Sakarya University

Journal of Computer and Information Sciences

e-ISSN 2636-8129

VOLUME 6 ISSUE 2 AUGUST 2023

SAUCIS

SAKARYA UNIVERSITY JOURNAL OF COMPUTER AND INFORMATION SCIENCES

ISSN 2636-8129

August 2023 Volume 6 Issue 2

SAUCIS

SAKARYA UNIVERSITY JOURNAL OF COMPUTER AND INFORMATION SCIENCES

ISSN 2636-8129

August 2023 Volume 6 Issue 2

SAUCIS

SAKARYA UNIVERSITY JOURNAL OF COMPUTER AND INFORMATION SCIENCES

ISSN 2636-8129

August 2023 Volume 6 Issue 2

CONTENTS

Sakarya University Journal of Computer and Information Sciences
http://saucis.sakarya.edu.tr/

DOI: 10.35377/saucis…1273536 RESEARCH ARTICLE

Classification of Malicious URLs Using Naive Bayes and Genetic Algorithm

Murat Koca[1*], İsa Avcı[2], Mohammed Abdulkareem Shakir Al-Hayani[2]

[1]Van Yuzuncu Yil University, Faculty of Engineering, Department of Computer Engineering Van/Türkiye
[2]Karabuk University, Faculty of Engineering, Department of Computer Engineering, Karabuk/Türkiye

Corresponding author:
Murat Koca,
Van Yuzuncu Yil University,
Faculty of Engineering,
Department of Computer Engineering
E-mail address:
muratkoca@yyu.edu.tr

Submitted: 30 March 2023
Revision Requested: 17 April 2023
Last Revision Received: 22 May 2023
Accepted: 27 May 2023
Published Online: 24 June 2023

Citation: Koca M. et al. (2023). Classification of Malicious URLs Using Naïve Bayes and Genetic Algorithm. *Sakarya University Journal of Computer and Information Sciences*. 6 (2) https://doi.org/10.35377/saucis...1273536

ABSTRACT

The financial losses of vulnerable and insecure websites are increasing day by day. The proposed system in this research presents a strategy based on factor analysis of website categories and accurate identification of unknown information to classify safe and dangerous websites and protect users from the previous one. Probability calculations based on Naive Bayes and other powerful approaches are used throughout the website classification procedure to evaluate and train the website classification model. According to our study, the Naive Bayes approach was benign and showed successful results compared to other tests. This strategy is best optimized to solve the problem of distinguishing secure websites from unsafe ones. The vulnerability data categorization training model included in this datasheet had a better degree of precision. In this study, the best accuracy probability of 96% was achieved in Naive Bayes' NSL-KDD data set categorization.

Keywords: HTML, Malicious, Naive Bayes, Machine learning, URL, Neural Network

1. Introduction

Hackers use commercial websites and random advertisements to spread their malicious links [1]. Because heavy internet users are persuaded to assume that their participation will yield financial gains, they fall prey to scam schemes, such as those that sell fake traditional fake ads, promote counterfeit loans, or sell cheap goods. Adequate expertise in avoiding actual physical injury and websites that pose a potential danger to our safety is not required [2].

Advertising exists for a variety of reasons, but its ultimate purpose is to get people to click through to the related sites and advertisements so they may read the content. In 2019, Symantec published a report on Internet security in which the company explained the existence of extensive and successive attacks on companies to steal information and cause significant losses, as well as large threats to personal and bank accounts and the threat of victims through threatening messages to pay a certain ransom using a variety of methods. Symantec also indicated the presence of widespread threats to personal and financial accounts, as well as the danger of victims receiving ransom demands. Clicking on harmful links in deceptive and fraudulent advertisements directs people to hostile websites [3].

At the moment, the technique that is employed to attack a network is also becoming more severe, and the difficulty of safeguarding worldwide networks is developing at the same rapid pace as the economy. The market for network security is expected to begin exhibiting signs of expansion around the year 2021, as stated by the projections [4].

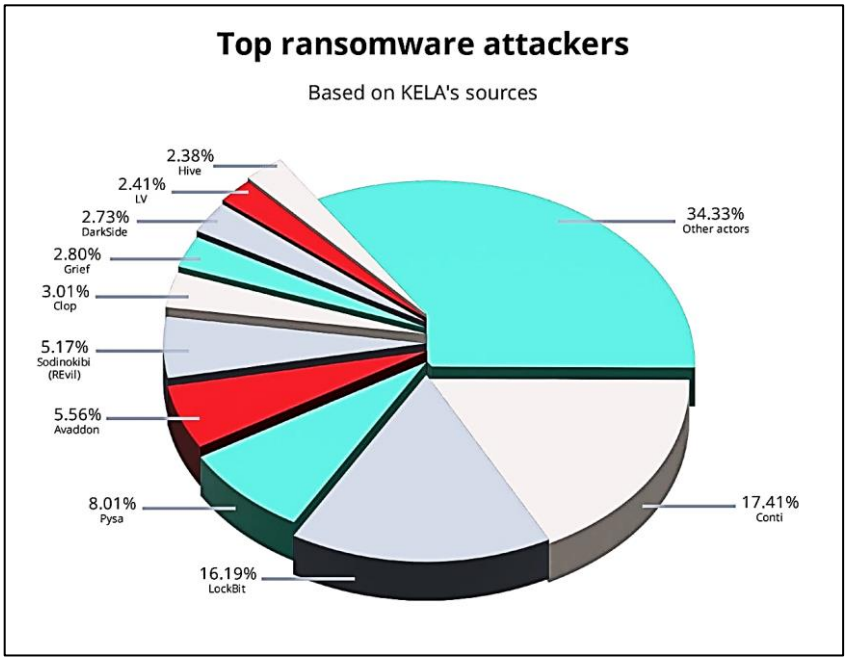


Figure 1. The proportion of ransomware attacks in malicious links [3].

The research shows that protecting and preserving the integrity of such networks has become an extremely important objective in light of the growing frequency with which cybercriminals attack networks. This is because cybercriminals are increasingly targeting networks. This is the case because con artists are placing a greater emphasis on networks, which has led to the current state of affairs in which the proposed system finds itself in. The majority of people will, throughout an ordinary day, navigate to a significant number of URLs on the internet. This activity is typically included as part of the activities that make up an average day. Unfortunately, within this ever-increasing multitude of URLs, there is now a considerable number of URLs that connect to dangerous websites. This is a very concerning trend. Because of the lightning-fast growth of the Internet, it is easy to mistakenly enter harmful URLs for those that are legitimate. Because it is possible to confound malicious URLs with genuine ones, making this error is not difficult to accomplish. As a result of this, it is much higher necessary to nurture the capabilities essential to discriminate between the two rapidly and exactly [5].

The goal of the academic community is to identify URLs that have the potential to be associated with fraudulent behavior and to do so, they make use of a wide variety of various benchmark models. The proposed system used a dataset that was comprised of URL occurrences so that the proposed system could assess the performance of K nearest neighbor KNN, support vector machine SVM, and Naive Bayes NB Tree. As a consequence of these studies, the proposed system realized that the application of this technology increased the accuracy with which the support vector machine and the KNN classified data. It was brought to our attention that decision Tree had the lowest degree of effectiveness when viewed in the context of the bigger picture. It has been claimed that a naïve Bayesian classifier may be used as a means of automatically classifying and determining which URLs have the potential to be fake. This could be accomplished through the use of a computer program. On a variety of benchmark data sets, the performance of the Navie Bayesian model, which was trained using probabilistic model learning, is superior to that of the support vector machine model [6]. This is the case even though the support vector machine model was trained using probabilistic model learning. This is the result of training having been done to improve the performance of the Naive Bayesian model. This is still the case even though the model of the support vector machine was constructed using probabilistic model learning. They developed a multi-stage filtering system that would detect potentially dangerous URLs by basing it on techniques that are associated with machine learning, which is a field of research that is abbreviated as ML [6].

It is based on techniques associated with deep learning, which is another acronym for deep learning. Because the classifier was trained with the critical threshold, it is now possible for the classifier to zero in on URLs at which it performs exceptionally well and pinpoints those. This is a direct effect of training the classifier with the critical threshold. Because of this, the classifier can function at its highest possible level of efficiency. If you discover that a certain group of classifiers is unable to accurately allocate a URL into one of their categories, you need to cast your vote with a number of the classifiers. In conclusion, it is vital to emphasize that the accuracy of spotting malicious URLs is enhanced when utilizing this method in comparison to the Bayesian model, the decision tree model, and the SVM model. This point cannot be emphasized enough. To correctly classify potentially hazardous URLs, logistic regression, neural networks, and three distinct iterations of the Naive Bayes approach were applied as analytical tools. According to the results of the study, the Naive Bayes strategies were the ones that had the highest success rate overall. Sheikh Shah Mohammad Motiur evaluated the efficacy of a large number of machine learning classifiers to determine whether or not they were capable of accurately identifying phishing URLs [7].

The metrics that he used to evaluate the effectiveness of these classifiers included the area under the receiver operating characteristic curve (AUC-ROC), accuracy, misclassification rate, and mean absolute error. When it comes to binary classification and feature sets that contain several different classes, stacking generalization provides more accurate results than random forest and multi-layer perceptron do. The impact that using a large number of machine learning models, in particular ML ensembles, to tackle the problem of locating phony URLs has. The random forest is the result of inheritance learning, and multiple metrics, including the recall rate, the accuracy rate, and the Area Under the Curve (AUC) value, have demonstrated that it is superior to the conventional ML model. In addition, the random forest is the product of inheritance learning. This state of affairs has arisen as a consequence of the fact that inheritance learning was the impetus behind the development of the random forest. Convolutional Neural Networks (CNN), Long Short-Term Memory Networks (LSTM), and Convolutional Neural Networks (CNN-LSTM) were the three distinct types of deep neural networks that were used in the process of identifying fake URLs [8].

However, to propose a YOLO-inspired multi-layer recursive convolutional neural network model for malicious URL identification, they did not conduct a comparison of the hidden layer and the number of neurons in the experiment. The abbreviations (CNN), (LSTM), and (CNN-LSTM) are the three that are most frequently utilized. The Text-RCNN and BRNN models, in addition to many other approaches, are not able to compete with the level of precision that can be accomplished by utilizing this method. During the study, each URL will have its length shortened in precisely the same manner until they are all the same length. This process will be repeated until there is no longer any distinction between any of them [9]. Working with longer URLs exposes you to a greater risk of losing data than doing the same activity with shorter URLs. There is a one-to-one relationship between the length of the URL and the degree of severity of this risk. The use of the benchmark machine learning model as a stepping stone in the process of developing more advanced feature engineering strategies to increase the rate at which potentially harmful URLs are detected is a possibility.

This would be done to boost the rate at which potentially harmful URLs are detected. A method was created for feature engineering that can change the spatial coordinates of the thing that is being generated in either a linear or nonlinear fashion, depending on what kind of change the proposed system wants to make. Both of these applications are viable options for making use of this technology. The recognition rates of KNN, linear support vector machine, and multi-layer perceptron are significantly improved when five separate spatial transformation models are utilized to generate and apply extra features to the classifier. This is because the classifier now contains a greater number of features. The generation of new capabilities and features is accomplished with the help of these five models.

The extraction of information from the text that is included within the URL is the primary focus of the great majority of the methods that may be employed in today's world to identify fake URLs. This holds for the vast majority of the different methods. The proposed system provides an association classification-based data mining method for identifying harmful URLs based on URLs and attributes acquired from online content. This method analyzes the relationships between the URLs and the attributes. This approach makes use of the URLs themselves, in addition to the attributes that may be extracted from the URLs. This strategy combines the usage of classification with that of association rules to accomplish what needs to be done. a set of instructions that, when combined, allow one to place things into categories and make relationships between those categories [10].

The proposed system first presents a weighted approach that extracts a fundamental set of characteristics for study, and then the proposed system evaluates machine learning algorithms based on how quickly and effectively they learn. This is done to further our understanding of the topic. This method collects lexical information from URLs and then analyzes it straightforwardly to locate links that can cause harm. The random forest algorithm and the KNN algorithm both have the potential to produce positive findings from the investigation. It was discovered that the pure lexical technique had the potential to enable rapid real-time determination of URLs in lightweight systems. This turned out to be the case [11].

This is accomplished by first collecting static lexical features from URL strings and then classifying them using an ensemble classification algorithm that has been trained by machine learning. This process is repeated until the desired result is achieved. This procedure is carried out numerous times until the desired outcome is accomplished. developed a method for detection that is wholly dependent on the lexical characteristics of the content they are looking to uncover. The convolutional neural network is now in a position to deliver a classification result that is more accurate because the URL strings have been gathered and processed. As a direct consequence of the increased precision of the classification, this is now conceivable. Developed a concatenated neural network technique model by combining Bi-Ind RNN and Caps Net to recognize malicious URLs, extract features (including vector and texture features at the character and word levels), and combine features [12].

Specifically, this model was created to recognize malicious URLs. Specifically, the identification of dangerous URLs was the motivation behind the creation of this approach. When a mixed neural network is used for the classification process, not only is there a significant rise in the speed with which potentially harmful URLs can be recognized but there is also a significant increase in the accuracy with which they can be identified. URL Net is a CNN-based deep neural network that was developed to determine whether or not a URL is fraudulent. The network was built to determine whether or not a URL is phony. The name of the network is derived from the term "universal resource locator," which describes its function. CNN and the word "CNN" are promoted, and the network is set up to manage the limits of manual feature engineering as well as the absence of visibility into the test URL [13].

2. Related Work

A lot of research has been done on the difficulty of categorizing a large number of websites currently available. The Naive Bayes classifier, which is commonly used to develop high-quality solutions to search problems using biologically inspired operators, can be combined with the Genetic Algorithm, which is commonly used to reduce the processing time by developing high-quality solutions to search problems using biologically inspired operators [14]. The Naive Bayes classifier would be able to develop high-quality solutions to search problems using biologically inspired operators. Using biologically inspired operators in this manner would allow the Naive Bayes classifier to generate high-quality solutions to search issues. In addition to that, the combination can cut down on the total amount of time that is needed for the process. They offer a variety of extra functions in addition to lexical and host-based capabilities. These features include the ability to enable or disable JS, the content of an HTML element, and a great deal more. Additional processing time is required for the categorization of web pages because there are a staggering 31 distinct attributes that can be utilized to accomplish this task [15].

On the other hand, it is possible to organize web pages. As a consequence of this, it is possible to classify websites according to a wide range of distinct characteristics. They used a range of tactics, such as recurrent feature switching and GA-based optimization, to achieve the desired level of accuracy, and they were successful in doing so [16]. The researchers observed that by employing a Naive Bayes classifier to identify websites based on URL properties, they were able to get an accuracy rate of 78% when utilizing this strategy. The results of the investigation led to this conclusion being drawn. This was a significant increase in accuracy in comparison to their earlier rates, which stood at 74% without GA and 87% with GA, respectively. This reflected a rise of 3% from the previous year. They are concentrating their efforts on discovering how people utilize websites, and the dataset that they are making use of is quite comprehensive. A key source of disappointment is the fact that the overall accuracy is lower than what was anticipated, even though the average number of recalls is higher than 88%. This is because the average number of recalls is higher than 88%.

This behavior involves pretending to be another website to steal sensitive information from its users. Even if the projects they're working on aren't specifically designed to find fraudulent websites, the strategies that they use to do so are significant in and of themselves. They were able to achieve ultimate accuracies of 89.1% and 88.5% with the Rotation Forest method, which proves that it is the best algorithm after undergoing a substantial amount of training and testing. After the installation of the feature ranking, they were successful in reaching an accuracy of 89% when utilizing MLP and an accuracy of 87.5 when utilizing REP Tree. Both of these results were achieved following the introduction of the feature ranking. It has been discovered that the utilization of multilayer perceptrons, which are one category of feedforward artificial neural networks, yields the most successful results that can be achieved in previous investigations. Because of this network, the number of dimensions that the data consists of will be cut down significantly. If the researchers use Rotation Forest on the entire training set, there is a good chance that they will achieve a significant improvement in their accuracy margin. The method of forest rotation would then be utilized in this scenario. This is because Rotation Forest is a method for supervised learning, which explains why these results were obtained. Rotation Forest beats other algorithms, such as Random Forest and Hyper Random Tree [2], in practical applications because it generates a bigger number of outcomes employing a lower number of trees. This is the primary reason why Rotation Forest is superior. This is the fundamental reason for the aforementioned advantage that it possesses in this regard.

To put it another way, the effectiveness of the Rotation Forest method is far higher. Nevertheless, the difficulty of their algorithms and the length of time that would be necessary to carry out their instructions will continue to be roadblocks. Support Vector Machine, J48, Naive Bayes, and Logistic regression are some of the approaches that should be utilized when searching websites for malware. Other methods that should be employed are Logistic regression and Naive Bayes. These are only a few of the available choices. They evaluated some different single-layer classifiers and concluded that J48 was the one that was the most effective at identifying websites that could contain malicious content. As a result of their additional research, they concluded that the XOR-aggregation cross-layer detection strategy is superior to the others because it rarely requires the utilization of the dynamic methodology [13]. This realization led them to the conclusion that the XOR-aggregation cross-layer detection strategy is superior to the others. After conducting the inquiry, they realized that this was the most likely explanation.

When applying the Naive Bayes classifier, it is necessary to bear in mind that the usage of multiple attributes does not result in particularly pleasing detection results. This is something that must be kept in mind at all times. It is imperative that this particular point not be ignored. Bear in mind that this is something that must be considered, as this must be done. Although it was possible to use a data aggregation cross-layer J48 classifier to achieve an astounding accuracy of 99.178%, the process took longer than four minutes, which may be uncomfortable for some customers. Because of the ever-increasing prevalence of social networks and the enormous amounts of data that they generate, researchers' curiosity has been piqued. This is because researchers have been able to see more data than ever before. In recent years, a significant amount of focus and research has been placed on a variety of topics, including the identification and filtering of spam, the localization of communities, and the dissemination of knowledge, to name just a few of these topics' specific manifestations.

Some different classification approaches, such as SMO, Naive Bayes, J48, and random forest, are utilized during the process of sorting spam emails into the right categories. These are but a few of the many different classification strategies that are put

into use. According to the results of the research, random forest performs much better than alternative classification approaches in terms of weighted precision (95.50%), recall (95.50%), accuracy (95.50%), and F-measure (95.50%). It was proposed that the S3D spam detection approach, which was cited in the previous sentence, could be employed instead of the semi-supervised spam detection method. S3D employs four unique lightweight classifiers to accomplish its goal of identifying spam tweets in real time. This is made possible by the platform's modular design. The authors of the article presented a strategy for recognizing Twitter spam that was dependent on the user's current mental state as the determining factor [4].

This strategy was utilized to determine the emotions that were mentioned in the Bengali text. It has been determined, based on the findings of the trials that were carried out, that the method that was proposed is capable of achieving an accuracy of 77.16 percent in the detection of two fundamental emotions (grief and happiness) in Bangla text. This conclusion was reached as a result of the findings that were obtained from the tests that were carried out. The results of the tests that were carried out served as the basis for this conclusion, which was arrived at as a result of those findings. In the course of these examinations, participants were tasked with locating instances of the aforementioned emotions within a Bangla text. According to the findings of an investigation that was carried out by Houshmand Craniometry using some different machine learning strategies on a dataset of SMS spam that was made available from the UCI Machine Learning repository[17], 10-fold cross-validation produced the highest level of accuracy[17]. This investigation was carried out using a dataset of SMS spam that was made available from the UCI Machine Learning repository [18]. It introduces a completely new supervised machine learning algorithm that is built on behavioral data as a tool for recognizing spam accounts in social networks. This algorithm is intended to be used as a way to eliminate unwanted accounts [8]. It is credited with developing the method, which may be found in [18]. Identifying accounts that are being used for spam requires the application of this method. They collected the data they required from Weibo and then used an ELM-based method to locate spam accounts among the user accounts they obtained.

The textual content, information regarding the user profile, and social interactions are the three types of attributes that should be chosen in the sequence presented here as part of this technique. Each of these three types of attributes is derived from a distinct source. This technique was implemented to identify user accounts that were being used to send spam. According to the source that was cited, the effectiveness of identifying spam SMS allegedly increased after the addition of a new content-based feature that was put into place. This was said by the source that was cited. The findings that were compiled from the application of a wide variety of classification strategies lend credence to the assertion that the suggested enhancements will lead to an increase in the level of precision achieved by SMS spam detection. These findings were compiled after applying the various classification strategies to a large number of messages. In addition, [10] discusses a spam detection system for social media that is both web-based and scalable.

The purpose of the system is to preserve the trustworthiness of social networks by warding off the creation of false posts and comments. Because of this, they were thus able to cluster vast volumes of data in a manner that was more efficient for them. The names of these algorithms, which are Decision Tree, KNN, Naive Bays, and Support Vector Machine, are how they are often referred to as (SVM). A training experiment simulation has been built as a means of emulating the progressive improvement that may be seen in individual spam filters. This improvement may be witnessed as the filters become more effective over time. To recreate the conditions of the training experiment, this has been carried out. It is possible that the training that was completed contributed to this progress. It was demonstrated that the SVM classification strategy was the most accurate of those that were used to evaluate whether or not the tone of a Bengali newspaper headline was negative or positive. The purpose of this evaluation was to determine whether or not the tone of the headline should be considered negative or positive. This analysis was conducted to determine whether or not the headline of a Bengali newspaper had a negative or a positive tone.

Logistic regression boosted tree, and support vector machine is just a few of the other methods of categorization that were applied here. The use of text processing software to perform semantic analysis and establish context comes highly recommended. They placed it to the test by utilizing a dataset that was open to the general public, did not include any information that was encrypted, and was genuine. They also integrated some additional well-established machine learning techniques, all of which have the potential to improve spam filtering in instant messaging as well as SMS. This was done in addition to the machine learning techniques already applied.

Malware is notoriously tough to eliminate due to its many manifestations and rapid dissemination. Listed below are our most important contributions to the field. The proposed system is currently constructing a model to identify URLs as either secure or dangerous using the Naive Bayes algorithm, a cutting-edge piece of technology that deviates from conventional methodologies.

3. Methodology

The only kind of file that can be loaded into the software at this time is a.csv. After doing individual tests on each of the features, the next step is to use Naive Bayes classifier to determine the five most important characteristics. In the meantime, these five characteristics would be used to evaluate which feature set had the greatest room for development. Following completion, the solution would be assessed using a test dataset. Figure 1 is a visual representation of the process.

3.1. Dataset

In the course of this particular experiment, the proposed system made use of the dataset that Kaggle made accessible to us for our research purposes as seen in Figure 2. The gathering of information was the most important task that needed to be accomplished. During our investigation on the internet, the proposed system came across some websites that contained links to a variety of other websites. Some of these other websites may contain content that is detrimental to users, and some of these other websites may also contain links to other potentially harmful websites [1].

The third step involved identifying URLs that were devoid of anything that would have contributed to the confusion that was being caused. The dataset was not only very simple to access but also did not require any sort of data processing or cleaning on our end because it was already in its finished form. In addition to that, the proposed system has produced a list of URLs, the vast majority of which go to malicious websites while others do not. Some of these URLs do not go to hazardous websites. Some of these URLs redirect to websites that are not malicious. Next, to determine which method is the most accurate in determining which URLs could potentially be harmful, a method of LR, Naive Bayes, and a CNN system was put through their paces. This study is to determine which method is the most accurate in making this determination.

This was done so that the proposed system could decide which method is the most trustworthy, therefore that is our motivation behind doing it. This was done to determine which method was the most accurate, and it was successful in that endeavor. This experiment takes a list of URLs as its input, and the degree to which it is successful or unsuccessful is determined by how accurately it guesses the locations of the web pages that are referred to by those URLs. This experiment accepts a list of URLs as its input. The experiment's input is a list of URLs if you care to provide one.

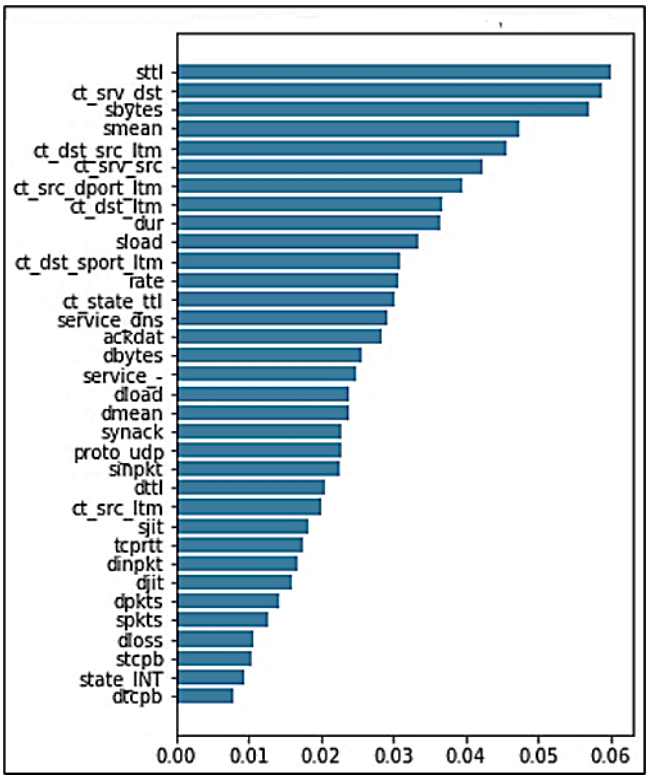


Figure 2. Features of the Dataset

3.2. Research Design

3.2.1. Naive Bayes Classifier

The Bayesian theorem and the premise of independence are the two foundations around which the Naive Bayes probabilistic model is constructed (the features being considered are unrelated to one another). Even though their seeming lack of complexity may be misleading, models that are trained with just two labels often perform rather well. Where x is a representation of the characteristic and C is the class being discussed. In light of this, the proposed system may describe C's conditional probability using the above as seen in Equation 1 when x is known.

$$\mathbf{P_r(C \mid x) = \frac{P_r(x|C)P_r(C)}{P_r(x)}} \quad (1)$$

3.2.2. Naive Bayes Classifier Probability-Based Prediction

Given that there are four distinct forms of harmful URLs, the software makes use of benign weight to filter them out. Naive Bayes assigns a probability of PPrr(CC00) to a benign URL and assigns PPrr(CC11), PPrr(CC22), PPrr(CC33), and PP(CC44) to each of four types of malicious URLs. The ultimate probability for good is PP(CC00), whereas for bad it is PP(CC11) as seen in Equations 2 and 3.

$$\mathbf{P(C_1) = P_r(C_1) + P_r(C_2) + P_r(C_3) + P_r(C_4)} \quad (2)$$

$$\mathbf{P(C_0) = wP_r(C_0)} \quad (3)$$

3.3. Proposed Methods

3.3.1. Choosing the Most Appropriate Features

A Naive Bayes classifier will be applied to each feature to identify a high-performing group. Then, the five characteristics with the greatest level of reliability would be selected. A Naive Bayes classifier would then randomly choose three of the aforementioned characteristics and utilize them to make a judgement. The conclusion is to choose the combination yielding the highest overall profit ratio.

3.3.2. Feature Extraction

During this research, the URL will be analyzed and interacted with in various ways. The data were first organized in a way that was easier to read. The first batch of feature data was produced by examining the content of the URL and making comparisons between possibly harmful and safe URLs. The evolutionary algorithm employs a method of feature extraction that helps in simplifying the feature vector, which in turn speeds up the processing of data. A "Genetic Algorithm (GA)" is a term that's used in the field of computer science to refer to the process of analyzing biological systems. The method of stochastic global search and optimization takes its inspiration from the natural setting in which the process of evolution takes place. An algorithm for a quick, exhaustive, and parallel global search serves as the foundation of this system.

This approach takes into account all of the potential solutions while avoiding the pitfalls associated with a locally optimal solution. In contrast, the genetic algorithm is not in any way restricted in any manner by considerations such as the requirement to keep function continuity or identify a given beginning point. Instead, the genetic algorithm is free to explore all possible starting points and preserve function continuity. Probabilistic optimization does not require the establishment of rules to automatically retrieve and navigate the optimal search space and to adapt the search direction as necessary. This is made possible by the fact that probabilistic optimization can adapt the search direction as needed.

The process began by seeding the population, encoding the attributes, and calculating the fitness of the individuals who would ultimately represent each chromosome. This allowed us to determine which individuals would best represent each chromosome. The crossover procedure was used to make children, and the chromosomes of the offspring were changed by the concept that the more fit an individual was, the better their chances were of being picked.

The crossover method was used to produce children after two members of the population were chosen at random from the population to take on the role of parents in the experiment. To develop a new population, one needs just to repeat the procedures from the approach that came before it. Last but not least, the proposed system might be modified to fit more closely the evaluation criteria established by the various frameworks (TPR, FPR, TNR, FNR). The following sequence presents the four rules:

The True Positive Rate is the percentage of potentially dangerous occurrences for which a positive detection was made by applying the analysis to the whole database of harmful examples as seen in Equation 4.

$$\mathbf{TP = \frac{N_{M\to M}}{N_{M\to M}+N_{M\to B}}} \quad (4)$$

4. Results and discussion

The dataset was evaluated using a total of three distinct approaches in its entirety. Following the completion of the data standardization, the proposed system proceeded to split into two distinct phases: the development phase and the assessment phase. When determining whether or not the results generated by a neural network are superior to those of the standard model, the performance of a conventional logistic regression model is used as a baseline. This allows one to determine whether or not the results generated by a neural network are superior. According to the findings as seen in Table 1, the accuracy score is 96%, the error rate is 0.04, and the recall score is 98%. It is successful in identifying safe websites with an F1 score of 95% as seen in Table 2. To determine whether or not a neural network can aid in the improvement of the issue, the proposed system employs one. The proposed system will begin by using the standard configuration of the SciKit learn class before moving on to exploring the various configuration choices in an attempt to get the maximum possible level of performance. The proposed algorithm results are shown in Table 1.

Table 1. Malicious website URL detection comparison using different methods.

Author	Algorithm	Accuracy	Error rate
Proposed algorithm	Modified NB	96%	0.04
Moruff Oyelakin et.al [5]	DT	88%	0.3
Subasi et.al [16]	KNN	87%	0.5
Jian et.al [18]	SVM	91%	0.2
Luo et.al [23]	CNN	93%	0.17

It can be seen with the classification of two classes where 0 means a successful classification of benign classes and 1 means a classification of malicious links as seen in Table 2.

Table 2. Malicious website URL identification using multinomial Naive Bayes: a confusion matrix.

Classes	Precision	Recall	F1-Score	Data
(0)	0.93	0.98	0.95	630
(1)	0.72	0.43	0.54	83
Accuracy	0.90	0.90	0.91	713
Macro avg.	0.82	0.71	0.75	713
Weighted	0.90	0.91	0.90	713

ML model performance over binary classification is shown comparatively on Naive Bayes, Decision Tree, KNN, Logistic Regression, and Random Forest algorithms as seen in Figure 3. In this experiment, Naive Bayes came to the fore as the most successful machine learning model.

