of adjusting the temperature of the ambient air and reducing the humidity value in the environment.

The fact that the system has more devices affects the temperature and humidity values of the environment and will cause problems in the operation and performance of the system. The stable operation of the devices in the data centers is achieved by the air conditioning of the environment [23].

For the air conditioning of the environment, the limit values recommended in the guidelines published by ASHRAE and specified in Table 1 are used.

3.3. Security

Security is one of the reasons data centers exist. Environment monitoring, fire protection and control of the input and output units are carried out within the security systems [12].

Fire is a big problem that environments should pay attention to. Any fire risk that may occur on data centers threatens the health of not only people but also their devices. Thanks to the detection systems, the pre-detection of the fire informs the formation of the gases that threaten the environment early [24].

4. Fuzzy Logic and Control

4.1. Fuzzy Logic

The solution of many problems that we encounter during the day can be done within the context of previously experienced situations and information. While qualifying as "right" or "wrong" for the solution of some problems, our answer to others may be "partly true" or "partly false". Fuzzy logic, which is a mathematical order, is used to shape an unclear situation [25]. The fuzzy logic algorithm, which is produced as an alternative to traditional and rule-based approaches, can achieve successful results with partial values in cases where the parameters of the problem are not clear. One of the most frequently given examples when trying to understand the basics of fuzzy logic is the weather. There are no two states that the weather is good or bad. There can be many intermediate situations between good and bad. One of the biggest reasons for choosing fuzzy logic in this study is that these intermediate values can be handled with fuzzy logic. The conditions inside the data centers can not only be described as good and bad, but it is aimed that the control center can take different precautionary positions autonomously according to many intermediate values between these two.

4.2. Design and Implementation of Fuzzy System

In systems using fuzzy logic, fuzzy inference process is examined in three main steps. These are blurring, fuzzy inference and defuzzification as seen in **Figure 2**.

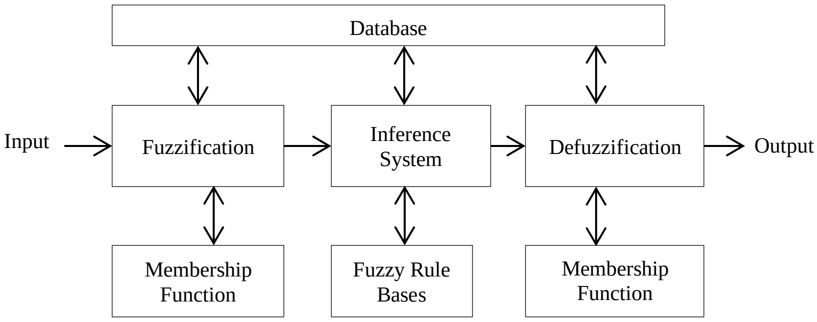


Figure 2. *Fuzzy logic system architecture*

Fuzzy subsets are created by matching the real data entered in the blur unit in the system architecture with fuzzy variables [26-28]. The subsets created by the membership function show a membership degree in the range of 0 – 1 [5, 29]. After fuzzification, fuzzy inference system is performed. In this unit, a decision is made about which rule is related to values and which one will be selected from the set of rules formed during fuzzification [30, 31]. The final stage of architecture is defuzzification. In the defuzzification system stage, the fuzzy inference system outputs are converted to a real value [26, 32].

5. Hardware Infrastructure

Data center control application; Raspberry pi is designed by developing temperature-humidity and flame sensor, peltier, fan and web interface. While the Raspberry Pi hardware was written with the Python program, the development of the web interface was provided with ASP.NET MVC. It shares a common database prepared with web interface and Raspberry pi MSSQL. Operations on the database are provided by web services using HTTPOST and HTTPGET structures.

6. Experimental Setup

Thanks to the sensors in the cabinets in the data center, the data was received and sent to the database with the Raspberry pi. In order to control the temperature of the center, rules were created by writing a fuzzy logic mechanism. Peltiers used for heating and cooling operate in line with these rules. If the humidity information in the system exceeds the threshold value, the fans operate and ventilate. In addition, possible fire situations are recorded with the flame sensor in the environment. Information is given to the determined personnel through the e-mail service included in the web interface of the system, which is seen and explained in **Figure 3**, in certain situations. While designing the interface, the interface library/framework created in the Koçak [4] study was used. In addition, fuzzy equations and calculations suitable for the data center control problem were performed on the server side using C# programming language.

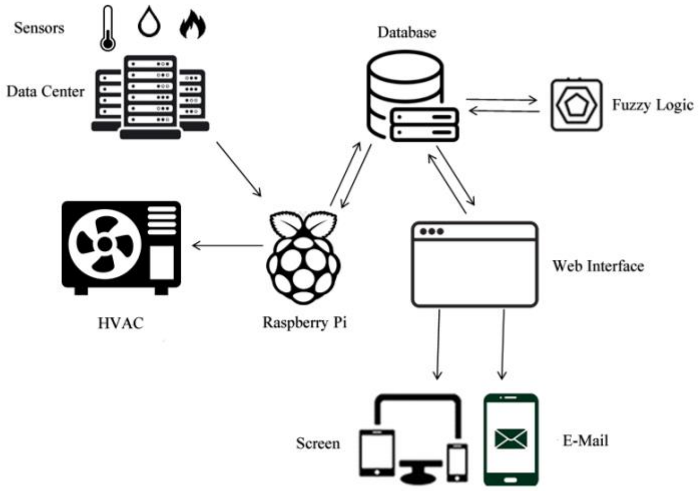


Figure 3. *Working principle of the system*

6.1. Fuzzy Set

"Hot, cold, little, much, extremely, a little, etc." that people use to explain certain situations and events in their living spaces. Expressions such as have a fuzzy state [4, 33]. Thus, the fuzzy expressions allow us to create fuzzy sets.

The lowest temperature measured between 1927-2021 in Turkey was -24.9°C, and the highest temperature was 41°C [34]. On the world, the lowest temperature was recorded in Greenland with -69.6°C, and the highest temperature was recorded in Jordan with 58°C [35]. The lowest and highest temperature ranges given in Table 2 have been determined based on the temperatures felt throughout Turkey and the world.

As seen in **Table 2**, fuzzy set expressions do not have a definite inference. For example, 16°C; It is in both the warm and slightly cold range. The rules created by the fuzzification process are evaluated and as a result of the inferences, the warm or slightly cold ranges are determined proportionally. If 30% is a little cold and 70% warm, different transactions can be created as a result of the transactions and membership values. **Table 2** emerges when the temperature values are considered to be matched with fuzzy expressions.

Temperature ranges are specified in the widest range and can also be evaluated regionally. The purpose of creating this chart is to determine the current situation by blurring the temperature ranges.

Table 2. *Fuzzy set representation of temperature ranges*

Fuzzy Set	Temperature Ranges
Extremely hot	50-70°C
So hot	35-55°C
Hot	28-45°C
A Little Hot	23-30°C
Warm	15-25°C
A little cold	10-18°C
Cold	8-13°C
So cold	0-10°C
Extremely Cold	(-70)-7°C

6.2. Membership Functions

Triangle membership function is used while creating membership functions in the interface. There are three variables in the triangle membership function. These; indicated as the start (a), the highest (b), and the end (c). μ(x) represents the calculated membership value for each element. Triangular membership degree was calculated as seen in Eq. (1).

$$\mu(x;a,b,c) = \begin{cases} x \gg c\,, x \ll a \rightarrow\ 0 \\ a < x \ll b \rightarrow \frac{x-a}{b-a} \\ b < x < c \rightarrow \frac{c-x}{c-b} \end{cases} \tag{1}$$

6.3. Fuzzy Rules

In the designed data center control system, there is no need for any changes on the source code with changes such as adding and removing through the interface. Rules are generated using the multiple-input, single-output (MISO) method. This method is illustrated by Eq. (2).

$$\text{If } X_1 = A \text{ and } X_2 = B \rightarrow Y = C \tag{2}$$

The X values in Equation 2 represent the inputs, the A, B and C values represent the states, and Y represents the output. In order to reach the Y output, A and B values must be calculated first. For each entry, it is necessary to find out how close to the statuses to be calculated, to be a member. As an exemplary rule, if the cabinet rear temperature is “Hot” and the cabinet front temperature is “Warm”, let the output be “Cool”. For the warm and warm expressions in this example, the membership degree of the cabinet rear and cabinet front temperature values is calculated with Equation 1. With the expression in Eq. (3), it is calculated which rule gives an output with the help of cabinet rear and cabinet front temperature membership values obtained from Eq. (2). The fuzzy set membership degree calculated with the help of this rule is truncated with the MIN-MAX method and the output value is finalized with the weight average method.

$$Y_i = MIN\left[\mu_{x_1}(x_1),\, \mu_{x_2}(x_2)\right] \tag{3}$$

The Y value given in Eq. (3) shows the output value *i* by calculating separately for each rule. The x values represent the inputs, while the μ(x) represent the mean. **Figure 4** shows an example graphic representation.

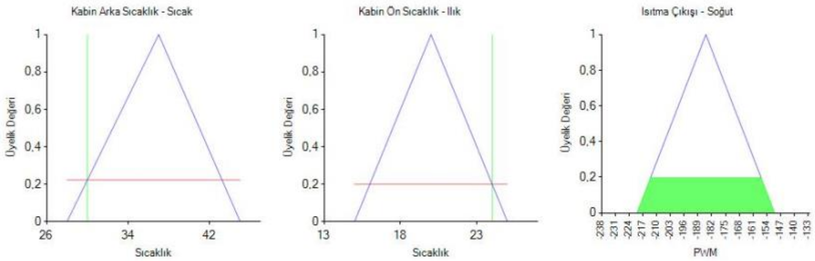


Figure 4. *An example fuzzy output graph (X1=30, X2=24)*

For the output seen in **Figure 5**, the calculation was made using Eq. (4). Each rule formed like the expression exemplified in **Figure 4** is included in the calculation. In Eq. (4), Y represents the output value, X the input value, and $\mu_n(x)$ represents the membership value calculated for each n value. As can be understood by Eq. (4), an output is produced as a result of these calculations by using the weighted average method.

$$Y = \frac{\sum_n \mu_n(x) * x}{\sum_n \mu_n(x)} \tag{4}$$

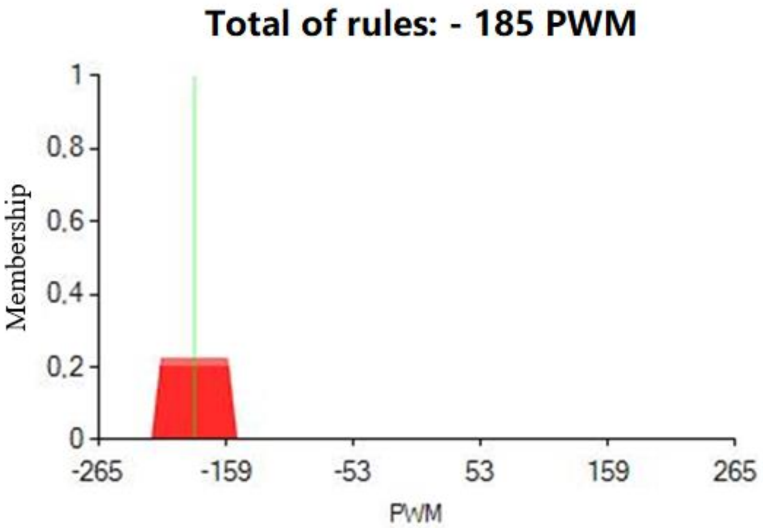


Figure 5. *An example rule sum post output graph*

6.4. Fuzzy Rules Table

Table 2 fuzzy set was used in the formation of fuzzy rules. A fuzzy rule table was created by defining 81 rules in total by matching the cabinet rear (9) and cabinet front (9) temperatures with fuzzy set.

6.5. Code Measurements

Software applications have tools that can prevent complexity with a secure and sustainable structure. With these tools, the developers monitor the problems that may occur and the development of the software being worked on [36].

Visual Studio 2019 application was used in the development of the web interface. With the Code Metrics tool included in the application, the maintenance index, cyclic complexity, inheritance depth, class link, source code lines and executable code lines are calculated. Calculated situations are shown in **Table 3**.

Table 3. *Code measurement results*

Maintainability Index	Cyclomatic Complexity	Depth of Inheritance	Class Coupling	Lines of Source code	Lines of Executable code
90	280	3	75	1284	402

Maintainability index; It represents the ease of code maintenance. A value in the range of 0 - 100 is calculated. If the calculated value is 20 and above, it is stated that there is a better ease of maintenance with the green color [37]. The fact that the maintenance index shown in Table 4 shows the value of 90 shows that it has a high ease of maintenance.

Cyclomatic complexity; Specifies the number of flow control mechanisms in the program. A range of 1-10 indicates a low risk, 11-20 indicates medium, 21-50 indicates high, and above 50 indicates a very high risk [38]. Although it is seen in Table 4 that the cyclomatic complexity is 280, this number represents the sum of all forms and pages. When the pages are evaluated individually, only one of the 307 values is at medium risk, while all other values seem to be at low risk.

Depth of inheritance; measures the object inheritance hierarchy. A low depth number indicates less complexity [39]. As seen in Table 4, it has a less complex structure.

Class coupling; shows how many classes are used in a class. A high value indicates that the classes interact more with each other [40]. Table 4 shows that the class coupling is 75. This number shows the total used class items, and it has been observed that it does not exceed the critical value when viewed individually.

Lines of source code specify the integer numbers of lines in the source code, while lines of executable code give the approximate number of executable processes or lines.

6.6. CPU Usage

Events, memory usage and CPU usage measurements are made with the diagnostic tool of Visual Studio 2019 application. Thanks to this tool, the performance of the designed interface can be measured. When the interface performances are examined, the highest category is IO (Input/Output) with 44%, followed by JIT (Just In Time) compiler with 32%. Data on CPU usage is shown in **Figure 6**.

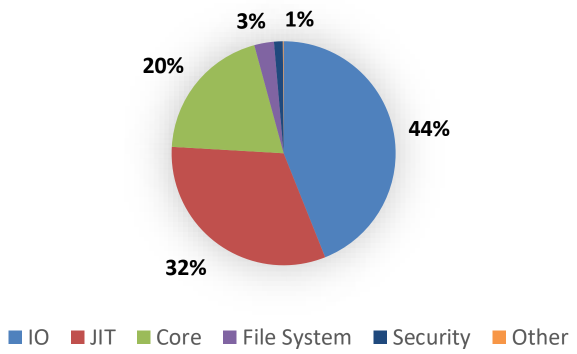


Figure 6. *CPU usage analysis top five categories*

6.7. Scenarios

Two different scenarios were considered in order to observe the working order of the designed system. With the scenarios written using VS.NET C#, the outputs of the system to be given by the method have been observed.

Scenario 1: The cabinet front temperature value, which represents the ambient temperature, is gradually increased linearly.

Output value of ambient temperature data given by fuzzy logic method is shown with PWM signals. The output values against the temperatures given for the scenario are shown with the PWM-temperature graph seen in **Figure 7**. Output signals give an output in the range of (-255) – 255. If the given output value has negative values, the cooling process must take place. Values where the system does not make heating or cooling and the ambient temperature is between 18-23 show the range in which the fuzzy method does not output.

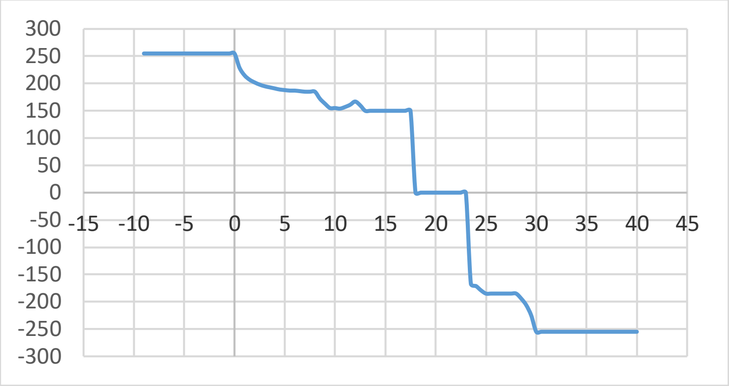


Figure 7. *PWM-temperature graph*

Scenario 2: In the case that the front temperature of the cabinet, which is the ambient temperature, increases linearly, the value of the cabinet rear temperature is increased in a parabolic way.

When the data of Konya province belonging to the General Directorate of Meteorology are examined, the range of (-10) - 40 is used for the web interface, based on the lowest and highest average values [41]. According to the scenario, the temperature was increased by 0.5 increase to the cabinet front temperature value. For the cabinet rear temperature, the increase amount is added with a decreasing value. The 1.0 increase value given for the cabinet rear temperature is determined by 0.0075 value decrease in each reading.

According to the scenario, the PWM output produced by the system was observed after the fuzzy logic process of the cabinet rear and cabinet front temperatures, which were examined in the simulation. Output values produced for cabinet rear and front temperatures are shown in **Figure 8**. When the observation data are examined, it is seen that no PWM output signal is produced at the temperature of 12 in the cabinet front and 26,905 in the back of the cabinet. Signal values are given as output for cooling and heating processes, albeit slightly at other temperature values.

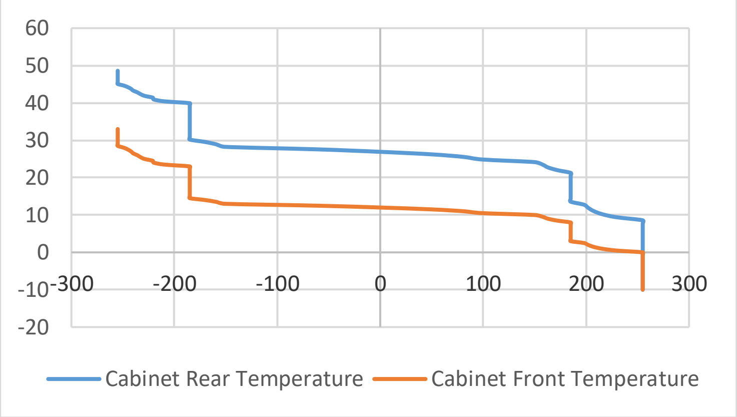


Figure 8. *PWM outputs produced for cabinet rear and front temperatures*

Scenario 3: The front and rear cabin temperature values were given to the system and a comparison was made.

The temperature values specified in Table 2 are given for the front and rear cabin temperature values. It has been determined that a total of 225 PWM signals are output by matching the cabin rear (15) and cabin front (15) temperatures within the specified (-70)-70℃ temperature range. These output signals are divided into heating (91), cooling (119), and HVAC system not operating (15). The temperature comparisons for heating (dark color) and cooling (light color) processes are shown in **Figure 9**. In addition, the sizes of the bubbles seen in **Figure 9 (a)** are specified according to the PWM signal values produced for heating and cooling. The representation of the generated signal values is shown in **Figure 9 (b)**.

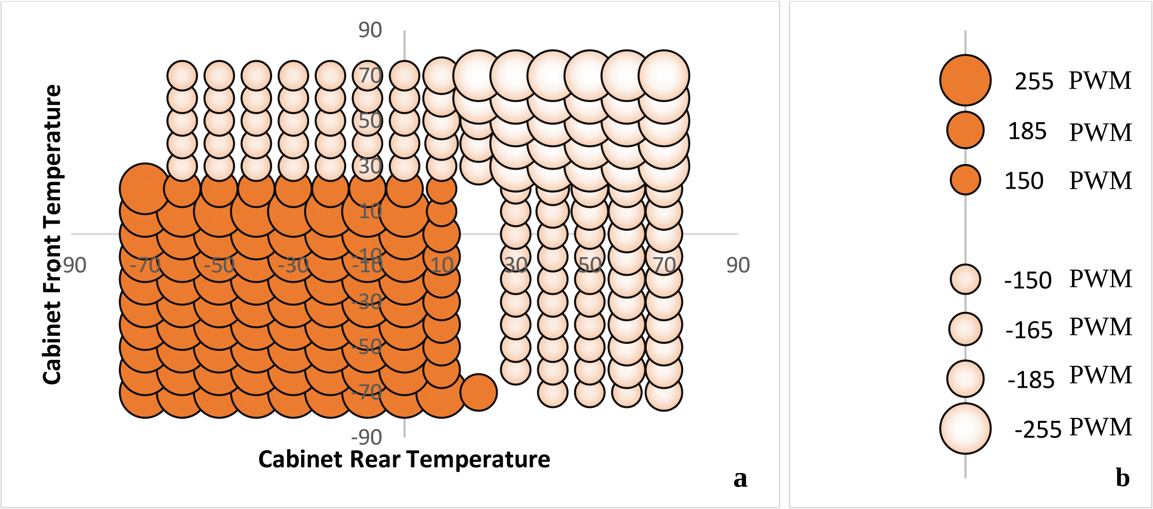


Figure 9. a) *PWM signal distributions according to temperature values* **b)** *The sizes of bubbles*

In **Figure 9 (a)**, the bubbles belonging to the cabin back temperature (20) and the cabin front temperature (40) value generate the -185 PWM signal as the system output value. With this value, it is seen that the cooling process is performed. As another example, 185 PWM signal is generated as the bubble output value of the cabin back temperature (10) and cabin front temperature (0) values. In this case, the heating process of the environment was performed.

Scenario 4 (Non-Fuzzy): Ambient humidity is linearly increased and decreased.

The output value given by the operation of the fan in the system is shown with PWM signals. The output values against the humidity values given as per the scenario are shown with the PWM-humidity graph in **Figure 10**. Output signals give 0 and 250 values. The threshold value was previously set as 50 by the user in the web interface. As can be seen in **Figure 10**, as a result of exceeding the threshold value of the humidity data of the environment, the PWM signal gives a value of 250 and enables the fan to operate. When the ambient humidity value falls below the threshold value, the PWM signal gives 0 value and the fan is stopped.

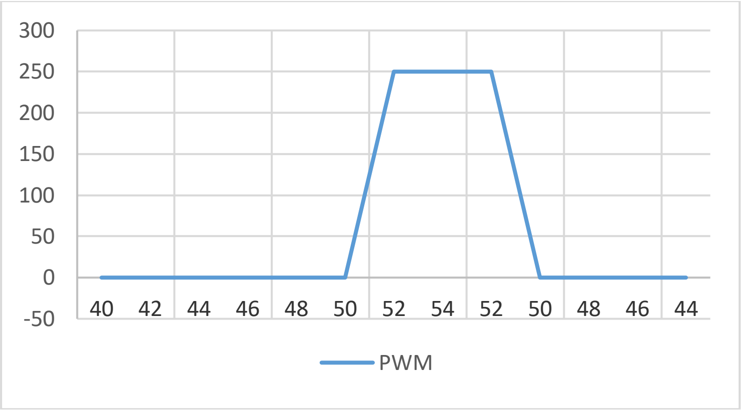


Figure 10. *PWM-humidity graph*

Scenario 5 (Non-Fuzzy): Any flame condition that may occur in the environment was observed with sensors.

The data received with the flame sensor in the environment is included in the system without the fuzzy logic method. By monitoring the system continuously, "There is a flame" or "No flame" warnings can be displayed on the interface in case of any flame that may occur. The outputs that may occur when a fire-like object is included in the system as per the scenario are shown in **Table 4**. As seen in **Table 4**, when the flame object is included in the system, the "Fire Detected" warning and the start and end times of the warning are also displayed. In this way, it can be reported how long the system has been exposed to fire.

Table 4. *Fire situation report*

Flame Detection	Start	Stop
Fire Detected	22.08.2022 14.17	22.08.2022 14.25

Fire Detected	13.08.2022 13.23	13.08.2022 13.31
Fire Detected	03.08.2022 12.34	03.08.2022 12.40
Fire Detected	16.07.2022 22.52	16.07.2022 22.57
Fire Detected	16.07.2022 11.08	16.07.2022 11.13

7. Discussion

There are many studies that perform data center control from industrial environments where control is performed. When the materials and methods used in the studies are examined, there are differences in the selection of the microcontroller, the programming language used and the method.

As a result of the literature research, it has been seen that the control controllers encountered in the data center are generally Arduino [5, 6, 42] and Raspberry Pi [6-8, 43]. Mostly Python [6-8] and MATLAB [5, 9] were chosen as programming languages in the system. The fuzzy logic chosen as the method has been applied less in the literature [5, 42, 44, 45] than the classical method in data centers.

Some situations encountered in the literature review and system adequacy and deficiencies were compared. In the designed control system, materials used more in data centers were selected in the literature review. It can be considered as an original work since there is no other study of the system using more modern programming languages and materials and fuzzy method selection.

Thanks to the fuzzy logic chosen as the control method, instead of heating or cooling the environment at an optimum temperature, how much it should be heated or cooled is automatically calculated according to the written fuzzy rules and variable temperature conditions. The fuzzy logic selection of the system instead of the classical logic creates a more qualified control in order to keep the environment at the optimum temperature.

Worldwide temperature values are variable. While 40 degrees is considered hot according to some geographical regions, according to some regions, this value is within the seasonal normals. For this reason, the optimum levels of data center installation needs may vary according to the environment. System values can be redesigned by examining the geographical conditions of the data center with the designed interface. Reshaping the place where the center is located according to the seasonal normals also ensures that the system is less damaged and uses energy.

8. Conclusion

In this study, necessary structures have been created for autonomous control of the data center with fuzzy logic. The Raspberry Pi used in the study is coded with Python. Temperature, humidity and flame status values were captured in ten-second intervals by sensors in the data center. One flame, one humidity and two temperature sensors are used in the system. Thanks to the fuzzy logic set, the captured temperature value is interpreted in the web interface written in C# and the possible states of the system are calculated. After the calculated situation, he gave command to the HVAC system and focused on keeping the system at the optimum level by heating and cooling. In case the humidity value is exceeded by the threshold value entered by the user from the web interface, it is desired to reach the optimum level by sending the PWM signal of the fan included in the system. When the system is observed at runtime, it is seen that the special case control, such as the fire situation, which was previously performed manually, is automatically managed by the developed application. Values taken from all sensors are kept on MSSQL database.

Data center temperature and humidity values are desired to be kept at optimum levels. Values were taken from temperature, humidity and flame sensors in the system. Fire detection for flame value, threshold value comparison for humidity value and fuzzy logic method for temperature values were used. The web interface designed in the study has a plain appearance away from complexity with ASP.NET MVC. The user gives an output corresponding to the values taken from the sensors with both the manual method and the fuzzy logic method. The synchronization of the sensor data received from the system and the determined short intervals followed the system with minimum delay. Any problems that occur in the system are recorded in the database and instant notifications are made to the users. In order to send the notifications to be made during the operation of the system, the administrator mail and the mailing list to be sent are saved from the web interface. Thus, possible situations such as temperature increase and decrease, and exceeding the humidity threshold value were informed to the people who should be informed by e-mail communication.

In line with the scenarios realized in the system; the fire situation was informed and the operation of the system was tested. When the humidity threshold value is exceeded, the fan is operated until the specified normal conditions are met. It has been observed that the fuzzy logic system responds to the values received from the temperature sensors, such as heating or cooling, according to the incoming values.

In this study, it is foreseen that the system will have a longer life due to the automatic management of the control mechanism and the control of it without causing any damage.

In studies to be conducted on data center management, it is recommended to consider the average temperature and humidity values throughout the year according to the geographical location of the data center.

In order to achieve more successful results in future studies, the system is planned to be built in larger data centers with more cabinets and devices. In Addition, incorporating machine learning techniques alongside fuzzy logic could potentially improve the system's adaptability to changing environmental conditions and enhance its overall performance. By learning from historical data and adjusting its parameters, the system could become more efficient in detecting and addressing undesirable situations in the data center.

Acknowledgements

We would like to thank Selçuk University Scientific Research Projects Coordinatorship for supporting this study with project number 21201044.

It was presented at the ICAIAME 2022 conference and published as a summary.

References

[1] Özdemir B. "Akıllı Ev Sistemlerinde Güvenlik Zafiyetleri ve Önlemleri", Yüksek Lisans Tezi, İstanbul Şehir Üniversitesi Fen Bilimleri Enstitüsü, İstanbul, 2019.

[2] Erışık Y. Ortam İzleme Kontrol Sistemleri [Online]. Available: https://www.cozumpark.com/ortam-zleme-kontrol-sistemleri (accessed: January 4, 2021).

[3] Gökmen HT, Küçüksille EU, "Veri Merkezi Tasarımı", In: XV. Akademik Bilişim Konferansı Bildirileri Akdeniz Üniversitesi, Antalya, (2013) 93-97.

[4] Koçak Ç. "Bulanık Mantık ile Arı Kovanlarının Uzaktan Takip ve Kontrol Sistemi", Yüksek Lisans Tezi, Mehmet Akif Ersoy Üniversitesi Fen Bilimleri Enstitüsü, Burdur, 2018.

[5] Purwanto FH, Utami E, Pramono E. "Design of server room temperature and humidity control system using fuzzy logic based on microcontroller", In: 2018 International Conference on Information and Communications Technology ICOIACT, (2018) 390-395.

[6] Baz FÇ, Uludağ K. "Veri Merkezi Güvenliğinin Sağlanmasında IoT Sensörlerinin Kullanımı Üzerine Bir Uygulama", Avrupa Bilim ve Teknoloji Dergisi 27 (2021) 392-397.

[7] Kurniawan DE, Iqbal M, Friadi J, Borman RI, Rinaldi R. "Smart monitoring temperature and humidity of the room server using raspberry pi and whatsapp notifications", In: Journal of Physics: Conference Series, (2019) doi:10.1088/1742-6596/1351/1/012006

[8] Balcı M. "Raspberry Pi Mini Bilgisayarı ile Ortam İzleme Sistemi Geliştirilmesi", Yüksek Lisans Tezi, Şeyh Edebali Üniversitesi Fen Bilimleri Enstitüsü, Bilecik, 2019.

[9] Abbas M, Khan MS, Zafar F. "Autonomous room air cooler using fuzzy logic control system", International Journal of Scientific & Engineering Research 2(5) (2011) 1-8.

[10] Das TK, Das Y. "Design of a room temperature and humidity controller using fuzzy logic", American Journal of Engineering Research 2(11) (2013) 86-97.

[11] Geng H. "Data Centers - Strategic Planning, Design, Construction, And Operations", In: Geng H. (Ed.), Data center handbook, John Wiley & Sons, New Jersey, USA, (2015) 3-14.

[12] Günel A. "Üniversitelere Yönelik Yeni Bir Veri Merkezi Tasarımı ve Uygulaması", Yüksek Lisans Tezi, Şeyh Edebali Üniversitesi Fen Bilimleri Enstitüsü, Bilecik, 2014.

[13] Dai J, Ohadi MM, Das D, Pecht MG. "Optimum cooling of data centers", Springer, New York, USA, 2014.

[14] Wikipedia, Data center [Online]. Available: https://en.wikipedia.org/wiki/Data_center (accessed: March 30, 2021).

[15] Cloudwolks, Understanding different types of data centers [Online]. Available: https://www.cloudwalks.com/blog/understanding-different-types-of-data-centers (accessed: March 30, 2021).

[16] Uptime, Data Center Certification [Online]. Available: https://uptimeinstitute.com/tier-certification (accessed: March 30, 2021).

[17] Uptime, Tier Classification System [Online]. Available: https://uptimeinstitute.com/tiers (accessed: March 30, 2021).

[18] Stansberry M. Explaining the Uptime Institute's Tier Classification System [Online]. Available: https://journal.uptimeinstitute.com/explaining-uptime-institutes-tier-classification-system/ (accessed: March 30, 2021).

[19] Uptime, Uptime Institute Issued Awards [Online]. Available: https://uptimeinstitute.com/uptime-institute-awards/list (accessed: March 30, 2021).

[20] Wikipedia, ASHRAE [Online]. Available: https://en.wikipedia.org/wiki/ASHRAE (accessed: March 30, 2021).

[21] ASHRAE TC 9.9, "2011 Thermal Guidelines for Data Processing Environments - Expanded Data Center Classes and Usage Guidance", USA, 2011.

[22] ASHRAE TC 9.9, "Data Center Power Equipment Thermal Guidelines and Best Practices", USA, 2016.

[23] Akın G, Ketenci S. Sistem Odaları İklimlendirme Sistemleri [Online]. Available: https://web.itu.edu.tr/akingok/doktora/iklimlendirme/Sistem_odasi_iklimlendirme_sistemleri.pdf (accessed: February 3, 2021).

[24] Donohue SS. "Fire Protection and Life Safety Design in Data Centers," In: Geng H. (Ed.), Data center handbook, John Wiley & Sons, New Jersey, USA, (2015) 229-243.

[25] Novák V, Perfilieva I, Mockor J. "Mathematical principles of fuzzy logic", Springer Science & Business Media,

LLC, USA, 1999.

[26] Klir GJ, Yuan B. "Fuzzy sets and fuzzy logic: theory and applications", Prentice Hall P T R, NJ, USA, 1995.

[27] Nhivekar G, Nirmale S, Mudholker R. "Implementation of fuzzy logic control algorithm in embedded microcomputers for dedicated application", International Journal of Engineering, Science and Technology 3(4) (2011) 276-283.

[28] Singhala P, Shah D, Patel B. "Temperature control using fuzzy logic", IJICS -International Journal of Instrumentation and Control Systems 4(1) (2014) 1-10; doi: 10.5121/ijics.2014.4101.

[29] Pricop E, Mihalache SF. "Fuzzy approach on modelling cyber attacks patterns on data transfer in industrial control systems," In: 2015 7th International Conference on Electronics, Computers and Artificial Intelligence, Bucharest, (2015) 1-6.

[30] Fahmi N, Huda S, Sudarsono A, Al Rasyid MUH. "Fuzzy logic for an implementation environment health monitoring system based on wireless sensor network", Journal of Telecommunication, Electronic and Computer Engineering, 9(2-4) (2017) 119-122.

[31] Kaur A, Kaur A. "Comparison of mamdani fuzzy model and neuro fuzzy model for air conditioning system", International Journal of Computer Science and Information Technologies 3(2) (2012) 3593-3596.

[32] Ross TJ. "Fuzzy Logic with Engineering Applications" (3nd Ed.), A John Wiley and Sons, Singapore, 2010.

[33] Altaş İH. "Bulanık Mantık: Bulanıklılık Kavramı", Enerji, Elektrik, Elektromekanik-3e 62 (1999) 80-85.

[34] Wikipedia. Fuzzy Logic [Online]. Available: https://en.wikipedia.org/wiki/Fuzzy_logic (accessed: March 30, 2021).

[35] Wikipedia. Set (mathematics) [Online]. Available: https://en.wikipedia.org/wiki/Set_(mathematics) (accessed: March 30, 2021).

[36] Jones M et al. Kod ölçümleri değerleri [Online]. Available: https://docs.microsoft.com/tr-tr/visualstudio/code-quality/code-metrics-values?view=vs-2022 (accessed: July 21, 2022).

[37] Jones M, Hogenson G. Kod ölçümleri - Bakım dizini aralığı ve anlamı [Online]. Available: https://docs.microsoft.com/tr-tr/visualstudio/code-quality/code-metrics-maintainability-index-range-and-meaning?view=vs-2019 (accessed: July 21, 2022).

[38] Karagedik Ö. Visual Studio 2008'de Kod Metrikleri ve Kod Analizi (Static and Dynamic Code Analyze) – Bölüm 1 [Online]. Available: http://univera-ng.blogspot.com/2010/03/visual-studio-2008de-kod-metrikleri-ve.html (accessed: July 21, 2022).

[39] Jones M, Hogenson G. Kod ölçümleri - Devralma derinliği (DIT) [Online]. Available: https://docs.microsoft.com/tr-tr/visualstudio/code-quality/code-metrics-depth-of-inheritance?view=vs-2022 (accessed: July 21, 2022).

[40] Bilen M. "Yapay Sinir Ağları İçin Web Tabanlı Bir Eğitim Yazılımı Geliştirilmesi", Yüksek Lisans Tezi, Süleyman Demirel Üniversitesi Fen Bilimleri Enstitüsü, Isparta, 2014.

[41] Wikipedia, Fan (machine) [Online]. Available: https://en.wikipedia.org/wiki/Fan_(machine) (accessed: April 10, 2022).

[42] Başcı MB. "Otonom Robotlar Aracılığıyla Veri Merkezlerindeki Kabinet İçi Sıcaklık Dağılımının Bulanık Mantık İle Kontrolü", Yüksek Lisans Tezi, Atatürk Üniversitesi Fen Bilimleri Enstitüsü, Erzurum, 2019.

[43] Utomo MAP, Aziz A, Winarno, Harjito B. "Server Room Temperature & Humidity Monitoring Based on Internet of Thing (IoT)", In: ICMETA 2018 - Journal of Physics: Conference Series 1306 (2019) 012030; doi:10.1088/1742-6596/1306/1/012030.

[44] Ko JS, Huh JH, Kim JC. "Improvement of energy efficiency and control performance of cooling system fan applied to Industry 4.0 data center", Electronics 8(5) (2019) 582.

[45] Tosun MF, Gençkal AA, Şenol R. "Modern Kontrol Yöntemleri ile Bulanık Mantık Temelli Oda Sıcaklık Kontrolü", Süleyman Demirel Üniversitesi Fen Bilimleri Enstitüsü Dergisi 23(3) (2019) 992-999.

Advances in Artificial Intelligence Research (AAIR)
ISSN 2757-7422
www.dergipark.com/aair/

Vol. 3 (No. 2), pp. 66-75, 2023
doi: 10.54569/aair.1184569
Published online: Oct 23, 2023

Development of a Traffic Speed Limit Sign Detection System Based on Yolov4 Network

Semih Selçuk [1,] Sefa Beker [1,*] Ömer Faruk Boyraz [1,]

[1] Anadolu Isuzu Automotive Industry and Trade Inc., R&D Center, Kocaeli, Turkey

Abstract

Recent advancements in artificial intelligence (AI) technologies have hastened the shift towards intelligent systems within the automotive industry. These systems enable the prevention of driver-related errors and accidents, as well as the provision of crucial information to drivers by proactively detecting road conditions. The present study focuses on the development of an AI-based system designed to furnish drivers with information regarding speed limit signs on the road, thereby enhancing traffic safety. The YOLOv4 model was employed in this system to achieve exceptional speed and accuracy levels. Following model training, rigorous validation was conducted, resulting in a remarkable test accuracy of 98%.

Keywords: *Speed Limit Sign, Object Detection, Deep Learning, YoloV4, Jetson Nano.*

1. Introduction

Studies on driver assistance systems and smart driving have been on the rise in recent times. Among these systems, the Intelligent Speed Assistance (ISA) system holds significance. According to the ISA regulation outlined in the General Vehicle Safety Regulation (EU) 2019/2144 [1], motor vehicles falling under categories M and N (such as buses and trucks) must be equipped with intelligent speed assistance systems, which will be mandated across all EU member states.

The importance of mechanical, electronic, and software advancements in enhancing driving safety cannot be overstated. Manufacturers are actively working to safeguard the lives and properties of drivers, passengers, and pedestrians alike [2]. In this regard, international authorities play a crucial role in fostering a shared understanding of security among manufacturers by establishing the general framework for these systems through relevant regulations [3].

Recent advancements in AI technologies have accelerated the integration of smart systems into transportation systems. These systems are aimed at preventing driver-related errors and accidents while providing advanced road condition detection to inform the driver proactively [4]. Modern vehicles now incorporate sophisticated driver assistance systems such as blind-spot detection, forward collision warning, driver fatigue detection, and intelligent speed assistance systems. Among these, the intelligent speed assistance system (ISA) stands out as a crucial component in providing speed-related information to the driver, ultimately ensuring a safer journey [5].

In the past, traffic sign detection relied on traditional computer vision algorithms. These algorithms typically utilized traffic sign features like color and shape and extracted various attributes from the signs. HSV (Hue, Saturation, Value) color space was often preferred over RGB due to its resilience to lighting variations. Machine learning techniques like template matching, support vector machines, or artificial neural networks were used to classify the signs based on the extracted features. However, these methods proved to be less efficient and slower in detecting traffic signs successfully [6].

Today, deep learning algorithms, specifically convolutional neural networks (CNNs), have revolutionized image recognition and object detection tasks. These deep learning methods enable quicker and more accurate detection by automatically extracting essential features from the target images without the need for extensive preprocessing [7].

In general, traffic speed sign recognition systems consist of two main components: sign detection and sign classification. Detecting the traffic speed signs accurately and swiftly, especially the smaller signs on the road, is a critical aspect of such systems. Numerous methods have been proposed in the literature to address this challenge, with SSD [8], Faster R-CNN [9], and Yolo algorithms [10] being among the most popular ones. The Yolo algorithm, in particular, stands out as it strikes a balance between speed and accuracy, making it highly suitable for this task.

Researchers have conducted extensive studies on traffic sign detection, often utilizing open source datasets due to the time-consuming nature of collecting traffic speed sign data. Some of the most well-known datasets used in these studies include the German Traffic Sign Dataset [11], Tsinghua Tencent Dataset [12], and the

*Corresponding author
E-mail address: sefabeker@gmail.com

Received: 6/Oct/2022; Received in revised form: 31/Jul/2023; Accepted: 19/Sep/2023.

Chinese Traffic Sign Datasets [13].

In their study, Rajendran et al. employed the YoloV3 model along with the German Traffic Sign Detection Benchmark Dataset. They conducted model training and evaluation using the Keras and Tensorflow frameworks, leveraging an Nvidia GTX 1060 GPU with 6 GB memory. The YoloV3 model achieved a speed of 10 frames per second (FPS) and demonstrated a satisfactory performance rate of 92.2% [14].

In contrast, Zuo et al. utilized the Faster R-CNN model for traffic sign detection in their study. The Faster R-CNN model training resulted in a mean average precision (mAP) value of 0.34493. Although the obtained results were not deemed satisfactory, the researchers emphasized the need for substantial efforts to enhance the model's performance [15].

This study presents the development of a cost-effective system designed to provide real-time speed limit information to drivers within vehicles. The system is specifically tailored to operate on the low-cost Jetson Nano embedded computer, rendering it feasible for integration into portable and commercial vehicles. In order to enhance the dataset utilized in the system's development, we supplemented open-source datasets with our own collected data. Notably, images containing speed limit signs were identified and extracted from the road images collected by Anadolu Isuzu test vehicles under nighttime conditions, thereby enriching the training dataset. Consequently, the system's nighttime performance has been substantially improved, as nocturnal images are often inadequately represented in open-source datasets.

The model training in this study involved multiple datasets, including the German Traffic Sign Dataset, Tsinghua Tencent Dataset (TT100K Dataset), Chinese Traffic Sign Dataset, and a dataset created by Google Maps and Anadolu Isuzu test vehicles. By leveraging the YoloV4 model, the researchers achieved real-time detection of traffic speed signs ranging from 20 km/h to 120 km/h with high performance.

The article is structured into five main sections. The introduction offers a comprehensive overview of the study. Section 2 encompasses an exploration of related studies in the literature. Subsequently, Section 3 outlines the dataset creation process, as well as the hardware and software methods adopted in the study. In the fourth section, the results of the software method are presented, which showcase the performance and accuracy of the developed system. Finally, the conclusion effectively summarizes the key findings and underscores the contributions made by the study.

2. Related Works

In recent years, significant progress has been made in traffic sign detection and intelligent driving systems, leveraging the advancements in artificial intelligence technologies. Numerous studies have explored different algorithms and approaches to improve the accuracy and real-time performance of traffic sign recognition systems. In this section, we present a review of relevant works in the literature, and compare the YoloV4 model used in our study with Faster R-CNN and SSD algorithms, which are widely recognized in the field of object detection for traffic sign recognition:

Rajendran et al. [14]: Rajendran et al. utilized the YoloV3 model in their study, employing the German Traffic Sign Detection Benchmark Dataset. They achieved a remarkable model speed of 10 FPS and an accuracy of 92.2%. While their results were promising, we sought to further enhance the performance of traffic sign detection with the improved YoloV4 model on a diverse dataset, which includes challenging night conditions.

Zuo et al. [15]: In their research, Zuo et al. adopted the Faster R-CNN model for traffic sign detection. However, their Faster R-CNN model yielded an mAP value of 0.34493, indicating suboptimal accuracy. Our comparative analysis confirmed the limitations of the Faster R-CNN model in real-time applications, particularly for traffic sign recognition.

In their research, Gao et al. [16] investigated the recognition of traffic signs, which involves two critical stages: detection and classification. For this study, the SSD (Single Shot MultiBox Detector) object detection algorithm was adopted to effectively identify traffic signs. The SSD model, a convolutional neural network, efficiently utilizes multiple feature maps for object detection. To address the challenges posed by the relatively small size of traffic signs compared to the entire image, the SSD model was further enhanced. This improvement encompassed model simplification and adjustments to the prior box sizes, resulting in superior detection performance, particularly for small targets. Extensive experiments were conducted on the test dataset, specifically the German Traffic Sign Dataset, highlighting the remarkable proficiency of the proposed algorithm in handling single-target, multi-target, and diverse lighting conditions. Notably, the precision and recall achieved on the test dataset were 91.09% and 88.06%, respectively.

Jiang et al. [17] proposed an enhanced traffic sign recognition method based on YOLOv3. They employed depthwise separable convolution to reduce parameters and computational complexity while maintaining accuracy. By replacing MSE loss with GIoU loss and incorporating Focal loss, they improved optimization and addressed class imbalance. Experiments on the TT100K dataset showed significant performance gains, achieving 89% mAP with reduced parameters and increased FPS, effectively improving detection speed and accuracy.

Shan et al. [18] conducted a study to enhance traffic sign detection using the SSD model, specifically ssd_300, in the context of Chinese road conditions. Training the model on the Chinese Traffic Sign Dataset, they achieved an improved mAP of 0.85 on the test dataset, outperforming ssd_300 by 0.13, while maintaining real-time detection capability. The model exhibited proficiency in detecting three categories of Chinese traffic signs and demonstrated strong robustness against different disturbances.

In conclusion, the reviewed literature emphasizes the importance of selecting appropriate algorithms for traffic sign detection, where the YoloV4 model stands out as a powerful solution, offering an optimal balance between speed and accuracy. Our study aims to build upon the existing research and contribute to the advancement of intelligent driver assistance systems through our carefully chosen YoloV4 model for real-time traffic sign recognition.

3. Materials and Methods

3.1 Dataset

In the study, a comprehensive dataset consisting of a total of 22,000 traffic speed sign images was compiled for training and evaluation purposes. Among these, 4,400 images were reserved for testing, and the remaining 17,600 images were used for training the model. The dataset covered speed limit signs ranging from 20 km/h to 120 km/h and encompassed various background, weather, and lighting conditions to ensure robustness. To ensure diversity and realism in the dataset, the researchers collected data from multiple sources. Approximately 50% of the data was collected using Anadolu Isuzu test vehicles on urban and interurban roads under various conditions, including day and night, rainy, sunny, and snowy weather. Approximately 25% of the data was obtained through screen capture using the Google Maps platform. The remaining portion was collected from the German Traffic Sign Dataset, Tsinghua Tencent Dataset, and Chinese Traffic Sign Datasets, specifically focusing on the recognition of speed limit signs.

Figure 1 in the study presents some sample images from the prepared dataset, providing a visual representation of the different traffic speed signs collected under various conditions.The large and diverse dataset allowed the researchers to effectively train and evaluate the performance of the YoloV4 model in real-time traffic speed sign detection, covering a wide range of speed limits and environmental scenarios.

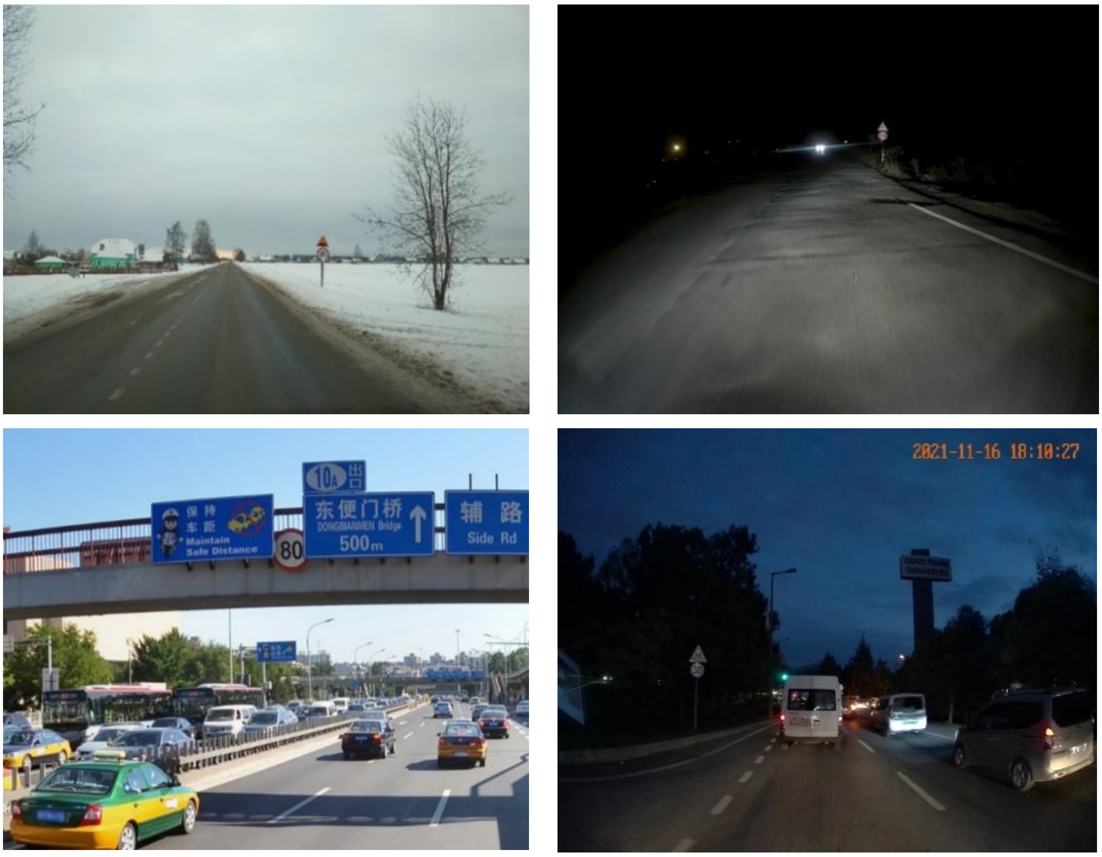


Figure 1. *Traffic speed sign sample images collected from different weather-environment conditions*

3.2. Materials

In the deep learning-based system, artificial intelligence model trainings were carried out on a computer equipped with Intel Core i7-vPRO, Nvidia GeForce RTX 2080 GPU, and 32GB RAM. Developed models have been tested in real time on both computer and embedded device. Nvidia Jetson Nano embedded device (**Figure 2**) used in the designed system has 4GB 64-bit LPDDR4 25.6GB/s memory, 4-core ARM A57 CPU and 128 CUDA core NVIDIA Maxwell™ GPU. With Nvidia Jetson Nano, real-time images from the camera are given as input to deep learning-based models. In addition CSI camera with Sony IMX219 sensor with 77 degree angle of view was used for creating traffic speed sign image dataset and for real-time speed limit sign detection.