5. Conclusion

The identification of potentially harmful URLs is one of the most important processes involved in the process of ensuring the safety of cybersecurity software. There is reason to be optimistic about the potential of machine learning algorithms The study was conducted by investigating the use of AI algorithms in the process of determining whether or not URLs might contain malicious content. The purpose of this study was to investigate the application of AI algorithms to the process of identifying whether or not URLs may include dangerous content. The results show a 98% recall percentage, an accuracy rate of 96%, and an error rate of 0.04. In this study, the proposed system was able to categorize potentially damaging URLs using

Logistic Regression, Neural Networks, and multiple Naive Bayes Algorithms. This allowed us to determine which URLs posed the greatest risk to users. Because the proposed model can identify which URLs constituted the biggest threat. When applied to the difficult distribution dataset, the results show that the Naive Bayes strategy performed noticeably better than both the logistic regression and neural network approaches. The proposed system has it in our plans to, among other things, do research on new datasets and experiment with a wide range of different machine-learning approaches.

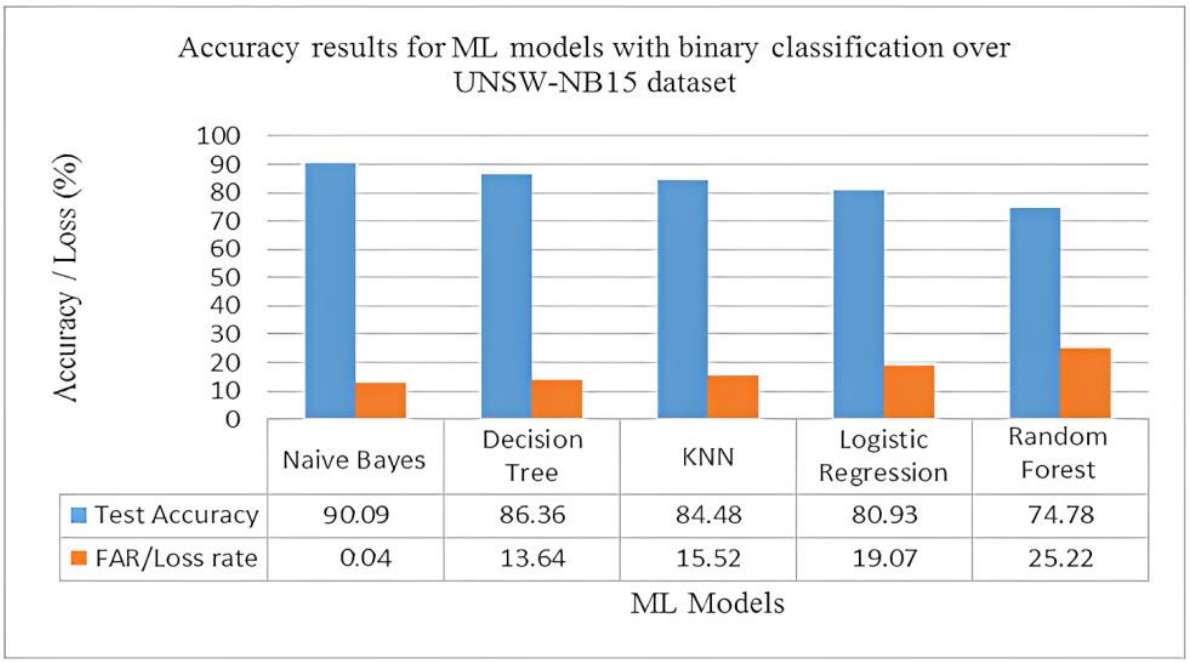


Figure 3. ML model performance over binary classification

References

[1] M. Tavallaee, E. Bagheri, W. Lu, and A. Ghorbani, "A Detailed Analysis of the KDD CUP 99 Data Set," *Submitted to Second IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA)*, 2009.

[2] A. Sharma and A. Thakral, "Malicious URL classification using machine learning algorithms and comparative analysis," Advances in Intelligent Systems and Computing, vol. 1090, pp. 791–799, 2020, doi: 10.1007/978-981-15-1480-7_73/COVER.

[3] K. U. Santoshi, S. S. Bhavya, Y. B. Sri, and B. Venkateswarlu, "Twitter Spam Detection Using Naïve Bayes Classifier," Proceedings of the 6th International Conference on Inventive Computation Technologies, ICICT 2021, pp. 773–777, Jan. 2021, doi: 10.1109/ICICT50816.2021.9358579.

[4] T. Islam, S. Latif, and N. Ahmed, "Using Social Networks to Detect Malicious Bangla Text Content," 1st International Conference on Advances in Science, Engineering and Robotics Technology 2019, ICASERT 2019, May 2019, doi: 10.1109/ICASERT.2019.8934841.

[5] A. Moruff Oyelakin, O. Akinyemi Moruff, A. Olasunkanmi Maruf, and A. Tosho, "Performance Analysis of Selected Machine Learning Algorithms for the Classification of Phishing URLs Machine Learning Techniques in building Predictive Models for COVID-19 View project Investigation of MANET security protocols and optimisation View project Performance Analysis of Selected Machine Learning Algorithms for the Classification of Phishing URLs", Accessed: Jan. 05, 2023. [Online]. Available: https://www.researchgate.net/publication/345161822

[6] Maciej Serda et al., "Synteza i aktywność biologiczna nowych analogów tiosemikarbazonowych chelatorów żelaza," Uniwersytet śląski, vol. 7, no. 1, pp. 343–354, 2013, doi: 10.2/JQUERY.MIN.JS.

[7] T. Wu, Y. Xi, M. Wang, and Z. Zhao, "Classification of Malicious URLs by CNN Model Based on Genetic Algorithm," Applied Sciences 2022, Vol. 12, Page 12030, vol. 12, no. 23, p. 12030, Nov. 2022, doi: 10.3390/APP122312030.

[8] R. Rajalakshmi, S. Ramraj, and R. Ramesh Kannan, "Transfer learning approach for identification of malicious domain names," Communications in Computer and Information Science, vol. 969, pp. 656–666, 2019, doi: 10.1007/978-981-13-5826-5_51/COVER.

[9] G. Wejinya and S. Bhatia, "Machine Learning for Malicious URL Detection," Advances in Intelligent Systems and Computing, vol. 1270, pp. 463–472, 2021, doi: 10.1007/978-981-15-8289-9_45/COVER.

[10] F. Alzubaidi, "DETECT MALWARE URL USING NAIVE BAYES ALGORITHM."
[11] A. E. El-Din, E. El-Din Hemdan, and A. El-Sayed, "Malweb: An efficient malicious websites detection system using machine learning algorithms," ICEEM 2021 - 2nd IEEE International Conference on Electronic Engineering, Jul. 2021, doi: 10.1109/ICEEM52022.2021.9480648.
[12] S. Wang, Y. Wang, and M. Tang, "Auto Malicious Websites Classification Based on Naive Bayes Classifier," Proceedings of 2020 IEEE 3rd International Conference on Information Systems and Computer Aided Education, ICISCAE 2020, pp. 443–447, Sep. 2020, doi: 10.1109/ICISCAE51034.2020.9236912.
[13] S. Wang, Y. Wang, and M. Tang, "Auto Malicious Websites Classification Based on Naive Bayes Classifier," Proceedings of 2020 IEEE 3rd International Conference on Information Systems and Computer Aided Education, ICISCAE 2020, pp. 443–447, Sep. 2020, doi: 10.1109/ICISCAE51034.2020.9236912.
[14] W. Fadheel, W. Al-Mawee, and S. Carr, "On Phishing: URL Lexical and Network Traffic Features Analysis and Knowledge Extraction using Machine Learning Algorithms (A Comparison Study)," 2022 5th International Conference on Data Science and Information Technology, DSIT 2022 - Proceedings, 2022, doi: 10.1109/DSIT55514.2022.9943832.
[15] C. Liu and G. Wang, "Analysis and detection of spam accounts in social networks," 2016 2nd IEEE International Conference on Computer and Communications, ICCC 2016 - Proceedings, pp. 2526–2530, May 2017, doi: 10.1109/COMPCOMM.2016.7925154.
[16] Subasi, A.; Balfaqih, M.; Balfagih, Z.; Alfawwaz, K. A comparative evaluation of ensemble classifiers for malicious webpage detection. Procedia Comput. Sci. 2021, 194, 272–279.
[17] Sayamber, A.B.; Dixit, A.M. Malicious URL detection and identification. Int. J. Comput. Appl. 2014, 99, 17–23.
[18] Jian, L.; Gang, Z.; Yunpeng, Z. Design and implementation of malicious URL multi-layer filtering detection model. Inf. Netw. Secur. 2016, 1, 6.
[19] Vundavalli, V.; Barsha, F.; Masum, M.; Shahriar, H.; Haddad, H. Malicious URL detection using supervised machine learning techniques. In Proceedings of the 13th International Conference on Security of Information and Networks, Merkez, Turkey, 4–7 November 2020; pp. 1–6.
[20] Rahman SS, M.M.; Islam, T.; Jabiullah, M.I. PhishStack: Evaluation of stacked generalization in phishing URLs detection. Procedia Comput. Sci. 2020, 167, 2410–2418.
[21] Zeyu, L.; Yong, S.; Zhi, X. Malicious URL recognition based on machine learning. Commun. Technol. 2020, 53.
[22] Pham TT, T.; Hoang, V.N.; Ha, T.N. Exploring efficiency of character-level convolution neuron network and long short-term memory on malicious URL detection. In Proceedings of the 2018 VII International Conference on Network, Communication and Computing, Taipei City, Taiwan, 14–16 December 2018; pp. 82–86.
[23] Chen, Z.; Liu, Y.; Chen, C.; Lu, M.; Zhang, X. Malicious URL detection based on improved multilayer recurrent convolutional neural network model. Secur. Commun. Netw. 2021, 2021, 9994127.
[24] Li, T.; Kou, G.; Peng, Y. Improving malicious URLs detection via feature engineering: Linear and nonlinear space transformation methods. Inf. Syst. 2020, 91, 101494.
[25] Kumi, S.; Lim, C.H.; Lee, S.G. Malicious URL detection based on associative classification. Entropy 2021, 23, 182.
[26] Raja, A.S.; Vinodini, R.; Kavitha, A. Lexical features based malicious URL detection using machine learning techniques. Mater. Today: Proc. 2021, 47 Pt 1, 163–166.
[27] Joshi, A.; Lloyd, L.; Westin, P.; Seethapathy, S. Using lexical features for malicious URL detection—A machine learning approach. arXiv 2019, arXiv:1910.06277.
[28] Kang, C.; Huazheng, F.; Yong, X. Malicious URL identification based on deep learning. Comput. Syst. Appl. 2018, 27, 27–33.
[29] Yuan, J.T.; Liu, Y.P.; Yu, L. A novel approach for malicious URL detection based on the joint model. Secur. Commun. Netw. 2021, 2021, 4917016.
[30] Le, H.; Pham, Q.; Sahoo, D.; Hoi, S.C. URLNet: Learning a URL representation with deep learning for malicious URL detection. arXiv 2018, arXiv:1802.03162.
[31] Yuan, J.; Chen, G.; Tian, S.; Pei, X. Malicious URL detection based on a parallel neural joint model. IEEE Access 2021, 9, 9464–9472.
[32] Zhao, G.; Wang, P.; Wang, X.; Jin, W.; Wu, X. Two-dimensional code malicious URL detection method based on decision tree. Inf. Secur. Technol. 2014, 5, 36–39.
[33] Liu, C.; Wang, L.; Lang, B.; Zhou, Y. Finding effective classifier for malicious URL detection. In Proceedings of the 2018 2nd International Conference on Management Engineering, Software Engineering and Service Sciences, Wuhan, China, 13–15 January 2018; pp. 240–244.
[34] Lin, H.L.; Li, Y.; Wang, W.P.; Yue, Y.L.; Lin, Z. Efficient malicious URL detection method based on segment pattern. Commun. J. 2015, 36, 141–148.
[35] Subasi, A.; Balfaqih, M.; Balfagih, Z.; Alfawwaz, K. A comparative evaluation of ensemble classifiers for malicious webpage detection. Procedia Comput. Sci. 2021, 194, 272–279.
[36] Sayamber, A.B.; Dixit, A.M. Malicious URL detection and identification. Int. J. Comput. Appl. 2014, 99, 17–23.
[37] Jian, L.; Gang, Z.; Yunpeng, Z. Design and implementation of malicious URL multi-layer filtering detection model. Inf. Netw. Secur. 2016, 1, 6.

[38] Vundavalli, V.; Barsha, F.; Masum, M.; Shahriar, H.; Haddad, H. Malicious URL detection using supervised machine learning techniques. In Proceedings of the 13th International Conference on Security of Information and Networks, Merkez, Turkey, 4–7 November 2020; pp. 1–6.
[39] Rahman SS, M.M.; Islam, T.; Jabiullah, M.I. PhishStack: Evaluation of stacked generalization in phishing URLs detection. Procedia Comput. Sci. 2020, 167, 2410–2418.
[40] Zeyu, L.; Yong, S.; Zhi, X. Malicious URL recognition based on machine learning. Commun. Technol. 2020.
[41] Pham TT, T.; Hoang, V.N.; Ha, T.N. Exploring efficiency of character-level convolution neuron network and long short-term memory on malicious URL detection. In Proceedings of the 2018 VII International Conference on Network, Communication and Computing, Taipei City, Taiwan, 14–16 December 2018; pp. 82–86.
[42] Chen, Z.; Liu, Y.; Chen, C.; Lu, M.; Zhang, X. Malicious URL detection based on improved multilayer recurrent convolutional neural network model. Secur. Commun. Netw. 2021, 2021, 9994127.
[43] Li, T.; Kou, G.; Peng, Y. Improving malicious URLs detection via feature engineering: Linear and nonlinear space transformation methods. Inf. Syst. 2020, 91, 101494.

Conflict of Interest Notice
The authors declare that there is no conflict of interest regarding the publication of this paper.

Ethical Approval and Informed Consent
It is declared that during the preparation process of this study, scientific and ethical principles were followed, and all the studies benefited from are stated in the bibliography.

Availability of data and material
Not applicable.

Plagiarism Statement
This article has been scanned by iThenticate ™.

Sakarya University Journal of Computer and Information Sciences
http://saucis.sakarya.edu.tr/

DOI: 10.35377/saucis…1257100 RESEARCH ARTICLE

Optimization of Several Deep CNN Models for Waste Classification

Mahir Kaya[1], Samet Ulutürk[1], Yasemin Çetin Kaya[1], Onur Altıntaş[1], Bülent Turan[1]

[1] Department of Computer Engineering, Faculty of Engineering and Architecture, Tokat Gaziosmanpaşa University, Tokat, Türkiye

Corresponding author:
Mahir Kaya,
Department of Computer Engineering,
Tokat Gaziosmanpaşa University
E-mail address: mahir.kaya@gop.edu.tr

Submitted: 28 February 2023
Revision Requested: 19 June 2023
Last Revision Received: 1 July 2023
Accepted: 04 July 2023
Published Online: 14 July 2023

Citation: Kaya M. et al. (2023). Optimization of Several Deep CNN Models for Waste Classification. *Sakarya University Journal of Computer and Information Sciences*. 6 (2) https://doi.org/10.35377/saucis...1257100

ABSTRACT

With urbanization, population, and consumption on the rise, urban waste generation is steadily increasing. Consequently, waste management systems have become integral to city life, playing a critical role in resource efficiency and environmental protection. Inadequate waste management systems can adversely affect the environment, human health, and the economy. Accurate and rapid automatic waste classification poses a significant challenge in recycling. Deep learning models have achieved successful image classification in various fields recently. However, the optimal determination of many hyperparameters is crucial in these models. In this study, we developed a deep learning model that achieves the best classification performance by optimizing the depth, width, and other hyperparameters. Our six-layer Convolutional Neural Network (CNN) model with the lowest depth and width produced a successful result with an accuracy value of 89% and an F1 score of 88%. Moreover, several state-of-the-art CNN models such as VGG19, DenseNet169, ResNet101, Xception, InceptionV3, RegnetX008, RegnetY008, EfficientNetV2S trained with transfer learning and fine-tuning. Extensive experimental work has been done to find the optimal hyperparameters with GridSearch. Our most comprehensive DenseNet169 model, which we trained with fine-tuning, provided an accuracy value of 96.42% and an F1 score of 96%. These models can be successfully used in a variety of waste classification automation.

Keywords: CNN, Deep learning, Trash, Optimization

1. Introduction

According to World Bank data, global urban waste generation annually amounts to 2.01 billion tons. Urbanization and consumption trends indicate a projected 69% increase by 2050, surpassing 3.4 billion tons [1]. Consequently, waste management systems have become indispensable for cities, ensuring efficient resource utilization and environmental protection. Inadequate waste management systems harm the environment and the economy [2]. Recycling is crucial, ranking as the second-best environmentally friendly method according to the Environmental Protection Agency [3]. The European Union achieved a 56% recycling rate in 2016, with ongoing efforts to raise this figure. Human resources are used in some aspects of recycling, which reduces efficiency, increases costs, and harms the health of those who work in this industry [4]. Intelligent systems are being used to reduce or eliminate these issues.

One of the critical stages in intelligent waste management systems is accurate and rapid waste classification. Convolutional Neural Networks (CNNs), which have end-to-end learning capability, are widely used to classify and segment images in many fields. In CNNs, multiple convolutional and max pooling layers are added sequentially to extract features from raw images. In the final stage, fully connected layers are employed for classification purposes. [5]. During the training phase, learning takes place through the iterative update of a multitude of filters and fully connected layer weights, utilizing the backpropagation algorithm [6]. At the onset of training, we establish numerous hyperparameters for the model architecture. These hyperparameters encompass various aspects such as model depth, number of filters, filter sizes, dropout rates, optimizers, learning rates, epochs, batch sizes, and more. CNN models often face challenges in achieving successful

classification in datasets with limited data and high inter-class similarity [7]. In CNN models, as the depth and width increase, overfitting is commonly encountered during the training phase, especially in datasets with limited labeled data. Despite exhibiting high accuracy on the training dataset, these models tend to perform poorly on previously unseen test datasets [8]. Avoiding overfitting and determining the optimal combination of hyperparameters play a crucial role in improving the classification performance of CNN models. Therefore, CNN models with various architectures and hyperparameters were used in the study for waste classification.

In this study, several models with varying depths, widths, and optimized hyperparameters were developed. An attempt was made to achieve the best model through hyperparameter optimization using GridSearch. Although the success rate of deep learning models increases as the depth and width increase, excessive depth can lead to gradient vanishing, which prevents the model from reaching the optimum [9, 10]. This problem is partially addressed in models such as ResNet [11] and DenseNet [12] through the use of residual connections, which allow the following inputs or blocks to receive information from the previous inputs. In waste management systems, real-time classification is performed, which makes the classification time of models crucial. While deep and complex models may achieve high classification accuracy, the time taken to classify a single image in real-time implementations can exceed expectations. To address this, we developed the best model using transfer learning and optimization techniques, which led to improved classification performance. Additionally, we successfully obtained the most effective shallow or small model in terms of both accuracy and prediction speed. The proposed model has the potential for successful utilization in diverse applications of waste classification automation.

The contributions of this study can be listed as follows:

- A novel waste classification CNN model has been developed to work in different embedded systems and be integrated into waste management systems.
- The performance of different architectures with various widths and depths in waste classification is presented.
- The model performance has been increased by optimizing many hyperparameters with GridSearch.
- With transfer learning, the performance of the state-of-the-art CNN models has been increased by fine-tuning and hyperparameter optimization methods.

Different CNN architectures have been optimized with extensive experimental studies, and the optimized models that make the most successful classification have been developed. These different CNN models can be integrated into various embedded systems and used practically in waste management systems.

2. Related works

CNN is a type of artificial neural network that processes images [11]. It extracts the features of the objects in the images and learns them using various learning algorithms. CNN provides effective results in many applications, including robotics, security cameras, license plate recognition, and face recognition. Since CNN is a technique that is widely used and has a high success rate in automatically classifying waste, the number of studies in this field is significant.

Transfer learning-based CNN models have shown successful results in image classification, especially when dealing with datasets with a limited number of samples. Bircanoğlu et al. [13] developed a waste classification model based on transfer learning. Several state-of-the-art CNN models were re-trained on the TrashNet dataset. The Densenet121 model gave the best result with 95% accuracy. The DenseNet121 model is reported to have an approximate CPU time of 649 ms. Wang et al. [14] proposed a system for waste management. The TrashNet dataset was expanded in the study, and the number of categories was increased from six to nine. MobileNetV3, MobileNetV2, InceptionV3, ResNet50, ResNet101, ResNet152, and Xception CNN architectures were used and compared for waste classification. In nine categories, the accuracy ranged from 91.9% to 94.6%. The MobileNetV3 architecture achieved the highest accuracy value of 94.26%. Furthermore, the MobileNetV3 architecture had the smallest size of 49.5 MB and the shortest duration of 261.7 ms. According to Aral et al. [15], waste recycling is critical for the global economy and climate balance, so classifying recyclable waste is critical for humanity. The architectures Densenet121, DenseNet169, InceptionResnetV2, MobileNet, and Xception were used in the study. The DenseNet121 architecture yielded the highest accuracy value of 95%. Zhang et al. [16] presented how smart systems can be used to classify waste, which is a crucial step toward achieving sustainable development for people. It was emphasized that smart systems should take the place of traditional waste classification because it is ineffective. With the transfer learning model, DenseNet169 CNN architecture was chosen as the most accurate, with an accuracy value of 82%. Gyawali et al. [17] proposed a CNN model for automated waste classification to assist recycling. The TrashNet dataset was expanded and used in the proposed model to eliminate human intervention. Using the Resnet18 architecture, an accuracy of 87.8% was achieved. Rabano et al. [18] stated that waste classification is the first step in recycling and reusing waste. For this purpose, MobileNet CNN architecture was chosen to classify waste. Transfer learning and TrashNet dataset were used. The accuracy value of 87.2% was reached on the test data. Feng et al. [19] proposed an enhanced GECM-EfficientNet model. In this model, they replaced the squeeze-and-excitation (SE) module from the EfficientNet architecture with the efficient channel attention (ECA) module. With the proposed model, they achieved a classification accuracy of 94.23% on the TrashNet test dataset.

Lin et al. [20] trained models using transfer learning with five different ResNet architectures on the TrashNet dataset. They used data augmentation techniques to increase the number of samples in the dataset before training the models. In this study, they achieved an accuracy of 88.8% and an F1 score of 88.9% with the RWNet-152 model. The ResNet architecture was one of the first architectures to address the vanishing gradients problem in CNN models by introducing residual connections.

In the existing studies, multilayer hybrid convolutional neural network-based customized CNN models have also been proposed. Shi et al. [21] emphasized that existing models for waste classification still struggle with issues like a poor success rate and a long working time. The study focused on the solution of these problems through the use of a Hybrid CNN network. CNN architecture is similar to VGGNet in structure, but there are fewer parameters. The accuracy rate was reported to be 92.6%. Yang et al. [22] proposed a model based on MLH-CNN. With the proposed model, they achieved a classification accuracy of 93.72% on the TrashNet test dataset. The accuracy of the proposed models was improved by applying image preprocessing techniques. Another custom CNN model was presented by Bobulski and Kubanek [23]. They conducted a study to increase the recycling rate by classifying plastic waste. In the study, the WaDaBa dataset was utilized. The system was developed to classify plastic waste using a microcomputer system with a color camera and image processing software. The study's success rate was 74%. Riba et al. [24] conducted a study to separate recyclable waste textile products. It aims to solve environmental problems while also producing high-quality recycling. For the study, a unique dataset was created. A new CNN architecture is proposed in the study to classify textile waste. The proposed system achieved 91.1% accuracy. Tran and Nguyen [25] proposed a customized CNN model consisting of five blocks that incorporate residual connections. The proposed model leverages the strengths of convolution layers, depthwise separable convolution layers, average pooling layers, batch normalization method, and two connectors to extract feature maps and optimize the network parameters. They achieved an accuracy of 90.71% on the TrashNet dataset.

Object detection-based waste management systems have also started to be widely implemented. Sallang et al. [26] created a smart trash can in their study. The Raspberry Pi 4 placed in this trash can was used to classify waste. TensorFlow Lite is used to create a new CNN architecture for the IoT system. The architecture was trained using a dataset created specifically for this study, and an accuracy of 87% was achieved. Melinte et al. [27] worked on improving the efficiency of object detectors by employing CNN architecture for waste classification in municipalities. The study's goal is to improve the sensitivity and performance of object detection devices, as well as their generalization and detection speed. The study made use of the TrashNet dataset. The accuracy of the R-CNN-created architecture is 95.76%. Nowakowski and Pamula [28] introduced two CNN architectures in their study for the classification of electrical and electronic waste. A mobile and web-based system was developed. After utilizing the dataset created specifically for this study, the first architecture achieved an accuracy of 96.7%, while the second architecture achieved an accuracy of 93.3%.

Alrayes et al. [7] proposed a Vision Transformer based on Multilayer Hybrid Convolution Neural Network for automatic waste classification. They achieved a classification accuracy of 95.8% on the TrashNet dataset. The Vision Transformer method has also become widely used in image classification. In this study, it has been stated that the Vision Transformer method outperforms transfer learning-based CNN methods in terms of performance. The iteration times during the training phase of the models were compared, but no information was provided regarding the prediction time of the models for classifying an image in real-time implementation.

Although many models have been developed in the literature to make automatic classifications in waste management systems, the success rates have not reached the desired level. Especially in waste classification, the similarity of glass and plastic wastes at some points affects the model performances. Further investigation is required to ascertain methods for improving the performance of the classifier when dealing with a dataset that has limited samples and significant similarities between different classes. In this study, architectures that will produce the optimum result have been obtained with extensive experimental studies.

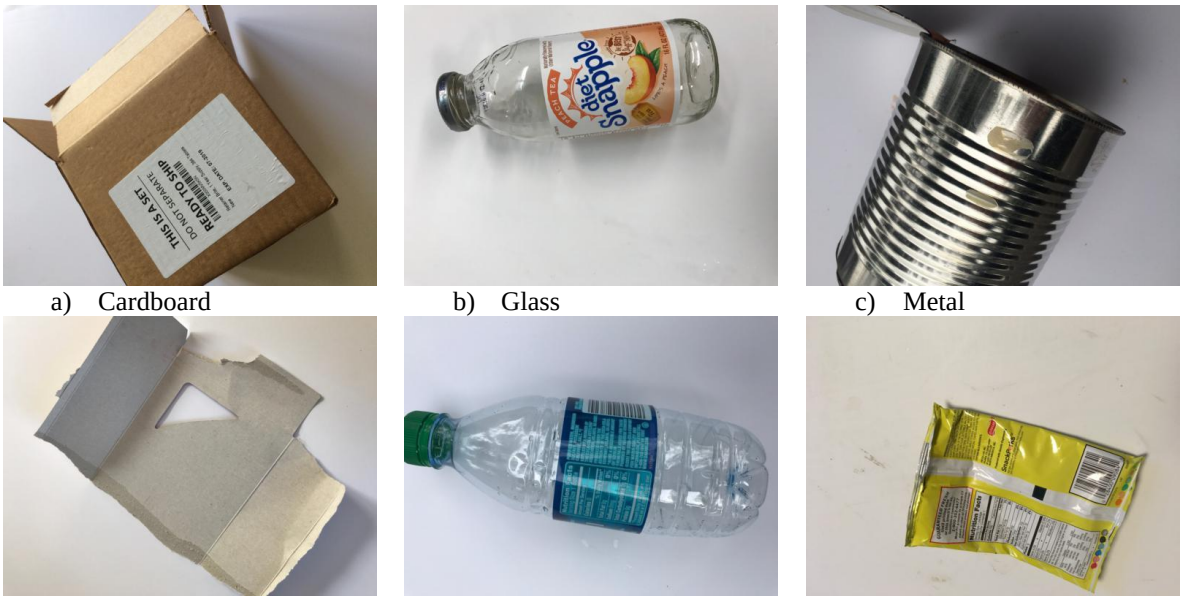
3. Materials and method

3.1 Dataset

The proposed study used the TrashNet [29] dataset. TrashNet data is divided into six categories: glass, paper, cardboard, plastic, metal, and other (garbage). The total number of images is 2527, with 501 in the glass category, 594 in the paper category, 403 in the cardboard category, 482 in the plastic category, 410 in the metal category, and 137 in the other category. The images in the dataset are 512x384 pixels in size. The dataset is 47 MB large. Figure 1 depicts some examples from the dataset.

The proposed model divides the dataset into training (80%) and test (20%) datasets. For the training, 323 cardboard, 401 glass, 328 metal, 476 paper, 386 plastic, and 110 garbage (other) images were used. Furthermore, 200 images representing 10% of the training images were used for validation. The test included 503 images, including 80 cardboard, 100 glass, 82 metal, 118 paper, 96 plastic, and 27 garbage (other). As a result, the proposed model's success was measured using data that had never been used before.

Data augmentation techniques were used to increase the number of images in the dataset. By doing this, the dataset was increased, and the model was kept from memorizing information during training. The employed techniques are as follows: 30-degree rotation (rotation_range), 0.2 percent shift (width_shift_range, height_shift_range), 0.2 shear_range, 0.2 zoom (zoom_range), and y-axis flip (horizontal flip). In this study, during the training phase, one of the specified data augmentation techniques was randomly applied to all images (the train dataset size) for each epoch. This approach ensures that the model sees different images in each epoch, thereby mitigating the problem of overfitting or memorizing the dataset. Data augmentation was not applied to the dataset before the training phase. In the test dataset, data augmentation techniques were not used.



a) Cardboard b) Glass c) Metal

d) Paper e) Plastic f) Other

Figure 1 Images from the TrashNet dataset

3.2 Convolutional neural network

Convolutional neural networks belong to a class of artificial neural networks. It is primarily employed in image processing and recognition applications. Using images as input, it learns the features of images through various layers. In this manner, it can perform image classification or object recognition.

CNNs have been used as image classifiers in most computer vision fields, requiring a simple and high-accuracy classifier [30]. In classical machine learning models, the problem-specific features were first determined manually. The feature vector was used to perform classification on the dataset. The biggest innovation that deep learning models add to this field is that feature vectors are generated automatically from the dataset during the training phase rather than manually. As a result, in CNN architectures, the number of filters and filter sizes in each convolutional layer is determined. During the training phase, the models will update these filter weights to determine the best features automatically for classification [6].

The AlexNet CNN was proposed in the ImageNet image recognition competition in 2012 and achieved the best performance [5]. This performance represents a breakthrough in computer vision. CNNs have since become the most widely used artificial neural network in image classification as a result of this research. CNN scans image segments and extracts features to recognize objects in those segments. Again, these features are used to define what the objects in the image are. CNN is built in layers, with each layer building on the previous layer's feature maps [5, 6]. It identifies objects in the image more accurately this way. A new CNN architecture is proposed in this study to classify waste.