Figure 2. *Nvidia Jetson Nano embedded device and CSI Camera [7->19]*

3.3. Method

After the images were collected, the traffic speed signs were labeled, and made ready for model training using the LabelImg toolbox.

The collected dataset was annotated using the open-source image labeling program, labelImg. Subsequently, data augmentation techniques were employed during the artificial intelligence training process of the developed system. These augmentation procedures encompassed angle changes, scaling, saturation, exposure, and hue adjustments to augment the dataset. While angle changes and scaling operations were integrated into the model training, the remaining parameters introduced random variations in terms of color saturation, brightness (exposure), and hue of the images, thereby enhancing the model's robustness and adaptability under diverse conditions. These enhancements contributed to the model's ability to operate with high performance, irrespective of environmental and lighting variations. The parameters, along with their corresponding values, were configured within the yolov4.cfg configuration file, as detailed in Table 1. To account for the relatively small area occupied by speed limit signs within the images, an image resolution of 608x608 was set. While higher resolutions in multiples of 32 could be employed, the resolution of 608x608 was deemed optimal in terms of both speed and accuracy, as higher resolutions may result in performance slowdowns for the model.

Table 1. *Data Augmentation Parameters for Network Training*

Parameter Name	Parameter Value
Width x Height (pixel)	608
Angle (degree)	5
Saturation	2
Exposure	2
Hue	0.1

The data prepared for the training were made ready as 20% test and 80% training set. To validate the performance impact on the created dataset, the system was compared using three different deep learning models, namely SSD and Faster RCNN, in addition to YOLOv4, as part of the study. By evaluating the speed (Frames Per Second - FPS) and accuracy (mean Average Precision - mAP) of these three distinct deep learning models, the system's performance was duly verified and confirmed.

3.3.1 SSD Algorithm

The SSD (Single Shot Multibox Detector) is a deep learning algorithm used for object detection in videos and images [8]. Similar to YOLO, SSD provides fast and accurate prediction performance. When compared to other algorithms such as Faster R-CNN and Mask R-CNN [20], SSD is preferred for its faster inference time and improved accuracy. The key feature that sets the YOLO algorithm apart from others is that it processes the entire image only once. The fundamental principle of the SSD algorithm is to use anchor boxes of various sizes and aspect ratios to detect objects in the image. These anchor boxes are fixed-size boxes that are employed to detect objects of different sizes and aspect ratios. The SSD architecture consists of the following steps: Input Image: SSD takes an input image for object detection. VGGNet and Feature Extractor: SSD is typically based on a pre-trained network like VGGNet, which is used to transform images into feature maps. The feature extractor network is utilized to obtain feature maps at different depths, which are essential for detecting objects. Feature Maps and Anchors: SSD makes predictions for object presence based on the anchor boxes present in the feature maps. Each anchor box predicts the class and location of potential objects using the features within it. Classification and Localization: SSD performs classification to determine the class of the objects and localization to accurately adjust their positions. Non-maximum Suppression (NMS): To handle the issue of multiple bounding boxes detecting the same object, SSD applies non-maximum suppression to remove redundant bounding boxes. In conclusion, the SSD algorithm accomplishes object detection efficiently by processing the image only once. Due to its real-time capabilities and high-performance nature, SSD is often employed in applications requiring fast object detection, including the detection of traffic speed signs. The structure of the SSD algorithm is shown in **Figure 3**.

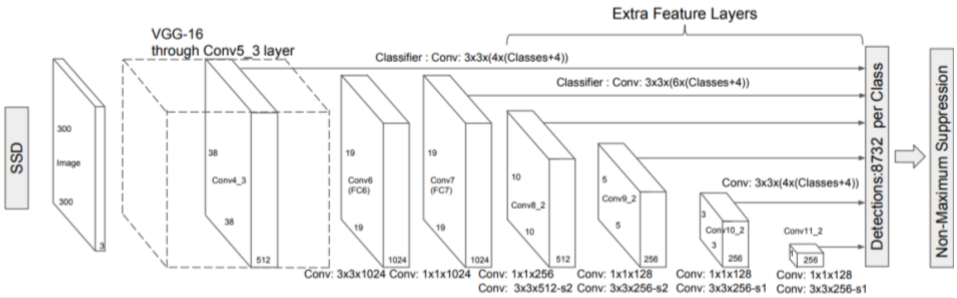


Figure 3. *Structure of the SSD algorithm [8]*

3.3.2 Faster RCNN

Faster R-CNN is a deep learning model developed for object detection tasks [9]. It offers higher detection accuracy compared to other algorithms like YOLO and SSD. It consists of two main components: a feature extractor network (backbone network) and a region proposal network (RPN). Feature Extractor Network (Backbone Network): Faster R-CNN utilizes a pre-trained convolutional neural network (CNN) such as VGGNet, ResNet, or other similar architectures as the feature extractor network. This network transforms the input image into feature maps, identifying important characteristics of objects. Region Proposal Network (RPN): Faster R-CNN employs a region proposal network (RPN) to suggest object candidate regions. RPN uses sliding windows on the feature maps to determine regions that potentially contain objects. These regions are later further examined in detail. The working principle of Faster R-CNN is as follows: Input Image: Faster R-CNN takes an input image for object detection. Feature Extractor: The input image is fed to the feature extractor network, generating feature maps. Region Proposals: RPN suggests object candidate regions using the feature maps. These regions represent areas that may contain objects and are further analyzed. Region-based CNN (RCNN): The proposed object regions are matched with the actual features in the feature maps and then provided to a Region-based CNN (RCNN). The RCNN is used to determine the class of each object region and refine the precise position of its bounding box. The structure of the Faster R-CNN algorithm is shown in **Figure 4**.

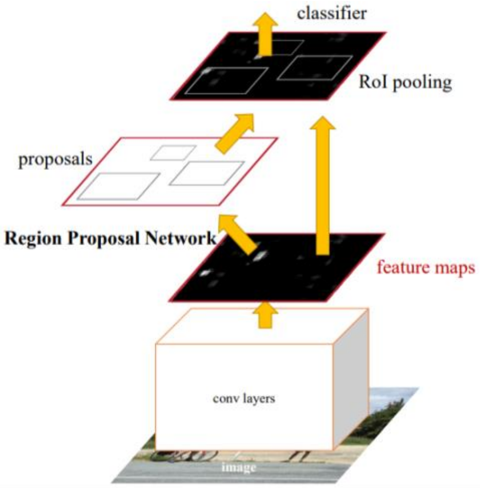


Figure 4. *Structure of the Faster R-CNN [9]*

Faster R-CNN provides high detection accuracy by effectively suggesting object candidate regions through the region proposal network and obtaining better features through the feature extractor network. As a result, it is a popular algorithm used successfully in more complex and detailed object detection tasks.

3.3.3 YoloV4 Algorithm

YOLO (You Look Only Once) is a deep learning algorithm that is built on convolutional neural networks and can detect objects from videos and images. Compared to algorithms such as Faster R-CNN and Mask R-CNN, the YoloV4 algorithm is preferred because it has faster and more accurate prediction performance. The biggest feature that distinguishes the Yolo algorithm from these algorithms is that it processes the image once. The structure of the YoloV4 algorithm, which consists of 106 layers, is shown in **Figure 5**. This structure consists of the input layer, feature extraction layers (Backbone-Neck and Dense Prediction), respectively.

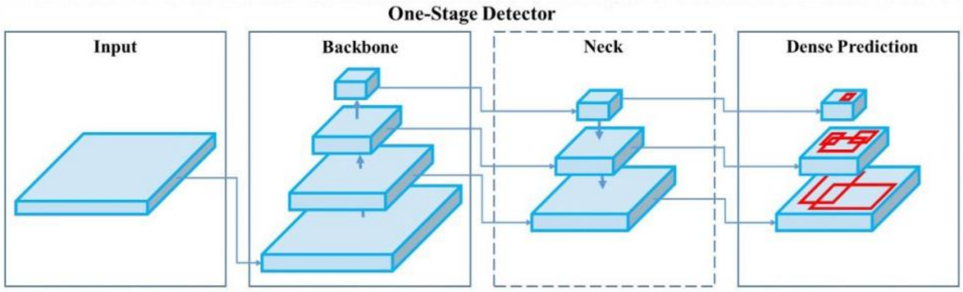
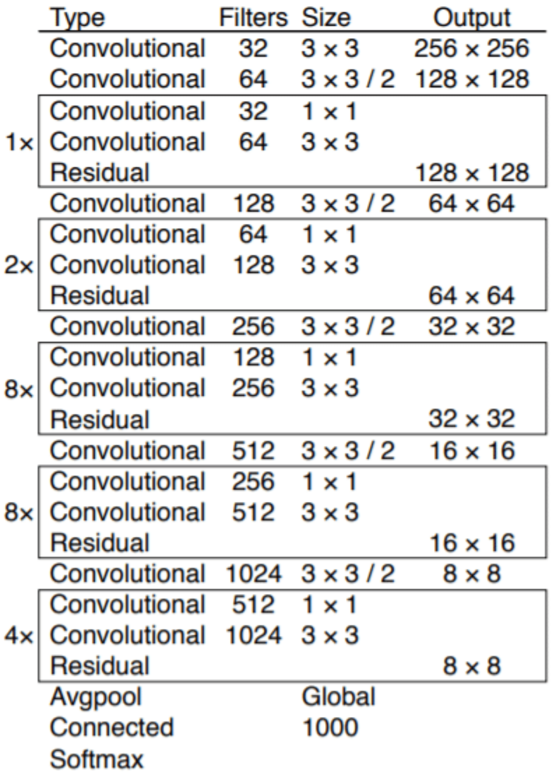


Figure 5. *Structure of the Yolov4 algorithm [21]*

Unlike the Yolov3 version, it includes features such as data augmentation, random image cropping, image blurring functions with CutMix and Mosaic functions. Thanks to these features, the variability of the input image can be increased so that the model is more robust against images obtained in different environments.

Yolov4 algorithm as feature extractor in backbone stage VGGNet [22] It uses the CSPDarknet-53 backbone shown in **Figure 6** to improve the learning capacity of convolutional neural networks similar to CNN architecture.



	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
	Convolutional	32	1 × 1	
1×	Convolutional	64	3 × 3	
	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
	Convolutional	64	1 × 1	
2×	Convolutional	128	3 × 3	
	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
	Convolutional	128	1 × 1	
8×	Convolutional	256	3 × 3	
	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
	Convolutional	256	1 × 1	
8×	Convolutional	512	3 × 3	
	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
	Convolutional	512	1 × 1	
4×	Convolutional	1024	3 × 3	
	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

Figure 6. *Darknet-53 structure [23]*

4. Results

The data generated for traffic speed sign detection are divided into test and training data sets. Then, Intel Core i7-vPRO was trained with the YoloV4 algorithm on a computer with Nvidia GeForce RTX 2080 GPU hardware. 300 images not included in the datasets were used to test the training process.

To thoroughly evaluate the detection performance on a consistent dataset, this study compares various models, including SSD and Faster R-CNN from the literature, using metrics such as mAP (mean Average Precision) and FPS (Frames Per Second). The number of recognized frames is employed as the evaluation metric for the detection efficacy. The experimental findings are presented in **Table 2**. According to the values seen in the table, Although the SSD model is faster, the accuracy of the model and the correct prediction of traffic signs are much more important. On the other hand, while the Faster R-CNN model achieved high accuracy rates, it performed quite poorly in terms of speed. As a result, the proposed Yolov4 algorithm exhibited a higher success rate and demonstrated a highly satisfactory Frames Per Second (FPS) in terms of speed.

Table 2. *Detection Performance Comparison of Various Mod*els

Model	mAP (%)	FPS (frame/second)
Faster R-CNN	93	7
SSD	68	**51**
YoloV4	**95**	42

The model was evaluated using various metrics, and the test results for the traffic speed sign dataset were analyzed with the aid of the confusion matrix presented in **Table 3**.

Table 3. Confusion Matrix

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

The Confusion Matrix is a table used to evaluate the performance of the object detection model on test data with known true values. The adapted version of the matrix for this study is presented in **Table 4**.

Table 4. *Confusion Matrix Explanation.*

Terms	Results
True Positive (TP)	Ex: 50 km/h speed sign detected and it is correct
True Negative (TN)	Presumed to have no speed sign in the image and it's true
False Positive (FP)	Ex: 50 km/h speed sign detected and this is wrong
False Negative (FN)	Presumed to have no speed sign in the image and this is incorrect

Precision Rate is one of the evaluation metrics utilized in object detection tasks. It represents the ratio of correctly predicted objects to all predicted objects. The figure displays the sum of True Positives (TP) and False Positives (FP) for all positive cases as assessed by the model. The Precision is calculated as the division of the number of True Positive cases by the number of all positive cases, and it is represented by Eq. (1).

$$Precision = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \tag{1}$$

The Recall Rate (Recall) is utilized to assess the model's capability to detect true instances within the test set. It is calculated as the sum of True Positives (TP) and False Negatives (FN) for all positive samples in the test set. The Recall Rate is expressed as the ratio of the number of true positive cases to all positive samples in the test set, as shown in Eq. (2).

$$Recall = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \tag{2}$$

The F1 Score value represents the harmonic average of the Precision and Recall values. It is calculated according to Eq. (3).

$$F1\ Score = 2 * \frac{Precision * Recall}{\mathrm{Precision} + \mathrm{Recall}} \tag{3}$$

The higher the F1 value, the more effective the model's testing will be [24]. The mAP (mean average precision) value is a commonly used metric for accuracy assessment in object detection models, representing the average of the average precision (AP) for each object class [25]. The IoU (Intersection over Union) value is another metric used to evaluate object detection systems. Typically, a prediction is considered correct when the IoU is greater than or equal to 0.5. For this study, the IoU threshold was set to 0.5.

The Yolov4 algorithm has been customized based on the dataset. Accordingly, the resizing is set to

608x608. The filters parameter is determined as 3*(5+classes), and for 4 classes, it is set to 48. The number of iterations is set to 30000, and in each iteration, 64 images are utilized, with each image divided into 8 grids. The learning rate parameter is set as 0.0013.

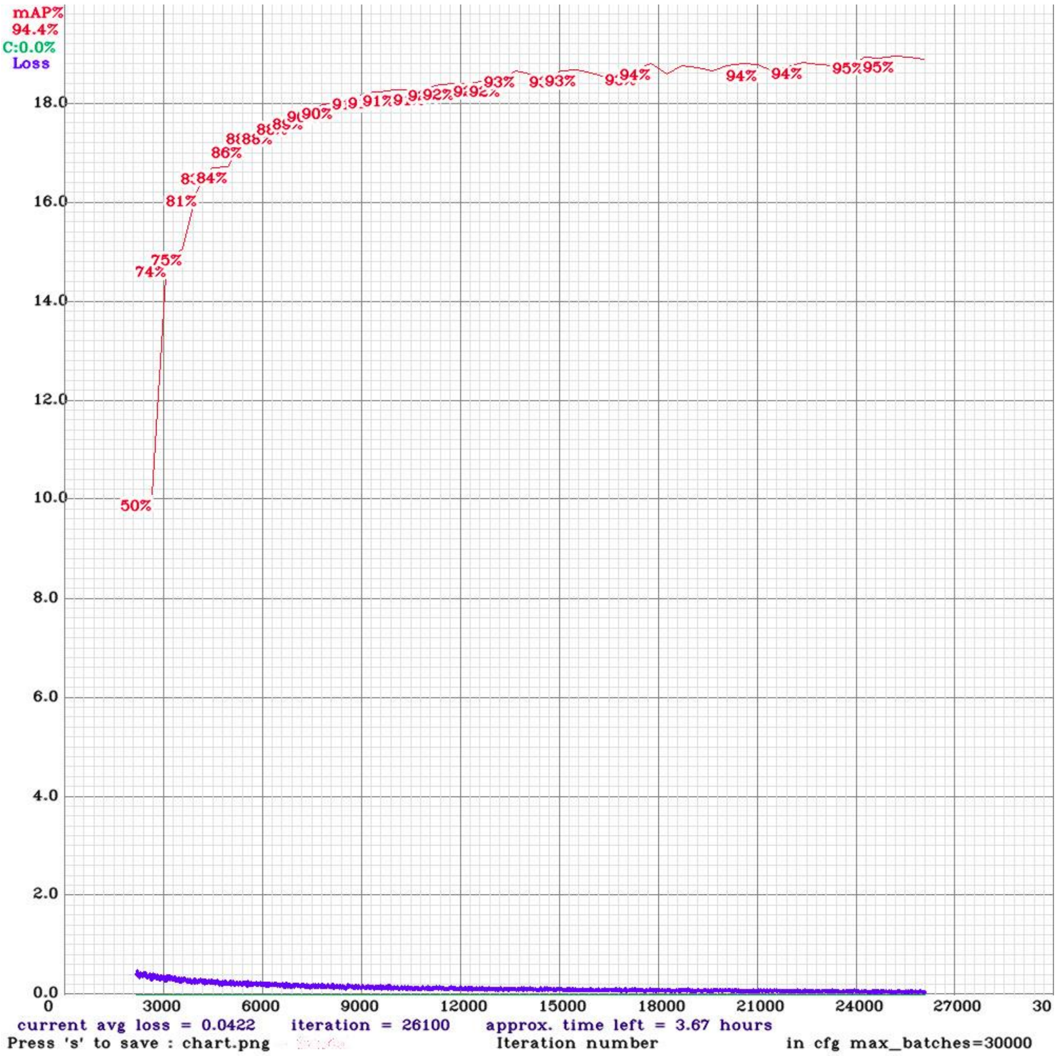


Figure 7. *Yolov4 Performance in a dataset with 11 Traffic Speed signs*

When examining the training result graph (**Figure 7**), it is observed that the error rate gradually decreases as the number of iterations progresses. Consequently, the average precision value, calculated after 3000 iterations, starts at 70% and progressively reaches 95% as the training advances.

Following the application steps, the model's IoU, True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN) values were examined to calculate the precision (Precision) of the model. Subsequently, the mAP and IoU values were determined. The evaluation results of the developed model are presented in **Table 5**.

Table 5. *Evaluation Results*

Evaluation Factors	Results
TP	274
TN	20
FP	2
FN	4
mAP	%95
Average IoU	%82
Precision	%99
Recall	%98
F1-score	%98

When evaluating the model results, it is observed that the F1-score value is 98%, the IoU value is 82%, and the mAP value is 95%. Based on the mAP result, it is concluded that the average sensitivity for each object class is adequate.

5.Conclusion

Mechanical, electronic, and software developments have gained paramount importance in enhancing driving safety within the automotive industry. For this reason, manufacturers undertake efforts to safeguard the lives and property of drivers, passengers, and pedestrians.

In this research, an artificial intelligence-based system was devised to furnish drivers with real-time information about speed signs on the road, with the primary goal of supporting traffic safety. A dataset consisting of 22,000 images was amassed for object detection, where 80% of these images were allocated as training data, and the remaining 20% served as test data.

The LabelImg Image Labeling Program was employed for annotating the objects, and the training process was executed on a computer equipped with an Intel Core i7-vPRO, Nvidia GeForce RTX 2080 GPU, and 32GB RAM using the Darknet Neural Network Framework. Python Programming Language facilitated result determination, and the OpenCV library was utilized for Image Processing algorithms. Python scripts were developed to detect objects in the images using the YOLOv4 Algorithm. Upon the completion of the study, 300 images were subjected to system evaluation, with 294 predictions accurately detected, indicating a 98% correctness rate. The performance metrics of the study yielded an mAP value of 0.95 and an IoU value of 0.82.

To effectively assess the detection performance of the YoloV4 model employed in the system, we additionally deployed the Faster R-CNN and SSD models, which are commonly used in the literature, on the same dataset. While the Faster R-CNN model exhibited commendable overall accuracy, its real-time performance was noticeably limited. Conversely, the SSD model ran faster than the YoloV4 model but demonstrated a significantly lower accuracy. Following the study, it was evident that the Yolov4 model displayed the most favorable performance in terms of both speed and accuracy.

References

[1] Type-approval requirements to ensure the general safety of vehicles and the protection of vulnerable road users [Online]. https://eur-lex.europa.eu/eli/reg/2019/2144/oj, (accessed: 29.09.2022)

[2] Parekh, D., Poddar, N., Rajpurkar, A., Chahal, M., Kumar, N., Joshi, G. P., & Cho, W. (2022). A review on autonomous vehicles: Progress, methods and challenges. Electronics, 11(14), 2162.

[3] Doecke, S. D., Raftery, S. J., Elsegood, M. E., & Mackenzie, J. R. (2021). Intelligent Speed Adaptation (ISA): benefit analysis using EDR data from real world crashes. Age, 26(21), 32.

[4] Dilek, E., & Dener, M. (2023). Computer vision applications in intelligent transportation systems: a survey. Sensors, 23(6), 2938.

[5] González-Saavedra, J. F., Figueroa, M., Céspedes, S., & Montejo-Sánchez, S. (2022). Survey of cooperative advanced driver assistance systems: from a holistic and systemic vision. Sensors, 22(8), 3040.

[6] Liu, C., Li, S., Chang, F., & Wang, Y. (2019). Machine vision based traffic sign detection methods: review, analyses and perspectives. Ieee Access, 7, 86578-86596.

[7] Turan, S., Milani, B., & Temurtaş, F. (2021). Different application areas of object detection with deep learning. Akıllı Ulaşım Sistemleri ve Uygulamaları Dergisi, 4(2), 148-164.

[8] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., & Berg, A. C. (2016). Ssd: Single shot multibox detector. In Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14 (pp. 21-37). Springer International Publishing.

[9] Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. Advances in neural information processing systems, 28.

[10] Jiang, P., Ergu, D., Liu, F., Cai, Y., & Ma, B. (2022). A Review of Yolo algorithm developments. Procedia Computer Science, 199, 1066-1073.

[11] Stallkamp, J., Schlipsing, M., Salmen, J., & Igel, C. (2011, July). The German traffic sign recognition benchmark: a multi-class classification competition. In The 2011 international joint conference on neural networks (pp. 1453-1460). IEEE.

[12] Dewi, C., Chen, R. C., Liu, Y. T., Liu, Y. S., & Jiang, L. Q. (2020, June). Taiwan stop sign recognition with customize anchor. In Proceedings of the 12th International Conference on Computer Modeling and Simulation (pp. 51-55).

[13] Zhu, Z., Liang, D., Zhang, S., Huang, X., Li, B., & Hu, S. (2016). Traffic-sign detection and classification in the wild. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 2110-2118).

[14] Rajendran, S. P., Shine, L., Pradeep, R., & Vijayaraghavan, S. (2019, July). Real-time traffic sign recognition using YOLOv3 based detector. In 2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT) (pp. 1-7). IEEE.

[15] Zuo, Z., Yu, K., Zhou, Q., Wang, X., & Li, T. (2017, June). Traffic signs detection based on faster r-cnn. In 2017 IEEE 37th International Conference on Distributed Computing Systems Workshops (ICDCSW) (pp. 286-288). IEEE.

[16] Gao, B., Jiang, Z., & zhang, J. (2019, July). Traffic sign detection based on ssd. In Proceedings of the 2019 4th International Conference on Automation, Control and Robotics Engineering (pp. 1-6).

[17] Jiang, J., Bao, S., Shi, W., & Wei, Z. (2020). Improved traffic sign recognition algorithm based on YOLO v3 algorithm. Journal of Computer Applications, 40(8), 2472.

[18] Shan, H., & Zhu, W. (2019, October). A small traffic sign detection algorithm based on modified ssd. In IOP Conference Series: Materials Science and Engineering (Vol. 646, No. 1, p. 012006). IOP Publishing.

[19] Jetson Nano [Online]. Available: https://www.nvidia.com/tr-tr/autonomous-machines/embedded-systems/jetson-nano/product-development/, (accessed: 29.09.2022)

[20] He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). Mask r-cnn. In Proceedings of the IEEE international conference on computer vision (pp. 2961-2969).

[21] Bochkovskiy A. 2020. Yolo v4, v3 and v2 for Windows and Linux. https://github.com/AlexeyAB/darknet, Date of access: 29.09.2022

[22] Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.

[23] Wang, C. Y., Liao, H. Y. M., Wu, Y. H., Chen, P. Y., Hsieh, J. W., & Yeh, I. H. (2020). CSPNet: A new backbone that can enhance learning capability of CNN. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops (pp. 390-391).

[24] Yu, J., & Zhang, W. (2021). Face mask wearing detection algorithm based on improved YOLO-v4. Sensors, 21(9), 3263.

[25] Hu, X., Liu, Y., Zhao, Z., Liu, J., Yang, X., Sun, C., ... & Zhou, C. (2021). Real-time detection of uneaten feed pellets in underwater images for aquaculture using an improved YOLO-V4 network. Computers and Electronics in Agriculture, 185, 106135.

Advances in Artificial Intelligence Research (AAIR)
ISSN 2757-7422
www.dergipark.com/aair/

Vol. 3 (No. 2), pp. 76-84, 2023
doi: 10.54569/aair.1303116
Published online: Oct 23, 2023

Deep Learning Ensemble Approach to Age Group Classification Based On Fingerprint Pattern

Olufunso Stephen Olorunsola [1,*], Olorunshola Oluwaseyi Ezekiel [2,]

[1] Department of Computer Science, Nigeria Defence Academy, Kaduna, Nigeria
[2] Airforce Institute of Technology, Kaduna, Nigeria

Abstract

The age distribution of a population is extremely valuable to any business or country. In order to make decisions with regard to facility allocations and other social economic developmental issues, determination of age group distribution information is essential. The attempt to deceive others about one's age is a significant problem in the sporting world, as well as in other organizations and electoral processes. Therefore, there is a requirement for an age detection system, which is required to authenticate individual claims. Fingerprint-based age estimate research is scarce due to paucity of dataset. However, there are indications that fingerprints can reveal age demographic. This study's objective is to live-scan fingerprint images in order to identify age groups. This study proposed novel Dynamic Horizontal Voting Ensemble (DHVE) with Hybrid of Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM) as the base learner. The method constructs a horizontal voting ensemble for prediction by dynamically determining proficient models based on the validation accuracy metric during base learner training on the training set. Accuracy, recall, precision, and the F1 score were employed as standard performance metrics to measures the model's performance analysis. According to this study, predicting individual age group was accurate to a degree of above 91%. The DHVE network performed well due to the design of the layers. Integration of dynamic selection approach to horizontal voting ensemble improved the average performance of the model output.

Keywords: *Deep learning, ensemble, age group, demographic, fingerprint, performance metric.*

1. Introduction

Very important information to every organization or a nation is the age bracket distribution. It is key in taking necessary decisions concerning facility distributions, restricted access and developmental issues. Falsification of age is a big challenge in sport, organizations and during elections. Hence, the need for an age detector system which is necessary to ensure the integrity of information available [1]. Prevalent methods for age estimation involve the use of physical features such as face, teeth, bones and other parts that support conventional methods [2]. Only few researches have been conducted in the estimation of age using the fingerprint. However, evidences abound that age bracket can be determined through individual fingerprint Fingerprints are one of those strange twists and wonders of nature. It is termed human built-in identity cards [3]. Fingerprints are considered as the oldest and most widely used in the world for biometric identification [4]. It is further described by [5] as the most extensively deployed recognition system due to its high accuracy and public acceptability compare with other biometric traits. It is the pattern of ridges and valleys on the surface of a fingertip [5], and no two individual has same fingerprint and it is unique to every man [6,7]. However, it has been found that having an efficient age group identification system is a key facilitator for achieving a number of key age group development results, including the elimination of age falsification in electioneering processes, sports, and other areas where age plays a role in the development.

Out of the several methods involved in verify human identity, biometric system offers a better approach due to it numerous advantageous features over other methods [8]. Biometrics is defined as the science of personal identification centered on their physical, behavioral, and physiological traits such as fingerprint, face, iris, gait and voice [9]. It is the science of using human body for identification purposes. Biometric can be sub-divided into hard and soft biometrics [8]. Primary or hard biometric deals with physical and behavioral traits such as hand geometry, voice, fingerprint, face, iris, Deoxyribonucleic acid (DNA), gait, palm print and keystroke dynamics [10]. Soft biometrics deals with secondary characteristics that provide other information that are not adequate to identify a person clearly such as age, gender, ethnicity, skin color, scars, and height [8]. They are soft because they are not sufficient enough to uniquely identify individual [9].

An exhaustive research study conducted by forensic scientists has resulted in the discovery of a unique pattern that is embedded in the fingerprint. The researchers came to the conclusion that a close examination of the minutia of the fingerprint can provide an insight to a person's age group as well as other vital information with regard to an individual [11]. These distinctive characteristics of fingerprints can be utilized to determine

*Corresponding author
E-mail address: stevenolorunsola@yahoo.com

Received: 26/May/2023; Received in revised form: 11/Aug/2023; Accepted: 28/Sep/2023.

an individual's age based on comparison to other people's prints.

Classifying human age groups based on fingerprint patterns has been the subject of several research, each of which has shown promising results. A method applying discrete wavelet transform (DWT) and the singular value decomposition (SVD) feature extractors to a person fingerprint image in other to estimate the age was proposed [12]. KNN was used as the classifier. The dataset used was collected using scanner with every image of size 260x300 pixels with resolution of 500 at 256 grey levels. All the ten fingers were captured making a total of 3570 fingerprints of which 1590 were female and 1980 were male fingerprints. 2/3 of the total images were used for the training and the remaining for classification. Fingerprints classification were grouped into five classes: up to 12, 13-19, 20-25, 26-35 and 36 and above. The performance of the system shows accuracy of 96.67%, 71.75%, 86.26%, 76.39% and 53.14% in the five groups for male and by 66.67%, 63.64%, 76.77%, 72.41% and 16.79% for female.

In a study, 500 fingerprints were captured from 10 finger of fifty Turkish individuals with the sole aim of estimating their age bracket from fingerprints [13]. The fingerprints obtained were transformed from features to binary images of 1x153600 matrix. KNN classification algorithm was used to classify based on distance measurement by means of "Euclidian distance" classifier. Performance rate of 93.3% for male and 83.0% for female within the age bracket 18- 24. A model to enhance fingerprint image quality to improve performance accuracy particularly in the elderly was designed [14]. Image enhancement was done by applying Local Binary Pattern (LBP) and the Local Phase Quantization (LPQ) operators. The model was evaluated using fingerprint images captured from 500 subjects. The method achieved success rate of 89.1% for age prediction.

In an investigation, the possibility of using human digital fingerprints to estimate human age-groups [15]. The motivation for the study was based on the fact that the variation in width of human digital fingerprint is limited to certain age group while the fingerprint pattern remains the same through life time. In the proposed method, discriminating features of the fingerprint were extracted using Curvelet Transform, with age estimation under consideration grouped into three (i.e. 6-10, 10-14 and 14-18). The approach uses the extracted features for training and testing classifications using Curvelet coefficients. The high dimensionality in the extracted features were removed by projecting them into principal component (PCA) subspace. K-nearest neighbor (KNN) was used as classifier. The experimental results show the little finger gives the best performance for all the classes under consideration. Age bracket 6-10 shows the best accuracy when only the thumb finger is used for training and testing.

In research conducted in an attempt to checkmate the problem of underage voting during election in Nigeria developed a system that predict human age estimation and gender using fingerprint analysis [1]. 280 fingerprints of various age group and gender were captured using life scan election. Discrete wavelet transform (DWT) + Principal Components Analysis (PCA) was used for age classification, while Back Propagation Neural Network was used for gender classification. 140 of the fingerprints were used for learning of which 70 were males and 70 females respectively. The age bracket was grouped thus; 1-10, 11-20, 21- 30, 31-40, 41-50, 51-60 and 61-70 accordingly. The discriminating feature used for gender classification is the Ridge Thickness Valley Thickness Ratio (RTVTR). Experiment Result shows accuracy of 82.14% for age estimation classification.

In research on a multi-resolution texture technique for automatic age-group estimation using digital fingerprints with reference point generation for poor quality images [16]. This study was motivated by the varying texture of human digital fingerprint as the person ages though the fingerprint pattern remains the same. 360 fingerprints images were captured from 36 subjects using live scan of which 18 were males and the other 18 females. The fingerprints obtained were distributed proportionally among the various gender to avoid distribution bias. The age bracket under consideration are 6-10, 10-14 and 14-18. 250 fingerprints were used for learning process and 100 for testing. The proposed novel approach Extracted features at different resolutions using Gabor filters and Wavelet based noise removal was applied to reduce the feature loss. Three other state-of-the-art classifiers were used to adjudge the performance accuracy of the proposed model. The results from the experiment shows the possibility of age-groups identification from digital fingerprint, particularly children. Classification accuracy of 80% for age group below 14 was achieved.

In a recent research, a comprehensive human demographic attributes of gender and age group were classified from multiple soft biometric traits [17]. Soft biometric traits considered in the research are hand, voice recordings and fingerprint. For the purpose of the research, hand videos, fingerprints and voice were obtained from 560 African subjects. The subject distribution consists of 197 females and 363 males. Right index finger of each of the subject were captured in two sessions. The images are in gray scale of 640x480 pixels. Convolutional neural networks (CNN) was used to train classifiers for fingerprint, voice and hand. The results show that age is best predicted from fingerprint.

To sum up, the bulk of the previous articles used ridge features for classification, while some used the ink method for fingerprint collection. One of the difficulties of this approach is that it is impossible to eliminate human error from the data collecting process. Like this, a lack of data makes it difficult to generalize the findings, which is a major limitation. Also, the deep learning technique, which can automatically recognize and define the underlying differences in data that are hard to measure, was not explored by the vast majority of the studies [18,19]. This is what inspired us to conduct this research, which employs a live-scan technology that accounts for the subject's age in order to create a comprehensive and realistic fingerprint dataset. Hopefully, the missing data in this introduction will be filled up by the dataset. An innovative deep learning model, the Dynamic Horizontal Voting Ensemble (DHVE), will be used to implement the proposed system.

2. The Proposed Method

Since fingerprint-based age group categorization is a complicated process, a powerful machine learning strategy needs to be developed for it. Details on the design process and model evaluation are provided here. The proposed Dynamic Horizontal Voting Ensemble (DHVE) system is shown in **Figure 1**. The implementation consists of four stages: stage one being the collecting and preparation of data. Stage two, development and training of the Model. The last stages are a dynamic ensemble selection followed by a prediction phase.

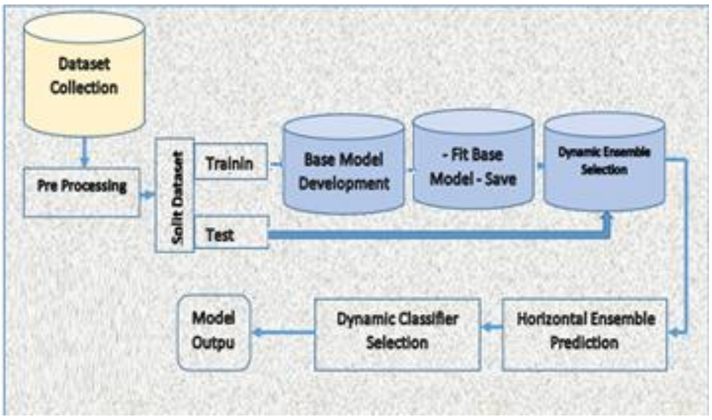


Figure 1. *Dynamic Horizontal Voting Ensemble Framework*

In this study, scanned fingerprint images were captured from a total of 453 Nigerian subjects. However, images from 450 Nigerian subjects were used in this study. Fingerprints from three (3) of the participants were left out due to poor quality. The ten (10) fingerprint of each subjects were captured making a total sample 4500 images 1170, 1190, 1290 and 850 belong to Child, Teen, Adult and Senior age groups respectively. To provide for dataset balance and fairness, only 850 subset images from Child, Teen, Adult and Senior subjects were utilized in this study (See **Table 1**).

Table 1. *Dataset distribution according to age group*

Biometric Class	Training Set	Test Set	Total
Child	680	170	850
Teen	680	170	850
Adult	680	170	850
Senior	680	170	850
Total	2,720	680	3,400

The fingerprint images were labeled with such attributes as image ID, gender, age group, ethnicity and finger type labels (i.e. the thumb, index, middle, ring and little finger labels) as shown in **Figure 2**. The age group categorization are Child (1-12), Teen (13-20), Adult (21-50) and Senior (above 50).

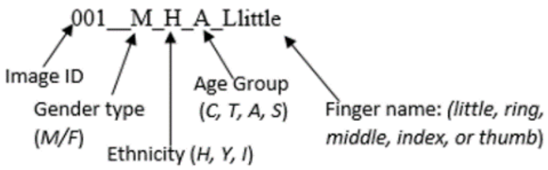


Figure 2. *Fingerprint image attribute*

2.3. Data Preprocessing Stage

The elimination of unwanted features and the minimization of noise are the primary goals of preprocessing [19]. In this study, the histogram equalization approach was utilized [20], which is a method that changes the histogram of an image into a uniform histogram by selecting all of the grey levels uniformly across the histogram of the image. Let f represent an image of dimensions mr by mc, where mr and mc are the rows and columns of a matrix containing integer pixels with intensities from 0 to L 1. Where L is the range of intensities that can be used (often 256). For convenience, we'll refer to p as the normalized histogram of f, where each intensity level is represented by a separate bin. So;

$$pn = \frac{\text{number of pixels with intensity } n}{total\ number\ of\ pixel!}$$

n = 0, 1...L-1

The histogram-equalized image g will be described by

$$\mathrm{g}_{i,j} = \mathrm{floor}((\mathrm{L} - 1) \sum_{n=0}^{f_{ij}} p_n), \quad (1)$$

where floor() round numbers down to the nearest integer. A bilateral filter smoothed and denoised the image while retaining edges. Bilateral filter improves Gaussian filter. Gaussian blurring formula:

$$GB[I]_p = \sum_{q \in S} G_\sigma(\| p - q \|)\ I_q \quad (2)$$

where Gσ(($\| p - q \|$)) is the 2D kernel Gaussian function. Gaussian filtering computes the image pixels weighted average of nearby spots with a declining weight pattern with respect to spatial distance from the midpoint p. Pixel q is given by the Gaussian G(||p q||), where σ is a neighborhood-size determining factor. Bilateral filters weight neighboring pixels like Gaussian convolution. However, the bilateral filter smooths while preserving edges by considering nearby pixels' value differences. The bilateral filter for the image I is indicated by BF[I], where Iq is the image pixel and Ip is the image midpoint.:

$$BF[I]_p = \frac{1}{W_P} \sum_{q \in S} G_{\sigma_s}(\| p - q)\ G_{\sigma_r}(I_p - I_q)\, I_q \quad (3)$$

Wp functions as a normalization parameter to ensure that pixel weights add up to 1

$$W_p = \sum_{q \in S} G_{\sigma_s}(\| p - q)\ G_{\sigma_r}(I_p - I_q)\, I_q \quad (4)$$

With the parameters σs and σr in the equations, we may determine the amount of filtering applied to the image I in Eq. (3)

2.2. Base Model Architecture

The Deep Convolutional Neural Network-Long Short-Term Memory (Deep CNN-LSTM) model was used as the base model in this study (see **Figure 3**). Two convolutional layers, two maxpooling layers, and two fully connected layers make up the CNN model structure. In order to function, the CNN and LSTM models convert the CNN output to (batch size, H, W*channel), where H and W stand for the image's height and width, respectively. This will provide the LSTM layer access to data in 3D. The reshape procedure is triggered by the lambda function. Layers of dense and softmax activation functions are employed by the LSTM model to make its predictions. The used LSTM layers each had 16 and 96 units. The output of the LSTM layer is then sent to the fully connected (FC) output layer, which is activated using a Softmax function, for final classification.

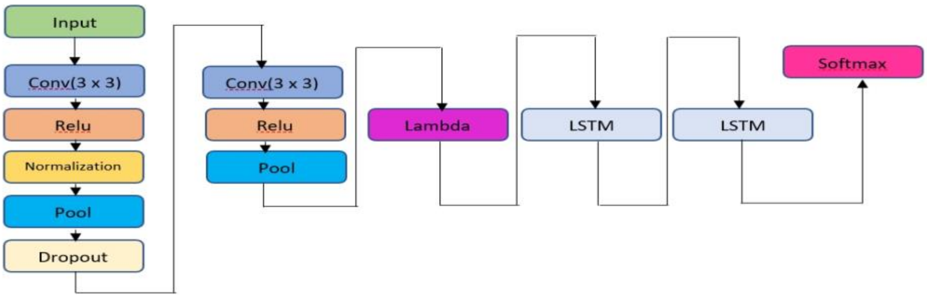


Figure 3. *Architecture of CNN-LSTM*

2.3. Dynamic Selection Scheme for Horizontal Voting Ensemble

Algorithm A describes the suggested method for selecting members of the horizontal voting ensemble dynamically. While training the base learner on the training set, the approach dynamically determines competent models based on the validation accuracy measure, allowing for the construction of a horizontal voting ensemble for prediction. Each training epoch is saved if model accuracy is over the threshold. To build a prediction ensemble, the ideal subset of stored models is chosen depending on ensemble size. Dynamic selection allows the best-performing models to join the ensemble during prediction, unlike the current technique in which ensemble members are deliberately chosen. Algorithm A describes the process.

```
Algorithm A: Dynamic stage
Input
Dataset: $Data_{set}$ = $Data_{Trn}$ U $Data_{Test}$
Component $Data_{set}$ Intersection in $Data_{Trn}$ & $Data_{Test}$ = Ø
Set initial values for J,j and $K_{set}$
Set list $En_{j}$ = [ ]
Set selection threshold_value
Procedure
While i ≤ J do
  Train $Data_{Trn}$ for one epoch
      if $epoch_{Acc}$ ≥ threshold_value:
            epoch.save(list(i))
      increment i by 1
  end while
Assign all_epoch saved to $K_{set}$
Arrange $K_{set}$ in ascending order of $epoch_{Acc}$
Assign $K_{set}$ to $En_j$ where j is the first $j^{th}$ elements of $K_{set}$
Output: $En_{j,}j$
```

The dynamic technique was used at two critical times in the development of the model. When choosing ensemble members and a single classifier's output surpasses the ensemble forecast, respectively.

The second algorithm dynamically selects a final prediction classifier. Algorithm B received ensemble member from algorithm1. The experiment used up to 150 epochs with ensemble sizes from 1 to 50. It provides the general dynamic selection method to horizontal voting ensemble. The dynamic technique was used in two key model development periods. First, when choosing a model for the ensemble, and second, when determining which classifier or ensemble score gives the best prediction score.

```
Algorithm B: Final Phase of DHV Ensemble Model
Input:
Build ensemble of varying size of $En_j$= {$e_1$,..$e_j$}
Evaluate varying size of ensemble $En_j$ on $Data_{test}$ = {$Data_{t1}$,..,$Data_{tj}$}
Set ensemble result $Ens_{result}$,
Set initial value for list variables Predicts, $Model_{result}$ = [ ]
Set parameters: $Model_{max-score}$, $Sub_{set}$, Sngle_result
Procedure:
While i ≤ j do:
   $Sub_{set}$ = $En_j$ [:i]
     for all epoch in range i:
       Train $Data_{test}$ in $Sub_{set}$(epoch), assign output to predicti
        predicti = predicti + Predicts
       epoch = epoch + 1
      end for
   P = $\sum_{\text{predicti}}^{\text{predicti } \in \text{ Predicts}}$ predicti
   $Ens_{result}$ = argmax(P)
   for all j in range (1, i+1):
       Sngle_result = En[j-1]. P($Data_{test}$)
       Append Sngle_result to $Model_{result}$
     end for
         epoch = 1
  end while
  Assign highest score in list $Model_{result}$ to $Model_{max\_score}$
  Set final $Ens_{result}$ value to most competent of $Model_{max\_score}$ and initial $Ens_{result}$
  score
 Output: $Ens_{result}$
```

3. Experimental Results

Table 2 illustrates the trained DHVE model's accuracy, precision, recall, and F1 score. Image quality and assessment methodologies impact algorithm performance measurement.

Table 2. *Classification performance of the DHVE model*

Fingerprints	Precision	Recall	F1 Score	Support
Adult	0.94	0.95	0.95	170
Child	0.77	0.76	0.75	170
Senior	0.79	0.74	0.74	170
Teen	0.99	1.00	0.99	170

From **Table 2**, The classification performance for teen in terms of recall equal to 1. The recall score of 1 signifies that all True Positives were identified and classified correctly. The precision and F1 score for Teen are near 1 at 0.99. This means that almost all the positive samples are classified to be positive and non-positive samples are classified as non-positive samples (for Precision values equal to 0.99). The F1 score of 0.99 means that the model was able to classify imbalance data perfectly. The DHVE model overall accuracy score for Age Group classification is 91% as shown in **Table 3**.

Table 3. *Overall Classifications Performance of DHVE model*

Classification Parameter	Precision	Recall	F1-Score	Support
Accuracy			0.91	680
macro avg	0.91	0.91	0.91	680
weighted avg	0.91	0.91	0.91	680

From **Table 3**, the macro and weighted average precision, recall, and F1 Score is 0.91, 0.91, and 0.91 respectively. Each metric performs well in classification. The macro and weighted average F1 score of 0.76 suggests that the proposed model effectively classified imbalance dataset. See **Figure 4** for the Average ROC-AUC.

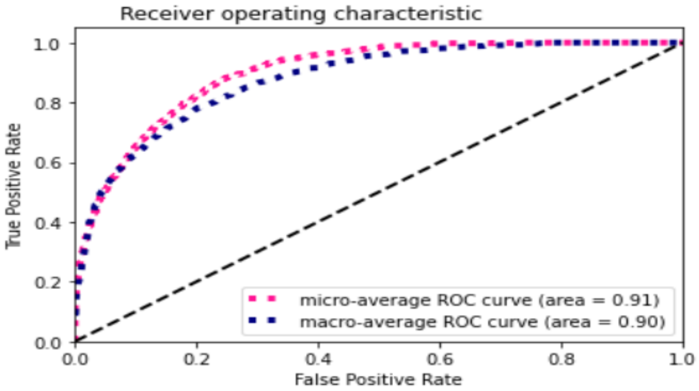


Figure 4. *Average ROC-AUC Plot of the DHVE Model for Age Group Classification*

Figure 4 depicts the average Receiver Operator Characteristics (ROC) Curve of the DHVE Model for multi-class. It depicts the Probability-based Curve that graphs the TPR vs the FPR at different levels, effectively isolating noise. The Area Under the Curve (AUC) measures the classifier's ability to distinguish across classes. The bigger the values, the stronger the classifier's ability to distinguish between positive and negative categories. Comparing the number of true positives against the rate of false positives, the Receiver Operating Characteristic (ROC) curve illustrates how effectively a model can classify. **Figure 4** illustrates the micro and macro averages for all examined classes. As seen by the average curve (across all classes), the model distinguishes between positive and negative classes with outstanding precision, as class averages are close to 1.00. **Figure 5** depicts the Confusion matrix for the DHVE classification.