3.3 Proposed model

Deep learning models can successfully classify the desired image according to different depth, width, and hyperparameter properties. Our aim in this study is to increase classification performance by designing these properties optimally. In the input layer of the proposed model, images were tested using 64x64, 128x128, 180x180, and 224x224 pixels. The most effective dimension for increasing the success rate was determined to be 180x180 pixels. As a result, 180x180 pixels images were used in the model's input layer. Figure 2 depicts the proposed model, which consists of four convolutional layers. The convolution layer’s filter numbers were determined to be 16, 32, 64, and 128, respectively. Each layer's filter (kernel) size was determined

to be 3x3. Activation functions were used to learn about any continuous and complex relationship between network variables [31]. To avoid linearity in the network, the ReLu activation function was used after all convolution layers.

The pooling layer was used in the network to reduce the feature map dimensions. The largest value in the filter window determines the size of the filter window in max pooling; in average pooling, the average of all pixels in the window is kept as a single value in the output pixel [32]. The maximum pooling layer was added after each convolution layer in the proposed model. The BatchNormalization layer was added after the activation layer. However, it was discovered that this layer had no positive effect on the model's success and was removed. Flattening took place after the last pooling layer but before the fully connected layer. The multidimensional feature array was thus reduced to a one-dimensional array and fed into the fully connected layers [33]. As a result, the flattening process was carried out after the last max-pooling layer, with the addition of a flattening layer.

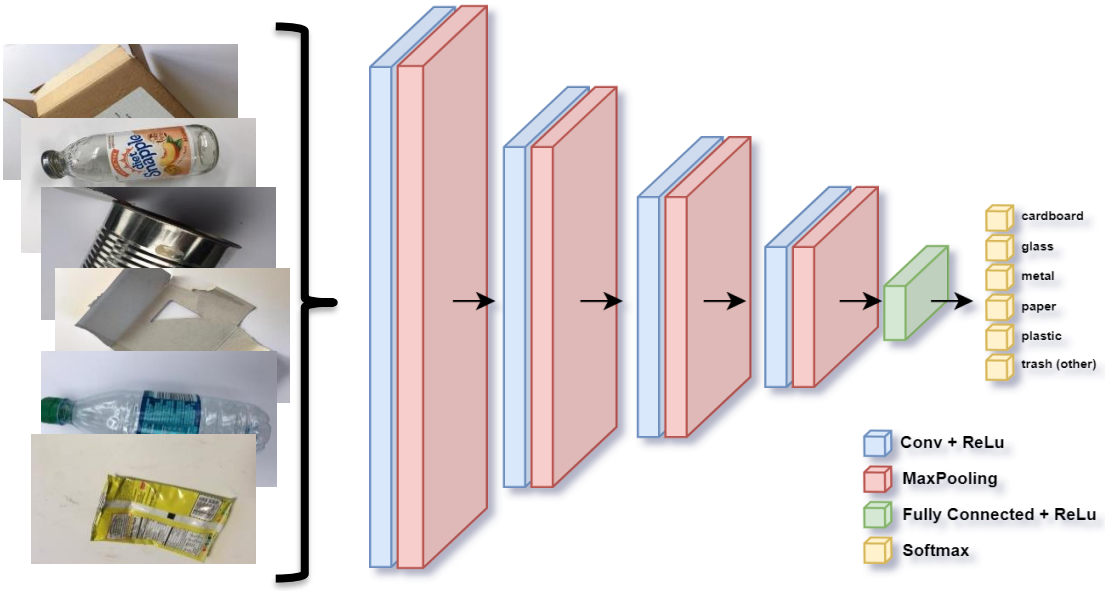


Figure 2 Proposed model

In CNN, the fully connected layer predicts the results using the features obtained by the convolution and pooling layers [33-35]. That is, the previous layers' output is used as the input in the fully connected layers. In this manner, it predicts the outcomes based on the features provided as input. The fully connected layer was represented by 512 units in our proposed model. The last layer added was the output layer. A dense layer with six neurons was added because the model will classify into six categories. The nonlinear Softmax activation function, which is used for multi-classification, was used in this layer. Softmax produces an output indicating the likelihood that the given input belongs to a class.

3.4 Performance metrics

The Confusion Matrix [36, 37] is a matrix created to interpret the results of a created model and to cross-examine the relationships between the actual and predicted values. This matrix contains four parameters.

• True Positive (TP): If a circumstance that is generally positive in the estimating process is projected to be positive.

• True Negative (TN): If the current situation is negative and the forecast is negative.

• False Positive (FP): If the current state is negative, but the estimating system expects a positive state.

• False Negative (FN): If the current state is positive and the estimator produces a negative outcome.

Accuracy, precision, recall, and F1-score metrics were used in the study [38-40]. Performance metrics are given in Equations 1-4, which are calculated from the confusion matrix.

$$\boldsymbol{Accuracy} = \frac{\boldsymbol{TP} + \boldsymbol{TN}}{(\boldsymbol{TP} + \boldsymbol{TN} + \boldsymbol{FP} + \boldsymbol{FN})} \quad (1)$$

$$Precision = \frac{TP}{TP + FP} \quad (2)$$

$$Recall = \frac{TP}{TP + FN} \quad (3)$$

$$F1\ Score = 2 * \frac{Recall * Precision}{Recall + Precision} \quad (4)$$

In cases where the class distribution of a dataset is imbalanced, it is necessary to consider the F1-score metric alongside accuracy for comparative purposes. To thoroughly assess the performance of a model, it is important to evaluate both precision and recall. The F1 score is a valuable metric that takes into account both of these measures. To summarize the model's performance in a more balanced manner, we utilize the F1-score, which is the harmonic mean of precision and recall values.

4. Experimentation and results

The TrashNet dataset was employed, utilizing pre-trained architectures from the ImageNet dataset. Initially, these architectures were loaded using the Keras deep learning framework, an open-source tool, with TensorFlow serving as the backend. Subsequently, the models were fine-tuned specifically for the TrashNet dataset. The experimental setup involved a Standard PC equipped with 16 GB RAM, an NVIDIA GeForce GTX 1080 Ti GPU with 11 GB memory, and an Intel i5-8400 processor operating at 2.80 GHz.

4.1 Hyperparameter optimization of the CNN architectures

In this study, many models with varying depth, width, and other hyperparameter values were developed. Table 1 displays the investigated hyperparameters and values. Table 2 shows the developed models with an accuracy value greater than 80% and the hyperparameter values for these models.

Table 1 Hiperparameters and values

Hyperparameter	Values
Number of convolutional layers (NCL)	4, 5, 6, 7, 9, 11, 13, 15
Number of filters (NF)	16, 32, 64, 128, 256, 512
Input Image Dimension, pixels (IID)	64, 128, 180, 224, 256, 512
Kernel size (KS)	3, 5, 7
Number of dense layers (NDS)	1, 2, 3, 4, 5
Epoch (Epo)	120, 150, 200, 300
Batch size (BS)	8, 16, 32
Optimizer (Opt)	Adam, Nadam, RmsProp, SGD, Adamax, Adadelta

The optimization strategy consists of generating CNNs with predetermined hyperparameters, training them iteratively, and identifying the optimal set of hyperparameters among these constructed networks. CNNs belong to the category of deep learning techniques that can learn end-to-end [5]. In the training phase, CNNs employ the backpropagation algorithm to update their filter weights and facilitate the learning process [6]. The training phase of CNNs is computationally expensive. In the case of waste classification, training a CNN network takes approximately 20 minutes. Considering all the hyperparameters in Table 1, there are a total of 181,440 different combinations of CNN networks. Training and comparing all these combinations would require a significant amount of time. Due to the large number of combinations, the Random Search method, which works on a limited number of combinations, can be applied. However, since Random Search does not remember past model results, we applied a different method. In this study, a prioritized Manuel Search approach was adopted by initially focusing on selected combinations based on our experience. Approximately 500 different CNN networks were trained, and out of these, 21 were reported.

In CNN architectures, increasing the depth (number of convolutional layers) and width (number of filters) hyperparameters enhances the model's learning capacity during the training phase [6]. However, as the models have more network weight parameters than the limited dataset, the risk of overfitting arises. To mitigate the problem of overfitting, regularization techniques such as L2 regularization, dropout, batch normalization, and data augmentation are employed [8]. However, these methods may not be sufficient for limited datasets. CNNs extract features from raw input images and perform classification tasks [6]. When we increase the size of input images, we often require deeper networks for better learning capacity.

In CNN architectures, the early layers learn basic features such as edges and color blobs, while the deeper layers learn more complex structures specific to the dataset [10]. In this study, we primarily focused on determining the depth of the model, the number of filters in each layer, and the size of the kernels for waste classification. Based on the obtained basic CNN architectures, we then observed the effects of other hyperparameters like optimizers, epochs, and batch sizes to achieve the optimal model. Increasing the depth and number of filters in a model leads to a higher success rate during the training phase. However, due to overfitting in the training phase, the performance of the model on unseen test datasets decreases. Deep models require a substantial amount of data to learn and generalize statistical patterns from the dataset during the training

phase. Therefore, pre-trained models trained on large-scale datasets (such as ImageNet) with millions of samples have been successfully utilized through transfer learning in solving current problems.

Table 2 Experimental studies

M No	NCL	NF	KS	IID	NDL	Epoch	BS	Opt	Acc	F1 score	CPU time (ms)
1	5	16, 16, 32, 32, 128	3, 3, 3, 3, 3	180	2	150	16	SGD	0.80	0.80	334.80
2	7	16,16,16,32,64,128, 256	3, 3, 3, 5, 5, 5, 7	224	3	150	16	SGD	0.81	0.81	443.76
3	5	16, 32, 32, 64, 64	3, 3, 5, 5, 5	128	4	150	16	Nadam	0.81	0.80	261.22
4	6	16, 32, 32, 64, 64, 128	3, 3, 3, 5, 7, 7	128	4	150	16	Nadam	0.82	0.81	301.89
5	6	16, 16, 32, 64, 128, 256	3, 3, 3, 3, 3, 7	224	3	150	16	SGD	0.82	0.82	410.76
6	6	16, 32, 32, 64, 128, 256	3, 3, 3, 3, 7, 7	224	3	150	16	SGD	0.82	0.81	442.71
7	6	8, 16, 32, 64, 128, 256	3, 3, 3, 3, 7, 7	224	3	150	16	SGD	0.83	0.83	432.75
8	5	8, 16, 32, 64, 128	3, 3, 3, 3, 3	256	4	120	16	Adam	0.83	0.83	401.56
9	6	16, 16, 32, 64, 128, 256	3, 3, 3, 3, 3, 7	256	3	150	16	SGD	0.84	0.83	421.47
10	7	16, 16,16,32,64,128, 128	3, 3, 5, 5, 5, 7, 7	128	2	150	16	Adam	0.84	0.83	337.80
11	6	16, 32, 32, 32, 64, 128	3, 3, 3, 3, 5, 5	224	3	150	16	RmsProp	0.84	0.84	379.88
12	5	16, 16, 32, 32, 128	3, 3, 3, 5, 7	180	3	150	32	Adam	0.84	0.83	358.13
13	5	16, 16, 32, 32, 128	3, 3, 3, 5, 5	180	2	150	16	SGD	0.84	0.83	340.62
14	6	16, 16, 32, 64, 128, 256	3, 3, 3, 3, 3, 7	224	3	150	16	SGD	0.85	0.84	410.73
15	5	16, 32, 64, 128, 256	3, 3, 3, 3, 7	224	3	150	16	SGD	0.86	0.86	332.77
16	5	16, 32, 64, 128, 256	3, 3, 3, 3, 7	224	3	150	16	Adamax	0.86	0.86	349.82
17	6	16, 32, 64, 128, 256, 256	3, 3, 3, 3, 7, 7	224	3	150	16	Adamax	0.86	0.86	426.80
18	5	16, 32, 64, 128, 256	7, 3, 3, 5, 7	180	3	300	16	Adamax	0.87	0.87	351.76
19	5	16, 32, 64, 128, 256	7, 3, 3, 5, 7	180	3	300	16	Adam	0.87	0.87	370.76
20	5	16, 32, 64, 128, 256	7, 3, 3, 5, 7	180	1	300	16	Nadam	0.88	0.87	248.63
21	4	16, 32, 64, 128	3, 3, 3, 3	180	1	300	16	Nadam	0.89	0.88	239.86

M No: Model No, Acc: Accuracy, and other abbreviations are given in Table 1.

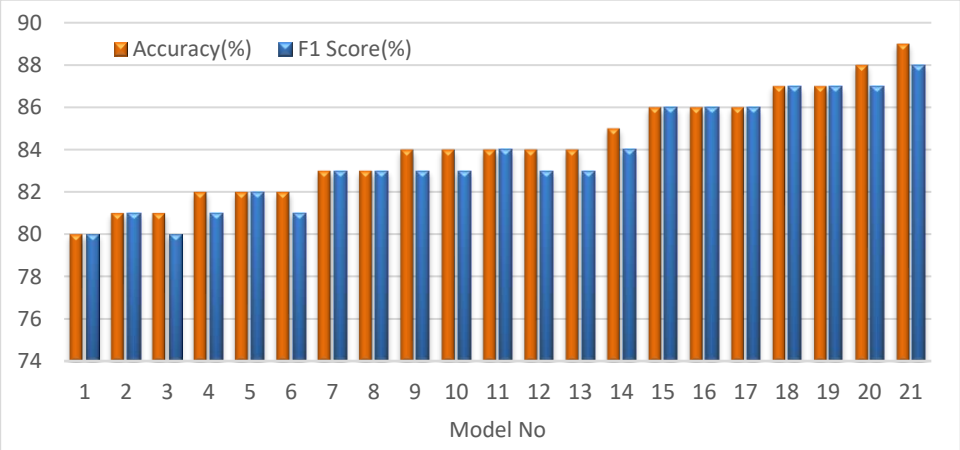


Figure 3 Accuracy and F1 score values of experimental studies for custom CNN models

The following study was carried out with the best-performing models among the derived models during the experimental studies shown in Table 2. The real-time classification time of models is important for waste classification. The best model, Model 21, achieved a classification time of 239.86 ms CPU time for a single image. The accuracy and F1 scores of the experimental studies can be analyzed in Figure 3. In the final proposed model (Model 21), four convolution layers were used to determine the best hyperparameters. Convolution layer filter numbers are 16, 32, 64, and 128, respectively. The size of the filters used in the layers is 3x3 on condition that all layers are equal. A 2x2 pooling (MaxPooling) layer was added after each conv layer. Because it produces better results in convolution layers, the ReLu activation function was used. In the classifier layer, the Softmax activation function was used. The epoch number was set to 300. Furthermore, the best performance was obtained with DenseNet169 architecture in the transfer learning study, with an accuracy of 96.42% and an F1 score of 96%.

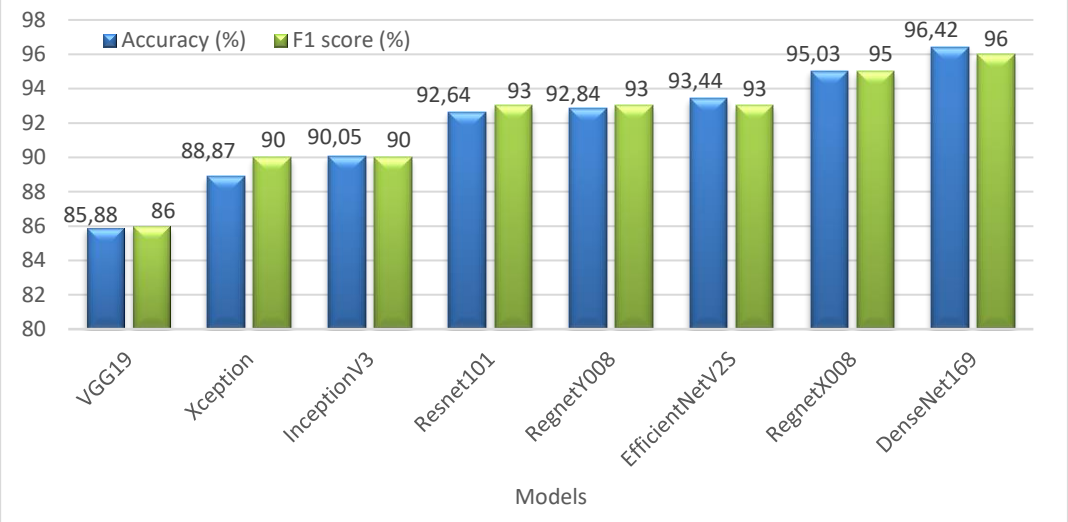


Figure 4 Accuracy and F1-scores of the state-of-the-art CNN models (with transfer learning and fine-tuning)

Figure 4 presents the accuracy and F1 scores of the state-of-the-art CNN models. Several state-of-the-art CNN models were trained through hyperparameter optimization. We keep the convolution layers of these models and remove the layers after the last convolution layer. After the last convolution layer, a Global Average Pooling, two dense and dropout layers are added. Finally, since we made a six-category classification, a dense layer with six neurons was added. Dense layers were optimized using different neuron numbers between 128 and 1024. In the dropout layer, the previous neurons were ignored with a ratio between 0.2 and 0.7, and the optimum ratio was found. In DenseNet169 architecture, the last fully connected layers were eliminated, a 256-node fully connected layer was created, and then a 128-node fully connected layer was added. A dropout with the 0.2 ratio was added after the 256-node fully connected layer. As a result, this architecture reduced the number of parameters from 12 million to about 5 million.

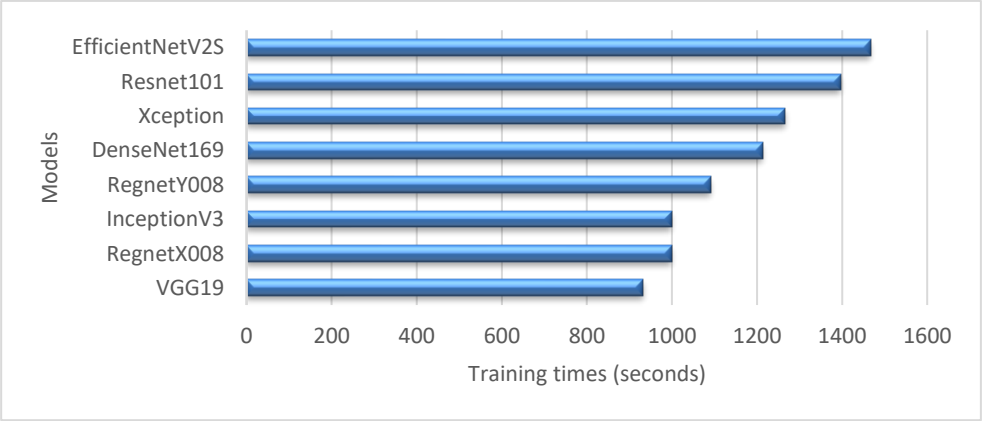


Figure 5 Training times of the state-of-the-art CNN models (with transfer learning and fine-tuning)

Figure 5 displays the training times of models on the TrashNet dataset using transfer learning. Since a large number of combinations were attempted in this study and the aim was to find the most optimal hyperparameter combination, training times became significant. Training the DenseNet169 model for 50 epochs takes approximately 20 minutes.

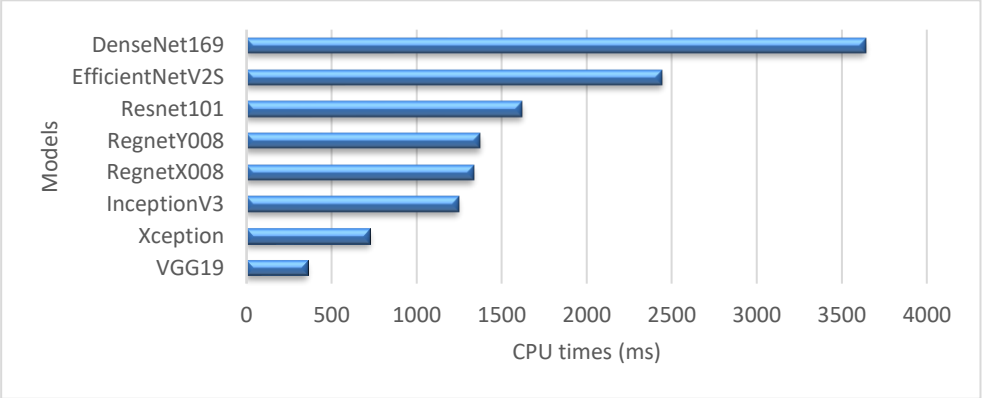


Figure 6 Prediction times (one image) of the state-of-the-art CNN models (with transfer learning and fine-tuning)

When examining Figure 6, it can be observed that the DenseNet169 model, which is the best-performing model, has a significantly longer inference time for predicting a single image compared to Model 21 in Table 2. The inference time for the DenseNet169 model to classify a single image is approximately 3.6 seconds. This duration is higher than the expected values for real-time implementations. Therefore, the proposed Model 21, which even predicts the class of an image in a shorter time (0.24 seconds) than the VGG19 model (0.36 seconds), can be utilized in waste management systems. Additionally, cross-platform support can be added as work moves forward. In this way, versions such as mobile and server-based systems can be derived [41, 42].

4.2 Confusion matrix

The confusion matrix of the proposed model is shown in Figure 7. When examined as an example in Figure 7, out of a total of 80 cardboard images in the test data, our model correctly predicted 72 of them. These 72 images correctly classified as Cardboard represent our true positive (TP) value. When analyzing the true labels in the horizontal row for Cardboard, our model made erroneous predictions for 4 images, classifying them as Glass instead of Cardboard. Similarly, it misclassified 2 images as Paper and 2 images as Plastic instead of correctly identifying them as Cardboard. These 8 images (4+2+2=8), which our model misclassified as Cardboard, represent the false negative (FN) value for our Cardboard class. Upon reviewing the column values for the Cardboard class in Figure 7, we can see the predictions made by our model. In these columns, our model erroneously labeled 3 images as Cardboard instead of Glass. Similarly, it misclassified 2 images as Cardboard instead of Plastic and 1 image as Cardboard instead of Trash. These 6 images (3+2+1=6) correspond to the false positive (FP) values for the Cardboard class. The same procedure is followed to interpret the confusion matrix values for the remaining classes, enabling the calculation of total FN and FP values for each class in Table 3. Precision, recall, and F1-score values for each class are computed by substituting the TP, FN, and FP values obtained from the confusion matrix into Equations 2-4.

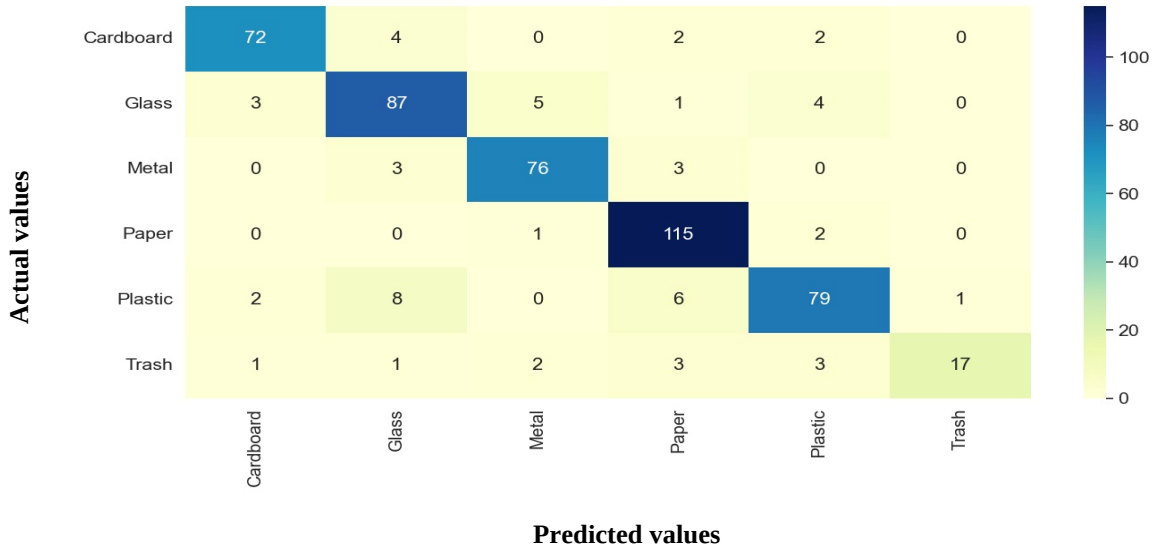


Figure 7 Proposed model confusion matrix

The confusion matrix of the study with the DenseNet169 architecture using the transfer learning method is presented in Figure 8. When the confusion matrix in both models is examined, it is seen that the most inaccurate cases are caused by the glass and plastic classifications.

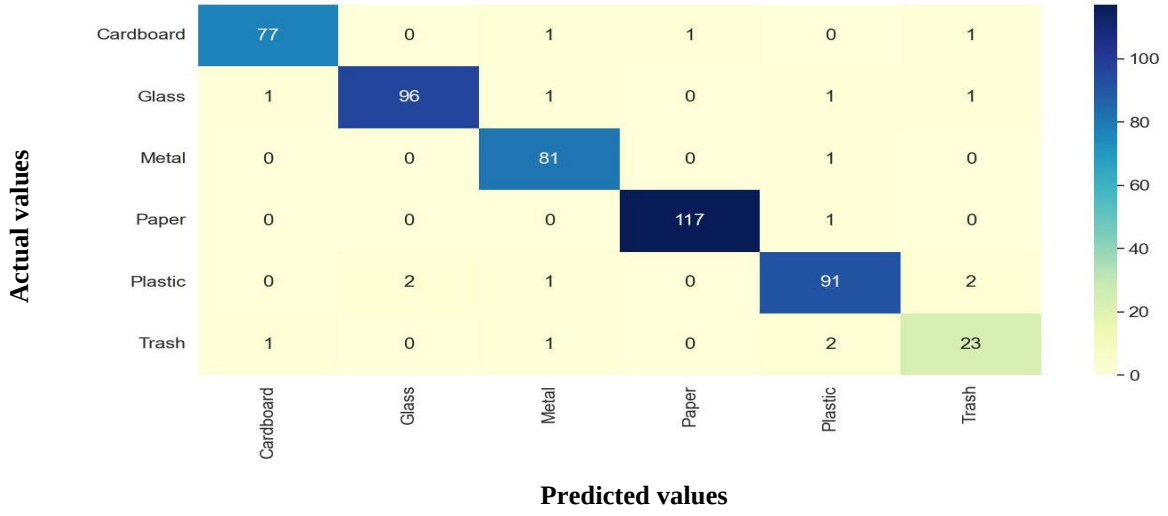


Figure 8 DenseNet169 confusion matrix

When the confusion matrix in both models is examined, the most incorrect cases are found in the glass and plastic classification. These two types of trash are quite similar to each other in some images. For this reason, it can be seen that the models are unstable in classifying them. For these two types of trash, solutions can be offered with data augmentation or handcrafted features. These handcrafted features can be integrated with CNN architecture.

4.3 Performance metrics

The proposed model was tested with 503 images. The model's performance metrics were measured as an accuracy of 88.66% and an F1 score of 88%. By utilizing the confusion matrix (see Section 4.2), we can substitute the TP, FN, and FP values obtained for each class into Equations 2-4 to determine the precision, recall, and F1-score values for each class. When we represent the F1-score using the True Positive (TP), False Positive (FP), and False Negative (FN) values, we obtain the following equation: F1 = TP / (TP + 0.5 (FP+FN)). Table 3 provides the calculated performance metric values for each class in multiclass datasets, utilizing the TP, FP, and FN values obtained from the confusion matrix table.

Table 3 Calculating precision, recall, and F1-score values for each class using the confusion matrix for model 21.

	TP	FP	FN	Precision	Recall	F1-score	Support
Cardboard	72	6	8	72/(72+6) = 0.92	72/(72+8) = 0.90	72 / (72 + 0.5 (6+8)) = 0.91	80 / 503 = 0.16
Glass	87	16	13	87/(87+16) = 0.84	87/(87+13) = 0.87	87 / (87 + 0.5 (16+13)) = 0.86	100 /503 = 0.20
Metal	76	8	6	76/(76+8) = 0.90	76/(76+6) = 0.93	76 / (76 + 0.5 (8+6)) = 0.92	82 /503 = 0.16
Paper	115	15	3	115/(115+15) = 0.88	115/(115+3) = 0.97	115 / (115 + 0.5 (15+3)) = 0.93	118/503 = 0.23
Plastic	79	11	17	79/(79+11) = 0.88	79/(79+17) = 0.82	79 / (79 + 0.5 (11+17)) = 0.85	96/503 = 0.19
Trash (Other)	17	1	10	17/(17+1) = 0.94	17/(17+10) = 0.63	17 / (17 + 0.5 (1 + 10)) = 0.76	27/503 = 0.05
Accuracy						0.89	503
Macro avg				0.90	0.85	0.87	503
Weighted avg				0.89	0.89	0.88	503

The results of the measured metrics are given in Table 3. To calculate the macro-averaged precision, recall, and F1 scores, the arithmetic mean (or unweighted mean) of the per-class F1 scores is computed. To calculate the weighted-averaged precision, recall, and F1 scores, the mean of all the per-class F1 scores is computed, considering each class's support. The individual metric value of each class is weighted according to the ratio of the number of images with their true labels. For example, according to the macro average, the F1 score value is calculated as (0.91+0.86+0.92+0.93+0.85+0.76)/6 = 0.87.

The performance metric results of the study with DenseNet169 were given in Table 4.

Table 4. DenseNet169 performance metrics

	Precision	Recall	F1-Score	Number of images
Cardboard	0.97	0.96	0.97	80
Glass	0.98	0.96	0.97	100
Metal	0.95	0.99	0.97	82
Paper	0.99	0.99	0.99	118
Plastic	0.95	0.95	0.95	96
Trash (Other)	0.85	0.85	0.85	27
-	-	-	-	-
Accuracy			0.96	503
Macro avg	0.95	0.95	0.95	503
Weighted avg	0.96	0.96	0.96	503

4.4 Accuracy and loss charts

When the accuracy graph shown in Figure 9 is examined, the training and test accuracy curves increase gradually. The validation curve follows the training curve somewhat from below. Accordingly, a small amount of memorization is observed, but the training of the model is successful as the validation curve follows the training curve in a parallel manner and there is no decrease in the validation curve in the following stages.

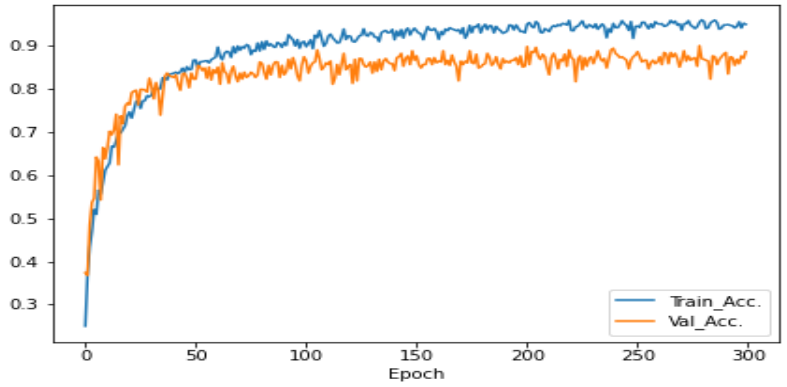


Figure 9. Training and test accuracy graph

In the loss graph in Figure 10, it is seen that the loss value decreases during the training and validation phases. In addition, the loss value in the validation phase follows the training phase from above. It has been tried to prevent memorization with Dropout methods.