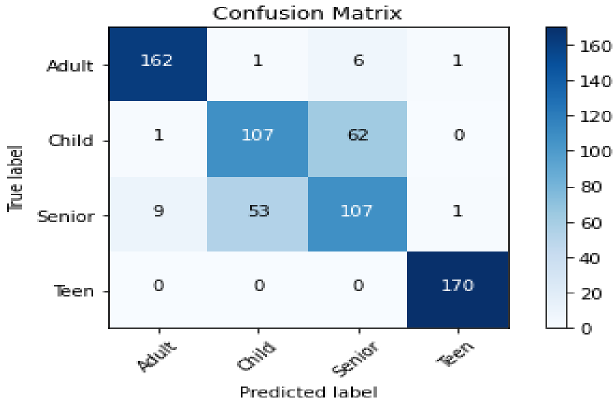


Figure 5. *Confusion Matrix of the DHVE Model for Age group*

Figure 5 illustrate the Confusion matrix for each Age group classification. The actual and predicted values for Adult, Child, Senior, and Teen are 162, 107, 107 and 170 respectively. The Confusion matrix for each class shows that the actual and predicted value for all Teen were correctly classified at 170. The model predicted 162 as Adult correctly while 1,6,1 Adult images were wrongly predicted as Child, Senior and Teen respectively. Likewise, 107 Child and 107 Senior were correctly classified. The model misclassified 62 Child as Senior, 9 Senior were wrongly classified as Adult. **Figure 6** shows the accuracy distribution of the proposed DHVE model with HVE and CNN-LSTM model.

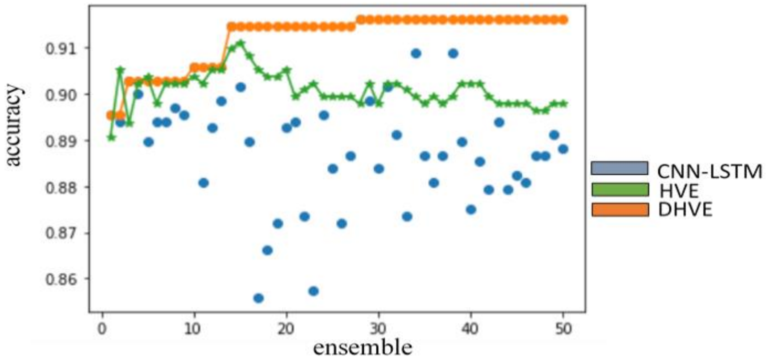


Figure 6. *Line Plot showing DHVE Performance comparison with other Model*

Figure 6 shows the accuracy distribution of the proposed DHVE model with HVE and CNN-LSTM model. The Blue legend in the plot is the result for CNN-LSTM and the Green is the HVE, Orange is the result for DHVE. The accuracy for CNN-LSTM, HVE and DHVE models are 0.878, 0.897 and 0.912 respectively. From **Figure 6**, the HVE and DHVE curves shows the performance of the ensemble models of various sizes from 1-50 members. The DHVE curve reveals that the performance of the ensemble improves as the number of ensemble increases from 12 until somewhere around 25 ensemble size where the performance became stable. The plot also reveals that the dynamic approach, which allow best performing models to participate in the prediction, enhances the performance of DHVE as against HVE in green colour which uses the static ensemble selection. The DHVE model outperforms all the other models with overall accuracy of 0.912. **Table 4** shows the performance metric of the models.

Table 4. *Models Performance on Precision, recall, F1 and Accuracy Metrics*

Models	Performance Metrics (%)			
	Precision	Recall	F1 Score	Accuracy
CNN-LSTM	88.0	87.0	87.0	87.0
HVE	89.0	87.0	87.0	89.0
DHVE	91.0	91.0	91.0	91.0

The superior performance of the DHVE model relative to other models demonstrates that the incorporation of a dynamic scheme increases the accuracy of the final prediction.

3.1. Comparison with existing networks

The performance of the proposed DHVE model was compared to that of k-Nearest Oracle algorithm (KNORA) and Dynamic Classifier Selection with Overall Local Accuracy (DCS-LA) algorithm using the data obtained for this study. DCS-LA algorithms that decide on a trained model from a large pool of candidates depending on the specifics of the input The k-nearest neighbor (kNN) approach is used to locate the k most comparable instances from the training dataset which correspond to the example when a prediction is needed. Evaluation of each model's classification accuracy on the vicinity of k training examples is known as local accuracy (OLA) [21]. The model chosen to make a forecast for the new example is the one that performs the best in this region. KNORA model is a dynamic selection-based oracle-based approach presented by [22]. The KNORA model's neighborhood boundaries are determined by the k-nearest neighbor's algorithm's parameters. Choosing a k number that fits the dataset is crucial since it determines the neighborhood size. When k is too little, relevant training set samples may be removed from the neighborhood, but when k is too huge, relevant samples may be masked by too many instances. This experiment tests ten different k values from 2 to 12 to get the optimum dataset outcome. This experiment will use the KNORA–Union algorithm. **Figure 7** and **8** illustrates KNORA-U and DCS-LA model accuracy distribution on age group dataset with k values from 2 to 12.

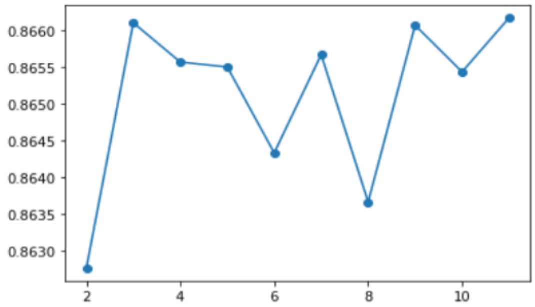


Figure 7. *Line Plot Showing Age Group Dataset Accuracy Distributions for KNORA-U*

The minimum performance occurs when k = 2, while the maximum accuracy is 0.866% at neighborhood size of k = 3 and 11.

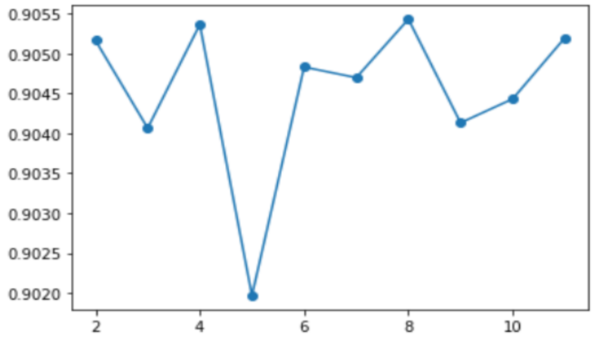


Figure 8. *Line Plot Showing Age Group Dataset Accuracy Distributions for DCS-LA*

The minimum performance occurs when k = 5, while the maximum accuracy is 0.90% at neighborhood size of k = 8

Table 5. *Comparison of proposed DHVE with KNORA-U and DCS-LA model.*

Dataset	Model Accuracy (%)		
	KNORA-U	DCS-LA	DHVE
Age group	86	90	91

4. Conclusion

Based on the outcomes of this study, a model has been developed that can determine age group estimations using fingerprint data. In addition, the results of this research have shown, via a series of controlled experiments, that the implementation of a dynamic scheme results in an improvement in the functionality of the horizontal voting ensemble method. The study's results also show that the DHVE Model is a good tool for recognizing fingerprints in a biometric system. Others advantages of this study shows improvement in the accuracy of the predictions. It reduces the variance of the predictions and makes the predictions more robust to noise. Based on how well the model worked, fingerprints could be put into groups based on age with little error. the disadvantage It is computationally expensive compare to known ensemble model. Training the model with a larger dataset can help it do a better job of classifying. In the future, researchers could also look into how adaptive preprocessing techniques could be used to improve low-quality fingerprint images and make classification models more accurate.

Declaration of interest: The authors declare that there is no conflict of interest.

References

[1] Galbally J, Haraksim R, Beslay L. “A Study of Age and Ageing in fingerprint Biometrics”, IEEE Transactions on Information Forensics and Security, 14(5), 1351–1365, 2019.

[2] Kumar S, Rani S, Jain A, Verma C, Raboaca MS, Illés Z, Neagu BC. “Face Spoofing, Age, Gender and Facial Expression Recognition Using Advance Neural Network Architecture-Based Biometric System”, Sensors, 22(14), 51-60, 2022.

[3] Medina-Sotomayor P, Pascual MA, Camps AI. “Accuracy of four digital scanners according to scanning strategy in complete-arch impressions”, PLOS ONE, 13(9), 2018.

[4] Al-Refoa A, Alshraideh M, Sharieh A. “A New Algorithm for Locating and Extracting Minutiae from Fingerprint Images”, Pattern Recognition and Image Analysis, 29(2), 268–279, 2019.

[5] Yang W, Wang S, Hu J, Zheng G, Valli C. “Security and Accuracy of Fingerprint-Based Biometrics: A Review”, Symmetry, 11(2), 141, 2019.

[6] Bahmani K, Plesh R, Johnson P, Schuckers S, Swyka T. “High Fidelity Fingerprint Generation: Quality, Uniqueness, And Privacy”, IEEE International Conference on Image Processing (ICIP) 2021.

[7] Faridah Y, Nasir H, Kushsairy AK, Safie SI, Khan S, Gunawan TS. “Fingerprint Biometric Systems”, Trends in Bioinformatics, 9(2), 52–58, 2016.

[8] Abdelwhab A, Viriri S. “A Survey on Soft Biometrics for Human Identification”, Machine Learning and Biometrics. 2(3), 2018.

[9] Das AK, Antitza D, Francois B. “Mitigating Bias in Gender, Age and Ethnicity Classification: A Multi-task Convolution Neural Network Approach”, Lecture Notes in Computer Science, 573–585, 2018.

[10] Lee H, Hwang JY, Kim DI, Lee S, Lee SH, Shin JS. “Understanding Keystroke Dynamics for Smartphone Users Authentication and Keystroke Dynamics on Smartphones Built-In Motion Sensors”, Security and Communication Networks, 2018.

[11] Kloppenburg, S., Van der Ploeg, I. “Securing Identities: Biometric Technologies and the Enactment of Human Bodily Differences. Science as Culture, 1–20, 2018.

[12] Gnanasivam P, Muttan S. “Fingerprint Gender Classification using Wavelet Transform and Singular Value Decomposition”, ArXiv (Cornell University) 2012.

[13] Eyüp BC, Seref S, Ramazan C, Oner K. “Age Estimation from Fingerprints: Examination of the Population in Turkey”, International Conference on Machine Learning and Applications”, 4(1), 2014.

[14] Marasco E, Luca L, Bojan C. “Exploiting quality and texture features to estimate age and gender from fingerprints”, Proceedings of SPIE 2014.

[15] Saxena A, Vijay KC. “Multi-resolution texture analysis for fingerprint based age-group estimation”, Multimedia Tools and Applications, 77(5), 2018.

[16] Das S, De Ghosh I, Chattopadhyay A. “Deep Age Estimation Using Sclera Images in Multiple Environment”, Advances in Intelligent Systems and Computing, 93–102, 2021.

[17] Iloanusi ON, Ejiogu UC. “Gender classification from fused multi-fingerprint types: A Global Perspective”, Information Security Journal, 1–11, 2020.

[18] Ibrahim AM, Eesee AK, Al-Nima RRO. “Deep fingerprint classification network”, TELKOMNIKA (Telecommunication Computing Electronics and Control), 19(3) 893-897, 2021.

[19] Deshmukh DK, Patil SS. “Fingerprint-Based Gender Classification by Using Neural Network Model”, Applied Computer Vision and Image Processing, 318–325, 2020.

[20] Xuan Z, Liu H, Li C, Liu Y. “Wavelet Bilateral Filter Algorithm-Based High-Frequency Ultrasound Image Analysis on Effects of Skin Scar Repair”, Scientific Programming, 1–7, 2021.

[21] Cruz, R. M. O., Sabourin, R., Cavalcanti, GDC. ‘Dynamic classifier selection: Recent advances and perspectives. Information Fusion, 41, 195–216, 2017.

[22] Ko AHR, Sabourin R, Britto Jr, Alceu S. “From dynamic classifier selection to dynamic ensemble selection”, Pattern Recognition, 41(5), 1718–1731, 2008.

Advances in Artificial Intelligence Research (AAIR)
ISSN 2757-7422
www.dergipark.com/aair/

Vol. 3 (No. 2), pp. 85-95, 2023
doi: 10.54569/aair.1254810
Published online: Oct 23, 2023

A Machine Learning Approach for Simultaneous Classification of Material Types and Cracks

Ömer Mintemur [1], *,

[1] Ankara Yıldırım Beyazıt University School of Engineering and Natural Sciences

Abstract

Exterior structures are susceptible to deformation, which can manifest as cracks on the surface. Deformations that occur on surfaces subjected to daily human use can exacerbate rapidly, potentially leading to irreversible structural damage. They have a potential to result in fatalities. Thus, continuous inspection of these deformations is of invaluable importance. In addition, the identification of the materials comprising the structures is essential to facilitate the implementation of appropriate precautionary measures. However, the inspections are hard to maintain with a solely human workforce. More advanced actions can be taken thanks to the developments in technology. Machine Learning methods could be used in this area where human workforce is ineffective. In this regard, an end-to-end Machine Learning approach was proposed in this study. The power of classical feature extraction methods and Artificial Neural Networks were combined to detect cracks and material of the surface simultaneously. The 2D Discrete Wavelet Transform and statistical properties gained from Gray Level Co-Occurrence Matrix were utilized in the feature extraction mechanism, and an ANN structure was designed. The findings of the study indicate that the proposed mechanism achieved an acceptable level of accuracy for recognizing the structural deformations, despite the challenges posed by the complexity of the problem.

Keywords: *Classification; material recognition; machine learning; crack detection.*

1. Introduction

Structures are susceptible to deformation (*cracks*) over an extended period of time and with frequent use [1,2]. The most common symptom of deformation is the appearance of cracks on the surface of the structure. A crack is a type of discontinuity that can form on the surface of a material, characterized by a break or separation along a portion of the surface, disrupting its uniformity and integrity [3]. The occurrence of surface cracks can have a considerable impact on the strength, stability, and aesthetic appearance of the material. In certain instances, surface cracks may also be indicative of deeper structural issues, requiring further investigation [4].

The structural deterioration tends to occur with greater frequency on exteriors such as walls, roads, and pavements. An example of the progression of surface degradation can be seen in the transformation of cracks on roads into holes over time [5]. The presence of holes on pavements presents a significant threat to public safety [6]. As such, it is essential to continuously monitor and promptly address these structural deficiencies to ensure the longevity and safety of the infrastructure.

However, conducting ongoing monitoring is not practical due to the constraints of the available workforce [7]. But advancements in technology allow for the delegation of ongoing monitoring tasks to computers. The utilization of computer vision techniques has become increasingly prevalent in the pursuit of continuous surface defect detection.

The established methodology for this task entails the extraction of salient features from an image, followed by an algorithm that categorizes the image as either exhibiting defects or being free from defects. The generation of features constitutes a component of the broader domain of image processing, encompassing various techniques including edge detection, extraction of color information, thresholding, and calculation of statistical characteristics such as energy and contrast etc. [8,9]. On the other hand, classification algorithms are situated within the domain of Artificial Intelligence (AI) and are commonly known as Machine Learning (ML) algorithms. The extracted features are input into an ML algorithm, which performs the classification.

There exists a substantial body of research that employs these conventional methodologies. An end-to-end methodology for identifying cracks in asphalt surfaces can be found in [10]. The methodology was divided into several sections. Prior to the feature extraction, thresholding and noise elimination were employed to enhance the quality of the acquired image. In the feature extraction phase, the Hough Transform (HT) [11] was utilized to extract features from the image. Finally, Bagging, which is an ensemble ML method [12] and Support Vector Machines (SVM) [13] were used to classify the cracks.

A study utilized Local Binary Pattern (LBP) [14] and Principal Component Analysis (PCA) [15] to identify

*Corresponding author
E-mail address: omintemur@ybu.edu.tr

Received: 22/Feb/2023; Received in revised form: 09/Aug/2023; Accepted: 01/Oct/2023.

cracks on the pavements can be found in [16]. The extracted features were classified using SVM, resulting in a high rate of classification accuracy, as indicated by the authors.

Statistical information extracted from a given image can serve as a significant feature for the purpose of classification, and such a study used these numerical values to detect the defect in pavements [17]. The authors used statistical properties that are calculated by constructing Gray-Level Co-Occurrence Matrix (GLCM) [18]. The implementation of this approach resulted in a substantial classification rate, estimated to be approximately 88%.

In addition to the conventional image processing (CIP) techniques, there has been a recent trend in the field towards the utilization of Convolutional Neural Networks (CNNs) as a means for image classification. CNNs can be classified as a sub class of Artificial Neural Networks (ANNs) and they offer a unique advantage as an end-to-end system, demonstrating exceptional performance in both feature extraction and classification due to its utilization of the backpropagation algorithm [19,20]. One of the key advantages of using CNNs in crack classification is the elimination of the manual feature extraction process. And these powerful networks have been widely utilized in the literature of crack detection [3, 21-24].

However, both CNNs and CIP techniques in this area have their own limitations that can be extended. The process of feature extraction in CNNs is automated and has the possibility to identify redundant features that may impede the efficiency of the training procedure. Also, it is noteworthy that there is currently no universally accepted method for the construction of these networks.

On the other hand, CIP techniques that have been proposed in this field are limited by their singular approach. Nevertheless, there exist potent image processing methods that can be combined to achieve more robust feature from the image data. Furthermore, classical ML algorithms are also limited by their finite number of parameters and are therefore not well-suited for modifications.

Therefore, a research inquiry emerges as to whether the capabilities of two sub-fields (CIP and ANN) of AI can be combined to produce competitive results in the area of crack detection.

The current research focus in this domain primarily centers around the detection of cracks on surfaces. However, the exclusive monitoring of cracks may not be beneficial in overall. Classifying the type of surfaces is also a crucial factor in ensuring appropriate maintenance and emergency response measures. This raises a second research question as to whether it is feasible to detect both cracks and the surface material with high accuracy.

This paper presents an end-to-end ML approach for detecting cracks and classifying the type of surface in question at the same time, in accordance with the mentioned research questions. The images of cracks on a surface tend to exhibit a distinctive linear orientation, which may be characterized as horizontal, vertical, or diagonal. These orientations were enhanced by using Discrete Wavelet Transform (DWT). GLCM features were utilized to extract statistical features from the enhanced images [17,25]. Finally, the obtained features were fed into an ANN structure to detect both cracks and surface material. The experimental evaluations were conducted using an image dataset produced by Utah State University, as it offers a diverse representation of various materials along with both cracked and non-cracked images.

The rest of the paper is organized as follows: The details regarding the dataset utilized and the methodology employed comprehensively described in Section 2. Section 3 outlines the experimental setup, configurations, and metrics employed to evaluate the efficacy of the proposed method. The results of the experiments, along with their interpretation, are presented in Section 4. Finally, the concluding remarks and suggestions for future work are presented in the last section of the paper.

2. Materials and Methods

This section briefly gives information to the reader about the dataset, the methods utilized, and the experimental setup in this paper.

2.1. Dataset

The present study utilized the SDNET2018 dataset to evaluate the efficacy of the proposed methodology. The dataset comprises three distinct surface types, namely, decks, walls, and pavements, each of which is categorized into two classes based on the presence or absence of cracks. The dataset consists of a total of 56,000 images depicting cracked and non-cracked decks, walls, and pavements, each with a resolution of 256x256 pixels. Total number of cracked images is 8484 and non-cracked images is 47607 [26,27]. Each image in the dataset is presented in RGB color format. Straightforward exemplary images from each class are given in **Figure 1**.

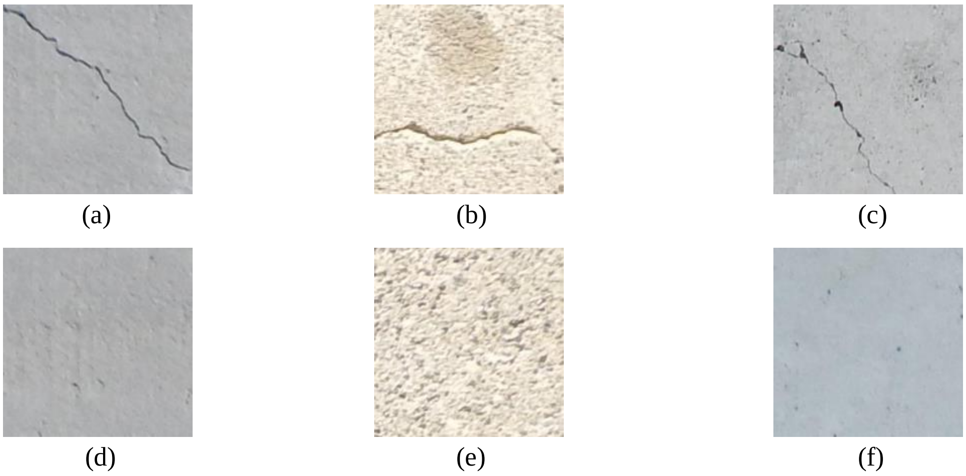


Figure 1. *Example Images From the SDNET2018. (a) Decks – Cracked (b) Pavements – Cracked (c) Walls – Cracked (d) Decks – Non Cracked (e) Pavements – Non Cracked (f) Walls – Non Cracked (Figure is in color in online version of paper)*

Upon examination of **Figure 1**, it becomes apparent that cracked and non-cracked images exhibit discernible patterns that differentiate them from one another. Nevertheless, the dataset includes images that exhibit considerable complexity and challenge even to human observers. Such examples are illustrated in **Figure 2**.

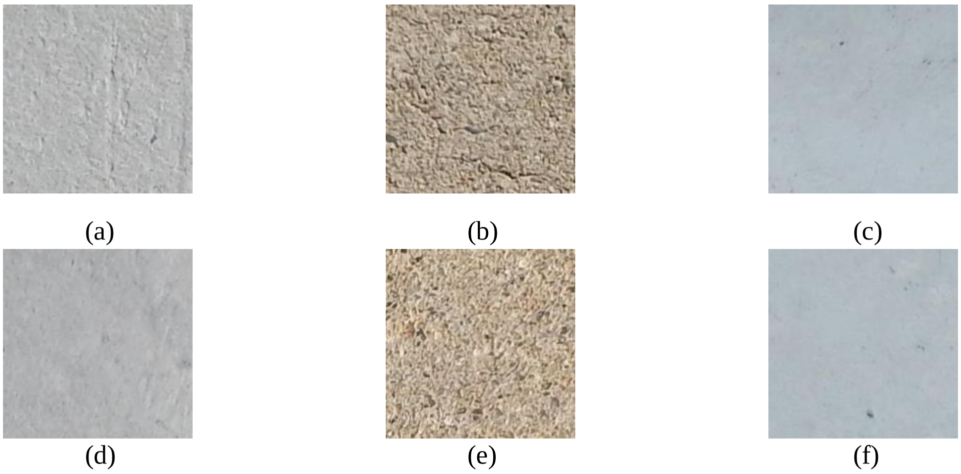


Figure 2. *Images with Considerable Complexity from the SDNET2018. (a) Decks – Cracked (b) Pavements – Cracked (c) Walls – Cracked (d) Decks – Non-Cracked (e) Pavements – Non-Cracked (f) Walls – Non-Cracked (Figure is in color in online version of paper)*

Cracked and non-cracked images exhibit differences in their linear or non-linear orientation perspectives. Additionally, these orientations reveal themselves at different angles. As can be seen from the example images, these orientations exhibit strong pattern changes (*edges*) in the images. These pattern changes could be enhanced, in other words, a mechanism could be employed to present strong pattern changes in these images. Thus, DWT was employed to enrich those pattern changes. The next subsection describes the DWT briefly and presents an example image which DWT applied.

2.2. 2D - Discrete Wavelet Transform

The Discrete Wavelet Transform (DWT) is a decomposition technique that enables the analysis of 1D and 2D signals in different frequency components. 2D-DWT allows for inspection of any image in multi-resolution. The 2D-DWT provides low and high frequency information about an image at different decomposition levels. In general, 2D-DWT divides images into four different frequency bands and each of them reveals different information about a given image:

1. LL – Low Frequency component of the image
2. LH – Horizontal Edges Enhanced
3. HL – Vertical Edges Enhanced

4. HH – Diagonal Edges Enhanced

While 1-Level of decomposition provides information about 3 different orientations, one advantage of 2D DWT is that it enables *n*-Level decomposition by decomposing the LL part of the image at each level, thus providing more substantial information about the image. The 2D-DWT algorithm initiates by selecting a wavelet basis function for the image at hand. The selected wavelet function is then used to decompose the image into its component frequency sub-bands. Given the long-standing history of 2D-DWT, numerous wavelet functions have been proposed in the literature, each of which enhances the LH, HL, and HH sub-bands differently [28]. An example 4-Level decomposition of a crack deck surface image is given in **Figure 3**.

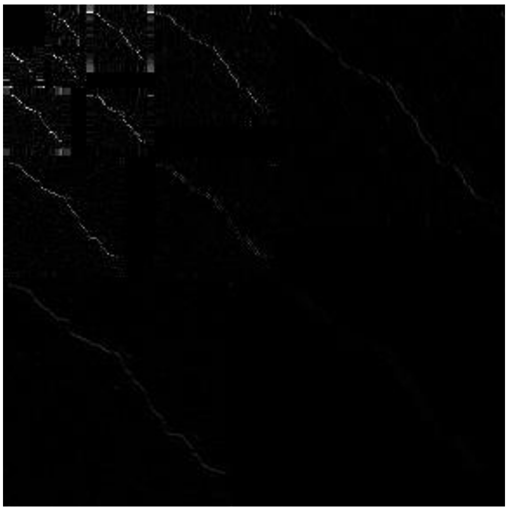
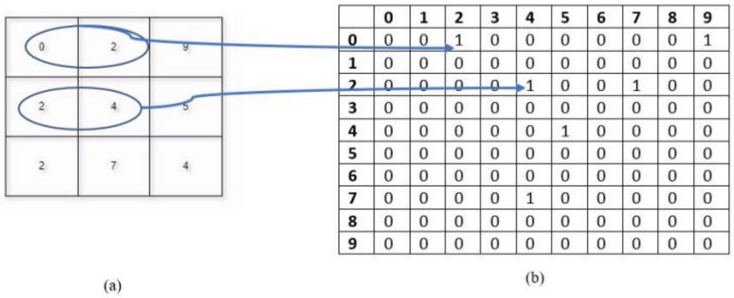


Figure 3. *4-Level Decomposition of Cracked Deck Image (Wavelet Function is Bior1.5)*

As illustrated in **Figure 3**, an increase in the level of decomposition leads to a more evident exposition of crack characteristics. Each decomposition in 2D-DWT can be used as raw features. However, the feature vector size may become excessively large for a given image, as the feature vector expands with each level of decomposition, in proportion to the image's shape. Another drawback of using 2D-DWT's results directly is the presence of unnecessary pixel information, as not every pixel is an important feature. Therefore, more meaningful features that summarize the image information into a single number could be effective for both interpretability and computational burden. For this reason, GLCM was employed in this study. The effectiveness of GLCM has been demonstrated in the literature [17, 25]. The next subsection gives brief information about GLCM to the reader.

2.3. Gray Level Co-Occurrence Matrix (GLCM)

GLCM is a statistical method that calculates correlation between pixels in a grayscale image. It evaluates the correlation between the gray level values of two pixels at a specific distance and angle in an image. It was proposed by Haralick [18]. The GLCM performs its operation using three parameters: a grayscale image, a distance parameter that determines how many pixels will contribute to correlation, and an angle parameter for which the correlation will be sought. It is capable of capturing texture changes effectively and presents this information in a number of properties: Dissimilarity, Correlation, Homogeneity, Contrast, ASM and Energy. General working mechanism and formula of each property is given in **Figure 4** and **Table 1** respectively.



0	2	9
2	4	5
2	7	4

(a)

	0	1	2	3	4	5	6	7	8	9
0	0	0	1	0	0	0	0	0	0	1
1	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	1	0	0	1	0	0
3	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	1	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	1	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0

(b)

Figure 4. *Working Mechanism of GLCM. (a) Gray values of the image. (b) Constructed GLCM of the image.*

Table 1. *GLCM Properties and Its' Respective Formulas.*

Properties	*Formula*
Dissimilarity	$\sum_{i,j=0}^{M-1} (Pij\|i-j\|)$
Correlation	$\sum_{i,j=0}^{M-1} Pij\left[\frac{(i-\mu_i)(j-\mu_j)}{(\sigma_i^2)(\sigma_j^2)}\right]$
Homogeneity	$\sum_{i,j=0}^{M-1} \frac{Pij}{1+(i-j)^2}$
Contrast	$\sum_{i,j=0}^{M-1} Pij\ (i-j)^2$
ASM	$\sum_{i=0}^{M-1}\sum_{j=0}^{M-1} P(i,j)^2$
Energy	$\sqrt{ASM}$

The present study employed 4-Level of 2D-DWT decomposition and for each level of decomposition properties which their equations are given in Table 1 were extracted for each image (LH, HL, HH). Pixels distance was selected as 5 and for angles of [0, π/2, π/3, π/4, π/6, 3π/4], the statistical features were extracted. Thus, rather than accepting whole decomposition as a feature, more discriminative and numerical features were extracted. These extracted features are suitable for a ML algorithm since they represent enough statistical information about both cracked and non-cracked images.

However, the classical ML algorithms may not be sufficient for capturing the underlying function required to discriminate between cracked and non-cracked images. Moreover, since the primary objective of this study is to classify both the surface material and the presence of cracks, more advanced AI approaches may be better suited for the task at hand. For these reasons, an ANN structure was employed, as it offers a more flexible structure that can be modified to fit the needs of the problem. The following subsection provides an overview of the background knowledge necessary for understanding ANNs.

2.4. Artificial Neural Networks (ANNs)

An ANN is an AI method that can often effectively capture the underlying discriminative properties of a feature set and is commonly used for tasks such as classification. ANNs consist of multiple layers, each with a specific number of neurons. Each neuron has an activation function that is triggered when certain threshold value is reached. Each layer in the ANN is fully interconnected with the subsequent layer and is updated based on the computed error rate of the network's output. This process of error backpropagation enables the network to adjust its weights and biases in a manner that minimizes the overall loss or error of the model during training. Due to the aforementioned characteristics, ANNs have often been compared to the structure and function of the human brain, as they both involve the processing of complex information through the use of interconnected units. A simple ANN structure is given in **Figure 5**.

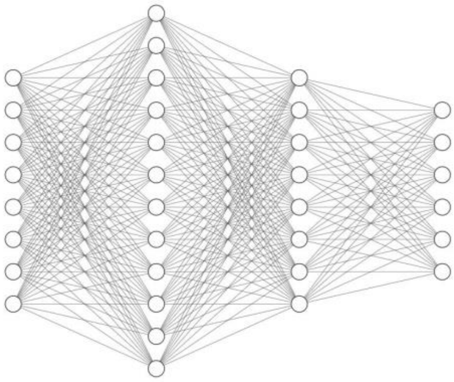


Figure 5. *An Example Artificial Neural Network Structure*

Figure 5 depicts the example architecture of an ANN with 8 input nodes and 6 output nodes, while the intermediary layers are referred to as hidden layers and may have varying numbers of neurons. Each neuron has an activation function that ensures the non-linearity through the network. Because of their easily modifiable structure and performance in classification tasks, ANNs were chosen for this study.

Each of the mentioned methods has its own set of parameters that must be determined prior to conducting the experiment. Thus, the next section defines the experimental settings for this study. Also, it presents the general overview of the proposed method.

3. Experimental Settings and Overview of the Proposed Method

The dataset used in this study is generally employed for classifying images as either cracked or non-cracked. However, in addition to the binary classification of surface cracks, determining the type of surface is also crucial for further analysis and emergency responses. To address this, the class labels in this study were modified to include not only cracked and non-cracked labels, but also information regarding the type of surface. To make labels suitable for the ANN outputs, they were modified using the Label Encoding. The encoded labels are given in **Table 2**.

Table 2. *Labels Modification*

Label Name	*Original Label*	*Encoded Label*
Decks – Cracked	0	[1 0 0 0 0 0]
Decks – Non - Cracked	1	[0 1 0 0 0 0]
Pavements – Cracked	2	[0 0 1 0 0 0]
Pavements – Non - Cracked	3	[0 0 0 1 0 0]
Walls - Cracked	4	[0 0 0 0 1 0]
Walls – Non – Cracked	5	[0 0 0 0 0 1]

As mentioned earlier, the overall distribution of the dataset is imbalanced, with significantly more non-cracked images than images with cracks. This data imbalance could potentially lead to poor results when training a model to classify crack images. To address the data imbalance, 2500 images were selected for each label to balance the number of samples in the dataset. The feature extraction process involved a 2D-DWT with a 4-Level decomposition using the Bior1.5 wavelet function. For each level of decomposition, the GLCM with parameters as specified in Section 2.3, was utilized to extract statistical properties of the resulting Wavelet coefficients. Finally, an ANN structure was constructed, with 432 inputs, 6 hidden layers, and 6 outputs. The activation function used for each hidden layer was the Hyperbolic Tangent (Tanh), while the output layer employed the SoftMax activation function to generate the probability of each label. The BinaryCrossentropy loss function and Adam optimizer were decided as hyperparameters. To prevent overfitting of the ANN, dropout and regularization techniques were employed on various layers. All features were normalized to ensure faster convergence of the ANN. Finally, the dataset was divided into train and test sets, with a ratio of 80% for training and 20% for testing. Parameters of the methods are detailed in **Table 3**.

Table 3. *Parameters of the Methods*

Method / Settings	*Parameter(s)*
Number of Images for Each Label	2500
2D DWT	4 Level Decomposition. Bior1.5 Wavelet Function for each level.
GLCM	Distance of 5 Pixels. Angles of [0, π/2, π/3, π/4, π/6, 3π/4]
ANN	6 Hidden Layers [512,256,128,256,64,32]
• Dropout	After 4th and 6th layers and ratio of 0.2 and 0.3 respectively.
• Regularization	First two layers have L1 and L2 regularization. The rest have L2 regularization except the final layer.
• Optimizer	Adam Optimizer – Learning Rate of 10^{-5}
• Number of Epoch	25
• Batch Size	8
• Loss Function	BinaryCrossentropy
• Train and Test set Ratio	80% - 20% respectively

3.1. Evaluation Metrics

The classical evaluation metrics for a classification problem were employed. The proposed method was evaluated in terms of accuracy, precision, recall, and F1 score. Additionally, the Area Under Curve (AUC) metric was employed, which serves as an indicator of the model's capacity to differentiate between classes. Moreover, the results of the method were presented in the form of a confusion matrix. The formulas for each metric are provided in **Table 4**.

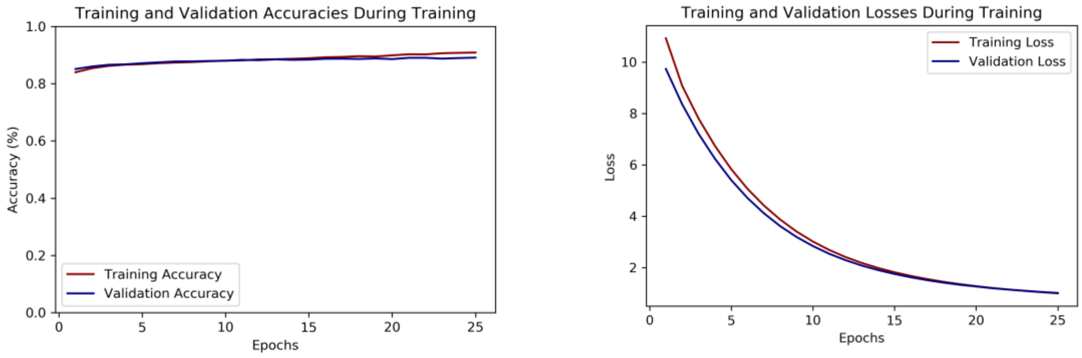
Table 4. *Evaluation Metrics*

Metric	*Formula*
Accuracy	$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$
Precision	$Precision = \frac{TP}{TP + FP}$
Recall	$Recall = \frac{TP}{TP + FN}$
F_1 Score	$F_1 = \frac{2}{\frac{1}{Recall} + \frac{1}{Precision}}$

To clarify the abbreviations used in **Table 4**, we draw an analogy between surface types, specifically Wall, and two categories of inner classes: cracked and non-cracked. The abbreviation 'TP' stands for True Positive, which is the number of correct predictions made by the model. This means that the model predicts Wall-Crack and the sample is actually Wall-Crack. Similarly, 'TN' stands for True Negative, which is the number of negative predictions made by the model. In this case, the model predicts the sample as non-Wall and non-cracked, and the sample is actually non-Wall and non-cracked. On the other hand, 'FP' and 'FN' are abbreviations for False Positive and False Negative, respectively. For False Positive, the model predicts Wall-Crack when the sample is actually not Wall-Crack. Finally, False Negative represents the number of predictions where the model predicts the sample as non-Wall and non-cracked when the sample is actually Wall-Crack.

4. Experimental Results

We first inspect the results by examining the accuracy and loss metrics during training for both the train set and test set. The behavior of accuracy and loss metrics are given in Figure 6 side by side.



(a) (b)

Figure 6. *Accuracies and Losses Per Epoch. (a) Accuracy (b) Loss (Figure is in color in online version of paper)*

Figure 6 (a) shows that the accuracy of both the train set and the test set (used as the validation set) exhibited high performance. Although heavy regularizations were employed, the constructed ANN model exhibited signs of overfitting at the end of the training phase. However, the same deductions cannot be made for **Figure 6 (b)**, which shows the model's loss performance. Both the training and validation set losses tended to smoothly converge to a minimum. Final accuracy rates for both the train set and the validation set were given in **Table 5**.

Table 5. *Train and Test Set Performances of the Model*

Set	*Accuracy*
Train	92.19%
Test	89.12%

To gain a comprehensive understanding of the model, additional metrics as specified in **Table 4** were assessed on both the train and test sets following the completion of the model's training process. The overall results in terms of metrics are given **Table 6.**

Table 6. *The Overall Results for the Model*

Set	*Metric*	*Result*
Train	Precision	83.08%
	Recall	66.72%
	F1 Score	73.88%
	AUC	96.80%
Test	Precision	72.33%
	Recall	56.14%
	F1 Score	63.07%
	AUC	93.69%

Upon examining the results provided in **Table 6**, it is evident that the model exhibits relatively better performance on the train set as compared to the test set. One of the apparent reasons for this could be attributed to the number of samples used during the training phase of the model. Increasing the amount of data available could be one solution to this overfitting problem. It is known that providing an ANN structure with a larger amount of data often results in improved performance. The model’s performance on precision was higher than other metrics which means that out of all the samples, 83% of them were classified correctly. The precision rate on the test set was 72.3%, which is a relatively good performance for such a challenging task. Finally, the model's performance was further evaluated using a confusion matrix, which is shown in **Figure 7**.

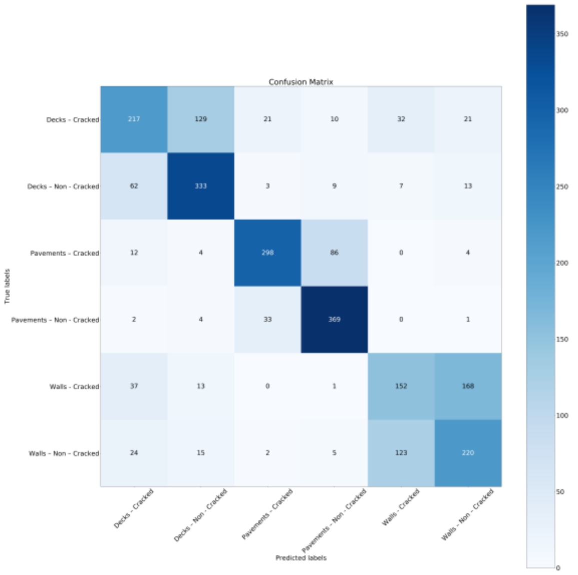


Figure 7. *Confusion Matrix for the Model (Figure is in color in online version of paper)*

Figure 7 highlights one of the primary factors that may have contributed to the relatively low performance of the model on the test set. Specifically, **Figure 7** indicates that while the model is generally effective at localizing surface materials, its ability to discriminate between inner-class categories appears to be not optimal. To provide a specific illustration, the model demonstrates a proficient capability in distinguishing between Wall and Pavement classes. However, the model appears to face considerable difficulty in effectively discerning between the cracked and non-cracked variants of the Wall surface category. As shown in **Figure 2**, hard examples of the dataset are one possible reason of dispersed inner class results in **Figure 7**. Despite the identified limitations, the findings suggest that the model's overall performance was satisfactory for the intended application.

To thoroughly assess the effectiveness of the proposed method, a quantitative analysis was conducted. This analysis involved a specific comparison of its performance against established methods documented in the existing literature, with a particular emphasis on CNNs. The comparison was based on the accuracy metric, as not all metrics presented in this paper were available in the literature within this context. Additionally, this comparison was limited to the SDNET 2018 dataset.

The methods introduced by Slonski [29] provide a comprehensive understanding of the behavior of Convolutional Neural Networks (CNNs) in the context of classification, specifically when applied to the SDNET 2018 dataset. Slonski's approach involved employing CNNs for the purpose of classifying cracks within the SDNET 2018. The method presented in this paper not only accomplishes crack classification but also extends its capability to discern the specific type of surface. Consequently, our proposed method demonstrates superior performance compared to the conventional binary crack classification achieved in [29]. Another strategy put forth by Chianese et al. [30] involves Transfer Learning. Their approach was utilization of three pre-established CNN architectures: AlexNet, Inception-V3, and ResNet-101According to the methodology presented in [30], our approach increased the performance. A tabulated representation of this comparative analysis is provided in Table 7.

Table 7. *Comparison with CNNs Approaches*

Method	*Accuracy*
Slonski,M. [29] – From Scratch CNN	86.00%
Slonski,M. [29] – Pretrained VGG16 – Data Augmentation	88.00%
Chianese, R. et al [30] - AlexNet	87.30%
Chianese, R. et al [30] – Inception-V3	84.67%
Chianese, R. et al [30] – ResNet-101	86.00%
Proposed Method	89.12%

5. Conclusion

Structural deformation is a natural phenomenon that occurs in most structures over time. These deformations typically manifest in the exterior materials of structures, such as roads and buildings. Persistent monitoring of structural degradation is of paramount importance in ensuring the safety and reliability of structures. Failure to regularly inspect and address such deformations can pose a serious threat to both the economy and, more importantly, human lives. Furthermore, merely detecting cracks may not be sufficient for taking appropriate actions, as the type of surface material can also significantly influence the structural degradation process. Nevertheless, conducting regular inspections of structures demands a significant number of human resources, which is often impractical to implement in real-world scenarios.

Given this challenge, there is a need for more practical solutions to address the issue, including advancements in technology. In this regard, AI solutions are employed. The spectrum of AI methods, that can be categorized as ML algorithms, is vast and varies from classical ML algorithm to more advanced techniques such as CNNs. The classical ML algorithms work based on the features that are extracted from a given image. However, they may not produce acceptable performance since they are not flexible in modification. Also, they are bound by the quality of the extracted features. On the other hand, mechanisms that do not require human intervention, such as CNNs, are both computationally expensive and may produce ineffective features.

For these challenges in mind, the present study proposed an end-to-end ML approach for classifying the cracks and non-cracked images and material of the surface parallel. For the feature extraction part, 2D-DWT was utilized, as it is known that DWT enhances the frequency changes in an image. For the ML part, an ANN structure was employed since it provides a more flexible mechanism and allows us to control all aspects of it.

The study could be enhanced in several ways. One possible enhancement could be using different Wavelet functions to assess to performance of the model more. Another proper improvement is to be adding color feature to enhance the approach, since the color of each surface differs.

Different from the other studies in the literature, the current work classifies both cracked and non-cracked images and surface material at the same time. The results suggested that the proposed mechanism achieved an acceptable level of accuracy, although the task at hand is difficult.

Declaration of interest

The authors declares that there is no conflict of interest.

Acknowledgements

The authors gratefully acknowledge the dataset provided by Utah State University.

Nomenclature

Abbreviations

AI	Artificial Intelligence
ANN	Artificial Neural Networks
CIP	Conventional Image Processing
CNN	Convolutional Neural Networks
DWT	Discrete Wavelet Transforms
GLCM	Gray Level Co-Occurence Matrix
NSGA	Hough Transform
HT	Hyperbolic Tangent

LBP	Local Binary Pattern
ML	Machine Learning
PCA	Principal Component Analysis
SVM	Support Vector Machines

References

[1] D. Ai, G. Jiang, S.-K. Lam, P. He, and C. Li, "Computer vision framework for crack detection of civil infrastructure—A review," *Engineering Applications of Artificial Intelligence*, 117 (2023) 10547; 10.1016/j.engappai.2022.105478.

[2] E. Mohammed Abdelkader, "On the hybridization of pre-trained deep learning and differential evolution algorithms for semantic crack detection and recognition in ensemble of infrastructures," *Smart and Sustainable Built Environment*, 11(3) (2022) 740–764; 10.1108/SASBE-01-2021-0010.

[3] L. Attard, C. J. Debono, G. Valentino, M. Di Castro, A. Masi, and L. Scibile, "Automatic crack detection using mask R-CNN," In: 11th international symposium on image and signal processing and analysis (ISPA), IEEE, (2019), 152–157.

[4] G. Lu, X. He, Q. Wang, F. Shao, J. Wang, and X. Zhao, "MSCNet: A Framework with a Texture Enhancement Mechanism and Feature Aggregation for Crack Detection," *IEEE Access*, 10 (2022) 26127–26139; 10.1109/ACCESS.2022.3156606.

[5] Z. Xu *et al.*, "Pavement crack detection from CCD images with a locally enhanced transformer network," *International Journal of Applied Earth Observation and Geoinformation*, 110 (2022) 102825; https://doi.org/10.1016/j.jag.2022.102825.

[6] P. Gupta and M. Dixit, "Image-based crack detection approaches: a comprehensive survey," *Multimedia Tools and Applications*, 81(28) (2022) 40181–40229; https://doi.org/10.1007/s11042-022-13152-z.

[7] L. Ali, F. Alnajjar, W. Khan, M. A. Serhani, and H. Al Jassmi, "Bibliometric analysis and review of deep learning-based crack detection literature published between 2010 and 2022," *Buildings*, 12(4) (2022) 432; https://doi.org/10.3390/buildings12040432.

[8] N. Safaei, O. Smadi, A. Masoud, and B. Safaei, "An automatic image processing algorithm based on crack pixel density for pavement crack detection and classification," *International Journal of Pavement Research and Technology*, 15(1) (2022) 159–172; https://doi.org/10.1007/s42947-021-00006-4.

[9] A. Mohan and S. Poobal, "Crack detection using image processing: A critical review and analysis," *Alexandria Engineering Journal*, 57(2) (2018) 787–798; https://doi.org/10.1016/j.aej.2017.01.020.

[10] A. Ahmadi, S. Khalesi, and A. Golroo, "An integrated machine learning model for automatic road crack detection and classification in urban areas," *International Journal of Pavement Engineering*, 23(10) (2022) 3536–3552; https://doi.org/10.1080/10298436.2021.1905808.

[11] J. Illingworth and J. Kittler, "A survey of the hough transform," *Computer Vision, Graphics, and Image Processing*, 44(1) (1988) 87–116; https://doi.org/10.1016/S0734-189X(88)80033-1.

[12] C. D. Sutton, "Classification and regression trees, bagging, and boosting," *Handbook of statistics*, 24 (2005) 303–329; https://doi.org/10.1016/S0169-7161(04)24011-1.

[13] V. Jakkula, "Tutorial on support vector machine (svm)," *School of EECS, Washington State University*, 37(2.5) (2006).

[14] G. Zhang, X. Huang, S. Z. Li, Y. Wang, and X. Wu, "Boosting Local Binary Pattern (LBP)-Based Face Recognition," In: Chinese Conference on Biometric Recognition, (2014) 179–186.

[15] R. Bro and A. K. Smilde, "Principal component analysis," *Analytical Methods*, 6(9) (2014) 2812–2831; https://doi.org/10.1039/C3AY41907J.

[16] C. Chen, H. Seo, C. H. Jun, and Y. Zhao, "Pavement crack detection and classification based on fusion feature of LBP and PCA with SVM," *International Journal of Pavement Engineering*, 23(9) (2022) 3274–3283; https://doi.org/10.1080/10298436.2021.1888092.

[17] N.-D. Hoang, "Automatic detection of asphalt pavement raveling using image texture based feature extraction and stochastic gradient descent logistic regression" *Automation in Construction*, 105 (2019) 102843; https://doi.org/10.1016/j.autcon.2019.102843

[18] R. M. Haralick, K. Shanmugam, and I. Dinstein, "Textural Features for Image Classification," *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-3 (6) (1973) 610–621;10.1109/TSMC.1973.4309314.

[19] Y. LeCun and Y. Bengio, "Convolutional networks for images, speech, and time series," *The handbook of brain theory and neural networks*, 3361 (10) (1995) 255-258, 1995.

[20] R. Rojas "The backpropagation algorithm," In: Neural Networks. Springer, Berlin, Heidelberg (1996) 149–182.

[21] C. Liu and B. Xu, "A night pavement crack detection method based on image-to-image translation," *Computer-Aided Civil and Infrastructure Engineering*, 37(13) (2022) 1737–1753; https://doi.org/10.1111/mice.12849.

[22] Q. Yang, W. Shi, J. Chen, and W. Lin, "Deep convolution neural network-based transfer learning method for civil infrastructure crack detection," *Automation in Construction*, 116 (2020) 103199; https://doi.org/10.1016/j.autcon.2020.103199.

[23] X. Zhang, D. Rajan, and B. Story, "Concrete crack detection using context-aware deep semantic segmentation network," *Computer-Aided Civil and Infrastructure Engineering,* 34(11) (2019) 951–971; https://doi.org/10.1111/mice.12477.