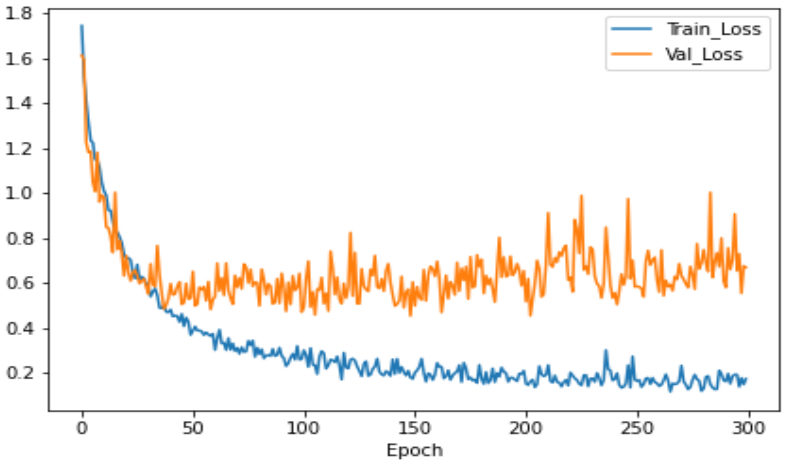


Figure 10. Training and test loss graph

Table 5 presents a comparison of the proposed model with existing studies. Most of the existing models have increased their model accuracy by transfer learning. Since state-of-the-art CNN models are very comprehensive models, good performances can be achieved in the desired dataset with the right hyperparameters. In this study, we achieved better results than existing

studies with hyperparameter optimization. All existing studies in Table 5 have used the TrashNet dataset, and the performance values on the TrashNet test dataset have been presented.

Table 5 Comparison of the recommended model with existing CNN methods using the TrashNet dataset

Model	Year	Accuracy (%)	F1-Score (%)
Rabano et al. [18]	2018	87.2	-
Gyawali et al. [17]	2020	87.8	-
Lin et al. [20]	2022	88.8	88.9
Tran and Nguyen [25]	2022	90.71	-
Shi et al. [21]	2021	92.6	91
Yang et al. [22]	2022	93.72	-
Feng et al. [19]	2022	94.23	-
Wang et al. [14]	2021	94.24	94
Alrayes et al. [7]	2023	94.7	-
Bircanoğlu et al. [13]	2018	95	-
Aral et al. [15]	2018	95	-
Proposed Model	2023	96.42	96

5. Conclusion

Urban waste is a major issue in many countries around the world. Recycling is regarded as the most effective method of reducing urban waste. This study aims to enhance the effectiveness of waste classification in recycling through the optimization of various CNN models. In the study, a new CNN architecture was proposed for the classification of wastes in six categories using the TrashNet dataset. The proposed model has achieved an accuracy of 88.66% and an F1 score of 88%. Using the DenseNet169 architecture and transfer learning, an accuracy of 96.42% and an F1 score of 96% were achieved. This research can help city governments and recycling facilities classify waste and create an efficient waste management system. Thanks to the small size and fast operation of the model, waste classification can be performed without the need to transport it to facilities. This can be achieved using smart garbage containers equipped with devices like Raspberry Pi. Additionally, reducing the need for manual labor in recycling, it will contribute to safeguarding the health of individuals involved in this sector. New models will continue to be evaluated in future studies. We also aim to expand the dataset to categorize waste into more specific categories and increase the accuracy rate.

References

[1] S. Kaza, L. C. Yao, P. Bhada-Tata and F. Van Woerden, "A Global Snapshot of Solid Waste Management to 2050," 2018, [Online]. Available: https://elibrary.worldbank.org/doi/abs/10.1596/978-1-4648-1329-0. [Accessed: 15-Dec-2022].

[2] D. Hoornweg and P. Bhada-Tata, *What a Waste: A Global Review of Solid Waste Management,* World Bank, Washington DC USA, 2012.

[3] R. E. Sanderson, *Environmental Protection Agency Office of Federal Activities' Guidance on Incorporating EPA's Pollution Prevention Strategy into the Environmental Review Process*, EPA, Washington, DC, USA, 1993.

[4] O. Adedeji and Z. Wang, "Intelligent Waste Classification System Using Deep Learning Convolutional Neural Network," *Procedia Manufacturing*, vol. 35, pp. 607-612, 2019.

[5] A. Krizhevsky, I. Sutskever and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Proc - Neural Information Processing System Conference*, pp. 1-9, 2012.

[6] Y. LeCun, Y. Bengio, & G. Hinton. "Deep learning," *Natur*e, vol. 521, pp. 436-444, 2015.

[7] F. S. Alrayes et al., "Waste classification using vision transformer based on multilayer hybrid convolution neural network," *Urban Climate*, vol. 49, pp. 1-14, 2023.

[8] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, & R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting.," *Journal of Machine Learning Research*, vol. 15, no.1, pp. 1929-1958, 2014.

[9] C. Tan, F. Sun, T. Kong, W. Zhang and C. Y. &. C. Liu, "A survey on deep transfer learning," *Proc. - 27th International Conference on Artificial Neural Networks*, pp. 270-279, 2018.

[10] J. Yosinski, C. Jeff , B. Yoshua ve L. Hod, "How transferable are features in deep neural networks?," *Advances in neural information processing systems*, 2014.

[11] K. He, X. Zhang, S. Ren and J. Sun, "Deep residual learning for image recognition" *Proc – IEEE conference onComputer Vision and Pattern Recognition*, pp. 770-778, 2016.

[12] G. Huang, Z. Liu, L. v. d. Maaten and K. Q. Weinberger, "Densely Connected Convolutional Networks," *Proc - IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4700-4708, 2017.

[13] C. Bircanoğlu, M. Atay, F. Beşer, Ö. Genç, & M. A. Kızrak, "RecycleNet: Intelligent waste sorting using deep neural networks," *Proc - 2018 Innovations in intelligent systems and applications*, pp. 1-7, 2018.

[14] C. Wang, J. Qin, C. Qu, X. Ran, C. L. b and B. Chen, "A Smart Municipal Waste Management System Based on Deep-Learning and Internet of Things," *Waste Management*, vol. 135, pp. 20-29, 2021.
[15] R. A. Aral, Ş. R. Keskin, M. Kaya and M. Hacıömeroğlu, "Classification of TrashNet Dataset Based on Deep Learning Models," *Proc - International Conference on Big Data*, pp. 2058-2062, 2018.
[16] Q. Zhang, Q. Yang, X. Zhang, Q. Bao, J. Su and X. Liu, "Waste image classification based on transfer learning and convolutional neural network," *Waste Management*, vol. 135, pp. 150-157, 2021.
[17] D. Gyawali, A. Regmi, A. Shakya, A. Gautam and S. Shrestha, "Comparative Analysis of Multiple Deep CNN Models for Waste Classification," 2020, [Online]. Available: https://arxiv.org/abs/2004.02168. [Accessed: 10-Dec-2022].
[18] S. L. Rabano, M. K. Cabatuan, E. Sybingco, E. P. Dadios and E. J. Calilung, "Common Garbage Classification Using MobileNet," *Proc - IEEE 10th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management*, pp. 1-4, 2018.
[19] Z. Feng, Yang, J., Chen, L., Chen, Z., & Li, L., "An Intelligent Waste-Sorting and Recycling Device Based on Improved EfficientNet," *International Journal of Environmental Research and Public Health*, vol. 19, no. 23, pp. 1-18, 2022.
[20] K. Lin, Zhao et al, "Applying a deep residual network coupling with transfer learning for recyclable waste sorting," *Environmental Science and Pollution Research*, vol. 29, no. 60, pp. 91081-91095, 2022.
[21] C. Shi, C. Tan, T. Wang and L. Wang, "A Waste Classification Method Based on a Multilayer Hybrid Convolution Neural Network," *Applied Science*, vol. 11, no 18, pp. 1-19, 2021.
[22] Z. Yang, Xia, Z., Yang, G., & Lv, Y. "A Garbage Classification Method Based on a Small Convolution Neural Network," *Sustainability*, vol. 14, no. 22, pp. 1-16, 2022.
[23] J. Bobulski and M. Kubanek, "Deep Learning for Plastic Waste Classification System," *Applied Computational Intelligence and Soft Computing*, vol. 2021, pp. 1-7, 2021.
[24] J.-R. Riba, R. Cantero, P. Riba-Mosoll and R. Puig, "Post-Consumer Textile Waste Classification through Near-Infrared Spectroscopy, using an Advanced Deep Learning Approach," *Polymers*, vol. 14, no. 12, pp. 1-14, 2022.
[25] B. G. Tran, & D. L. Nguyen. "Simple and Efficient Convolutional Neural Network for Trash Classification," *Proc - Annals of Computer Science and Information Systems*, pp. 255-260, 2022.
[26] N. C. A. Sallang, M. T. Islam, M. S. Islam and H. Arshad, " A CNN-Based Smart Waste Management System Using TensorFlow Lite and LoRa-GPS Shield in Internet of Things Environment," *IEEE Access*, vol. 9, pp. 153560-153574, 2021.
[27] D. O. Melinte, A.-M. Travediu and D. N. Dumitriu, "Deep Convolutional Neural Networks Object Detector for Real-Time Waste Identification," *Applied Sciences*, vol. 10, no. 20, pp. 1-18, 2020.
[28] P. Nowakowski and T. Pamula, "Application of Deep Learning Object Classifier to Improve E-waste Collection Planning" *Waste Management*, vol. 109, pp. 1-9, 2020.
[29] M. Yang, and G. Thung, "Classification of trash for recyclability status." CS229 project report 2016.1 (2016): 3.
[30] F. Hu, G.-S. Xia, J. Hu and L. Zhang, "Transfering Deep Convolutional Neural Networks for the Scene Classification of High-Resolution Remote Sensing Imagery," *Remote Sensing*, vol. 7, no. 11, pp. 14680-14707, 2011.
[31] R. Jain, P. Nagrath, G. Kataria, V. S. Kaushik, & D. J. Hemanth, "Pneumonia detection in chest X-ray images using convolutional neural networks and transfer learning," *Measurement*, vol. 165, pp. 1-10, 2020.
[32] L. Alzubaidi, J. Zhang, A. J. Humaidi, A. Al-Dujaili, Y. Duan, O. Al-Shamma, J. Santamaría, M. A. Fadhel and M. A.-A. &. L. Farhan, "Review of deep learning: concepts, CNN architectures, challenges, applications, future directions," *Journal of Big Data*, vol. 8, pp. 1-74, 2021.
[33] S. K. Sundararajan, B. Sankaragomathi ve D. S. Priya, "Deep Belief CNN Feature Representation Based Content Based Image Retrieval for Medical Images," *Journal of Medical Systems*, vol. 43, pp. 1-9, 2019.
[34] G. Li and N. Li, "Customs classification for cross-border e-commerce based on text-image adaptive convolutional neural network," *Electronic Commerce Research*, vol. 19, pp. 779-800, 2019.
[35] X. Y. Wu, "A hand gesture recognition algorithm based on DC-CNN," *Multimedia Tools and Applications*, vol. 79, pp. 9193-9205, 2020.
[36] S. V. Stehman, "Selecting and interpreting measures of thematic classification accuracy," *Remote Sensing of Environment*, vol. 62, no .1, pp. 77-89, 1997.
[37] S. M. Piryonesi and T. E. El-Diraby, "Data Analytics in Asset Management: Cost-Effective Prediction of the Pavement Condition Index," *Journal of Infrastructure Systems*, vol. 26, no. 1, pp. 1-25, 2020.
[38] D. M. W. Powers, "Evaluation: From Precision, Recall and F-Measure to ROC, Informedness, Markedness & Correlation," *Journal of Machine Learning Technologies*, vol. 2, pp. 37-63, 2011.
[39] K. M. Ting, C. Sammut and G. I. Webb, *Encyclopedia of machine learning*, New York: Springer Science & Business Media, 2011.
[40] M. Talo, U. B. Baloglu, Ö. Yıldırım and U. R. Acharya, "Application of deep transfer learning for automated brain abnormality classification using MR images," *Cognitive Systems Research*, vol. 54, pp. 176-188, 2019.
[41] Y. Çetin-Kaya, M. Kaya & A. Akdağ, "Route Optimization for Medication Delivery of Covid-19 Patients with Drones," *Gazi University Journal of Science Part C: Design and Technology*, vol. 9, no. 3, pp. 478-491, 2021.
[42] M. Kaya, and Y. Çetin-Kaya, "Seamless computation offloading for mobile applications using an online learning algorithm," *Computing*, vol. 103, no.5, pp. 771-799, 2021.

Conflict of interest
The author declares that there are no potential conflicts of interest.

Funding
This research did not receive a specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

Data availability
The datasets generated during and/or analyzed during the current study are available from the corresponding author upon reasonable request.

Ethical approval and informed consent
It is declared that during the preparation process of this study, scientific and ethical principles were followed, and all the studies benefited from are stated in the bibliography.

Sakarya University Journal of Computer and Information Sciences
http://saucis.sakarya.edu.tr/

DOI: 10.35377/saucis...1311014 RESEARCH ARTICLE

Predicting Effective Efficiency of the Engine for Environmental Sustainability: A Neural Network Approach

Beytullah Eren[1], **İdris Cesur**[2]

[1]Sakarya University, Faculty of Engineering, Department of Environmental Engineering, 54187, Sakarya, Türkiye
[2]Sakarya University of Applied Sciences, Faculty of Technology, Department of Mechanical Engineering, 54187, Sakarya, Türkiye

Corresponding author:
Beytullah Eren, Department of Environmental Engineering, Faculty of Engineering, Sakarya University, Sakarya, Türkiye
E-mail address:
beren@sakarya.edu.tr

Submitted: 08 June 2023
Revision Requested: 04 July 2023
Last Revision Received: 07 July 2023
Accepted: 17 July 2023
Published Online: 24 July 2023

Citation: Eren B. and Cesur I.. (2023). Predicting Effective Efficiency of the Engine for Environmental Sustainability: A Neural Network Approach. *Sakarya University Journal of Computer and Information Sciences.* 6 (2) https://doi.org/10.35377/saucis...1311014

ABSTRACT

Predicting engine efficiency for environmental sustainability is crucial in the automotive industry. Accurate estimation and optimization of engine efficiency aid in vehicle design decisions, fuel efficiency enhancement, and emission reduction. Traditional methods for predicting efficiency are challenging and time-consuming, leading to the adoption of artificial intelligence techniques like artificial neural networks (ANN). Neural networks can learn from complex datasets and model intricate relationships, making them promise for accurate predictions. By analyzing engine parameters such as fuel type, air-fuel ratio, speed, load, and temperature, neural networks can identify patterns influencing emission levels. These models enable engineers to optimize efficiency and reduce harmful emissions. ANN offers advantages in predicting efficiency by learning from vast amounts of data, extracting meaningful patterns, and identifying complex relationships. Accurate predictions result in better performance, fuel economy, and reduced environmental impacts. Studies have successfully employed ANN to estimate engine emissions and performance, showcasing its reliability in predicting engine characteristics. By leveraging ANN, informed decisions can be made regarding engine design, adjustments, and optimization techniques, leading to enhanced fuel efficiency and reduced emissions. Predicting engine efficiency using ANN holds promise for achieving environmental sustainability in the automotive sector.

Keywords: Engine efficiency prediction, Environmental sustainability, Artificial neural networks (ANN), Emission reduction, Fuel efficiency enhancement

1. Introduction

Predicting the effective efficiency of engines for environmental sustainability has become a crucial area of research and development in the automotive industry. The accurate estimation and optimization of the effective efficiency of engines play a significant role in making informed decisions for vehicle design and adjustments, enhancing fuel efficiency, and reducing emissions. The automotive industry is constantly seeking innovations and improvements due to the impact of rapidly advancing technologies. In this process, accurately predicting and optimizing automobile efficiency is crucial. Accurately evaluating effective efficiency can aid in making better engine design and adjustment decisions, increasing fuel efficiency, and reducing emissions. Predicting effective efficiency is critical for the automotive industry to reduce costs and offer more competitive and innovative products. Accurate predictions result in better performance, fuel economy, and driving experience while reducing environmental impacts. Therefore, predicting effective efficiency plays a significant role in the automotive sector's research and development efforts and decision-making processes.

Predicting automobile effective efficiency using traditional methods can be challenging and often time-consuming. As a result, artificial intelligence techniques, particularly artificial neural networks (ANN), have become an attractive research topic in this field. ANN are powerful learning algorithms capable of learning from complex datasets and modeling complex relationships. Therefore, using ANN to predict automobile effective efficiency holds promise for achieving more accurate and effective results. Emissions from internal combustion engines, such as vehicles, contribute significantly to air pollution and adversely affect the environment and human health. Neural networks provide a powerful tool for modeling and predicting

engine emissions accurately. Neural networks can learn complex patterns and relationships from large datasets, making them well-suited for predicting emissions. By analyzing various engine parameters, such as fuel type, air-fuel ratio, engine speed, load, and temperature, neural networks can identify correlations and patterns influencing emission levels. These models can then estimate emissions based on input parameters, enabling engineers to optimize effective efficiency and reduce harmful pollutant releases. By utilizing a neural network approach, the prediction of effective efficiency can be effectively carried out, leading to more environmentally friendly outcomes. ANN offer several advantages in predicting effective efficiency from an environmental sustainability perspective. These models can learn from vast amounts of data, extract meaningful patterns, and identify complex relationships. By accurately predicting effective efficiency, decision-makers can make informed choices regarding engine design, adjustments, and optimization techniques. This leads to enhanced fuel efficiency, reduced emissions, and improved environmental impact.

Uslu and Celik [1] utilized ANN to estimate the operational efficiency and emissions of a single-cylinder, direct-injection, air-cooled diesel engine using fuel mixtures of diethyl ether (DEE) and diesel. The ANN model accurately predicted the engine's performance and emissions, achieving regression coefficients (R^2) ranging from 0.964 to 0.9878. The mean relative error (MRE) values also ranged from 0.51% to 4.8%. Fu et al. [2] aimed to evaluate the effectiveness of a statistical modeling approach employing ANN in predicting the efficiency and emissions of a calibrated spark ignition (SI) engine. The ANN algorithm utilized engine speed and load as input variables, while fuel consumption and emissions were the output variables. The results demonstrated that the well-trained network accurately forecasted engine efficiency, unburned hydrocarbons, carbon monoxide, and nitrogen oxide emissions with minimal errors and a high coefficient of determination. Another study focused on using an ANN to predict diesel engine performance using biodiesel, bioethanol, and biogas. The researchers developed the ANN model to overcome the challenges and costs associated with traditional engine experiments. The ANN incorporated fuel mixtures with varying percentages of biofuels, and experimental tests were conducted to collect reference values. The estimated values from the ANN model were compared with the experimental results, demonstrating the model's reliability in estimating engine performance. Statistical analyses indicated a reliability value of 99.94% for the ANN model, supporting its effectiveness in predicting engine performance [3]. Another study focused on applying ANN modeling to predict engine performance parameters, specifically brake-specific fuel consumption, effective power, average effective pressure, and exhaust gas temperature, for a methanol engine. Experimental data from tests conducted on a four-cylinder, four-stroke engine at various speeds and torques were used to train the ANN model using a backpropagation algorithm. The accuracy of the ANN predictions was evaluated by comparing them with the experimental results, revealing high R^2 values close to 1, small RMS values, and mean errors. This demonstrated that the developed ANN model effectively predicted engine performance parameters for internal combustion engines [4]. The article focused on using ANN to model a diesel engine fueled with waste-cooking biodiesel to predict engine performance parameters and exhaust emissions. Experimental data from tests conducted on a two-cylinder, four-stroke diesel engine operating at different speeds using blends of waste vegetable cooking biodiesel and diesel fuel were used to train the ANN model. The study found that the ANN model effectively predicted engine performance and exhaust emissions, with high correlation coefficients and low mean square error values. The results indicated that blends of waste vegetable oil methyl ester with diesel fuel improved engine performance and emission characteristics [5].

In conclusion, the prediction of the effective efficiency of the engine using a neural network approach is a promising method for achieving environmental sustainability in the automotive industry [6], [7]. By leveraging the capabilities of ANN, accurate predictions can be made, enabling better decision-making processes, and contributing to the development of eco-friendly vehicles[8]–[10]. This approach can potentially drive positive change by optimizing effective efficiency for reducing environmental impact and promoting a sustainable future. This research aims to predict the effective efficiency of the engine using ANN for the purpose of environmental sustainability. Utilizing ANN to forecast the efficiency, emissions, and other performance characteristics of engines can assist in developing strategies to reduce environmental impacts. This study aims to explore the potential of ANN to provide insights into engine technologies' sustainability and environmental effects.

2. Material and Methods

2.1. Artificial Neural Networks (ANN)

Artificial neural networks, also known as ANN or neural networks, are artificial models designed for computer systems to solve complex problems. Inspired by the functioning principles of the human brain, ANN can recognize patterns in large datasets, analyze complex relationships, and make predictions. Neural networks comprise neurons (nerve cells) and the network structure where these neurons communicate. Figure 1 depicts the structure of a basic artificial neuron.

ANN is constructed by linking artificial neural cells together, forming intricate structures. This network stands out with its layered structure while processing input data. It generally consists of three fundamental layers: the input, hidden, and output. Figure 2 shows the basic structure of an ANN. The input layer receives data from the external world and initiates the processing by transmitting it to the neurons. Each input neuron represents different features of the data. Hidden layers identify learned patterns from input data and extract features. Each hidden layer receives outputs from the neurons in the previous layer and continues the process by transmitting them to its neurons. The output layer utilizes the information from the hidden

layers to generate results or make decisions regarding a specific output. The complexity of the problem determines the variation in the number of hidden layers and neurons within an ANN [12], [13].

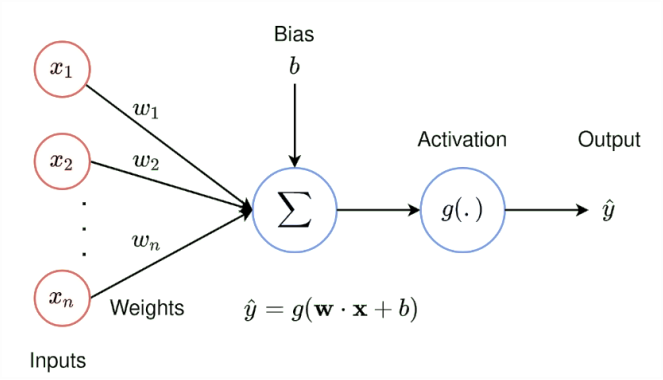


Figure 1 A basic structure of an artificial neuron [11]

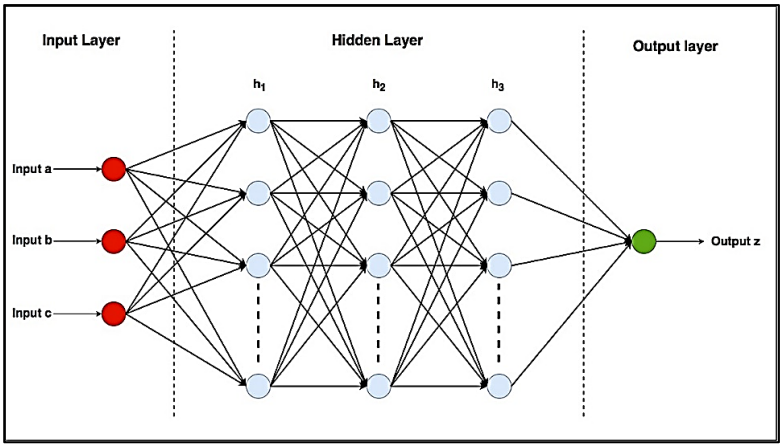


Figure 2 The basic structure of an ANN [14]

ANN employ activation functions to perform computations within neurons. Activation functions take the total input of each neuron and produce an output value. These functions possess non-linear characteristics that enhance the decision-making ability of the neural network. Common activation functions include sigmoid, ReLU (Rectified Linear Unit), and tanh (hyperbolic tangent) [15]. During the training of ANN, datasets are utilized, and parameters such as weights and biases are adjusted to improve the network's performance. This process is typically accomplished using the backpropagation algorithm. Backpropagation involves calculating the error by comparing the network's predictions with the actual results and then propagating these errors backward to update the weights.

The steps of the ANN modeling methodology are explained in Figure 3:

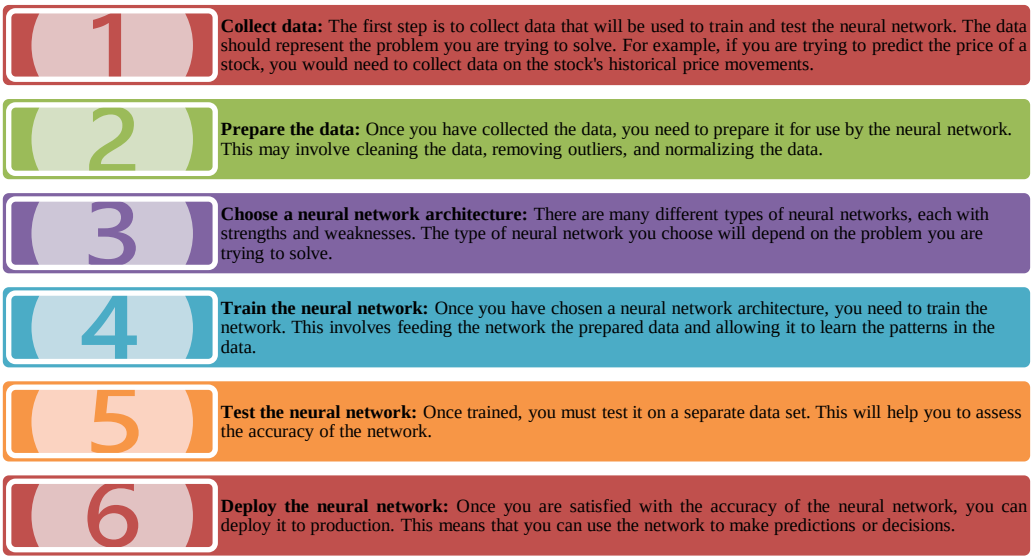


Figure 3 The Steps Involved in ANN Modeling Methodology

2.2. Dataset

The dataset used in this study was collected through experiments. The experiments were conducted to determine the performance of the engine. The engine was tested under different conditions, including different types of engines (standard, e10, e20), fuel ratios (gasoline and ethanol), speeds (rpm), torques (Nm), effective powers (kW), and specific fuel consumptions (g/kWh) to determine effective efficiency (%). The values of effective efficiency (%) were obtained from the experiments. The dataset consists of 90 rows of data. Statistical information regarding the dataset is provided in Table 1.

Table 1 Statistical information related to the dataset utilized in this study

	Features	Unit	Minimum	Maximum	Average	Explanation
Input	Engine Types	-	-	-	-	Standard, e10, e20
	Fuel Ratio Gasoline	-	0.8	1	0.9	
	Fuel Ratio Ethanol	-	0	0.2	0.1	
	Engine Speeds	rpm	1400	3400	2400	
	Torque	Nm	29.27	34.95	31.74	
	Effective Power	kW	4.21	11.46	7.98	
	Specific Fuel Consumptions	g/kWh	270.47	348.66	303.34	
Output		%	24.28	31.12	28.06	

2.3. Data Normalization

Data normalization is a process of scaling the values of the data so that they fall within a specific range. This is often done to improve the performance of machine learning algorithms, such as ANNs. Normalizing the data allows us to mitigate the influence of redundant data repetition and confine the data within a range compatible with ANNs. This normalization process enhances the accuracy and efficiency of the ANN model, resulting in improved performance. In this study, the data is normalized using the min-max normalization method described in Equation 1[16].

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}}(u - l) + l \quad (1)$$

where:

- x is the original data.
- x' is the normalized data.
- x_{max}, x_{min} are the original vector's maximum and minimum values.
- u, l are respectively the upper and lower values of the new range for normalized data.

3. Results and Discussion

The dataset utilized in this study was obtained through experiments to evaluate the engine's effective efficiency. The dataset consists of one dependent variable and seven independent variables. An overview of the dataset is provided in Table 1. In total, there are 90 rows of data. After applying min-max normalization using Equation 1, the dataset was randomly split into training (75%, 62 data), validation (15%, 14 data), and testing (15%, 14 data) sets.

To find the best number of neurons in the hidden layer, a single-hidden-layer ANN architecture was constructed. The number of neurons in the hidden layer was systematically varied from 5 to 50 in increments of 5. The network was trained using the Levenberg-Marquardt algorithm as the learning algorithm. The hidden layer utilized the sigmoid activation function, while the output layer employed the pureline function. During training, a set of input-output pairs was presented, where the inputs represented engine parameters and the outputs represented corresponding emission levels. Through backpropagation, the network adjusted its internal weights and biases to minimize the difference between predicted and actual emission values. Once trained, the neural network demonstrated accurate predictions for unseen input data. The network’s performance was evaluated using the mean square error (MSE) index, and the results indicated that the optimal number of neurons in the hidden layer for predicting engine performance was determined to be 10, as depicted in Figure 4.

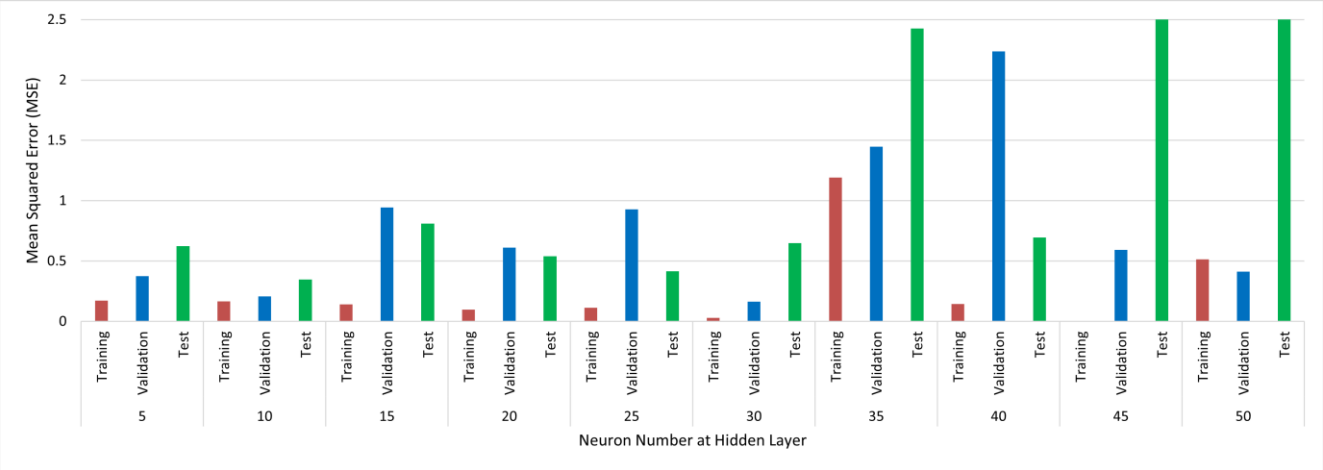


Figure 4 Determination of Neuron Number at the Hidden Layer

In Figure 5, an ANN structure consists of seven independent variables and one dependent variable, with a hidden layer containing 10 neurons. The proposed ANN model is illustrated in Figure 6.

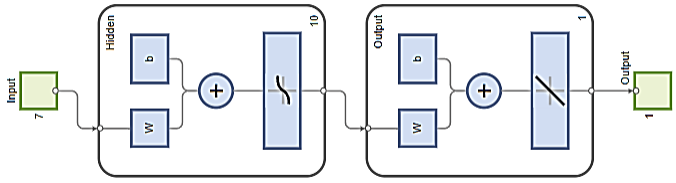


Figure 5 ANN Architecture Utilized for Prediction of the Effective Efficiency of the Engine

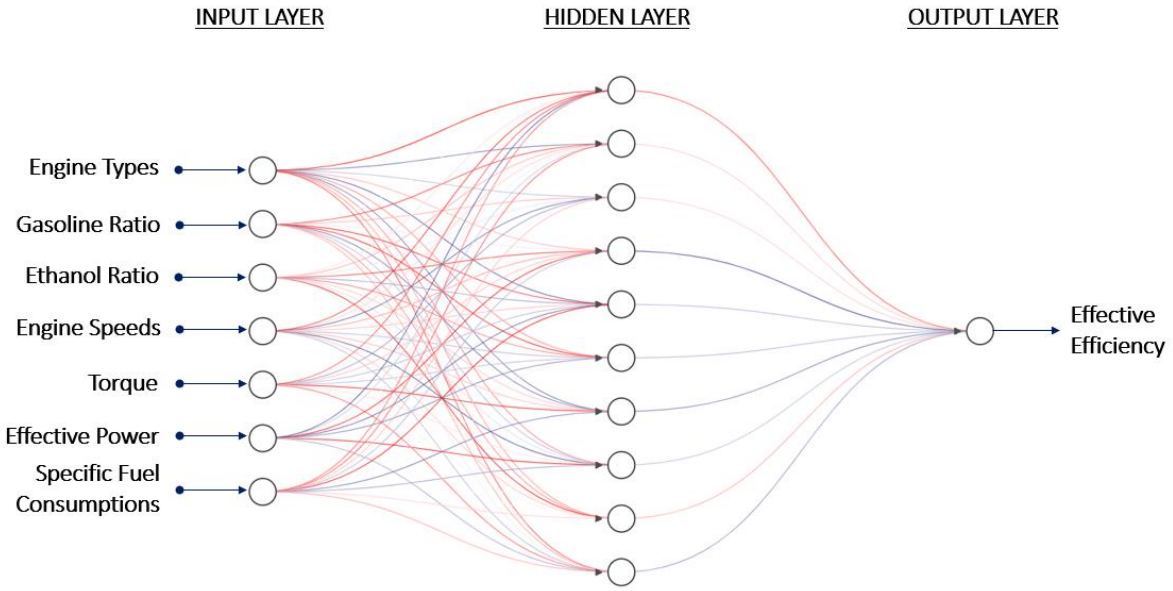


Figure 6 The Proposed ANN Model.

Table 2 Training Progress of the Network

Unit	Initial Value	Stopped Value	Target Value
Epoch	0	13	1000
Elapsed Time	-	00:00:00	-
Performance	27.1	0.101	0
Gradient	54.9	0.0638	1e-07
Mu	0.001	0.01	1e+10
Validation Checks	0	6	6

The data about the training progress of the network is presented in Table 3. It was observed that the network exhibited significantly good performance, with a performance error of 0.101, which is closer to the target error of 0.01. The network achieved this level of performance within 13 iterations, which is considerably fewer than the initial 1000 epochs. The gradient function was calculated to be 0.0638, and the training gain (Mu) was set to 0.01. A validation check of six (6) was recorded, aligning with expectations as weight bias had been addressed through the normalization of the raw data. The Mean Squared Error (MSE) and correlation coefficient (R) values associated with the network training are displayed in Table 3. These values demonstrate that the network training has been remarkably successful, as indicated by the small MSE values and the high R values.

Table 3 The Mean Squared Error (MSE) and Correlation Coefficient (R) Values of the Neural Network

	Observations	%	MSE	R
Training	62	70	0.1667	0.9658
Validation	14	15	0.2064	0.9703
Test	14	15	0.3452	0.9527

Figure 7 shows that the network completed its training in the 11th iteration, and the best validation performance was achieved in the 5th iteration. Lower mean square error is a fundamental criterion used to assess the training accuracy of a network. An error value of 0.20641 at epoch 7 is evidence of a network with a strong capacity to predict performance.

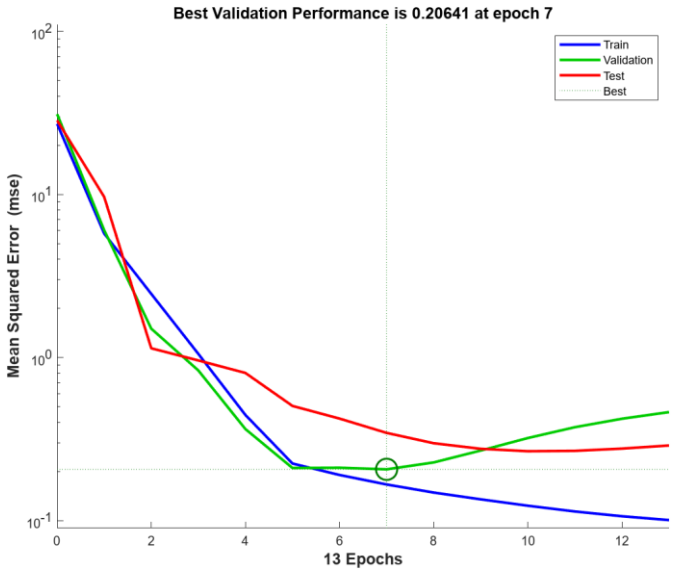


Figure 7 Error Performances for Training, Validation, and Test Data Sets

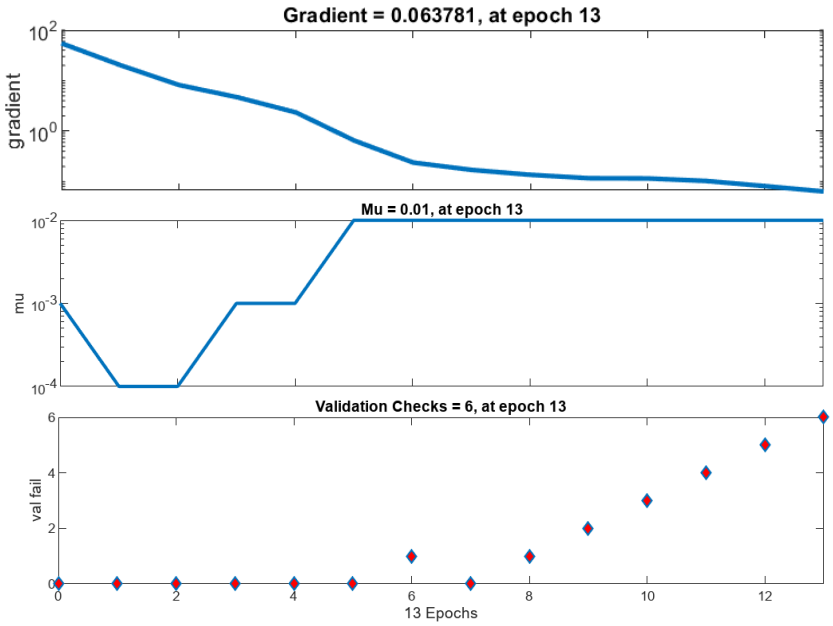


Figure 8 Training State Plot

Figure 8 illustrates the training process progress, where the gradient function depicts the change in model error during parameter updates, the momentum gain (Mu) measures the model's learning performance on the training data, and the validation check assesses the model's ability to generalize to new data. Essentially, the network calculates the loss function's gradient to assess the error contributions of the selected neurons. A lower error indicates better performance. As seen in Figure 8, the computed gradient value of 0.063781 suggests that the error contributions of the chosen neurons are minimal. represents the training gains and should be less than one. A momentum gain of 0.01 reflects a high-capacity network for performance prediction.

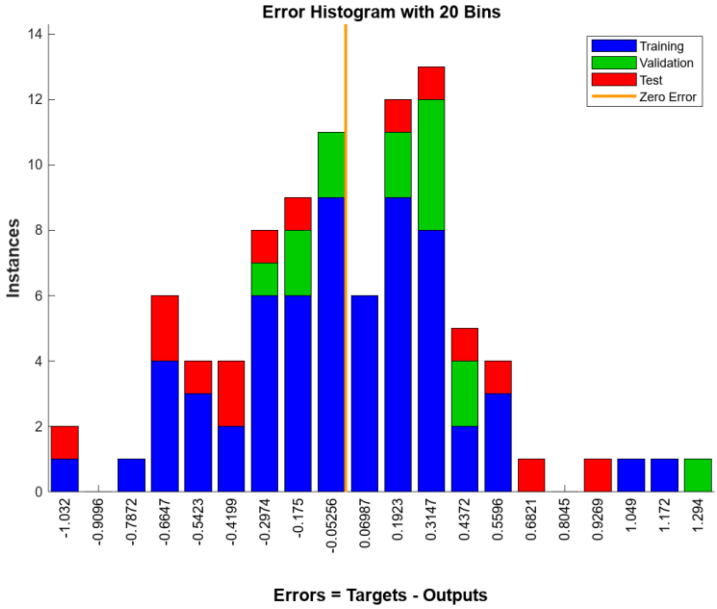


Figure 9 Error Histogram Plot

The error histogram of the trained neural network for the training, validation, and testing steps is depicted in Figure 9. These error values reflect the disparities between predicted and target values and can be negative. The graph consists of vertical bars, referred to as bins, representing the number of samples falling within each bin. In this case, the total error range is divided into 20 smaller bins. The y-axis indicates the number of dataset samples within each bin. The figure demonstrates that the errors in fitting the data are reasonably distributed around zero within an acceptable range.

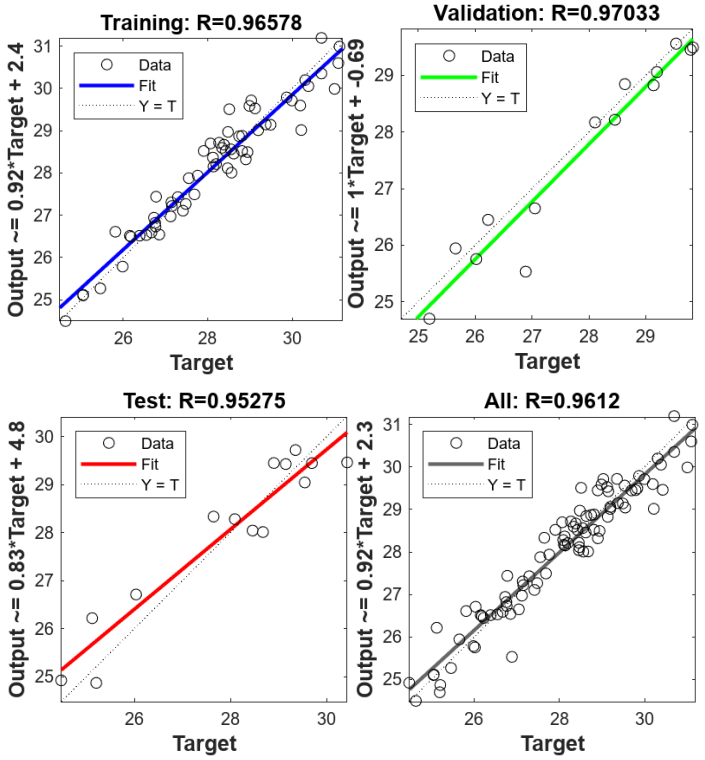


Figure 10 Regression Plot

Figure 10 displays a regression plot depicting the relationship between the input variables (types of engines (standard, e10, e20), fuel ratios (gasoline and ethanol), speeds (rpm), torques (Nm), effective powers (kW), and specific fuel consumptions (g/kWh)) and the target variable (effective efficiency of the engine (%)), alongside the training, validation, and testing progress. The calculated values of the correlation coefficient (R), observed in Figure 10, indicate that the network has been effectively trained and can be utilized for rainfall prediction.

4. Conclusions

In this study, our main objective was to utilize an ANN to predict the effective efficiency of engines. We aimed to establish an appropriate ANN architecture that could accurately predict the engine's effective efficiency based on specific input parameters, including engine types (standard, e10, e20), fuel ratios (gasoline and ethanol), speeds (rpm), torques (Nm), effective powers (kW), and specific fuel consumptions (g/kWh). Accurate prediction of the effective efficiency of the engine through a neural network approach enables the optimization of engine performance, facilitates the development of eco-friendly vehicles, ensures compliance with environmental regulations, and contributes to a sustainable future.

Optimizing engine performance is achieved through accurate prediction of effective efficiency, allowing for the identification of optimal operating conditions. This optimization leads to improved fuel efficiency, reduced emissions, and minimized environmental impact. Furthermore, the study contributes to developing eco-friendly vehicles by providing insights into the factors influencing effective efficiency. This knowledge can be applied in designing and manufacturing engines that are more sustainable and energy efficient. Moreover, the accurate prediction of engine efficiency and emissions ensures compliance with strict environmental regulations imposed by governing bodies. Meeting these regulations not only avoids penalties but also actively contributes to reducing the overall environmental impact of the automotive industry.

Ultimately, the study's focus on enhancing engine efficiency and reducing emissions aligns with the broader objective of achieving environmental sustainability in the automotive sector. By prioritizing energy efficiency, reduced emissions, and environmental responsibility, a sustainable future can be realized.

References

[1] S. Uslu and M. B. Celik, “Prediction of engine emissions and performance with artificial neural networks in a single cylinder diesel engine using diethyl ether,” *Engineering Science and Technology, an International Journal*, vol. 21, no. 6, pp. 1194–1201, Dec. 2018, doi: 10.1016/J.JESTCH.2018.08.017.

[2] J. Fu *et al.*, “Application of artificial neural network to forecast engine performance and emissions of a spark ignition engine,” *Appl Therm Eng*, vol. 201, p. 117749, Jan. 2022, doi: 10.1016/J.APPLTHERMALENG.2021.117749.

[3] H. Oğuz, I. Saritas, and H. E. Baydan, “Prediction of diesel engine performance using biofuels with artificial neural network,” *Expert Syst Appl*, vol. 37, no. 9, pp. 6579–6586, Sep. 2010, doi: 10.1016/J.ESWA.2010.02.128.

[4] Y. Çay, A. Çiçek, F. Kara, and S. Sağiroğlu, “Prediction of engine performance for an alternative fuel using artificial neural network,” *Appl Therm Eng*, vol. 37, pp. 217–225, May 2012, doi: 10.1016/J.APPLTHERMALENG.2011.11.019.

[5] B. Ghobadian, H. Rahimi, A. M. Nikbakht, G. Najafi, and T. F. Yusaf, “Diesel engine performance and exhaust emission analysis using waste cooking biodiesel fuel with an artificial neural network,” *Renew Energy*, vol. 34, no. 4, pp. 976–982, Apr. 2009, doi: 10.1016/J.RENENE.2008.08.008.

[6] Y. Cay, “Prediction of a gasoline engine performance with artificial neural network,” *Fuel*, vol. 111, pp. 324–331, Sep. 2013, doi: 10.1016/J.FUEL.2012.12.040.

[7] Y. Çay, I. Korkmaz, A. Çiçek, and F. Kara, “Prediction of engine performance and exhaust emissions for gasoline and methanol using artificial neural network,” *Energy*, vol. 50, no. 1, pp. 177–186, Feb. 2013, doi: 10.1016/J.ENERGY.2012.10.052.

[8] M. I. Arbab, H. H. Masjuki, M. Varman, M. A. Kalam, S. Imtenan, and H. Sajjad, “Fuel properties, engine performance and emission characteristic of common biodiesels as a renewable and sustainable source of fuel,” *Renewable and Sustainable Energy Reviews*, vol. 22, pp. 133–147, Jun. 2013, doi: 10.1016/J.RSER.2013.01.046.

[9] R. K. Rai and R. R. Sahoo, "Engine performance, emission, and sustainability analysis with diesel fuel-based Shorea robusta methyl ester biodiesel blends," *Fuel*, vol. 292, p. 120234, May 2021, doi: 10.1016/J.FUEL.2021.120234.

[10] A. Tuan Hoang *et al.*, "A review on application of artificial neural network (ANN) for performance and emission characteristics of diesel engine fueled with biodiesel-based fuels," *Sustainable Energy Technologies and Assessments*, vol. 47, p. 101416, Oct. 2021, doi: 10.1016/J.SETA.2021.101416.

[11] URL-1, "The Basics of Neural Networks (Neural Network Series) — Part 1 | by Kiprono Elijah Koech | Towards Data Science." https://towardsdatascience.com/the-basics-of-neural-networks-neural-network-series-part-1-4419e343b2b (accessed May 29, 2023).

[12] H. Kahraman, İ. Cesur, B. Eren, and A. Çoban, "Biyodizel Yakıt Kullanan İçten Yanmalı Motorlarda Aşınma-Sürtünme Optimizasyonu ve Tahmini için Taguchi ve Yapay Sinir Ağı Uygulaması," *Politeknik Dergisi*, pp. 1–1, Dec. 2023, doi: 10.2339/POLITEKNIK.1216411.

[13] B. Oyar, B. Eren, and A. Özdemir, "Removal of Reactive Black 5 from Polluted Solutions by Electrocoagulation: Modelling Experimental Data Using Artificial Neural Networks," *Sakarya University Journal of Science*, vol. 24, no. 4, pp. 712–724, Aug. 2020, doi: 10.16984/SAUFENBILDER.698146.

[14] K. C. Yao, W. T. Huang, C. C. Wu, and T. Y. Chen, "Establishing an AI Model on Data Sensing and Prediction for Smart Home Environment Control Based on LabVIEW," *Math Probl Eng*, vol. 2021, 2021, doi: 10.1155/2021/7572818.

[15] S. Kılıçarslan and K. Adem, "An overview of the activation functions used in deep learning algorithms," *Journal of New Results in Science*, vol. 10, no. 3, pp. 75–88, 2021, doi: 10.54187/jnrs.1011739.

[16] B. Eren, M. Yaqub, and V. Eyupoglu, "A comparative study of artificial neural network models for the predictioof Cd removal efficiency of polymer inclusion membranes," *Desalination Water Treat*, vol. 143, 2019, doi: 10.5004/dwt.2019.23531.

Conflict of Interest Notice
The authors declare that there is no conflict of interest regarding the publication of this paper.

Ethical Approval and Informed Consent
It is declared that during the preparation process of this study, scientific and ethical principles were followed, and all the studies benefited from are stated in the bibliography.

Availability of data and material
Not applicable.

Plagiarism Statement
This article has been scanned by iThenticate ™.

Sakarya University Journal of Computer and Information Sciences
http://saucis.sakarya.edu.tr/

DOI: 10.35377/saucis…1314638 RESEARCH ARTICLE

Deep Learning-Based Classification of Dermoscopic Images for Skin Lesions

Ahmet Furkan SÖNMEZ[1], Serap ÇAKAR[2], Feyza SELAMET[2], Muhammed KOTAN[3], İbrahim DELİBAŞOĞLU[4], Gülüzar ÇİT[4]

[1]Department of Computer Engineering, Zonguldak Bulent Ecevit University, Zonguldak, Türkiye
[2]Department of Computer Engineering, Sakarya University, Sakarya, Türkiye
[3]Department of Information Systems Engineering, Sakarya University, Sakarya, Türkiye
[4]Department of Software Engineering, Sakarya University, Sakarya, Türkiye

Corresponding author:
Ahmet Furkan SONMEZ,
Department of Computer Engineering,
Bulent Ecevit University Zonguldak, Türkiye
E-mail address:
afsonmez@beun.edu.tr

Received: 14 June 2023
Revised: 21 July 2023
Accepted: 26 July 2023
Published Online: 17 August 2023

Citation: Sönmez AF. et al. (2023). Deep Learning-Based Classification of Dermoscopic Images for Skin Lesions. *Sakarya University Journal of Computer and Information Sciences*. 6 (2) https://doi.org/10.35377/saucis...1314638

ABSTRACT

Skin cancer has emerged as a grave health concern leading to significant mortality rates. Diagnosis of this disease traditionally relies on specialist dermatologists who interpret dermoscopy images using the ABCD rule. However, the integration of computer-aided diagnosis technologies is gaining popularity as a means to assist clinicians in accurate skin cancer diagnosis, overcoming potential challenges associated with human error. The objective of this research is to develop a robust system for the detection of skin cancer by employing machine learning algorithms for skin lesion classification and detection. The proposed system utilizes Convolutional Neural Network (CNN), a highly accurate and efficient deep learning technique well-suited for image classification tasks. By using the power of CNN, this system effectively classifies various skin diseases in dermoscopic images associated with skin cancer The MNIST HAM10000 dataset, comprising 10015 images, serves as the foundation for this study. The dataset encompasses seven distinct skin diseases falling within the realm of skin cancer. In this study, diverse transfer learning methods were used and evaluated to enhance the performance of the system. By comparing and analyzing these approaches the highest accuracy rate was obtained using the MobileNetV2 model with a rate of 80.79% accuracy.

Keywords: Convolutional Neural Network, Transfer Learning, Image Classification, Skin Cancer

1. Introduction

Skin cancer is the most common type of cancer worldwide, and melanoma is the deadliest form. Cancer, as a term, refers to a malignant tumor that develops when cells in an organ or tissue divide and multiply irregularly. Skin cancer occurs when skin cells grow abnormally, and melanoma is defined as a type of skin cancer resulting from uncontrolled division and proliferation of these cells. Early detection of such skin cancers increases the likelihood of successful treatment.

There are two examination methods commonly used by doctors for skin diseases. One of them is dermoscopy, also known as dermatoscopy, which involves superficial microscopic inspection of the skin. It is used to identify abnormalities in moles and other skin lesions. This type of examination magnifies moles and allows for accurately evaluating subtle details that are not visible to the human eye. Dermoscopy can be performed using a handheld device or by capturing images through computerized systems. It is a frequently preferred diagnostic method because it provides early diagnosis without causing any adverse side effects.

The other method is histopathological examination, which employs various techniques to examine changes in organs, tissues, and cells under a microscope. The tissues to be studied are first sliced into suitable thicknesses using a small cutting

instrument called a microtome. They are then evaluated in a laboratory by doctors. This method is a critical diagnostic tool for confirming the diagnosis of melanoma.

When diagnosing malignant melanoma, dermatologists examine the blemish on the skin with the eye or pre-taken photographs of the blemish and look at four essential parameters. These parameters are called the ABCD rule. ABCD rule is applied for easy detection of differentiation in the follow-up of moles.

- A (Asymmetry): If one half of the mole is not similar to the other half (in terms of color and/or shape)
- B (Border): If the borders of the mole are irregular (indented)
- C (Color): If the color of the mole is not homogeneous (two or more colors such as brown, black, red, gray, and white are present together or if there is a mottled appearance)
- D (Diameter/Diameter): If the diameter of the mole is larger than 6 mm (roughly larger than the diameter of an eraser pencil) [1,2].

In this study, transfer learning models of CNN that analyze skin lesion images and classify them according to seven skin diseases were tested using a publicly available dataset. The results obtained with different models and parameters are added to the table in detail.

2. Literature Review

In the realm of dermatology research, a multitude of studies have been conducted regarding skin diseases. Kiran Pai and Anandi Giridharan embarked on a study where they employed CNNs to discern and accurately diagnose seven distinct types of skin lesions. A web application has been developed to accurately predict the top three potential types of skin lesions for a given image and present the corresponding top three classes. The model was trained using VGGNet, a transfer learning method based on the CNN architecture, carefully selected for its effectiveness and reliability in achieving optimal performance. The model was trained for 50 epochs on the HAM10000 dataset [3], yielding a test accuracy of 78 percent. Ketut Eddy Purnama et al. suggested an advanced system for precisely classifying and detecting skin diseases. Using CNN with the InceptionV3 model, dermatological diseases in dermoscopic images were accurately classified. The web classifier utilizing the CNN Inception V3 model achieved an impressive accuracy of 72 percent, while the web classifier using the MobileNetV1 model attained a good accuracy of 58 percent [4].

Harsh Gupta et al. embarked on a comprehensive study to analyze images depicting infected regions of the skin and, additionally, classify skin cancer into a unified category. A range of pre-processing techniques were used on the skin cancer images. By harnessing the power of CNN and leveraging transfer learning models, the accuracy of the classification process was significantly enhanced. Utilizing EfficientNet B1, an outstanding accuracy rate of 94.1 percent was accomplished [5]. Xingmei Cao et al. opted for generating mixed skin lesion images to address the data imbalance issue. This method is a variation of Mask Recurrent Convolutional Neural Network (Mask R-CNN), and it involves the creation of a melanoma detection framework. Through the utilization of Mask R-CNN alongside the concept of community learning, the accuracy of the generated classification was experimentally enhanced by 2.56 percent. The study was conducted on the ISIC dataset, and the proposed algorithm achieved an accuracy of 90.61 percent [6]. Attik et al. suggested a computer-aided diagnosis (CAD) system based on deep learning. RGB images were used to train the Mask R-CNN model. The ISBI2016 and ISIC2017 datasets were selected for these images. The Least Squares Support Vector Machine (LS-SVM) technique was also employed. Three distinct datasets (ISBI2016, ISBI2017, and HAM10000) were utilized for validation. The obtained accuracies were 96.3%, 94.8%, and 88.5%, respectively [7]. Khan *et al*. presented a state-of-the-art deep learning framework, leveraging Mask R-CNN for precisely segmenting and classifying skin lesions. This framework surpasses existing techniques in sensitivity, precision, F1 score, and accuracy [8], establishing itself as a significant breakthrough in dermatological research. Srivastava et al. proposed a texture based feature extraction framework for classifying dermatoscopic images of skin cancer. For their technique, the average accuracy, average precision, and average recall value were found to be 96%, 95.44% and 75.20%, respectively [9]. Alam et al. proposed a skin cancer classifier based on deep learning for the HAM10000 dataset. They found that in both unbalanced and balanced datasets, the results of RegNetY-320 outperformed those of AlexNet and InceptionV3 in terms of receiver operating characteristic (ROC) curve, F1 score and accuracy [10]. Bassel et al. suggested a hybrid deep learning model for the automatic classification of benign and malignant skin cancers using various methods such as Resnet50, Xception, and VGG16. The proposed method achieved 90.9% accuracy and could provide a robust and reliable classification system with a large training dataset [11]. Salma and Eltrass presented an automated CAD system for the classification of skin lesions using deep learning techniques. The paper also mentions the use of dermoscopy, a noninvasive skin imaging technique, for early identification of skin cancer [12]. Shetty et al. focused on the classification of skin lesions using machine learning and CNN. The paper concluded that CNN provides better accuracy compared to other machine learning algorithms used in their work [13]. Aladhadh et al. suggested a two-tier framework for the classification of skin cancer using Medical Vision Transformer (MVT) and data augmentation techniques. In their study, the MVT-based model achieved better results than other techniques for skin cancer classification [14]. Iqbal et al. proposed a hybrid approach using CNN and local binary patterns (LBP) for the classification of melanoma images. The approach was evaluated on publicly accessible datasets and

showed promising results with an average sensitivity of 95.63%, accuracy of 97.29% and specificity of 97.90% [15]. Ahmad et al. developed hybrid techniques SVM-MobileNet, SVM-ResNet101and SVM-MobileNet-ResNet101 to classify two datasets, HAM10000 and PH2, of skin lesions. Their results showed better performance than pre-trained CNN models [16]. Alwakid et al. employed Inception-V3 and Inception Resnet-V2 models for melanoma recognition using the HAM10000 dataset. Their models performed the results of 0.89 for Inception-V3 and 0.91 for InceptionResnet-V2 [17].

3. Methodology

3.1. Convolutional Neural Network (CNN)

CNNs are improved versions of Artificial Neural Networks (ANNs). CNNs expand on the concept of ANNs by increasing the depth of the network through the addition of more hidden layers. CNNs can be seen as an example of this deepening network structure.

The key difference that sets CNNs apart from ANNs is the use of the DropOut method in CNNs, which helps prevent overfitting and memorization during the training process. By dropping out randomly selected neurons during training, CNNs encourage the network to learn more robust and generalizable features.

The architecture of CNNs forms the foundation of the Deep Learning concept. It consists of various layers that perform different tasks in a sequential manner. The initial stages involve Convolutional and Pooling layers, which extract important features from the input data (such as images) in a hierarchical manner. These layers capture local patterns and gradually build up a more abstract representation of the data.

The last phase of a CNN typically involves Fully Connected layers, which connect all the neurons from the previous layers to form a dense network. It is followed by a Classification layer that produces the final output, usually in the form of probabilities for different classes or categories.

In summary, CNNs can be viewed as a series of trainable components arranged in a sequential manner, with an informative classifier at the end. The training process occurs through layer-by-layer processing as the input data flows through the network. Eventually, a final output is generated, and a comparison is made with the correct result to evaluate the performance of the network.

CNNs, like CNNs, can handle various types of input data, including signals such as audio, images, or videos. This flexibility allows CNNs to be applied to a wide range of problems across different domains [18].

Figure 1 shows the general overview of a CNN architecture. The layers used in a CNN are Convolution Layer, ReLU, Pooling Layer, Fully Connected Layer, DropOut Layer and Classification Layer, respectively. Convolution Layer is the layer that forms the basis of CNN. This operation is done by applying a specific filter over the entire image.

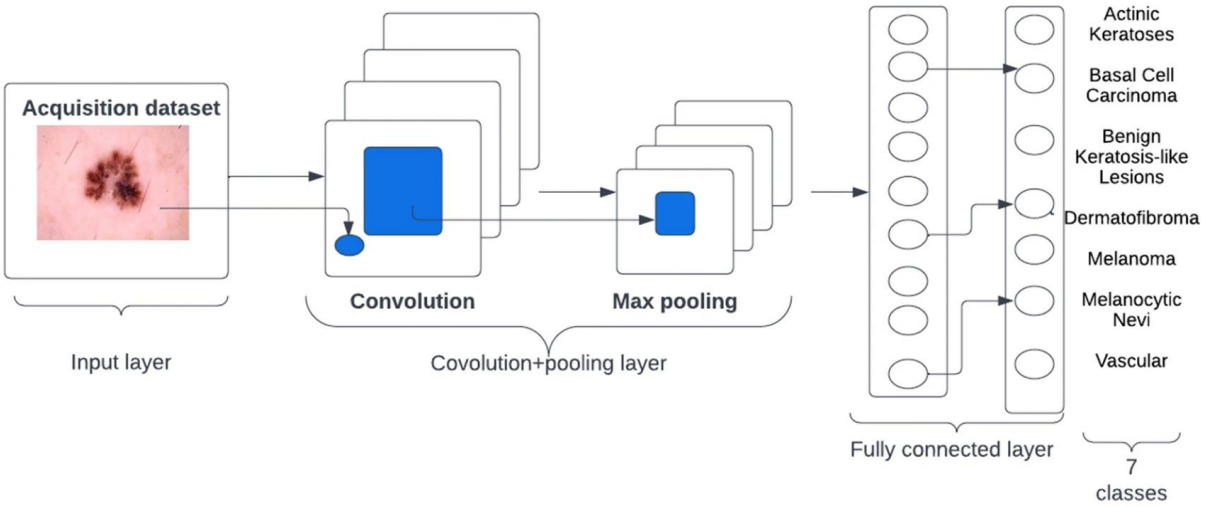


Figure 1 Overview of CNN Architecture [13]

ReLU is the most often utilized rectifier unit for CNN neuron outputs and comes after the convolution layers. Pooling Layer usually occurs after the ReLU layer. Its primary goal is to lower the input size (Width x Height). Fully Connected Layer is connected to all areas of the previous layer, and it works on an input where each input is connected to all neurons. DropOut Layer is used to prevent the network from memorizing and some nodes of the network are removed in this layer. Classification is done in the Classification Layer. This layer's output value and the number of objects to be categorized are both equal [19].