[24] A. Chordia, S. Sarah, M. K. Gourisaria, R. Agrawal, and P. Adhikary, "Surface Crack Detection Using Data Mining and Feature Engineering Techniques," In: IEEE 4th International Conference on Computing, Power and Communication Technologies (GUCON), (2021) 1–7.

[25] L. Cong, J. Shi, T. Wang, F. Yang, and T. Zhu, "A method to evaluate the segregation of compacted asphalt pavement by processing the images of paved asphalt mixture," *Construction and Building Materials*, 224 (2019) 622–629; https://doi.org/10.1016/j.conbuildmat.2019.07.041.

[26] S. Dorafshan, R. J. Thomas, and M. Maguire, "SDNET2018: An annotated image dataset for non-contact concrete crack detection using deep convolutional neural networks," *Data in brief,* 21 (2018) 1664–1668; https://doi.org/10.1016/j.dib.2018.11.015.

[27] M. Mirbod and M. Shoar, "Intelligent Concrete Surface Cracks Detection using Computer Vision, Pattern Recognition, and Artificial Neural Networks," *Procedia Computer Science,* 217 (2023) 52–61; https://doi.org/10.1016/j.procs.2022.12.201.

[28] I. Daubechies, "Ten lectures on wavelets", SIAM, 1992.

[29] M. Słoński, "A comparison of deep convolutional neural networks for image-based detection of concrete surface cracks," *Computer assisted methods in Engineering and Science,* 26 (2) (2019) 105–112; http://dx.doi.org/10.24423/cames.267.

[30] R. Chianese, A. Nguyen, V. Gharehbaghi, T. Aravinthan, and M. Noori, "Influence of image noise on crack detection performance of deep convolutional neural networks," In: Proceedings of the 10th International Conference on Structural Health Monitoring of Intelligent Infrastructure (SHMII 10), (2021) 1681 – 1688.

Advances in Artificial Intelligence Research (AAIR)
ISSN 2757-7422
www.dergipark.com/aair/

Vol. 3 (No. 2), pp. 96-107, 2023
doi: 10.54569/aair.1329048
Published online: Oct 23, 2023

Analyzing the Impact of Augmentation Techniques on Deep Learning Models for Deceptive Review Detection: A Comparative Study

Anusuya Krishnan [1,*], Kennedyraj Mariafrancis [2,]

[1] United Arab Emirates University, UAE
[2] Noorul Islam University, India

Abstract

Deep Learning has brought forth captivating applications, and among them, Natural Language Processing (NLP) stands out. This study delves into the role of the data augmentation training strategy in advancing NLP. Data augmentation involves the creation of synthetic training data through transformations, and it is a well-explored research area across various machine learning domains. Apart from enhancing a model's generalization capabilities, data augmentation addresses a wide range of challenges, such as limited training data, regularization of the learning objective, and privacy protection by limiting data usage. The objective of this study is to investigate how data augmentation improves model accuracy and precise predictions, specifically using deep learning-based models. Furthermore, the study also conducts a comparative analysis between deep learning models without data augmentation and those with data augmentation. Our proposed method, combining RoBERTa with data augmentation, achieves a remarkable 94% accuracy, underscoring the significant effectiveness of this approach in improving NLP model performance.

Keywords: *Deep learning techniques; deceptive review detection; augmentation.*

1. Introduction

Text augmentation techniques play a vital role in Natural Language Processing (NLP) tasks by expanding the diversity and quantity of training data. With the increasing availability of large text corpora and the advancements in deep learning models, researchers and practitioners have recognized the significance of text augmentation in improving the performance of various NLP applications. Text augmentation involves generating new instances of text by applying a series of linguistic transformations while preserving the original meaning and context. These transformations can range from simple operations such as synonym replacement and random word deletion to more complex techniques like paraphrasing and back-translation. By augmenting the training data, models can learn to generalize better, capture a wider range of language patterns, and become more robust to variations in input [1].

In recent years, text augmentation techniques have gained considerable attention and have been successfully applied to a wide range of NLP tasks, including sentiment analysis, text classification, machine translation, and named entity recognition, among others. Researchers have explored various augmentation strategies, leveraging linguistic rules, pre-trained language models, and domain-specific knowledge to generate augmented data that mimics the characteristics of real-world text [2].

The benefits of text augmentation extend beyond the augmentation of model performance. It can also help mitigate data scarcity challenges, particularly in low-resource domains, where collecting a large, annotated dataset is often impractical or expensive. Furthermore, text augmentation can address issues related to data bias, as it can help balance the representation of different classes and reduce the risk of overfitting to specific patterns in the training data. Despite the widespread use of text augmentation in NLP, there exists a compelling need for a comprehensive understanding of its impact on model performance. Diverse factors, including the selection of augmentation techniques, the extent of augmentation, and the intricate interplay between augmentation and model architecture, can collectively shape overall effectiveness. Additionally, a critical examination of the constraints and potential risks linked with text augmentation, such as the introduction of synthetic artifacts or inadvertent amplification of inherent biases in the original data, remains imperative [3-6].

In this study, we aim to provide a comprehensive analysis of text augmentation techniques in the context of NLP tasks. We will explore the existing augmentation methods, categorize them based on their underlying principles, and discuss their advantages and limitations. Furthermore, we will conduct a comparative evaluation of without augmentation and with augmentation techniques on NLP tasks, investigating their impact on model performance, generalization, and robustness.

The following sections of this paper provide a comprehensive examination of the research findings. Section 2 presents a concise overview of relevant works in the field, drawing insights from existing literature. Moving forward, Section 3 delves into the background and intricacies of our proposed machine learning approach

*Corresponding author
E-mail address: anusuyababy18@gmail.com

Received: 18/Jul/2023; Received in revised form: 06/Sep/2023; Accepted: 18/Oct/2023.

designed for detecting deceptive reviews. The section elaborates on the methodology and underlying principles of our approach. Section 4 focuses on presenting the detailed results and analysis conducted to evaluate the accuracy of our model in identifying deceptive reviews. These experiments offer valuable insights into the efficacy of our approach. Finally, in Section 5, the paper concludes by summarizing the key findings and contributions of our work, while also outlining potential avenues for future research and development in this domain.

2. Related works

The development of universal data augmentation techniques in the field of natural language processing (NLP) has encountered challenges due to the complexity of devising generalized rules for transforming languages. Our survey introduces various methodologies for implementing data augmentation in textual data. While some prior research has proposed methods for augmenting data in NLP, there remains a noticeable gap in a comprehensive exploration of this area. Notably, one study generated new data by translating sentences into French and then back into English [7]. Furthermore, alternative approaches encompass introducing noise to the data to enhance smoothness and employing predictive language models to replace synonyms [8-10]. However, despite the validity of these techniques, their practical adoption is limited due to substantial implementation costs in relation to the performance improvements they yield. Another study laid the groundwork for incorporating formal causal language into data augmentation, involving the use of structured causal models and the process of abduction, action, and prediction to generate counterfactual instances. Their experiments encompass aligning phrases within sequences in neural machine translation to extract counterfactual substitutions [10].

One of the prominent challenges when applying machine learning methods, including artificial neural networks, to small datasets is the issue of learning stability. This instability can manifest in a strong dependence on parameter selection, training batch order, and other factors [11]. It also encompasses challenges such as overfitting and the inability to achieve effective generalization. As a result, the volatility inherent in small datasets can lead to inconsistent outcomes when employing models of similar architecture. This, in turn, can limit generalization and accuracy, impeding overall performance [12]. Furthermore, ensuring the reproducibility of results can become problematic, even when employing the same architecture and dataset, making comparisons, enhancements, and optimizations more challenging [13]. Previous studies have made efforts to address the stability challenge in machine learning with small datasets using diverse techniques [14]. These methods include k-fold cross-validation, ensemble methods, Radial-Basis Function (RBF) neural networks, and other approaches [15-17]. While many of these methods have demonstrated success in specific applications, their applicability across various datasets and problems remains uncertain [18-20]. This uncertainty arises from the specific architectural requirements and assumptions concerning the data distribution.

In a previous study, the author introduced an innovative approach to ensemble learning using augmentation for the detection of stance and fake news [22]. Their method involves data augmentation, which entails creating new training instances sharing the same true labels as their source instances [23]. Although data augmentation is widely embraced in computer vision (CV) as a cornerstone of robust predictive performance, its exploration in the realm of natural language processing (NLP) has been comparatively limited. In the context of NLP, it is often seen as an incremental improvement, affording modest but consistent performance enhancements. This distinction can be attributed to the inherent characteristics of textual data, such as polysemy, which renders the formulation of label-preserving transformations notably more intricate [24]. Another author proposed augmentation rules rooted in syntactic heuristics including strategies like inversion, where subject and object are swapped in sentences, and passivization, which transforms hypothesis sentences in premise-hypothesis natural language inference (NLI) pairs into their passive counterparts [25]. Another noteworthy approach by researchers involves constructing a knowledge graph from the extensive input context for abstractive summarization. This graph facilitates semantic swaps that uphold overall coherence [26].

In recent developments, some studies highlighted the limited benefits of supervised syntactic parsing in contemporary pre-training and fine-tuning pipelines with large language models [27]. Some researchers extended these ideas to domains such as molecules, genomics, therapeutics, and healthcare, proposing the integration of such structures with text data to potentially enhance text representations [28]. They have introduced an edge augmentation technique that exposes graph neural networks (GNNs) to likely yet nonexistent edges while limiting exposure to existing but improbable ones [29]. This augmentation results leads to a 5% average accuracy enhancement across six prominent node classification datasets. Building upon this, researchers demonstrate the effectiveness of adversarially controlled node feature augmentation for graph classification [30]. Similarly, another author created an embedding graph to enforce coherence between predictions from strongly and weakly augmented data [31].

Another author delves into the application of data augmentation (DA) for the detection of stance and fake news. In the initial segment of their study, they examine the impact of diverse DA techniques on the efficacy of standard classification algorithms. Their research capitalizes on the insights gleaned from this analysis to introduce a novel ensemble learning methodology rooted in augmentation. Their approach harnesses text augmentation to enrich the diversity and accuracy of the base learners, thereby elevating the predictive capabilities of the ensemble [32]. Furthermore, they investigated to address the challenge of class imbalance, a prevalent issue in the realm of stance and fake news detection that often leads to biased models. Also, they empirically demonstrate how text augmentation can be instrumental in mitigating moderate and severe class imbalances, elucidating its potential in rectifying this problem. Another study conducted an analysis of small datasets poses several significant challenges, primarily due to the limited sampling of characteristic patterns. As a result, drawing confident conclusions about the unknown distribution becomes elusive, leading to reduced statistical confidence and increased errors. However, small datasets can be crucial in scenarios involving novel or rare conditions where large amounts of data are unavailable or yet to be accumulated [33-34].

On the other hand, unsupervised machine learning methods have demonstrated effective capabilities in reducing dimensionality and eliminating redundancy in the observable parameter space [34]. These methods have played a crucial role in analyzing and identifying characteristic patterns and trends in complex data, including constrained datasets. Importantly, these methods do not rely on labeled data with known outcomes and can be applied with smaller sample sizes [35]. We believe that these characteristics make unsupervised machine learning methods suitable for analyzing early and rare conditions, scenarios, and situations where large amounts of confidently labeled data have not yet been accumulated. Furthermore, these methods enable the aggregation of data for later stages of statistical analysis using conventional techniques.

To overcome the challenges associated with analyzing small datasets, a proposed solution involves the utilization of different augmentation techniques. These techniques aim to identify characteristic structures within the input data, effectively addressing both the issues of limited labels and training instability when working with minimal datasets. By examining the latent representations of the augmented data, it becomes possible to identify underlying structures that can be used to generate new data points.

3. Methodology

In this section, we present a comprehensive exposition of our proposed framework, drawing comparisons with existing methodologies. The primary objective of this research is to delve into the impact of data augmentation on enhancing model accuracy and the precision of predictions, with a specific focus on deep learning-based models. Additionally, this study undertakes a comparative analysis between deep learning models that do not employ data augmentation and those that integrate this technique. The overall architecture of the proposed methodology can be observed in **Figure 1**.

3.1. Data Collection

In our research, we utilized the “Deceptive opinion spam corpus” dataset, which is publicly available on Kaggle, comprises a total of 1600 reviews [1]. These reviews are divided into two categories: 800 truthful reviews and 800 fake reviews. The dataset focuses specifically on the top twenty hotels in Chicago. The reviews within this dataset were sourced from platforms such as TripAdvisor and Amazon Mechanical Turk. Each review entry in the dataset contains the following information: the review label indicating its authenticity (whether it is fake or truthful), the corresponding hotel name, the sentiment or polarity of the review classified as positive or negative, the review source, which can either be TripAdvisor or Mechanical Turk, and lastly, the actual review text itself.

3.2. Data Preprocessing

In machine learning tasks, data preprocessing plays a critical role, especially when dealing with unstructured data. It involves a diverse set of techniques aimed at cleaning and refining the data, which includes removing punctuation, URLs, stop words, converting to lowercase, tokenization, stemming, and lemmatization. These techniques effectively eliminate irrelevant information and prepare the data for feature extraction. Tokenization, a fundamental technique in natural language processing, breaks down the text into smaller units referred to as tokens. These tokens can encompass alphanumeric characters, punctuation marks, or special characters. For example, the sentence “the desert is tasty” would be tokenized into “the,” “desert,” “is,” and “tasty.” Stop words, such as articles, conjunctions, prepositions, and pronouns, are commonly used words in everyday English. These words lack significant meaning and are typically removed during the preprocessing stage. Lemmatization is the process of transforming tokenized words into their base or root forms to enhance human comprehension. It reduces words to their common root form, effectively eliminating variations in tense or form. For instance, words like “dancing,” “danced,” and dancer” would all be reduced to “dance.” In this paper, we utilize lemmatization as an integral part of our data preprocessing approach.

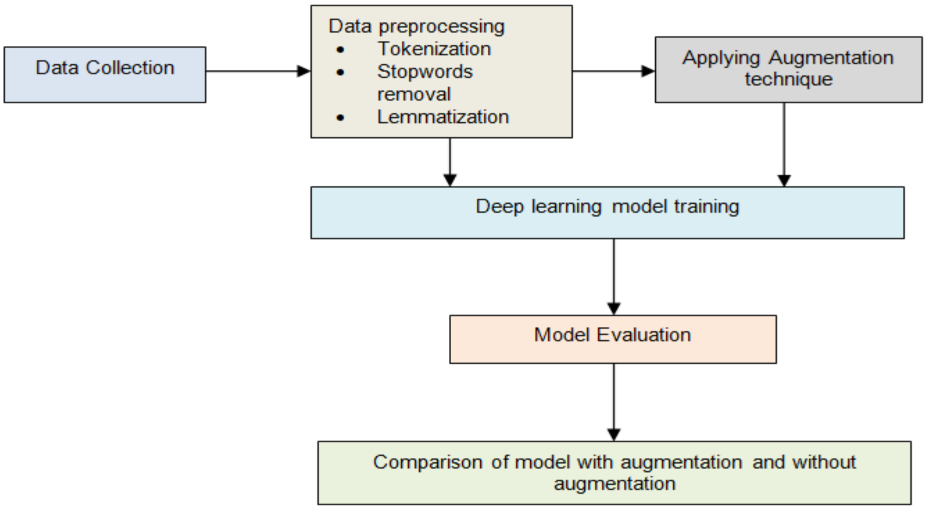

Data Collection
Data preprocessing
Tokenization
Stopwords removal
Lemmatization
Applying Augmentation technique
Deep learning model training
Model Evaluation
Comparison of model with augmentation and without augmentation

Figure 1. *Proposed Methodology*

3.3. Augmentation Technique

Text augmentation is an effective method utilized to amplify the diversity and volume of a text dataset. It entails employing various transformations to the text, resulting in augmented versions that can improve the performance of machine learning models. Commonly employed techniques for text augmentation include synonym replacement, random insertion, random deletion, and random swap. The synonym replacement technique entails substituting words in the text with their synonyms. This expands the vocabulary and introduces semantic variations, thereby enriching the text. On the other hand, random insertion involves the random insertion of words at different positions within the text. This technique increases the length of the text and introduces noise, which enhances the model's ability to handle variations in input length. Random deletion, as the name suggests, involves the removal of random words from the text. This simulates missing or incomplete information, compelling the model to learn more resilient representations. Lastly, random swap entails swapping the positions of two randomly selected words within the text. This introduces different word orderings, thereby enhancing the model's capacity to accommodate variations in sentence structure. The example of four methods in data augmentation is provided in the **Table 1**.

Table 1. *Example of data augmentation of four methods*

Example	Sentences
Original data	I love to eat pizza.
Synonym replacement	I adore indulging in pizza.
Random insertion	I absolutely love to eat delicious pizza every weekend.
Random swap	I pizza to eat love.
Random deletion	I to eat pizza.

Following data cleaning, we partitioned the dataset into an 80:20 ratio using the train-test split method. Subsequently, we exclusively applied all four augmentation techniques to the training data.

3.4. Deep learning model training

Then we employed different deep learning techniques. Deep learning model training for text data using word embedding is a powerful technique that enables the effective representation and analysis of textual information. By employing word embeddings, which are dense vector representations of words, the model can capture semantic relationships and contextual information. The process of training a deep learning model for text data using word embedding involves several steps. First, the text data is preprocessed by removing unnecessary characters, tokenizing the text into individual words or subword units, and performing other necessary preprocessing steps. Next, we split the dataset into 80:20 using train test split. Following that, augmentation technique applied only on training dataset. Then word embedding technique is chosen, such as Word2Vec or GloVe. These techniques create word embeddings by considering the co-occurrence statistics of words in a large corpus. Alternatively, pretrained word embeddings can be utilized, which have been trained on extensive datasets and capture general language semantics. Once the word embeddings are obtained, they are used to represent the text data. Each word in the input text is mapped to its corresponding word embedding vector. The resulting sequence of word embeddings forms the input to the deep learning model. In this study, we utilized different deep learning models like recurrent neural networks (RNNs) such as LSTM and transformer-based models like BERT, DistilBERT, XLNET, RoBERTa [36-40]. These models take the word embeddings as input and employ layers to capture the hierarchical structure and dependencies within the text data.

During training, the model parameters are optimized by backpropagation and gradient descent methods. The model is presented with the input word embeddings, and the predicted outputs are compared to the true labels or targets. The difference between the predictions and the targets is measured using a suitable loss function, such as categorical cross-entropy for multi-class classification or binary cross-entropy for binary classification. The model parameters are updated iteratively based on the gradients of the loss function with respect to the model's parameters. This process continues for multiple epochs, where each epoch represents a complete pass through the training data. The model learns to adjust its parameters to minimize the loss and improve its performance on the given task. Hyperparameter tuning is an essential step to optimize the model's performance. Parameters such as learning rate, batch size, number of layers, and regularization techniques can be adjusted to improve the model's generalization ability and prevent overfitting. In this paper, we configured the hyperparameters for transformer models as follows: learning rate = 5e-5, weight decay = 0, number of epochs = 20, and batch size = 64. Additionally, for LSTM models, we utilized dropout rate = 0.2, number of layers = 64, activation = sigmoid, optimizer = Adam, and epochs = 20.

3.5. Model Evaluation

The evaluation of a deep learning model is a vital stage in determining its performance and effectiveness. It revolves around assessing the model’s ability to generalize to new and unseen data, as well as its accuracy in accomplishing the designated task. In this research, we have selected accuracy, F1-score, and the loss function as our performance metrics. Accuracy is a widely used metric for assessing classification models, representing the percentage of correctly predicted labels. Additionally, the loss, exemplified by cross-entropy loss, quantifies the degree of prediction error in the model. In this study, we used binary cross-entropy loss function for binary classification problems, where the goal is to assign one of two classes to each input sample. By monitoring the loss throughout the training process, we gain insights into the model's convergence and progress.

3.6. Hardware and software used:

We utilized a high-performance Linux Ubuntu 18.04 server with 40 CPU cores, powered by an Nvidia DGX-1, and equipped with 8 NVIDIA Tesla V100 GPUs, each boasting 32 GB of memory. This server also featured a web-based multi-user concurrent job scheduling system. All experiments and training were carried out using Python 3.8.16. The libraries used for Deep Learning, Data Processing, and Data Visualization, including tensorflow-gpu v2.3.1, keras v2.4.3, SciPy v1.16.4, NumPy, Pandas, Matplotlib, and Seaborn, were integrated into the environment.

4. Experimental results

In this section, we delve into the experimental assessment of our proposed approach and provide an analysis of the acquired results. To conduct this evaluation, our primary focus during the comparative performance analysis was the utilization of the Opinion spam dataset, comprising a modest 1,600 reviews. Despite its relatively small size, this dataset presented a formidable challenge when implementing deep learning models.

4.1. Original Data without Augmentation

The original data undergoes a preprocessing phase to ensure cleanliness and standardization. This involves various techniques such as tokenization, lowercasing, punctuation removal, handling special characters, and employing methods like stopword removal and lemmatization. Once the data is preprocessed, we split the dataset into 80:20 using train test split. Then it is transformed into word embeddings, which are compact vector representations capturing the semantic and contextual information of words. Each word in the preprocessed data is associated with its corresponding word embedding vector, resulting in sequences or matrices of word embeddings. These word embeddings act as input features for deep learning models. The models leverage their layers and parameters to make predictions and conduct tasks such as text classification. By utilizing the learned representations from the word embeddings, the deep learning models effectively interpret and analyze the textual data, enabling accurate predictions and effective text classification. We have used accuracy and loss metrics for model evaluation. **Table 2** shows the deep learning model performance without augmentation.

Table 2. *Deep learning models without Augmentation.*

Deep learning model	Library	Training		Testing	
		Accuracy	Loss	Accuracy	Loss
LSTM	Word2Vec	92%	0.05	86%	0.49
LSTM	GloVe	93%	0.08	82%	0.82
BERT	Transformer	91%	0.09	78%	0.77
DistilBERT	Transformer	96%	0.03	85%	0.76
XLNET	Transformer	95%	0.06	88%	0.43
RoBERTa	Transformer	97%	0.01	91%	0.31

The table provides an overview of various deep learning models trained on a specific task, including their architecture, word embedding library, training accuracy, training loss, testing accuracy, and testing loss. The LSTM models with Word2Vec and GloVe libraries achieved accuracies ranging from 92% to 93% during training and 86% to 82% during testing. The BERT model with the Transformer library achieved a training accuracy of 91% but dropped to 78% on the testing dataset. The DistilBERT model achieved a high training accuracy of 96% and 85% accuracy on the testing dataset. The XLNET model achieved accuracies of 95% during training and 88% during testing. Finally, the RoBERTa model outperformed all others with a training accuracy of 97% and a testing accuracy of 91%. **Figure 2** shows the performance analysis of training accuracy and testing accuracy of the deep learning models. From this **Figure**, we observed that RoBERTa model performed well compared to other deep learning models.