3.2. Dataset

The images used in this study were taken from the “Skin Cancer MNIST: HAM10000” dataset [20]. The HAM10000 dataset is a widely used dataset in the field of dermatology and machine learning. It stands for "Human Against Machine with 10000 training images" and consists of a collection of 10015 images of skin lesions. The dataset was created to facilitate the research and development of automated algorithms for the diagnosis of skin cancer.

Each image in the HAM10000 dataset is accompanied by various metadata, including information such as the lesion's clinical diagnosis, anatomical location, patient information, and other attributes. The dataset covers a range of skin lesion types, including melanoma and other types of benign and malignant lesions.

Researchers and developers often use the HAM10000 dataset to train machine learning models or develop computer vision algorithms that can accurately classify and diagnose skin lesions. The goal is to create automated systems that can assist dermatologists in the early detection and diagnosis of skin cancer, potentially improving patient outcomes and reducing the burden on healthcare systems.

It's important to note that while the HAM10000 dataset is a valuable resource, any real-world application of machine learning algorithms for medical diagnosis should involve rigorous validation, clinical studies, and integration with healthcare professionals to ensure safety and accuracy. The sample skin lesion types collected from the HAM10000 dataset are shown in Figure 2. These skin lesion types include cases belonging to seven different classes. In the dataset, the HAM10000_metadata.csv file contains all the information about the lesions, such as classes and properties. These classes are akiec, bcc, bkl, df, mel, nv and vasc, respectively.

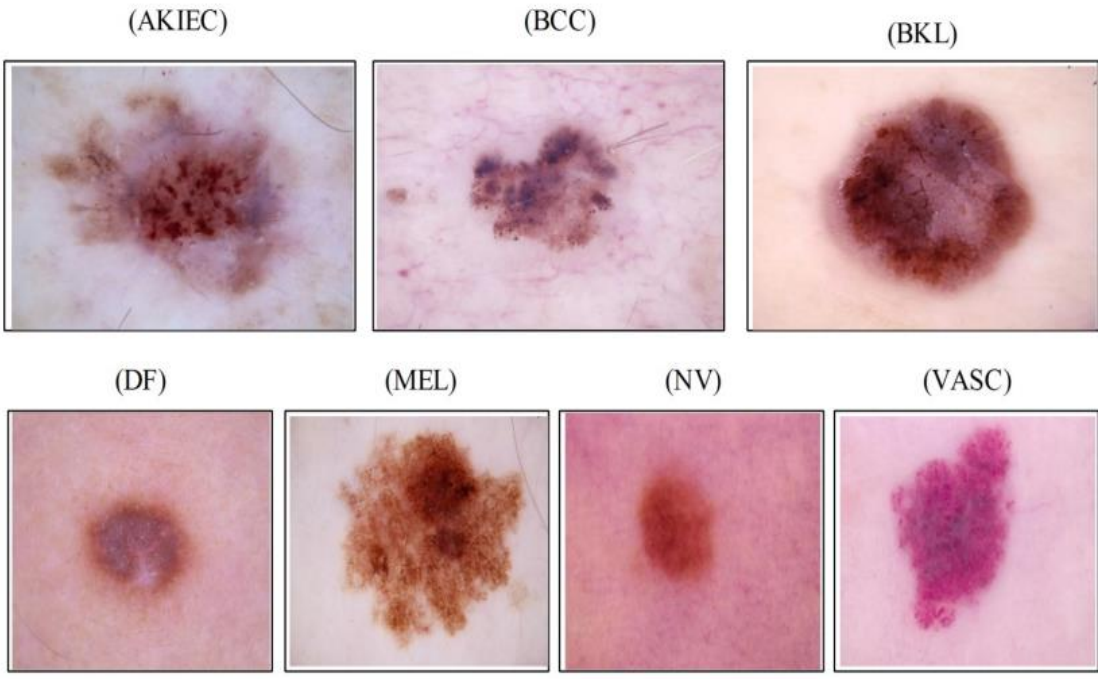


Figure 2 Sample skin lesion types collected from the HAM10000 dataset [20]

3. 3. Data Augmentation

Data augmentation is the process of adding data to enhance the number of data to be used. In the data augmentation procedure, the new data is added to the training data. New data is created by changing the attributes of images, such as horizontal/vertical rotations, brightness changes, horizontal /vertical shifts and zoom [21, 22]. Figure 3 shows the augmentation techniques applied in this study and an example result on an original image. The classes applied to images from the data augmentation techniques listed in Figure 3 are shown in Table 1.

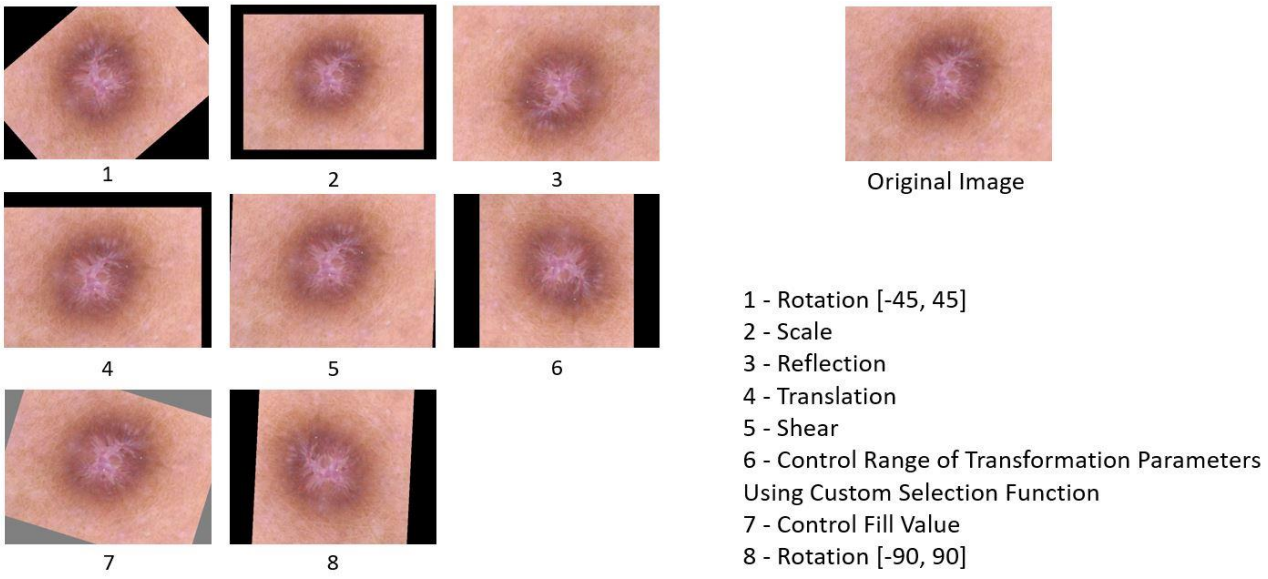


Figure 3 Data augmentation techniques applied in this study and an example result on an original image.

The alternative purpose of the data augmentation process is to balance the amount of data for each class of dataset. As can be seen in Table 1, the distribution of the number of images of the lesions is quite irregular. While the total number of images of the Melanocytic Nevi (nv) skin disease class in Table 1 is 6705, the total number of images of the Dermatofibroma (df) skin disease class is 115. There are several methods to balance this situation. In this study, 1000 images from each class were used. In the classes with a sufficient number of images, 1000 images were randomly selected, and in the other classes, the number of images was increased using augmentation. As stated in Table 1, 7000 images constitute our dataset.

Table 1 Image distributions of the dataset before and after augmentation

Skin Lesion	**Number of Images (Before Augmentation)**	**Balanced Number of Images (After Augmentation)**	**Type of Augmentation Technique**
Melanocytic Nevi (nv)	6705	1000	None
Melanoma (mel)	1113	1000	None
Benign Keratosis (bkl)	1099	1000	None
Basal Cell Carcinoma (bcc)	514	1000	1
Actinic Keratoses (akiec)	327	1000	1,2 and 3
Vascular Lesions (vas)	142	1000	1,2,3,4,5,6 and 7
Dermatofibroma (df)	115	1000	1,2,3,4,5,6,7 and 8
Total	10015	1000	

4. Experimental Observations and Results

First, the increased dataset is divided into three parts: training, testing and validation. As a result of pre-processing, each class consists of 1000 images. It was found appropriate to allocate 70% of the images for training, 20% for testing and 10% for validation. Thus, for each class, there are 700 images in the training set, 200 images in the test set, and 100 images in the validation set. It should be noted that none of the images reserved for training or the reproduced versions of that image with data augmentation methods are not included in the test images or validation images. That is, the images of the training group, test group and validation group do not have the same or similar images.

As indicated in Table 2, many Transfer Learning (TL) models have been tested with our dataset. Early stopping criteria is applied during the training phase. In this way, if it is noticed that there is no increase in accuracy with the patience value entered as a parameter, the training is completed without waiting until the entered epoch value. Table 2 also indicates how many steps each model terminates. Another parameter is the trainable status of the feature extraction layers transferred from pre-trained models. In the model, the trainable part is tested separately as both false and true and added to the table. In addition, two different frameworks were used. One of them is TensorFlow and the other is PyTorch. It is seen that the highest accuracy values are reached when PyTorch is selected as the framework and True is selected as trainable. While there is no significant difference between True and False values in TensorFlow, there is quite a difference between True and False in PyTorch.

After trial and error processes, we determined appropriate hyperparameters such as learning rate and optimizer. SGD is used for optimization during the training, and the learning rate is set to 0.001. Cross entropy loss is used to measure the error between prediction and real class values. We did not use a fixed number of epochs, as we ended the training process by looking at validation. The training was terminated automatically according to the validation loss value via early stopping.

Table 2 Applied Transfer Learning models and results

TL Model	Number of epochs	Accuracy	Trainable	Framework
VGG16	13	57%	FALSE	TensorFlow
VGG16	25	54.57%	TRUE	TensorFlow
ResNet50V2	12	62.71%	FALSE	TensorFlow
ResNet50V2	27	38.43%	TRUE	TensorFlow
MobileNet	12	61.64%	FALSE	TensorFlow
MobileNet	25	70.36%	TRUE	TensorFlow
MobileNetV2	14	63.36%	FALSE	TensorFlow
MobileNetV2	25	55%	TRUE	TensorFlow
DenseNet169	12	59.07%	FALSE	TensorFlow
DenseNet169	25	48.43%	TRUE	TensorFlow
NASNetMobile	12	56.36%	FALSE	TensorFlow
Xception	14	58.64%	FALSE	TensorFlow
Xception	29	68.43%	TRUE	TensorFlow
InceptionResNetV2	11	60.5%	FALSE	TensorFlow
InceptionResNetV2	16	73%	TRUE	TensorFlow
VGG19	19	62.43%	FALSE	TensorFlow
ResNet18	18	66.71%	FALSE	PyTorch
MobileNetV2	20	65.86%	FALSE	PyTorch
EfficientNetB0	27	66.21%	FALSE	PyTorch
InceptionResNetV2	22	60.07%	FALSE	PyTorch
ResNet18	19	79.43%	TRUE	PyTorch
MobileNetV2	**22**	**80.79%**	**TRUE**	**PyTorch**
EfficientNetB0	23	79.71%	TRUE	PyTorch
InceptionResNetV2	24	76.43%	TRUE	PyTorch

Since the highest accuracy rate was obtained with the MobileNetV2 model, the accuracy and loss graphs of this model are shown in Figure 4. As can be seen from the graphs, the training was completed automatically at 22 epochs since there was no change in the values.

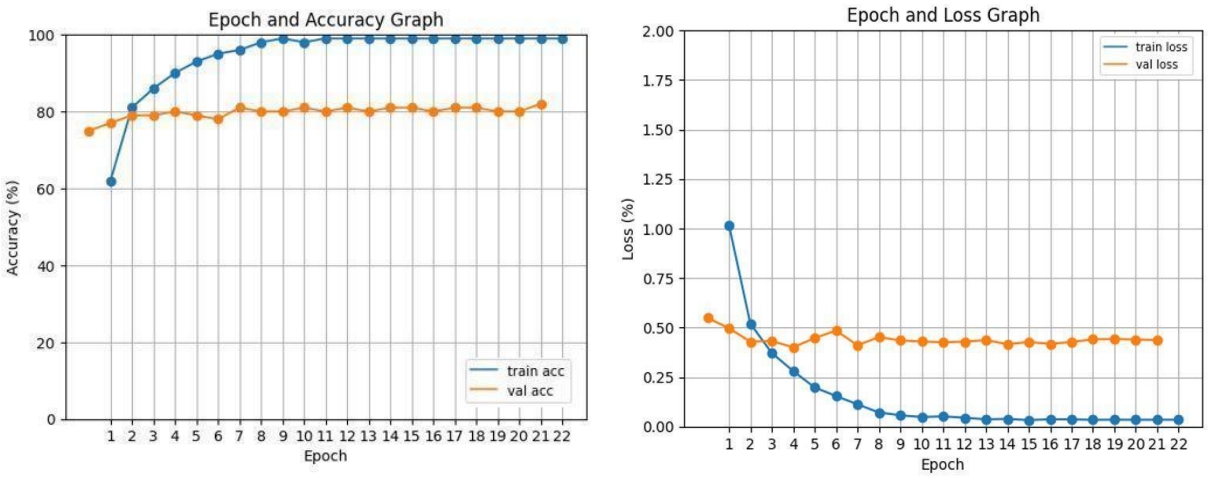


Figure 4. Accuracy and loss graph of MobileNetV2

Figure 5 shows the confusion matrix created for the seven lesion classes in the data set for MobileNetV2. In the confusion matrix, label 0 indicates akiec lesion, while label 6 indicates vasc lesion. Other labels also continue in alphabetical order as Table 1. Mel indicated with label number 4 is the lesion class with the lowest accuracy value, with an accuracy value of 59%. Vasc indicated with label number 6 is the lesion class with the highest accuracy value of 93%.

Figure 5. Confusion Matrix of MobileNetV2

Table 3 shows the comparison of our study with other studies in the literature. It is not correct to make a complete comparison here because the number of classes and data numbers in other studies are different from each other.

Table 3 Comparison with other studies in the literature

Reference	Year	Method	Dataset	Accuracy
[1]	2020	Decision Tree	ISIC	80%
[2]	2017	Support Vector Machine	ISIC and Skinvision website	92.1%
[3]	2019	VGGNet CNN	HAM10000	78%
[4]	2019	InceptionV3	HAM10000	72%
[5]	2020	EfficientNetB1	HAM10000	94% (F1-Score)
[6]	2021	Mask R-CNN	ISIC2017 and ISIC2018	90.61%,
[7]	2021	MASK RCNN-DenseNet	ISBI2016, ISBI2017 and HAM10000	96,3%
[8]	2021	Mask R-CNN	PH2, ISBI2016, ISIC2017 and HAM10000	86,5%
[9]	2022	M-QuadLTQP with CNN	HAM10000 and ISICUDA11	96%
[10]	2022	RegNetY320	HAM10000	91%
[11]	2022	StackingCV	ISIC	90.9%
[12]	2022	ResNet50+SVM	ISIC2017 and HAM10000	99.87%
[13]	2022	Machine Learning+CNN	HAM10000	95.18%
[14]	2022	MVT(Medical Vision Transformer)+MLP	HAM10000	96.14%
[15]	2022	LBPCNN	ISIC2017, ISIC2018, ISIC2019 and HAM10000	97.29%
[16]	2023	MobileNet+Handcrafted	HAM10000 and PH2	100%
[17]	2022	InceptionResNetV2	HAM10000	91.26%
Ours	2023	MobileNetV2	HAM10000	80.79%

5. Conclusions

Skin cancer is a common and serious disease that can cause death if left untreated. When skin cancer is detected early from dermatoscopic images, the probability of definitive treatment is high. Manual diagnosis of skin cancer is a time- and cost-intensive process. Therefore, it is of great importance to develop automatic diagnostic methods to classify multiclass skin lesions with higher accuracy. Recently, deep learning-based models have demonstrated above-human-level accuracy in classification tasks. CNNs outperform human vision and can significantly reduce a dermatologist's or specialist's efforts to predict a possible worsening. In this study, seven different skin cancer types in the MNIST HAM10000 dataset were classified and compared using different CNN models. Since the HAM10000 dataset is unbalanced, the data augmentation process was done in a balanced way before the classification. The highest accuracy rate was obtained with the MobileNetV2 model. The observed 80.79% accuracy when tested with real samples may not be sufficient. This ratio is clearly due to the imbalance in the number of images belonging to the classes, since the total number of images of the Melanocytic Nevi (nv) skin disease class is 6705, while the total number of images of the Dermatofibroma (df) skin disease class is 115. Adding new images to skin disease classes with a low number of images can increase the accuracy and reliability of the model. In addition, the accuracy values can be increased when pre-processing methods such as hair removal and contrast stretching are applied to the images while editing the data set. Hybrid models can also be tried to achieve higher results in future studies.

References

[1] A. R. Ali, J. Li, and S. J. O'Shea, "Towards the automatic detection of skin lesion shape asymmetry, color variegation and diameter in dermoscopic images," *Plos one*, 2020.

[2] H. Alquran, I. A. Qasmieh, A. M. Alqudah, S. Alhammouri, E. Alawneh, A. Abughazaleh, and F. Hasayen, "The melanoma skin cancer detection and classification using support vector machine," *2017 IEEE Jordan Conference on Applied Electrical Engineering and Computing Technologies (AEECT)*, IEEE, pp. 1-5, 2017.

[3] K. Pai, and A. Giridharan, "Convolutional Neural Networks for classifying skin lesions," *TENCON 2019-2019 IEEE Region 10 Conference (TENCON)*, IEEE, pp. 1794-1796, 2019.

[4] I. K. E. Purnama, *et al*., "Disease classification based on dermoscopic skin images using convolutional neural network in teledermatology system," *2019 International Conference on Computer Engineering, Network, and Intelligent Multimedia (CENIM)*. IEEE, pp. 1-5, 2019.

[5] H. Gupta, H. Bhatia, D. Giri, R. Saxena, and R. Singh, "Comparison and Analysis of Skin Lesion on Pretrained Architectures," *International Research Journal of Engineering and Technology (IRJET)*, pp. 2704-2707, 2020.

[6] X. Cao, J. S. Pan, Z. Wang, Z. Sun, A. Haq, W. Deng, and S. Yang, "Application of generated mask method based on Mask R-CNN in classification and detection of melanoma," *Computer Methods and Programs in Biomedicine*, vol. 207, 2021.

[7] M. A. Khan, T. Akram, Y. D. Zhang, and M. Sharif, "Attributes based skin lesion detection and recognition: A mask RCNN and transfer learning-based deep learning framework," *Pattern Recognition Letters*, vol. 143, pp. 58-66, 2021.

[8] M. A. Khan, Y. D. Zhang, M. Sharif, and T. Akram, "Pixels to classes: intelligent learning framework for multiclass skin lesion localization and classification," *Computers & Electrical Engineering*, vol. 90, 2021.

[9] V. Srivastava, D. Kumar, and S. Roy, "A median based quadrilateral local quantized ternary pattern technique for the classification of dermatoscopic images of skin cancer," *Computers and Electrical Engineering*, vol. 102, 2022.

[10] T. M. Alam, K. Shaukat, W. A. Khan, I. A. Hameed, L. A. Almuqren, M. A. Raza, M. Aslam, and S. Luo, "An Efficient Deep Learning-Based Skin Cancer Classifier for an Imbalanced Dataset," *Diagnostics*, vol. 12, no. 9, pp. 2115-2131, 2022.

[11] A. Bassel, A. B. Abdulkareem, Z. A. A. Alyasseri, N. S. Sani, and H. J. Mohammed, "Automatic malignant and benign skin cancer classification using a hybrid deep learning approach," *Diagnostics*, vol. 12, no. 10, 2022.

[12] W. Salma and A. S. Eltrass, "Automated deep learning approach for classification of malignant melanoma and benign skin lesions," *Multimedia Tools and Applications*, vol. 81, no. 22, pp. 32643-32660, 2022.

[13] B. Shetty, R. Fernandes, A. P. Rodrigues, R. Chengoden, S. Bhattacharya, and K. Lakshmanna, "Skin lesion classification of dermoscopic images using machine learning and convolutional neural network," *Scientific Reports*, vol. 12, 2022.

[14] S. Aladhadh, M. Alsanea, M. Aloraini, T. Khan, S. Habib, and M. Islam, "An effective skin cancer classification mechanism via medical vision transformer," *Sensors*, vol. 22, no. 11, 2022.

[15] S. Iqbal, A. N. Qureshi, and G. Mustafa, "Hybridization of CNN with LBP for Classification of Melanoma Images," *Computers, Materials & Continua*, vol. 71, no. 3, 2022.

[16] I. A. Ahmed, E. M. Senan, H. S. A. Shatnawi, Z. M. Alkhraisha, M. M. A. Al-Azzam, "Multi-Models of Analyzing Dermoscopy Images for Early Detection of Multi-Class Skin Lesions Based on Fused Features," Processes, vol. 11, no. 3, 2023.

[17] G. Alwakid, W. Gouda, M. Humayun, NZ. Jhanjhi, "Diagnosing Melanomas in Dermoscopy Images Using Deep Learning," *Diagnostics*, vol. 13, no. 10, 2023.

[18] A. W. Setiawan, "Effect of Color Enhancement on Early Detection of Skin Cancer using Convolutional Neural

Network," *2020 IEEE International Conference on Informatics, IoT, and Enabling Technologies (ICIoT)*, pp. 100-103, IEEE, 2020.
[19] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, 2012.
[20] P. Tschandl, C. Rosendahl, and H. Kittler, "The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions," *Scientific Data*, vol. 5, no. 1, pp.1-9, 2018.
[21] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *Journal of big data*, vol. 6, no. 1, pp. 1-48, 2019.
[22] A. Mikołajczyk and M. Grochowski, "Data augmentation for improving deep learning in image classification problem," *2018 international interdisciplinary PhD workshop (IIPhDW)*. IEEE, pp. 117-122, 2018.

Conflict of Interest Notice
The authors declare that there is no conflict of interest regarding the publication of this paper.

Ethical Approval and Informed Consent
It is declared that during the preparation process of this study, scientific and ethical principles were followed, and all the studies benefited from are stated in the bibliography.

Availability of data and material
Not applicable

Plagiarism Statement
This article has been scanned by iThenticate ™.

Sakarya University Journal of Computer and Information Sciences
http://saucis.sakarya.edu.tr/

DOI: 10.35377/saucis…1309970 RESEARCH ARTICLE

Rapid and Precise Identification of COVID-19 through Segmentation and Classification of CT and X-ray Images

Ahmet SAYGILI [1]

[1]Tekirdağ Namık Kemal University, Çorlu Faculty of Engineering, Department of Computer Engineering, Çorlu/Türkiye

Corresponding author:
Ahmet SAYGILI
asaygili@nku.edu.tr

Submitted: 05 June 2023
Revision Requested: 03 August 2023
Last Revision Received: 06 August 2023
Accepted: 10 August 2023
Published Online: 10 August 2023

Citation: Saygılı A. (2023). Rapid and Precise Identification of COVID-19 through Segmentation and Classification of CT and X-ray Images. *Sakarya University Journal of Computer and Information Sciences*. 6 (2) https://doi.org/10.35377/saucis...1309970

ABSTRACT

The COVID-19 pandemic, caused by a novel coronavirus, has become a global epidemic. Although the reverse transcription-polymerase chain reaction (RT-PCR) test is the current gold standard for detecting the virus, its low reliability has led to the use of CT and X-ray imaging in diagnostics. As limited vaccine availability necessitates rapid and accurate detection, this study applies k-means and fuzzy c-means segmentation to CT and X-ray images to classify COVID-19 cases as either diseased or healthy for CT scans and diseased, healthy, or non-COVID pneumonia for X-rays. Our research employs four open-access, widely used datasets and is conducted in four stages: preprocessing, segmentation, feature extraction, and classification. During feature extraction, we employ the Gray-Level Co-Occurrence Matrix (GLCM), Local Binary Pattern (LBP), and Histogram of Oriented Gradients (HOG). In the classification process, our approach involves utilizing k-Nearest Neighbor (kNN), Support Vector Machines (SVM), and Extreme Learning Machines (ELM) techniques. Our research achieved a sensitivity rate exceeding 99%, which is higher than the 60-70% sensitivity rate of PCR tests. As a result, our study can serve as a decision support system that can help medical professionals make rapid and precise diagnoses with a high level of sensitivity.

Keywords: COVID-19, Diagnosis, Imaging techniques, Segmentation methods, Machine learning-based classification

1. Introduction

COVID-19, which emerged in Wuhan, China, has rapidly spread globally, affecting a vast number of individuals. As of May 5, 2021, over 153 million people have been afflicted by the disease, and the death toll has surpassed 3.2 million [1]. This pandemic has caused profound impacts on society, manifesting various physical, mental, psychological, and sociological repercussions. A study conducted in early 2020 on 1,210 participants from 194 cities in China revealed that 54% of respondents rated the psychological impact of the COVID-19 outbreak as moderate to severe, with 29% exhibiting moderate to severe anxiety symptoms [2]. COVID-19 can present diverse symptoms in individuals, with fever and cough being the most prevalent [3]. However, the disease's asymptomatic transmission necessitates early diagnosis and isolation to interrupt the transmission chain and contain the epidemic [4]. It is imperative to resolve COVID-19 as soon as possible due to its detrimental effects on humanity.

The gold standard for diagnosing COVID-19 is the reverse transcription-polymerase chain reaction (RT-PCR) test [5], [6]. However, this testing method has limitations, including high false-negative rates and sensitivity rates that can sometimes be as low as 60% [7]. COVID-19 primarily affects the lungs and can be detected through radiological imaging, such as X-rays and computed tomography (CT) scans [8], [9]. Although effective, manual evaluation of these images can be time-consuming, particularly for a widespread disease like COVID-19. To overcome this issue, a computer-aided diagnostic method has been

developed to predict COVID-19 from radiological images, using distinct patterns such as ground-glass opacity and embolism showing linear consolidation [10]. COVID-19 and pneumonia are diseases that have similar symptoms that affect the lungs and cause respiratory problems. Therefore, it is a very difficult process to distinguish between COVID-19 and non-COVID-19 Pneumonia. As a result of all these diagnostic and diagnostic difficulties and findings, our study has been carried out with the idea that a fast and effective automatic diagnosis system can be established. This study builds upon the knowledge and expertise obtained from previous studies [11–13], but it differs in several ways. First, it compares different clustering methods to generate parameters for various classification methods. Second, it uses three different feature selection methods and models with both CT and X-ray images, as well as four different datasets. Another distinguishing feature of this study from [11–13] is the analysis of combined data sets from different image sources, where both healthy and diseased images were merged and classified. Efforts were made to improve the performance of the system in this context.

The fundamental objective of this research is to create an automated system based on segmentation and machine learning techniques, capable of identifying COVID-19 disease on CT and X-ray images. We consolidated four datasets from different COVID-19 patient studies into one study. We performed binary classification to distinguish COVID-19 cases from non-COVID-19 cases, and COVID-19 positive cases from non-COVID-19 pneumonia cases. In the three-class classification process, we classified cases as COVID-19, non-COVID-induced pneumonia, and non-COVID-19. The data sets used were publicly accessible and were chosen for their comprehensiveness. Our study outperforms previous studies in terms of accuracy, speed, and other performance metrics. Furthermore, our study's high-performance rates across different imaging modalities and data sets indicate its adaptability to various data sets.

The subsequent sections of this paper are structured as follows. Section 2 comprises a comprehensive review of pertinent literature regarding COVID-19 diagnosis. Section 3 delineates the clinical aspects of the issue, outlines the utilized methodologies for detecting COVID-19 cases, and describes the datasets used in the experiment. Section 4 presents the experimental design and results of the COVID-19 detection. In Section 5, the outcomes of this study are analyzed and compared with those of other relevant studies in the literature. Finally, Section 6 concludes the article and proposes future research directions.

2. Literature Review

Numerous computer-based studies have been conducted in the literature for the detection and diagnosis of the COVID-19 virus, utilizing either CT or X-ray images. Specifically, studies employing X-ray images for the diagnosis of COVID-19 are as follows:

- Abraham et al. [14] conducted a study on the diagnosis of COVID-19 using two different X-ray datasets, where they employed Multi Convolutional Neural Networks (CNN) and BayesNet classifier. The initial dataset in the study was comprised of 453 samples positive for COVID-19 and 497 samples negative for COVID-19. The second dataset included 71 samples positive for COVID-19 and 7 samples negative for COVID-19. The study was able to achieve success rates of 91.16% and 97.44%, respectively, for these datasets. Joshi et al. utilized deep learning techniques for binary and four-class classification processes consisting of COVID-19, non-COVID-19, bacterial pneumonia, and viral pneumonia [15]. The accuracy rate for the binary classification process was determined to be 99.61%, whereas the multiclass classification process obtained an accuracy rate of 94.79%.
- Aslan et al. conducted a study in which they used X-ray images for detecting COVID-19 infection and segmenting lung images [16]. The study used 219 COVID-19 positive, 1341 COVID-19 negative, and 1345 viral pneumonia X-ray images, achieving an accuracy rate of 98.7%.
- Demir suggested an LSTM model-based technique for the automatic detection of COVID-19 cases using X-ray images [17]. The images were subjected to Sobel gradient and watershed segmentation during the preprocessing stage to improve the model's performance.
- Öztürk et al. conducted the DarkCovidNet study, which achieved a success rate of 98.08% using 1125 X-ray images, including 125 COVID-19 positive, 500 No Findings, and 500 Pneumonia images [18].
- Altan and Karasu utilized deep learning techniques in their study and achieved an accuracy rate of 99.69% using positive, negative, and viral pneumonia images [19].
- Nour et al. proposed a system that could classify X-ray images into three categories, COVID-19 positive, COVID-19 negative, and viral pneumonia, using Deep Features and Bayesian Optimization methods [20].
- Gupta et al. obtained a 99% recall and precision value in their study, named Integrated Stacking InstaCovNet-19, using a dataset consisting of 361 COVID-19-positive, 365 healthy, and 362 pneumonia images [21].
- Juel et al. propose a deep learning approach for recognizing Covid-19, Viral Pneumonia, Lung-Opacity, and normal chest from chest X-ray images using a Modified Convolutional Neural Network (M-CNN) and Bidirectional LSTM (BiLSTM) with a Multi-Support Vector Machine (M-SVM) classifier. The proposed method achieves an accuracy of 98.67% [22].
- Kong and Cheng propose a method for classifying chest X-ray images of COVID-19 patients using a fusion of DenseNet and VGG16 networks with an attention mechanism and ResNet segmentation. The proposed model

achieved 98% accuracy in binary and 97.3% three-category classification, making it a useful tool for clinicians and radiologists [23].

The studies utilizing CT images for the detection of COVID-19 in computer-aided studies in the literature are as follows:

- Mishra et al. used VGG16 and ResNet50 transfer learning methods for the detection of COVID-19 in CT images. In addition to a binary classification of COVID-19 and non-COVID-19, the study also performed multiclass classification as COVID-19, normal, and pneumonia. The study achieved 99% success in binary classification with VGG16 and ResNet50. The multiple classification performance results for VGG16 and ResNet50 were 86.74% and 88.52%, respectively [24].
- Chakraborty et al. used meta-heuristic and fuzzy segmentation methods to segment COVID-19 CT images and anomalies [25].
- Ardakani et al. classified COVID-19 and non-COVID-19 CT images using 10 deep-learning models and found that ResNet and Xception models were the most successful ones [26].
- Gilanie et al. classified three different public datasets and achieved accuracy, specificity, and sensitivity values of 96.68%, 95.65%, and 96.24%, respectively [27].
- Kalane et al. used a fully convolutional network method based on U-Net architecture for the automatic detection of COVID-19 in 1000 chest CT images and obtained an overall accuracy value of 94.10% [28].
- In their study, Li et al. utilized the 3D ResNet-18 model and reported a precision rate of 89.6% for detecting COVID-19 cases. The dataset they used consisted of 305 COVID-19-positive cases, 872 Community-Acquired Pneumonia (CAP) cases, and 1498 non-pneumonia CT images [29].
- Xu et al. proposed a new method for the detection of COVID-19 based on segmentation and feature extraction of CT images using the 3D-CNN model [30]. The accuracy rate for the binary classification process was determined to be 99.61%, whereas the multiclass classification process obtained an accuracy rate of 94.79%.
- Jaiswal et al. used the DenseNet201-based deep learning method for the classification of COVID-19 CT images as diseased or healthy and achieved an accuracy of 97% in the study [31].
- Kathamuthu et al. propose a deep transfer learning-based convolution neural network model for COVID-19 detection using computed tomography scan images for medical applications. The model aims to detect the presence of COVID-19 in chest CT images and has shown promising outcomes in detecting and monitoring COVID-19 patients. The VGG16 model in the paper achieved an accuracy of 98.00% [32].

Numerous studies have been conducted in the literature focusing on the detection, segmentation, and classification of COVID-19. In addition to the aforementioned studies, Göreke et al. proposed a system that employs blood data of COVID-19 patients to assist experts [33]. Using deep neural networks, they determined that ethnic and genetic differences have an impact on disease diagnosis. Based on the literature review, it can be observed that chest CT and X-ray imaging play a crucial role in identifying abnormalities for detecting COVID-19, and the implementation of image processing methods with machine learning algorithms has enabled this capability.

3. Material and Methods

3.1 Dataset

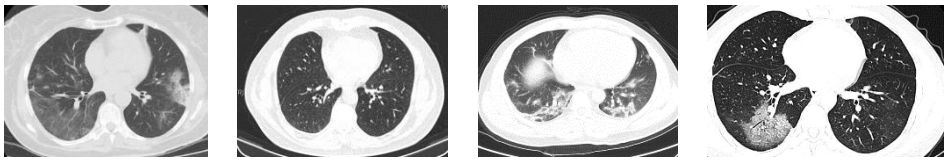
In our study, we performed analysis procedures on four different publicly accessible datasets, consisting of two X-ray and two CT image datasets. We aimed to select datasets frequently used in the literature to ensure easy comparisons of success rates. Table 1 provides details on the datasets, which contain varying numbers of samples from different locations. The study results are expected to be suitable for different COVID-19 variants represented in the datasets from various locations. Notably, the X-ray datasets include three distinct classes, unlike the CT datasets, comprising healthy and diseased lung images and pneumonia class not caused by COVID-19. This differentiation can aid healthcare professionals in distinguishing between pneumonia associated with COVID-19 and pneumonia that is not related to it, which can be difficult due to the severity of COVID-19. Our research aims to make a valuable contribution to the process of expert diagnosis in this regard.

Our study conducted analysis procedures on four different publicly accessible datasets, comprising two X-ray and two CT image datasets. We have selected datasets that are widely used in the literature to ensure that the obtained success rates can be easily compared. The details of the data used in the study are presented in Table 1. As shown in the table, the data sets consist of different numbers of samples obtained from various locations. It can be inferred that the findings of the study apply to different variants of COVID-19 based on the data sets from diverse locations. Moreover, the X-ray data sets consist of three distinct categories, which differ from the CT data sets. Whereas the CT images include healthy and diseased lung images, the X-ray images also incorporate the pneumonia category caused by reasons other than COVID-19. This distinction between COVID-19-induced pneumonia and pneumonia caused by other factors is significant, as medical professionals may sometimes misinterpret this due to the severity of the virus. Hence, this study can potentially help medical professionals to diagnose accurately.

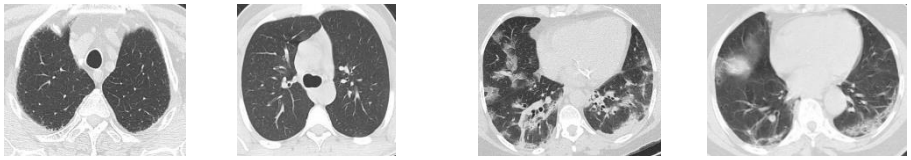
Table 1. Dataset information

	Type	# of class	# of cases	Location	Citation
Dataset 1	CT	2	(+) 349 (-) 397	China	[34]
Dataset 2	CT	2	(+) 1252 (-) 1230	Sao Paulo, Brazil	[35]
Dataset 3	X-ray	3	(+) 125 (-) 500 Non- COVID Pneumonia 500	Mixed (Spain, Canada, United Kingdom, etc.)	[18], [36], [37]
Dataset 4	X-ray	3	(+) 1200 (-) 1341 Non- COVID Pneumonia 1345	Italy	[38], [39]

In our study, four distinct data sets containing CT and X-ray images were utilized. Sample images from these data sets are illustrated in Figures 1 and 2. Figure 1 presents the normal and COVID-19 images of Datasets 1 and 2, which include CT images. The contrast between CT images of healthy individuals and those diagnosed with COVID-19 can be observed in the same figure.



Normal Cases (Dataset 1) COVID-19 affected cases (Dataset 1)

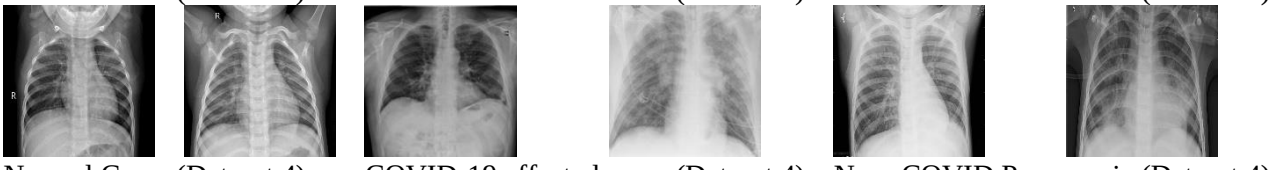


Normal Cases (Dataset 2) COVID-19 affected cases (Dataset 2)

Figure 1. Sample CT images from Dataset 1 [34] and Dataset 2 [35]

Datasets 3 and 4, comprising X-ray images, contain three distinct classes. Figure 2 displays representative images from these datasets that depict normal lung X-rays as well as those depicting COVID-19 and non-COVID-19 pneumonia. Notably, it is essential to develop methods to differentiate between pneumonia caused by COVID-19 and other etiologies.

Normal Cases (Dataset 3) COVID-19 affected cases (Dataset 3) Non- COVID Pneumonia (Dataset 3)



Normal Cases (Dataset 4) COVID-19 affected cases (Dataset 4) Non- COVID Pneumonia (Dataset 4)

Figure 2. Sample X-ray images from Dataset 3 [18], [36], [37] and Dataset 4 [38], [39]

3.2 Preprocessing

The data sets utilized in our study contained images with diverse dimensions and sizes. Therefore, we first converted all images to the same format, namely PNG format, and subsequently transformed them into gray levels. This resulted in 2-channel images that are less computationally expensive to process compared to 3-channel images. The sizes of the images in

the four different datasets varied significantly. This guarantee that the study can be used for various data sets, all images were resized and converted to the dimensions of 256 x 256. In the next step, we performed image sharpening by setting the Standard deviation of the Gaussian low pass filter value and sharpening value as '2'. Following that, the image intensity values were normalized from the range of [0, 1] to the range of [0, 0.87], which accentuates the dark regions relatively more. Our final preprocessing step was the application of the Wiener filter [40], which we applied after segmentation because it improved the success of the experimental studies in our research. The Wiener filter is expressed by the formula below:

$$W(u,v) = \frac{H(u,v)}{|H(u,v)|^2 + S_{nx}(u,v)} \quad (1)$$

In the context of image processing, the noise ratio is denoted as $S_{nx}(u,v)$, while the sinc function of the target pixel is represented by H (u, v). A neighborhood dimension of [4 4] is set for the Wiener filter. The parameters of the processes applied in the preprocessing stage were determined experimentally. After conducting experimental studies, the most appropriate values were determined.

3.3 Segmentation

The images in the datasets used in our study have homogenous regions due to their characteristics such as gray tones and texture features [41]. These features make them suitable for various clustering or segmentation methods to detect desired regions. In our study, we used k-means (KM) and fuzzy c-means (FCM) segmentation methods to identify anomalies in CT and X-ray images.

The FCM segmentation method is an unsupervised method that has been successfully applied in various fields such as medical imaging, target recognition, and data analysis [42]. This method does not require any class tags since its primary purpose is to group images based on their specific features. The objective of the algorithm is to minimize the J objective function in equation number 2, where X=(x_1, x_2, …, x_N) is an image with N pixels that we want to divide into c sets.

$$J = \sum_{j=1}^{N}\sum_{i=1}^{c} \mathrm{u}_{\mathrm{ij}}^{\mathrm{m}} \left\| x_j - v_i \right\|^2 \quad (2)$$

Here u_{ij}, denotes the membership value of the j_{th} pixel belonging to the i_{th} cluster, v_i denotes the center point of the i_{th} cluster, and m denotes the blur constantly.

In our study, the blur constant m was chosen as the default value of 2. As can be seen from Formula 2, the FCM method allows a data point to be included in more than one cluster. The decisive factor here is the membership value. The higher the membership value a data point is connected to a cluster, the more similar it is to that cluster. The total membership value of pixels in different clusters is always 1. The membership values and cluster centers are calculated according to formulas 3 and 4;

$$u_{ij} = \frac{1}{\sum_{\mathrm{k=1}}^{\mathrm{c}} \left(\frac{\left\| x_j - v_i \right\|}{\left\| x_j - v_k \right\|} \right)^{2/(m-1)}} \quad (3)$$

$$v_i = \frac{\sum_{j=1}^{N} u_{ij}^m x_j}{\sum_{j=1}^{N} u_{ij}^m} \quad (4)$$

The algorithm starts the first iteration with randomly selected cluster centers, and for the termination process, it decides according to the number of changes in the membership function or cluster centers in two consecutive iterations. Or, it can be terminated after a certain number of iterations.

In our study, we also employ the k-means (KM) clustering algorithm, which was developed by J.B. MacQueen in 1967, making it one of the oldest clustering algorithms [43]. It is a widely used non-parametric learning method. Unlike fuzzy c-means, the KM algorithm allows each data point to belong to only one cluster [44]. The steps of the k-means algorithm are as follows:

- First, the number of cluster centers is selected, and an equal number of samples are randomly chosen to be the centers.
- The remaining samples are assigned to the cluster they are closest to based on their distances from the cluster centers.
- The center points of the clusters are then recalculated based on the new sample assignments, and the distances of the samples to the centers are re-examined.
- The algorithm continues to run until there is no change in the cluster assignments.

The sum squared error (SSE) is used to evaluate the k-means clustering method. The clustering result with the lowest SSE value is considered to be the most successful result. SSE calculation is performed using the following formula [45];

$$SSE = \sum_{i=1}^{k} \sum_{x \in Ci} dist2(\mathrm{mi}, \mathrm{x}) \quad (5)$$

x: An object in C_i cluster, mi: Center point of C_i cluster, k: Number of clusters. Here, dist is the standard Euclidean Distance between two objects, an object whose x value is in the C_i set, and the mi value is the center point of the set C_i. The number

of clusters was determined to be 2 in both KM and FCM methods because of the black background and lungs in the anterior part in CT and X-ray images.

3.4 Feature extraction

Feature extraction is a process of identifying distinctive parts of an image from a large number of pixels [46]. In our study, we utilized three different feature extraction methods: Gray Level Co-occurrence Matrix (GLCM), Histogram of Oriented Gradients (HOG), and Local Binary Pattern (LBP).

In the first method used in our study, seven textural features based on GLCM were calculated for four different directions (0º, 45º, 90º, and 135º) from each image. Seven Haralick texture descriptors, including Angular Second Moment, Contrast, Inverse Difference Moment/Homogeneity, Dissimilarity, Entropy, Maximum Probability, and Inverse, were extracted from each co-occurrence matrix calculated at each of the four angles [47].

Histogram of Oriented Gradients (HOG) is a feature extraction method that has shown effective results in object and pattern recognition. The primary objective of the HOG method is to display the image as local histograms [48]. The process of extracting HOG features from an image involves applying horizontal and vertical Sobel filters. This is followed by calculating the gradient size and orientation angle during the feature extraction process.

The extraction of features using a Local Binary Pattern (LBP) is a non-parametric approach [49], [50]. LBP has important advantages, such as high tolerance to images with different lighting conditions, sensitivity to small changes in gray-level images, and low calculation cost, making it frequently preferred, especially in image processing studies. The process of calculating the LBP identifier obtained from a 3x3 frame according to the 8-neighborhood is computed using Equation 6.

$$\mathrm{LBP}_{P,R}(x_c) = \sum_{p=0}^{P-1} u(x_p - x_c)2^p, \qquad u(y) = \begin{cases} 0, & \text{if } y < 0 \\ 1, & \text{if } y \geq 0 \end{cases} \tag{6}$$

The notation used in our study defines x_c as the central pixel, x_p as the neighboring pixels surrounding the central pixel, R as the distance between the central pixel and its neighboring pixels, and P as the number of neighbors considered.

3.5 Classification

In our study, we employed three different classifiers to classify the features obtained from segmented images: Extreme Learning Machines (ELM), Support Vector Machines (SVM), and the k-nearest neighbor (kNN) method, which is a simple yet effective approach.

ELM is a type of single hidden layer feed-forward neural network (SLFN) [51], [52]. In this method, input weights are randomly determined, while output weights are calculated analytically [53]. In our study, we experimented with many hidden layers set to 4096 and a regularization parameter of 1e-1 for the ELM method. SVM, which stands for Support Vector Machine, is a supervised classification approach created for binary classification tasks but has proven to be effective in handling multi-class problems as well. The primary goal of SVM is to identify the hyperplane that can best differentiate the two classes, as noted in [54]. For multiclass SVM, we followed the one-versus-all approach.

$(x_i, y_i)_{1 \leq i \leq N}$ shows the training examples. Each example shows the size of the $x_i \in R^d$, d feature space. y_i shows class labels. SVM aims to find a hyperplane that separates the data from each other, where the samples with the same label will stay on the same side. For this, a line equation is defined as in 7.

$$y_i(w.x_i + b) > 0, \qquad \text{i=1, … , N} \tag{7}$$

If a hyperplane exists, as shown in Formula 7, it indicates that the data can be separated linearly. However, if the data is distributed nonlinearly, it can still be separated linearly by transforming the feature space into a high-dimensional feature spaceThe Support Vector Machines (SVM) is a type of classifier that is based on kernels. The literature on SVM commonly uses radial, polynomial, and linear kernels as kernel functions [55]. In our study, we employed the polynomial kernel as it performed better than the other kernels for our datasets.

On the other hand, k-Nearest Neighbor (kNN) is a non-parametric classification method that has been widely applied in various classification problems [56]. The classification process is performed by examining the nearest neighbors of the samples and assigning them to the cluster they are most similar. The kNN classifier generates feature vectors and labels of the training samples and applies the same features to the test sample whose class is unknown. Distances between data points are calculated, and the k closest samples are selected. Among these samples, the majority class is assigned to the new instance. In our study, we set the k value to 1 and used Euclidean distance for distance measurement. Despite its simplicity, kNN is a popular and effective method as it produces successful results and is simple and easy to apply.

3.6 Modeling

There are various methods for modeling data, with the most popular ones involving partitioning data into separate sets for training and testing. However, in cases where there are imbalanced class distributions, this partitioning approach may not be suitable. Instead, cross-validation (CV) is a widely used method where the entire dataset can be utilized for both training and testing. In our classification studies, we adopt the 10-fold cross-validation method, which involves dividing the dataset into 10 equal parts. During each iteration, one part is used for testing while the other nine parts are used for training. This process is repeated ten times, with the testing part being changed at each iteration. Finally, the average of the values obtained from

each iteration is calculated. This ensures that all data is utilized for both training and testing. A visual representation of the tenfold cross-validation method is illustrated in Figure 3.

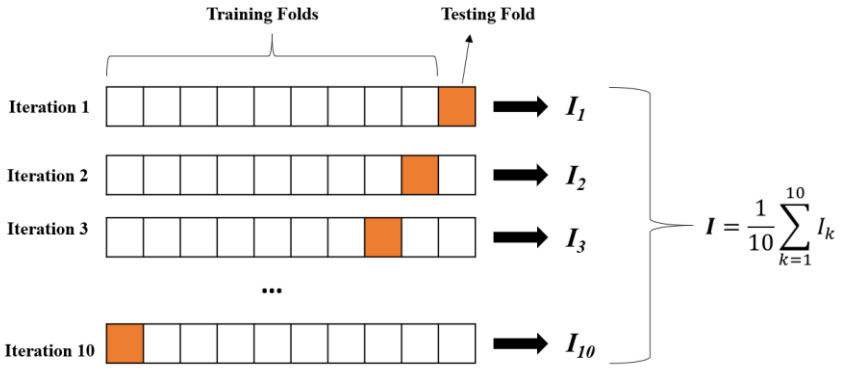


Figure 3. 10-fold cross-validation schema

3.7 Performance measurement

The metrics given below will be used to measure the success of the classification processes in our study.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{(\text{TP} + \text{TN} + \text{FP} + \text{FN})} * 100 \quad (8)$$

$$\text{Recall} = \frac{\text{TP}}{(\text{TP+FN})} * 100 \quad (9)$$

$$\text{Specificity} = \frac{\text{TN}}{(\text{TN} + \text{FP})} * 100 \quad (10)$$

$$\text{Precision} = \frac{\text{TP}}{(\text{TP+FP})} * 100 \quad (11)$$

$$\text{NPV} = \frac{\text{TN}}{(\text{TN} + \text{FN})} * 100 \quad (12)$$

TP shows correctly predicted positive samples, TN correctly predicted negative samples, FP incorrectly predicted positive samples, and FN incorrectly predicted negative samples. The choice of which metric to focus on depends on the particular aspect of the test under consideration. For instance, if identifying positive samples is more important, then the recall metric would be a suitable choice. This was the case in our study. On the other hand, the accuracy metric would be more appropriate if we want to assess the correct identification of both positive and negative samples. However, in cases where the class distribution is unbalanced, accuracy may not be a suitable evaluation metric [57].
The flow chart showing the methods and implementation process described in the third section is shown in Figure 4.

4. Experimental results

We conducted experimental studies using four different datasets with varying sizes and characteristics. The experiments were performed on a desktop computer equipped with an i5 processor and 4 GB graphic cards. To differentiate between healthy and patient samples, we employed five performance metrics to assess the outcomes. We will report all five metrics in our results; however, Recall and Accuracy metrics are of primary importance to our study.
As depicted in Figure 4, our study included stages of data acquisition, preprocessing, segmentation, feature extraction, classification, and modeling, as well as performance evaluation. The feature extraction process was conducted on segmented images, and the segmented images obtained by employing KM and FCM methods are presented in Figures 5 and 6. The FCM method divided CT images into two clusters, as shown in Figure 5. The segmentation results indicate significant differences between diseased and healthy images. As previously mentioned, COVID-19 anomalies in the lungs become more apparent after segmentation, which is evident in both Figure 5 and Figure 6.

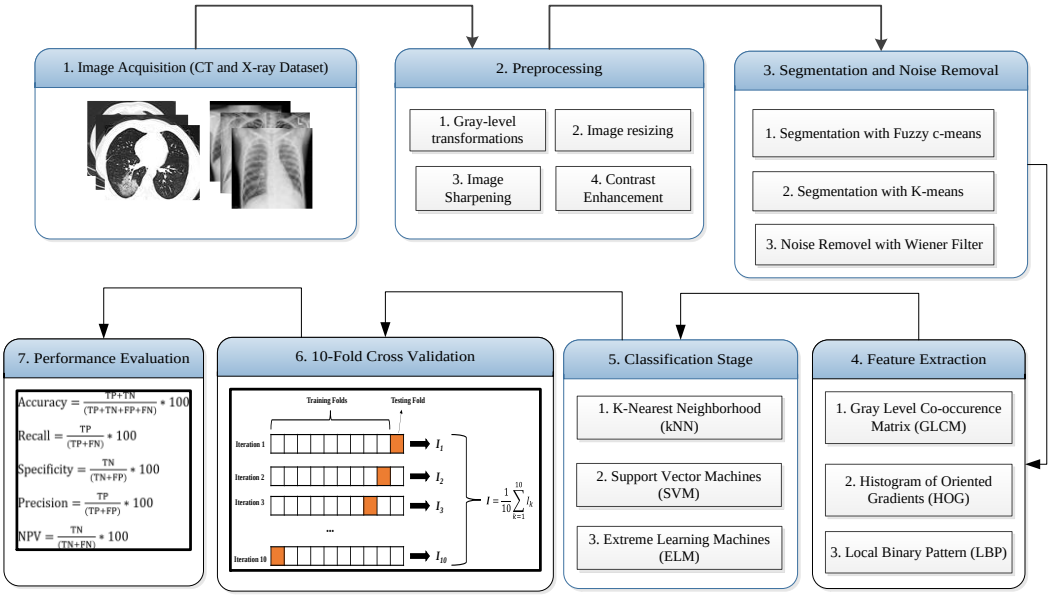


Figure 4. A diagram illustrating the sequence of actions performed by the system.

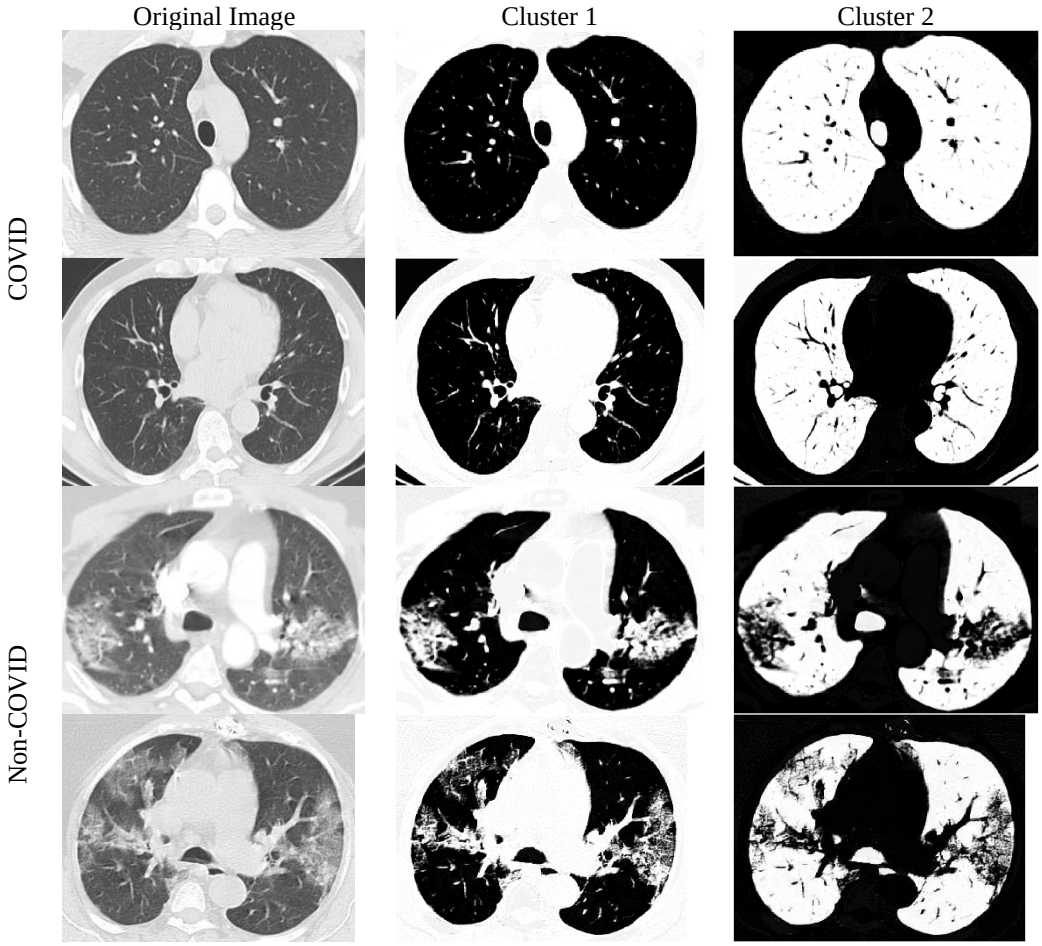


Figure 5. Segmented Sample CT images with FCM method

Figure 6 shows a few examples of diseased and healthy images for obtaining the KM segmentation method on X-ray images. Here, too, the dashed structures on the new images obtained as a result of the segmentation of the images in the first two lines, which are COVID-19, stand out. However, the result sets obtained by segmentation of healthy images seen in the last two lines are seen smoothly and clearly.

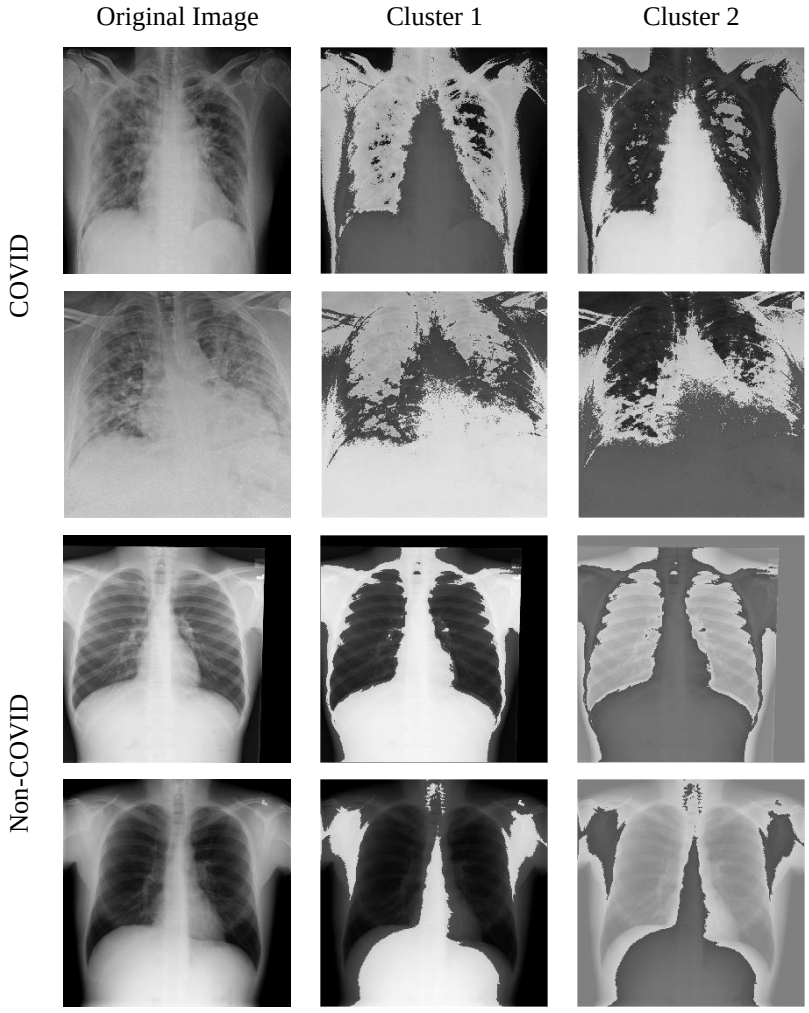


Figure 6. Sample segmented X-ray images with the k-means method

The performance measurement results obtained for four different datasets in the study are presented in Tables 2, 3, 4, and 5. All the detailed results obtained by combining three different classifiers, three different feature extraction methods, and two different segmentation methods are shown in all four tables. Table 2 displays the binary classification results for CT datasets. Upon inspection, it is observed that Dataset 1 yields a Recall value of 90.8% and an Accuracy value of 90.2%. For Dataset 2, these values increase to 99.4% and 98.8%, respectively. The images in Dataset 1 are low resolution and noisy, while Dataset 2 comprises images with higher resolution and less noise. It is believed that this difference accounts for the performance discrepancy. Additionally, Table 2 indicates that the KM and FCM segmentation methods do not offer a significant advantage over one another. Both methods exhibit superior results in different scenarios. FCM segmentation produces better results for Dataset 1, whereas the KM method performs better for Dataset 2. Upon examining the feature extraction methods, it is observed that the GLCM method and its classification results are subpar compared to the HOG and LBP methods, which yield better results across all models. Comparing the HOG and LBP methods, Table 2 demonstrates that the LBP method delivers relatively superior results. Although the HOG method produces comparable results, the LBP method stands out as a feature extraction method in the model that yields the highest values in both CT datasets. Looking at the classification methods in Table 2, it is evident that the kNN method is the most effective, consistent with many prior studies. Additionally, ELM yields relatively better results than SVM.

Table 3 presents the three-class classification results for X-ray data sets, including COVID-19 positive, healthy, and non-COVID-19-derived pneumonia classes. As shown in Table 3, the best results are achieved with the ELM-SVM methods, which differ from Table 2, which presents the results for CT data sets. For Dataset 3, the highest values are obtained with the SVM, LBP, and KM combination, with a Recall value of 84.0% and an accuracy value of 83.9%. However, for Dataset 4, the triplet of ELM, LBP, and KM produced higher results, with a Recall value of 98.7% and an accuracy value of 99.6%. The results for Dataset 4 are considerably better than those for Dataset 3 because the images in Dataset 4 are cleaner, less noisy, and higher in resolution. In particular, Dataset 3 contains many images with various artifacts, which negatively affects the success. Table 3 also shows that the KM segmentation method performs better than the FCM method in X-ray images. Additionally, the LBP method is more successful than the HOG method in both X-ray and CT images, while the GLCM method yields the least successful results as a feature extraction method.