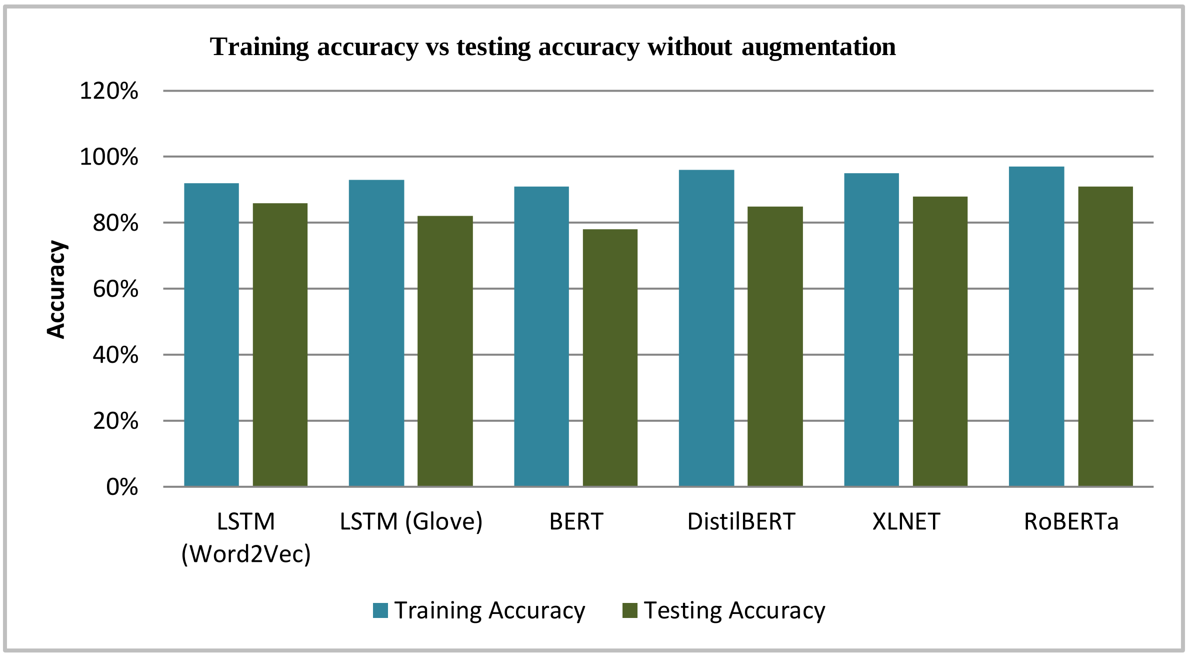


Figure 2. *Training vs testing accuracy without augmentation technique*

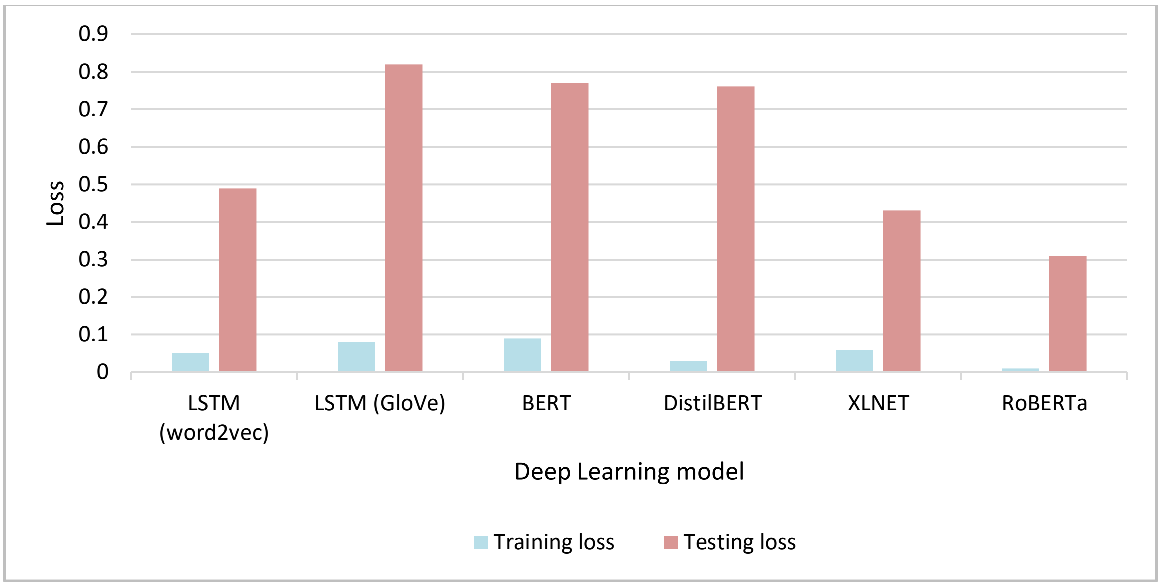


Figure 3. *Training vs testing loss without augmentation technique*

Figure 3 shows the results of a deep learning model with a training loss and testing loss. The training loss is a measure of how well the model fits the training data, while the testing loss is a measure of how well the model fits new data. The lower the loss, the better the model is performing. The chart explains that the RoBERTa model has the lowest training loss and testing loss, followed by DistilBERT, XLNET, and LSTM (Word2Vec). This suggests that RoBERTa is the best-performing model out of the ones tested. The LSTM models (Glove) and BERT have higher training and testing losses than the other models. This suggests that LSTM and BERT models are not as good at fitting the data as the other models. Overall, the table shows that the RoBERTa model is the best-performing model out of the ones tested. However, it is important to note that the results of this table are based on a small dataset, so more testing is needed to confirm these results.

4.2. Applying augmentation in training data

Due to the limited size of the dataset, training deep learning models directly may not yield satisfactory results. To address this, we employed various techniques to augment the data and improve its size and diversity. The data was cleaned and split into an 80:20 ratio for training and testing. We applied the synonym replacement technique to augment the training data, expanding each sentence by a factor of ten, resulting in an augmented dataset of 14,400 samples from the original 1,600. For augmentation, we utilized the nlpaug library's wordnet parameter in the nlpaug.augmenter.word module. However, the test data remained unaltered, with augmentation exclusively applied to the training data. The augmented data was preprocessed by removing punctuation and stop words.

Table 3. *Deep leaning models with Augmentation.*

Deep learning model	Library	Training		Testing	
		Accuracy	Loss	Accuracy	Loss
LSTM	Word2Vec	95%	0.01	88%	0.39
LSTM	GloVe	96%	0.005	89%	0.32
BERT	Transformer	98%	0.009	91%	0.27
DistilBERT	Transformer	97%	0.001	90%	0.26
XLNET	Transformer	98%	0.02	89%	0.29
RoBERTa	Transformer	99%	0.001	94%	0.15

Subsequently, we transformed the augmented data into word vectors using either the Word2Vec technique from the gensim library or the word embeddings for transformer method. Tokenization was applied to further process the vector data, preparing it for input into the deep learning models. The **Table 3** summarizes the performance of different deep learning models on a specific task. We have evaluated our model using accuracy and loss metrics. Each model is associated with a specific word embedding library, and the table provides information on their training and testing accuracy as well as training and testing loss. The LSTM model, when combined with the Word2Vec library, achieved a training accuracy of 95% with a training loss of 0.01. When evaluated on a separate testing dataset, it attained an accuracy of 88% with a testing loss of 0.39. Similarly, when the LSTM model was paired with the GloVe library, it achieved a slightly higher training accuracy of 96% and a lower training loss of 0.005. On the testing dataset, it achieved an accuracy of 89% with a testing loss of 0.32. **Table 3** shows the deep learning model performance with augmentation.

The BERT model achieved a higher training accuracy of 98% with a training loss of 0.009. During testing, it reached an accuracy of 91% with a testing loss of 0.27. The DistilBERT model achieved a slightly lower training accuracy of 97% with a training loss of 0.001. On the testing dataset, it obtained an accuracy of 90% with a testing loss of 0.26. The XLNET model achieved a training accuracy of 98% with a higher training loss of 0.02. When tested, it achieved an accuracy of 89% with a testing loss of 0.29. Finally, the RoBERTa model, using the Transformer library, achieved the highest training accuracy of 99% with a training loss of 0.001. On the testing dataset, it achieved an impressive accuracy of 94% with the lowest testing loss of 0.15. Overall, we observed from the table that the RoBERTa model achieved the highest accuracy, followed closely by the BERT model.

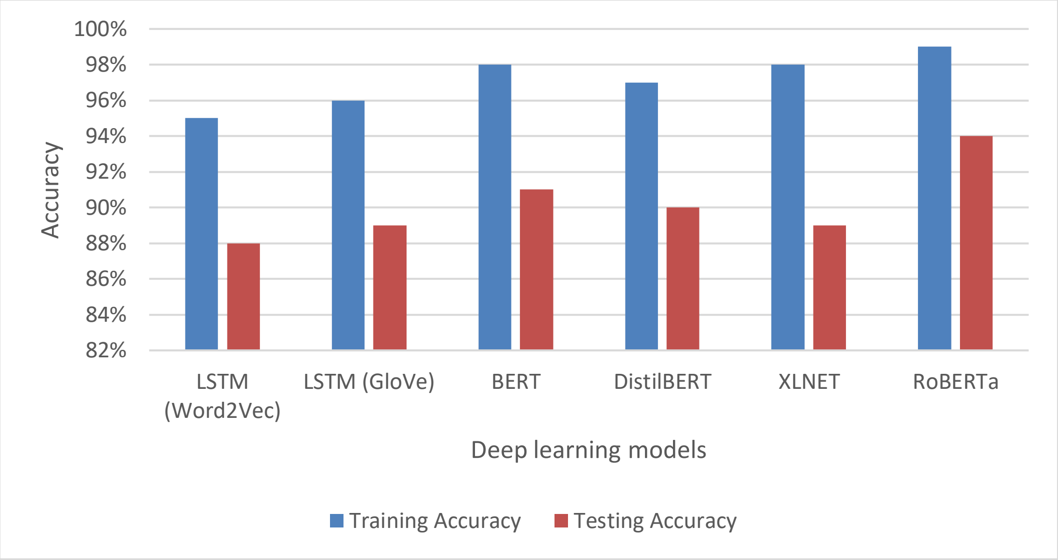


Figure 4. *Training vs testing accuracy with augmentation technique*

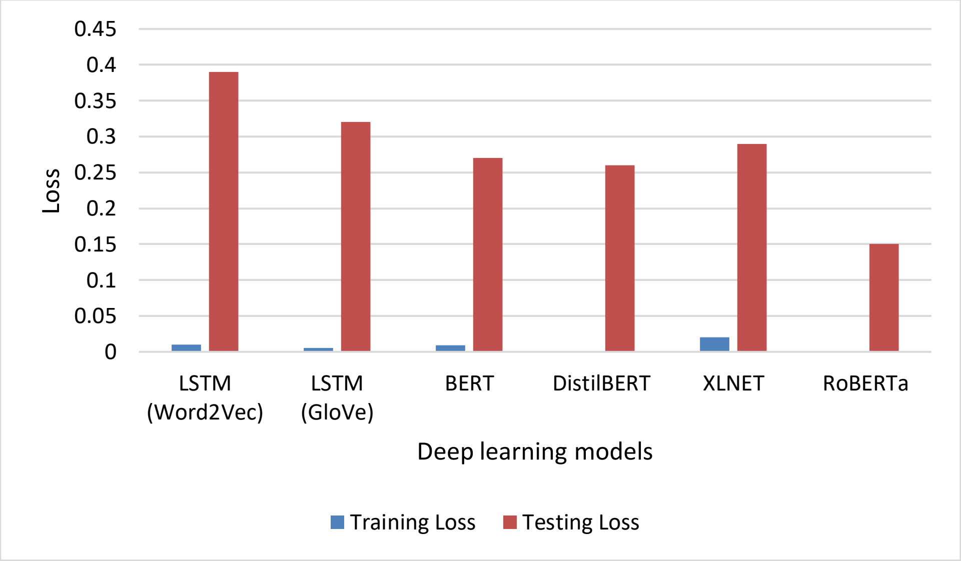


Figure 5. *Training vs testing loss with augmentation technique*

Figure 4 showcases the training and testing accuracy of deep learning models. Also, **Figure 5** shows that the RoBERTa model has the lowest training loss and testing loss, followed by DistilBERT, LSTM (GloVe), and BERT. This suggests that BERT is the best-performing model out of the ones tested. The LSTM model (Word2Vec) and XLNET have higher training and testing losses than the other models. This suggests that

LSTM model (Word2Vec) and XLNET are not as good at fitting the data as the other models. Overall, the barchart shows that the RoBERTa model is the best-performing model out of the ones tested.

4.3. Performance comparison of proposed model with machine learning models and other existing works

In **Table 4**, we present a comparative analysis of our model with existing machine learning approaches. Our model has demonstrated superior performance when compared to these methods. Table 3 provides an overview of different models along with their associated libraries and accuracy scores. The passive aggressive classifier, utilizing the TF-IDF library, achieved an accuracy of 92.5%. This model employs an online learning algorithm that makes updates based on the observed data. The linear support vector machine (LSVM), using the bag of words approach, obtained an accuracy of 91.8%. SVM is a supervised learning algorithm that separates data points using hyperplanes in a high-dimensional feature space. RoBERTa, utilizing the transformer library, achieved an accuracy of 91% without augmentation and 94% with augmentation. RoBERTa utilizes self-attention mechanisms to capture contextual dependencies in the input text. These results demonstrate the performance of the models on the given task, using RoBERTa with augmentation showing the highest accuracy, followed by the passive aggressive classifier and the linear SVM.

Table 4. *Performance comparison of proposed methodology with machine learning approach*

Input data	Model	Library	Accuracy
Original data	Passive aggressive classifier	TF-IDF	92.5%
Original data	Linear support vector machine	Bag of words	91.8%
Original data	RoBERTa	Transformer	91%
Original data (augmentation only on training data)	RoBERTa	Transformer	94%

It's essential to consider specific factors when assessing the performance of our proposed methods in comparison to other existing studies. Firstly, our results can only be directly compared to studies that have employed the same dataset, namely OpSpam. Secondly, we adopted well-established performance metrics for binary classification problems, consistent with those found in the scikit-learn library. Lastly, for the sake of ensuring a fair basis for comparison, we have exclusively presented the best results from each of the studies used in our evaluation. **Table 5** provides a comprehensive comparison between our proposed methods and existing research work that utilized the OpSpam dataset for fake review detection. Most of these studies have employed machine learning classifiers as their modeling approach. Regarding feature extraction, methods such as TF-IDF, LIWC, unigrams, and bigrams have been widely used in existing works. Notably, our proposed methods, leveraging transformer model with augmentation, have achieved the highest levels of performance in these comparisons.

Table 5. *Performance comparison of proposed methodology with existing works on the opinion spam dataset*

Model	Augmentation Techniques	Feature vectorization	Accuracy	F1-score	References
Linear SVM	No	TF-IDF	84%	83.6%	[9]
Spam GAN	Yes	Bag of words	86.8%	87.8%	[29]
Naïve Bayes	No	Bigram	93%	93%	[5]
SVM	No	Unigram + Bigram	90.9%	91%	[2]
SVM	No	LIWC + Bigram	91.2%	91%	[28]
Linear SVM	No	LDA + WSM	86%	86%	[24]
RoBERTa	Yes	Word embeddings	94%	94%	Our work

5. Discussion

The paper introduces a novel method for detecting fake reviews, particularly tailored for small datasets, through the application of augmentation techniques. The proposed method outperforms state-of-the-art approaches on the Deceptive Opinion Spam Corpus (OpSpam) dataset. It utilizes four distinct text augmentation techniques, including synonym replacement, random deletion, random insertion, and random swap. These techniques are applied to the training data, which is subsequently converted into vector representations for input into deep learning models. The proposed method offers several notable strengths. First, it excels in identifying subtle fake review activities with a higher degree of accuracy than existing state-of-the-art methods. Second, the incorporation of augmentation techniques has notably enhanced the precision of fake review detection. Third, the research underscores the significance of introducing artificial data into the training set, a factor that greatly improves the overall performance of the method.

Detecting fake reviews continues to pose a formidable challenge, particularly when dealing with smaller datasets. Artificial intelligence (AI) has exhibited promising outcomes within this domain, but the deficiency in interpretability and transparency of machine learning models, especially in the context of smaller datasets, raises doubts about the credibility of these proposed models. In alignment with the objective of developing models that not only excel in performance but also prioritize interpretability and transparency in their decision-making processes, the proposed model attains superior results and effectively tackles the constraint of working with smaller datasets by integrating augmentation techniques.

Overall, the proposed model presents a promising solution to the challenge of detecting fake reviews in smaller datasets, with the potential to make a significant impact on the realms of marketing and e-commerce. By enhancing the accuracy and dependability of online reviews using limited data, the approach addresses a critical need. Despite the promising results of this study, several limitations warrant further investigation in future research. One limitation pertains to the reliance on a single dataset, which may not adequately represent all categories of fake reviews. Additionally, the method primarily focuses on textual features, leaving room for exploration of other feature types such as images or user behavior.

Another noteworthy limitation lies in the interpretability of the proposed approach, which could constrain its applicability in contexts where transparency and interpretability are paramount. Additionally, the proposed method may still be susceptible to sophisticated fake review attacks, prompting further investigation into methods for bolstering its resilience against such threats. Despite these limitations, our proposed model presents a valuable contribution to the field of fake review detection, particularly in the context of small datasets. It outperforms state-of-the-art approaches and offers several notable strengths, including its ability to identify subtle fake review activities and its resilience to overfitting.

Future studies should delve into graph mining and machine learning techniques to further enhance the performance and generalizability of the method, as well as to address the limitations of our study. For example, future studies could investigate the use of multiple datasets from different domains to improve the robustness of the model to a wider range of fake review attacks.

6. Conclusion

Text augmentation techniques are pivotal for enhancing the performance and generalization capabilities of natural language processing models, particularly in challenging scenarios like detecting fake reviews with limited labeled data. Most existing fake review detection methods rely on supervised machine learning models due to the scarcity of data. In this study, we harnessed diverse data augmentation methods and incorporated them into various deep learning models to address data scarcity and counteract overfitting. Text augmentation not only mitigated these challenges but also bolstered the models' ability to handle out-of-domain or previously unseen examples. Our research introduces an innovative approach that harnesses augmentation techniques tailored for smaller datasets, using the OpSpam dataset for evaluation. The methodology involved text data preprocessing, augmentation application, transformation into vector representations, and training deep learning classifiers with transformer models. Our results demonstrate that augmentation significantly enhances the accuracy of detecting subtle fake review patterns, outperforming existing state-of-the-art methods.

In conclusion, our proposed approach highlights the efficacy of augmentation-based strategies and emphasizes the importance of employing these techniques, especially when dealing with limited datasets in the context of fake review detection. Our future research efforts will focus on enhancing our approach by integrating text-based features and investigating the utilization of advanced deep learning graph models like GCN, graphSAGE, or GNN, with the goal of further improving performance metrics. Additionally, we intend to explore additional graph theory techniques that hold promise for application in our dataset.

Declaration of interest

The authors declare that there is no conflict of interest.

Acknowledgements

The authors would like to acknowledge the support provided by United Arab Emirates University.

References

[1] Ahmed H, Traore I, Saad S. “Detecting opinion spams and fake news using text classification”, Security and Privacy 1.1 (2018) : e9.

[2] Bengio Y. “Learning deep architectures for AI”, Foundations and trends® in Machine Learning 2.1 (2009): 1-127.

[3] Algur SP, Patil AP, Hiremath PS, Shivashankar S. “Conceptual level similarity measure-based review spam detection”, International Conference on Signal and Image Processing, pp. 416-423. IEEE, 2010.

[4] Lau RY, Liao SY, Kwok RC, Xu K, Xia Y, Li Y. “Text mining and probabilistic language modeling for online review spam detection”, ACM Transactions on Management Information Systems (TMIS) 2, no. 4: 1-30, 2012.

[5] Jindal Nitin, Bing Liu. “Opinion spam and analysis”, In Proceedings of the international conference on web search and data mining, pp. 219-230, 2008.

[6] Choi Wonil, Kyungmin Nam, Minwoo Park, Seoyi Yang, Sangyoon Hwang, Hayoung Oh. “Fake review identification and utility evaluation model using machine learning”, Frontiers in artificial intelligence 5: 1064371, 2023.

[7] Yu AW, Dohan D, Luong MT, Zhao R, Chen K, Norouzi M, Le QV. “Qanet: Combining local convolution with global self-attention for reading comprehension”, 2018. CoRR aba/1804.09541. URL: https://arxiv.org/pdf/1804.09541.

[8] Kobayashi. “Contextual augmentation: Data augmentation by words with paradigmatic relations”, In: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers), Association for Computational Linguistics, New Orleans, Louisiana, pp. 452–457, 2018.

[9] Xie Z, Wang SI, Li J, Lévy D, Nie A, Jurafsky D, Ng AY. “Data noising as smoothing in neural network language models”, In: 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017.

[10] LeBaron B, Weigend AS. “A bootstrap evaluation of the effect of data splitting on financial time series”, IEEE Transactions on Neural Networks 9.1 (1998): 213-220.

[11] Coates A, Ng A, Lee H. “An analysis of single-layer networks in unsupervised feature learning”, Proceedings of the fourteenth international conference on artificial intelligence and statistics. JMLR Workshop and Conference Proceedings, 2011.

[12] Cunningham P, Carney J, Jacob S. “Stability problems with artificial neural networks and the ensemble solution”, Artificial Intelligence in medicine 20.3 (2000): 217-225.

[13] Dolgikh S. “Identifying explosive epidemiological cases with unsupervised machine learning”, medRxiv (2020): 2020-05.

[14] Hornik K, Stinchcombe M, White H. “Multilayer feedforward networks are universal approximators”, Neural networks 2.5 (1989): 359-366.

[15] Izonin I, Tkachenko R, Dronyuk I, Tkachenko P, Gregus M, and Rashkevych M. “Predictive modeling based on small data in clinical medicine: RBF-based additive input-doubling method”, Mathematical Biosciences and Engineering 18.3 (2021): 2599-2613.

[16] Karar ME. “Robust RBF neural network–based backstepping controller for implantable cardiac pacemakers”, International Journal of Adaptive Control and Signal Processing 32.7 (2018): 1040-1051.

[17] Ott M, Choi Y, Cardie C, Hancock JT. “Finding deceptive opinion spam by any stretch of the imagination”, arXiv preprint arXiv:1107.4557 (2011).

[18] Prystavka P, Cholyshkina O, Dolgikh S, Karpenko D. “Automated object recognition system based on convolutional autoencoder”, In 2020 10th international conference on advanced computer information technologies (ACIT). IEEE, 2020.

[19] Corona Rodriguez R, Alaniz S, Akata Z. “Modeling conceptual understanding in image reference games”, Advances in Neural Information Processing Systems 32 (2019).

[20] Li J, Ott M, Cardie C, Hovy E. “Towards a general rule for identifying deceptive opinion spam”, In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, pp. 1566-1576, 2014.

[21] Salah I, Jouini K, Korbaa O. “Augmentation-based ensemble learning for stance and fake news detection”, In Advances in Computational Collective Intelligence – 14th International Conference, ICCCI 2022, Proceedings of Communications in Computer and Information Science (Vol. 1653, pp. 29–41). 2022.

[22] Xie Q, Dai Z, Hovy E, Luong T, Le Q. “Unsupervised data augmentation for consistency training”, Advances in neural information processing systems 33, pp. 6256-6268, 2020.

[23] Shorten C, Khoshgoftaar TM, Furht B. “Text data augmentation for deep learning”, Journal of Big Data, 8(1), 1–34, 2021.

[24] Min J, McCoy RT, Das D, Pitler E, Linzen T. “Syntactic data augmentation increases robustness to inference heuristics”, In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 2339–2352, 2020.

[25] Huang L, Wu L, Wang L. “Knowledge graph-augmented abstractive summarization with semantic-driven cloze reward”, In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 5094–510, 2020.

[26] Glavaš G, Vulić I. “Is supervised syntactic parsing beneficial for language understanding tasks? An empirical investigation”, In: Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, pp. 3090–3104, 2021.

[27] Li MM, Huang K, Zitnik M. “Representation learning for networks in biology and medicine: advancements, challenges, and opportunities”, arXiv preprint arXiv:2104.04883 (2021).

[28] Zhao T, Liu Y, Neves L, Woodford O, Jiang M, Shah N. Data augmentation for graph neural networks. In Proceedings of the AAAI conference on artificial intelligence 2021 May 18 (Vol. 35, No. 12, pp. 11015-11023).

[29] Kong K, Li G, Ding M, Wu Z, Zhu C, Ghanem B, Taylor G, Goldstein T. “FLAG: adversarial data augmentation for graph neural networks”, arXiv:2010.09891 (2020).

[30] Devlin J, Chang MW, Lee K, Toutanova K. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”, In Proceedings of NAACL-HLT 2019 Jun 2 (Vol. 1, p. 2).

[31] Ester M, Kriegel HP, Sander J, Xu X. “A density-based algorithm for discovering clusters in large spatial databases with noise”, In KDD, vol. 96, no. 34, pp. 226-231. 1996.

[32] Forman, George, Ira Cohen. “Learning from little: Comparison of classifiers given little training”, In European Conference on Principles of Data Mining and Knowledge Discovery. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004.

[33] Fischer A, Igel C. “Training restricted Boltzmann machines: An introduction”, Pattern Recognition 47.1 (2014): 25-39.

[34] Hekler EB, Klasnja P, Chevance G, Golaszewski NM, Lewis D, Sim I. “Why we need a small data paradigm”, BMC medicine 17.1 (2019): 1-9.

[35] Mukherjee A, Liu B, Glance N. “Spotting fake reviewer groups in consumer reviews”, In Proceedings of the 21st international conference on World Wide Web, pp. 191-200, 2012.

[36] Shojaee S, Murad MA, Azman AB, Sharef NM, Nadali S. “Detecting deceptive reviews using lexical and syntactic features”, In 2013 13th International Conference on Intellient Systems Design and Applications, pp. 53-58. IEEE, 2013.

[37] Sanh V, Debut L, Chaumond J, Wolf T. “DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter”, arXiv preprint arXiv:1910.01108 (2019).

[38] Liu Y, Ott M, Goyal N, Du J, Joshi M, Chen D, Levy O, Lewis M, Zettlemoyer L, Stoyanov V. “RoBERTa: A Robustly Optimized BERT Pretraining Approach”, arXiv preprint arXiv:1907.11692 (2019).

[39] Clark K, Luong MT, Le QV, Manning CD. “ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators”, arXiv preprint arXiv:2003.10555 (2020).

[40] Greff K, Srivastava RK, Koutník J, Steunebrink BR, Schmidhuber J. “LSTM: A search space odyssey”, IEEE transactions on neural networks and learning systems 28, no. 10: 2222-2232, 2016.