As previously mentioned, our X-ray data sets include three different classes. In contradistinction to the CT image sets, the X-ray datasets comprise images of non-COVID-19 pneumonia, enabling the differentiation between images of COVID-19 and those of healthy individuals, as well as discerning between images of COVID-19 and pneumonia resulting from other causes. The results for these scenarios are presented in Tables 4 and 5. Table 4 displays the classification results of X-ray images according to COVID-19 and pneumonia classes, whereas Table 5 presents the results for COVID-19 and healthy classes. As known, COVID-19 disease causes pneumonia, making it a time-consuming process for an expert to differentiate between pneumonia and not COVID-19, which can lead to errors. As shown in Table 4, while Dataset 3's Recall value was 92.8%, the accuracy value was 97.7%. For Dataset 4, these values were 99.1% and 99.3%, respectively. This finding suggests that the system we have created can accurately differentiate between pneumonia caused by COVID-19 and pneumonia caused by other factors, which is a noteworthy and valuable outcome. When we examine Table 5, which is related to the detection of COVID-19 and healthy individuals from X-ray images, an accuracy rate of 98.6% was achieved for Dataset 3, and an accuracy rate of 99.5% was achieved for Dataset 4. These findings demonstrate that the distinction between healthy and sick individuals can be made with even higher accuracy.
Table 6 presents the average execution time of the classification method utilized in this research. The ELM method, which includes a large number of hidden layers, requires a considerably longer processing time. On the other hand, the kNN classifier provides the shortest processing times. Moreover, the SVM method produces results similar to those of the kNN method. As can be seen from Table 6, the size of the data set also affects the classification process.

Table 7 demonstrates the highest accuracy rates obtained for the four distinct data sets in this study through the confusion matrix. The confusion matrices c and d indicate erroneous classifications between COVID-19 positive and negative, as well as COVID-19 negative and non-COVID-19 ones. Another noteworthy finding is that COVID-19-positive cases and pneumonia can be more successfully distinguished.

Table 7. Confusion Matrices with Performance Metrics for Binary and Multiclass Classification in Four Different Datasets. Yellow values show Accuracy, Bottom-Left blue value Precision, Bottom-right blue value NPV, Right-upper blue recall, and Right-bottom blue specificity values.

Dataset 1

Actual Class		COVID (+)	COVID (-)	
	COVID (+)	317	41	88.5
	COVID (-)	32	356	91.8
		90.8	89.7	90.2

Predicted Class

Dataset 2

Actual Class		COVID (+)	COVID (-)	
	COVID (+)	1207	7	99.4
	COVID (-)	22	1245	98.2
		98.3	99.4	98.8

Predicted Class

Dataset 3

Actual Class		COVID (+)	Non-COVID	Pneumonia	
	COVID (+)	105	2	4	94.6
	No Findings	4	425	82	83.2
	Pneumonia	16	73	414	82.3
		84.0	85.0	82.8	83.9

Predicted Class

Dataset 4

Actual Class		COVID (+)	Non-COVID	Pneumonia	
	COVID (+)	1184	5	7	99.0
	No Findings	9	1295	79	93.8
	Pneumonia	7	41	1259	96.3
		98.7	96.6	93.6	96.2

Predicted Class

Table 2. Mixed performance measurement results for Dataset 1 and 2 (CT datasets) were obtained with three different classifiers, three different feature extraction, and two different segmentation methods (The highest performance values are shown in bold)

Classification Methods	Feature Extraction Methods	Segmentation Methods	Dataset 1					Dataset 2				
			Accuracy	Recall	Specificity	Precision	NPV	Accuracy	Recall	Specificity	Precision	NPV
ELM	GLCM	KM	71.3	69.3	73.0	69.3	73.0	86.9	86.9	86.9	87.1	86.7
		FCM	60.3	54.7	65.2	58.1	62.1	82.0	80.4	83.6	83.3	80.7
	HOG	KM	79.5	77.9	80.9	78.2	80.7	93.6	92.5	94.6	94.6	92.5
		FCM	82.0	81.1	82.9	80.6	83.3	94.6	94.4	94.8	94.8	94.3
	LBP	KM	84.0	83.4	84.6	82.7	85.3	94.5	93.8	95.3	95.3	93.8
		FCM	84.2	82.5	85.6	83.5	84.8	93.3	92.8	93.8	93.9	92.8
kNN	GLCM	KM	76.4	77.1	75.8	73.7	79.0	94.5	96.0	93.0	93.3	95.8
		FCM	67.0	65.9	68.0	64.4	69.4	88.7	90.2	87.2	87.8	89.7
	HOG	KM	79.2	75.9	82.1	78.9	79.5	97.3	98.7	95.9	96.1	98.7
		FCM	79.8	77.1	82.1	79.1	80.3	97.1	98.6	95.5	95.7	98.5
	LBP	KM	86.6	88.0	85.4	84.1	89.0	**98.8**	**99.4**	**98.2**	**98.3**	**99.4**
		FCM	**90.2**	**90.8**	**89.7**	**88.5**	**91.8**	97.9	99.0	96.9	97.0	98.9
SVM	GLCM	KM	56.0	65.3	47.9	52.4	61.1	63.3	61.6	65.0	64.2	62.4
		FCM	49.5	63.3	37.2	47.0	53.6	62.2	65.6	58.8	61.9	62.7
	HOG	KM	74.9	74.5	75.3	72.6	77.1	89.8	86.5	93.2	92.9	87.1
		FCM	76.7	70.8	81.9	77.4	76.1	89.0	87.1	91.0	90.8	87.4
	LBP	KM	75.7	73.1	78.1	74.6	76.7	91.8	89.9	93.7	93.5	90.1
		FCM	77.9	75.4	80.1	76.9	78.7	88.7	87.9	89.5	89.5	87.9

Table 3. Multi-class mixed performance measurement results for Datasets 3 and 4 (X-ray datasets) were obtained with three different classifiers, three different feature extraction, and two different segmentation methods (The highest performance values are shown in bold)

Classification Methods	Feature Extraction Methods	Segmentation Methods	Dataset 3 (Three Class)					Dataset 4 (Three Class)				
			Accuracy	Recall	Specificity	Precision	NPV	Accuracy	Recall	Specificity	Precision	NPV
ELM	GLCM	KM	72.2	54.4	97.6	73.9	94.5	74.7	78.3	91.8	81.0	90.4
		FCM	65.2	33.6	96.1	51.9	92.0	70.5	72.5	89.4	75.3	87.9
	HOG	KM	80.2	76.0	99.2	92.2	97.1	96.0	98.6	99.3	98.5	99.4
		FCM	77.2	82.4	98.7	88.8	97.8	96.3	98.3	99.6	99.1	99.3
	LBP	KM	83.1	85.6	99.3	93.9	98.2	**98.7**	**99.6**	**99.6**	**99.0**	**99.4**
		FCM	82.0	84.8	98.9	90.6	98.1	96.8	98.8	99.7	99.4	99.5
kNN	GLCM	KM	71.6	48.8	97.3	69.3	93.8	80.6	86.3	95.4	89.4	94.0

		FCM	67.7	60.8	95.2	61.3	95.1	79.9	86.0	97.1	92.9	93.9
	HOG	KM	73.9	72.8	96.7	73.4	96.6	92.1	96.0	99.3	98.4	98.2
		FCM	72.5	77.6	96.1	71.3	97.2	90.7	94.0	99.7	99.3	97.4
	LBP	KM	78.8	80.8	99.2	92.7	97.6	94.4	97.3	99.7	99.2	98.8
		FCM	77.2	83.2	99.1	92.0	97.9	94.1	97.9	99.8	99.5	99.1
SVM	GLCM	KM	43.1	33.6	86.9	24.3	91.3	80.6	86.3	95.4	89.4	94.0
		FCM	43.8	48.8	91.5	41.8	93.5	79.9	86.0	97.1	92.9	93.9
	HOG	KM	81.2	71.2	99.1	90.8	96.5	92.1	96.0	99.3	98.4	98.2
		FCM	79.4	81.6	99.5	95.3	97.7	90.7	94.0	99.7	99.3	97.4
	LBP	KM	**83.9**	**84.0**	**99.4**	**94.6**	**98.0**	96.1	98.3	99.4	98.6	99.3
		FCM	83.2	83.2	99.7	97.2	97.9	96.1	98.0	99.2	98.2	99.1

Table 4. Binary classification results in which the distinction between COVID-19 and pneumonia is addressed in X-ray data sets (The highest performance values are shown in bold)

			Dataset 3 (Binary Class)					**Dataset 4 (Binary Class)**				
Classification Methods	Feature Extraction Methods	Segmentation Methods	Accuracy	Recall	Specificity	Precision	NPV	Accuracy	Recall	Specificity	Precision	NPV
ELM	GLCM	KM	88.6	61.6	95.4	77.0	90.9	84.4	84.1	84.8	83.1	85.6
		FCM	82.7	36.8	94.2	61.3	85.6	78.7	75.7	81.4	78.4	78.9
	HOG	KM	95.4	84.0	98.2	92.1	96.1	98.9	98.8	99.1	99.0	98.9
		FCM	97.3	92.0	98.6	94.3	98.0	99.3	98.9	99.6	99.5	99.0
	LBP	KM	96.8	90.4	98.4	93.4	97.6	**99.3**	**99.1**	99.4	99.3	**99.2**
		FCM	**97.7**	**92.8**	**99.0**	**95.9**	**98.2**	99.3	98.9	**99.6**	**99.6**	99.0
kNN	GLCM	KM	87.8	93.2	94.0	72.5	91.1	88.4	87.2	89.5	88.1	88.7
		FCM	86.4	64.8	91.8	66.4	91.3	91.7	89.1	94.1	93.0	90.6
	HOG	KM	90.2	77.6	93.4	74.6	94.3	97.8	96.7	98.9	98.7	97.1
		FCM	90.7	81.6	93.0	74.5	95.3	98.2	96.3	99.8	99.7	96.8
	LBP	KM	95.4	84.8	98.0	91.4	96.3	98.8	97.9	99.6	99.5	98.2
		FCM	94.2	82.4	97.2	88.0	95.7	98.9	98.3	99.6	99.5	98.5
SVM	GLCM	KM	51.8	54.4	51.2	21.8	81.8	49.9	59.8	41.1	47.5	53.4
		FCM	70.7	62.4	72.8	36.4	88.6	57.2	65.6	49.8	53.8	61.9
	HOG	KM	93.3	76.0	97.6	88.8	94.2	98.9	98.5	99.2	99.1	98.7
		FCM	95.8	85.6	98.4	93.0	96.5	99.1	98.8	99.3	99.2	98.9
	LBP	KM	96.6	88.0	98.8	94.8	97.1	98.7	98.5	99.0	98.8	98.7
		FCM	96.8	89.6	98.6	94.1	97.4	98.9	98.5	99.2	99.1	98.7

Table 5. Binary classification results in which the distinction between COVID-19 and healthy is discussed in the X-ray data sets (The highest performance values are shown in bold)

			Dataset 3 (Binary Class)					**Dataset 4 (Binary Class)**				
Classification Methods	Feature Extraction Methods	Segmentation Methods	Accuracy	Recall	Specificity	Precision	NPV	Accuracy	Recall	Specificity	Precision	NPV
ELM	GLCM	KM	92.0	72.0	97.0	85.7	93.3	93.0	91.9	94.0	93.2	92.9
		FCM	91.5	68.0	97.4	86.7	92.4	87.4	85.3	89.3	87.7	87.2
	HOG	KM	97.3	89.6	99.2	96.6	97.4	99.4	99.2	99.6	99.6	99.3
		FCM	98.4	95.2	99.2	96.7	98.8	99.3	99.0	99.5	99.4	99.1
	LBP	KM	98.2	92.0	99.8	99.1	98.0	**99.5**	**99.4**	99.6	99.5	**99.5**
		FCM	**98.6**	**95.2**	**99.4**	**97.5**	**98.8**	99.3	98.8	99.7	99.7	99.0
kNN	GLCM	KM	90.4	64.8	96.8	83.5	91.7	95.9	93.8	97.8	97.5	94.6
		FCM	91.2	84.8	92.8	74.6	96.1	95.6	92.6	98.2	97.9	93.7
	HOG	KM	95.7	88.8	97.4	89.5	97.2	98.3	97.1	99.4	99.3	97.4
		FCM	95.2	90.4	96.4	86.3	97.6	97.8	95.6	99.7	99.7	96.2
	LBP	KM	97.9	92.8	99.2	96.7	98.2	99.2	98.5	99.8	99.7	98.7
		FCM	97.0	90.4	98.6	94.2	97.6	99.4	98.8	**99.9**	**99.8**	99.0
SVM	GLCM	KM	57.9	52.0	59.4	24.3	83.2	55.8	68.6	44.4	52.5	61.3
		FCM	91.2	83.2	93.2	75.4	95.7	63.7	67.3	60.5	60.4	67.4
	HOG	KM	97.7	91.2	99.4	97.4	97.8	99.2	98.8	99.5	99.4	99.0
		FCM	98.1	95.2	98.8	95.2	98.8	99.2	98.9	99.4	99.3	99.0
	LBP	KM	98.1	92.8	99.4	97.5	98.2	99.4	99.1	99.7	99.7	99.2
		FCM	97.8	90.4	99.6	98.3	97.6	99.0	98.4	99.6	99.5	98.6

Table 6. Classification average execution time of the method

		Dataset-1	**Dataset-2**	**Dataset-3**	**Dataset-4**
Time (s)	ELM	33.9	59.3	42.2	91.2
	kNN	29.9	3.1	0.4	2.5
	SVM	45.6	5.3	1.0	4.7

4. Discussion

Our study aims to propose an approach that can achieve successful results with various imaging modes and datasets and support early diagnosis of COVID-19. In the modeling, a system has been developed to be applied to datasets that are independent of the dataset, and the results demonstrate success on all four different datasets. Table 8 compares our study with other studies in the literature, focusing on the use of one of the four datasets used in our study. However, it should be noted that comparing different studies using the same dataset may not be entirely accurate since they may use different subsets of the same dataset. Despite this limitation, Table 8 provides a general comparison of our study with other studies. Upon reviewing Table 8, it can be observed that our study achieved a higher success rate than the state-of-the-art studies in nearly all of the four distinct datasets. For instance, in the study of Ozturk et al., while an accuracy of 87.02% was obtained in the multiclass classification of X-ray images, this rate was 83.9% in our study. However, our study produced far more successful results in terms of recall values in the same comparison. Therefore, our study can be considered very successful in detecting COVID-19-positive samples. Similarly, while Chowdhury et al. achieved an accuracy rate of 99.41% for multiclass classification in their study, our study obtained an accuracy rate of 99.3%. Nonetheless, our study achieved a recall value of 99.6%, whereas Chowdhury et al. achieved a recall value of 96.61%.

Table 8. Comparison of the performances of the state of the art studies

Author(s)	Dataset	Accuracy (%)	Recall (%)	Precision (%)
Yang et al. [34]	Dataset-1	89.0	NA	NA
Soares et al. [35]	Dataset-2	97.38	95.53	99.16
Jaiswal et al. [31]	Dataset-2	96.25	96.29	96.29
Ozturk et al. [18]	Dataset-3 (Binary)	98.08	95.13	98.03
Ozturk et al. [18]	Dataset-3 (Multiclass)	87.02	85.35	89.96
Rahman et al. [39]	Dataset-4 (Multiclass)	96.29	97.28	NA
Chowdhury et al. [38]	Dataset-4 (Binary)	99.41	99.41	99.42
Chowdhury et al. [38]	Dataset-4 (Multiclass)	97.74	96.61	96.61
This Study	Dataset-1	90.2	90.8	88.5
This Study	Dataset-2	98.8	99.4	98.3
This Study	Dataset-3 (Binary)	97.7	92.8	95.9
This Study	Dataset-3 (Multiclass)	83.9	94.0	94.6
This Study	Dataset-4 (Binary)	99.3	99.1	99.6
This Study	Dataset-4 (Multiclass)	98.7	99.6	99.0

In general, we can say that a more effective system has been produced compared to other studies. It is noteworthy that the developed system operates on four distinct datasets acquired through various imaging modes, delivering superior performance when compared to prior studies in the field. Furthermore, the classification times presented in Table 6 indicate that the evaluation duration of the system is less than 1 minute. This is seen as another positive aspect that needs attention.

5. Conclusion

Our research yielded a comprehensive system that can diagnose COVID-19 from datasets captured through various imaging modes. This system was created by adopting a holistic approach that integrates image processing, segmentation, feature extraction, and machine learning methods. Our model can be applied to any dataset and has demonstrated its efficacy by achieving successful results. We developed a holistic approach that is not limited to any specific dataset. Our study not only distinguished between COVID-positive and negative cases but also conducted a multi-class classification process, where we identified COVID-19-positive, healthy, and pneumonia classes. Our research achieved a remarkable accuracy rate of 97.7% and 99.3% in two distinct datasets in identifying individuals with COVID-19 and those with pneumonia but without COVID-19. Considering that non-COVID-19 pneumonia and COVID-19 pneumonia exhibit comparable characteristics, making it challenging for healthcare experts to differentiate between them, the achievements of this research are noteworthy.
For more than two years, the entire world has been mobilized to detect and treat COVID-19. Although the contagiousness of the virus has decreased, pneumonia still affects people due to various factors. Therefore, the results obtained from our study are considered important, and they could provide support for medical professionals in their decision-making. A limitation of our study is the lack of information on the period of the disease in which the data were collected. Future studies could create datasets containing this information to provide a more in-depth analysis of the effects of the disease.

References

[1] "World Health Organization Coronavirus (COVID-19) Dashboard, Retrieved 05th May 2021. (Web Link: https://covid19.who.int/)." https://covid19.who.int/

[2] C. Wang *et al.*, "Immediate psychological responses and associated factors during the initial stage of the 2019 coronavirus disease (COVID-19) epidemic among the general population in China," *Int J Environ Res Public Health*, vol. 17, no. 5, p. 1729, 2020.

[3] C. M. A. de O. Lima, "Information about the new coronavirus disease (COVID-19)," *Radiol Bras*, vol. 53, no. 2, pp. V–VI, 2020.

[4] M. W. M. Mustafa, "Audiological profile of asymptomatic Covid-19 PCR-positive cases," *Am J Otolaryngol*, vol. 41, no. 3, p. 102483, 2020.

[5] Y. Fang *et al.*, "Sensitivity of chest CT for COVID-19: comparison to RT-PCR," *Radiology*, vol. 296, no. 2, pp. E115–E117, 2020.

[6] L. Lan *et al.*, "Positive RT-PCR test results in patients recovered from COVID-19," *JAMA*, vol. 323, no. 15, pp. 1502–1503, 2020.

[7] T. Ai *et al.*, "Correlation of chest CT and RT-PCR testing for coronavirus disease 2019 (COVID-19) in China: a report of 1014 cases," *Radiology*, vol. 296, no. 2, pp. E32–E40, 2020.

[8] H. Shi *et al.*, "Radiological findings from 81 patients with COVID-19 pneumonia in Wuhan, China: a descriptive study," *Lancet Infect Dis*, vol. 20, no. 4, pp. 425–434, 2020.

[9] D. Dong *et al.*, "The role of imaging in the detection and management of COVID-19: a review," *IEEE Rev Biomed Eng*, 2020.

[10] C. Hani *et al.*, "COVID-19 pneumonia: a review of typical CT findings and differential diagnosis," *Diagn Interv Imaging*, vol. 101, no. 5, pp. 263–268, 2020.

[11] A. SAYGILI, "Analysis and Segmentation of X-ray Images of COVID-19 Patients using the k-means Algorithm," *Veri Bilimi*, vol. 4, no. 3, pp. 1–6.

[12] A. Saygılı, "A new approach for computer-aided detection of coronavirus (COVID-19) from CT and X-ray images using machine learning methods," *Appl Soft Comput*, vol. 105, p. 107323, 2021.

[13] A. Saygılı, "Computer-aided detection of COVID-19 from CT images based on Gaussian mixture model and kernel support vector machines classifier," *Arab J Sci Eng*, vol. 47, no. 2, pp. 2435–2453, 2022.

[14] B. Abraham and M. S. Nair, "Computer-aided detection of COVID-19 from X-ray images using multi-CNN and Bayesnet classifier," *Biocybern Biomed Eng*, vol. 40, no. 4, pp. 1436–1445, 2020.

[15] R. C. Joshi *et al.*, "A deep learning-based COVID-19 automatic diagnostic framework using chest X-ray images," *Biocybern Biomed Eng*, vol. 41, no. 1, pp. 239–254, 2021.

[16] M. F. Aslan, M. F. Unlersen, K. Sabanci, and A. Durdu, "CNN-based transfer learning–BiLSTM network: A novel approach for COVID-19 infection detection," *Appl Soft Comput*, vol. 98, p. 106912, 2021, doi: https://doi.org/10.1016/j.asoc.2020.106912.

[17] F. Demir, "DeepCoroNet: A deep LSTM approach for automated detection of COVID-19 cases from chest X-ray images," *Appl Soft Comput*, vol. 103, p. 107160, 2021.

[18] T. Ozturk, M. Talo, E. A. Yildirim, U. B. Baloglu, O. Yildirim, and U. R. Acharya, "Automated detection of COVID-19 cases using deep neural networks with X-ray images," *Comput Biol Med*, vol. 121, p. 103792, 2020.

[19] A. Altan and S. Karasu, "Recognition of COVID-19 disease from X-ray images by hybrid model consisting of 2D curvelet transform, chaotic salp swarm algorithm and deep learning technique," *Chaos Solitons Fractals*, vol. 140, p. 110071, 2020.

[20] M. Nour, Z. Cömert, and K. Polat, "A novel medical diagnosis model for COVID-19 infection detection based on deep features and Bayesian optimization," *Appl Soft Comput*, vol. 97, p. 106580, 2020.

[21] A. Gupta, S. Gupta, and R. Katarya, "InstaCovNet-19: A deep learning classification model for the detection of COVID-19 patients using Chest X-ray," *Appl Soft Comput*, vol. 99, p. 106859, 2021.

[22] J. Sikder, N. Datta, and D. Tripura, "A Deep Learning Approach for Recognizing Covid-19 from Chest X-ray using Modified CNN-BiLSTM with M-SVM," in *2022 International Conference on Electrical, Computer and Energy Technologies (ICECET)*, IEEE, Jul. 2022, pp. 1–6. doi: 10.1109/ICECET55527.2022.9872776.

[23] L. Kong and J. Cheng, "Classification and detection of COVID-19 X-Ray images based on DenseNet and VGG16 feature fusion," *Biomed Signal Process Control*, vol. 77, p. 103772, Aug. 2022, doi: 10.1016/j.bspc.2022.103772.

[24] N. K. Mishra, P. Singh, and S. D. Joshi, "Automated Detection of COVID-19 from CT scan using Convolutional Neural Network," *Biocybern Biomed Eng*, 2021.
[25] S. Chakraborty and K. Mali, "SuFMoFPA: A superpixel and meta-heuristic based fuzzy image segmentation approach to explicate COVID-19 radiological images," *Expert Syst Appl*, vol. 167, p. 114142, 2021.
[26] A. A. Ardakani, A. R. Kanafi, U. R. Acharya, N. Khadem, and A. Mohammadi, "Application of deep learning technique to manage COVID-19 in routine clinical practice using CT images: Results of 10 convolutional neural networks," *Comput Biol Med*, vol. 121, p. 103795, 2020.
[27] G. Gilanie *et al.*, "Coronavirus (COVID-19) detection from chest radiology images using convolutional neural networks," *Biomed Signal Process Control*, vol. 66, p. 102490, 2021.
[28] P. Kalane, S. Patil, B. P. Patil, and D. P. Sharma, "Automatic detection of COVID-19 disease using U-Net architecture based fully convolutional network," *Biomed Signal Process Control*, vol. 67, p. 102518, 2021.
[29] Y. Li *et al.*, "Efficient and effective training of COVID-19 classification networks with self-supervised dual-track learning to rank," *IEEE J Biomed Health Inform*, vol. 24, no. 10, pp. 2787–2797, 2020.
[30] X. Xu *et al.*, "A deep learning system to screen novel coronavirus disease 2019 pneumonia," *Engineering*, vol. 6, no. 10, pp. 1122–1129, 2020.
[31] A. Jaiswal, N. Gianchandani, D. Singh, V. Kumar, and M. Kaur, "Classification of the COVID-19 infected patients using DenseNet201 based deep transfer learning," *J Biomol Struct Dyn*, pp. 1–8, 2020.
[32] N. D. Kathamuthu *et al.*, "A deep transfer learning-based convolution neural network model for COVID-19 detection using computed tomography scan images for medical applications," *Advances in Engineering Software*, vol. 175, p. 103317, Jan. 2023, doi: 10.1016/j.advengsoft.2022.103317.
[33] V. Göreke, V. Sarı, and S. Kockanat, "A novel classifier architecture based on deep neural network for COVID-19 detection using laboratory findings," *Appl Soft Comput*, vol. 106, p. 107329, 2021.
[34] J. Zhao, Y. Zhang, X. He, and P. Xie, "Covid-ct-dataset: a ct scan dataset about covid-19," *arXiv preprint arXiv:2003.13865*, vol. 490, 2020.
[35] P. Angelov and E. Almeida Soares, "SARS-CoV-2 CT-scan dataset: A large dataset of real patients CT scans for SARS-CoV-2 identification," *medRxiv*, 2020.
[36] J. P. Cohen, P. Morrison, L. Dao, K. Roth, T. Q. Duong, and M. Ghassemi, "Covid-19 image data collection: Prospective predictions are the future," *arXiv preprint arXiv:2006.11988*, 2020.
[37] X. Wang, Y. Peng, L. Lu, Z. Lu, M. Bagheri, and R. M. Summers, "Chestx-ray8: Hospital-scale chest x-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2097–2106.
[38] M. E. H. Chowdhury *et al.*, "Can AI help in screening viral and COVID-19 pneumonia?," *IEEE Access*, vol. 8, pp. 132665–132676, 2020.
[39] T. Rahman *et al.*, "Exploring the effect of image enhancement techniques on COVID-19 detection using chest X-ray images," *Comput Biol Med*, vol. 132, p. 104319, 2021.
[40] J. S. Lim and A. V Oppenheim, "Enhancement and bandwidth compression of noisy speech," *Proceedings of the IEEE*, vol. 67, no. 12, pp. 1586–1604, 1979.
[41] R. M. Haralick and L. G. Shapiro, "Image segmentation techniques," *Comput Vis Graph Image Process*, vol. 29, no. 1, pp. 100–132, 1985.
[42] J. C. Bezdek, R. Ehrlich, and W. Full, "FCM: The fuzzy c-means clustering algorithm," *Comput Geosci*, vol. 10, no. 2–3, pp. 191–203, 1984.
[43] A. Likas, N. Vlassis, and J. J. Verbeek, "The global k-means clustering algorithm," *Pattern Recognit*, vol. 36, no. 2, pp. 451–461, 2003.
[44] I. Davidson, "Understanding K-means non-hierarchical clustering," *SUNY Albany Technical Report*, vol. 2, pp. 2–14, 2002.
[45] T. Pang-Ning, M. Steinbach, and V. Kumar, "Introduction to data mining Addison-Wesley," 2005.
[46] P. K. Bhagat, P. Choudhary, and K. M. Singh, "Chapter 13 - A comparative study for brain tumor detection in MRI images using texture features," in *Sensors for Health Monitoring*, N. Dey, J. Chaki, and R. Kumar, Eds., Academic Press, 2019, pp. 259–287. doi: https://doi.org/10.1016/B978-0-12-819361-7.00013-0.
[47] R. M. Haralick, K. Shanmugam, and I. H. Dinstein, "Textural features for image classification," *IEEE Trans Syst Man Cybern*, no. 6, pp. 610–621, 1973.
[48] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, Ieee, 2005, pp. 886–893.

[49] T. Ojala, M. Pietikainen, and D. Harwood, "Performance evaluation of texture measures with classification based on Kullback discrimination of distributions," in *Proceedings of 12th international conference on pattern recognition*, IEEE, 1994, pp. 582–585.
[50] T. Ojala, M. Pietikäinen, and D. Harwood, "A comparative study of texture measures with classification based on featured distributions," *Pattern Recognit*, vol. 29, no. 1, pp. 51–59, 1996.
[51] G.-B. Huang, H. Zhou, X. Ding, and R. Zhang, "Extreme learning machine for regression and multiclass classification," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 42, no. 2, pp. 513–529, 2011.
[52] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: theory and applications," *Neurocomputing*, vol. 70, no. 1–3, pp. 489–501, 2006.
[53] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: a new learning scheme of feedforward neural networks," in *2004 IEEE international joint conference on neural networks (IEEE Cat. No. 04CH37541)*, Ieee, 2004, pp. 985–990.
[54] V. Vapnik, *The nature of statistical learning theory*. Springer science & business media, 2013.
[55] F. H. Garabaghi, R. Benzer, S. Benzer, and A. Ç. Günal, "Effect of polynomial, radial basis, and Pearson VII function kernels in support vector machine algorithm for classification of crayfish," *Ecol Inform*, vol. 72, p. 101911, Dec. 2022, doi: 10.1016/J.ECOINF.2022.101911.
[56] S. Arya, D. M. Mount, N. S. Netanyahu, R. Silverman, and A. Y. Wu, "An optimal algorithm for approximate nearest neighbor searching fixed dimensions," *Journal of the ACM (JACM)*, vol. 45, no. 6, pp. 891–923, 1998.
[57] A.-M. Šimundić, "Measures of diagnostic accuracy: basic definitions," *EJIFCC*, vol. 19, no. 4, p. 203, 2009.

Acknowledgement

This study was funded by the Scientific Research Projects Coordination Unit of Tekirdağ Namık Kemal University. Project number: NKUBAP.06.GA.21.317

Data availability
Data sharing is not applicable to this article as no datasets were generated or analyzed during the current study.

Conflict of Interest Notice
The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Ethical Approval and Informed Consent
It is declared that during the preparation process of this study, scientific and ethical principles were followed, and all the studies benefited from are stated in the bibliography.

Plagiarism Statement
This article has been scanned by iThenticate ™.

Sakarya University Journal of Computer and Information Sciences
http://saucis.sakarya.edu.tr/

DOI: 10.35377/saucis...1309103 RESEARCH ARTICLE

Price Prediction Using Web Scraping and Machine Learning Algorithms in the Used Car Market

Seda Yılmaz [1], İhsan Hakan Selvi [1]

[1] Information Systems Engineering Department, Institute of Natural Sciences, Sakarya University, Sakarya, Türkiye.

Corresponding author:
Seda YILMAZ, Information Systems Engineering Department, Institute of Natural Sciences, Sakarya University, Sakarya, Türkiye
E-mail address:
seda.yilmaz11@ogr.sakarya.edu.tr

Submitted: 02 June 2023
Revision Requested: 17 August 2023
Last Revision Received: 25 August 2023
Accepted: 28 August 2023
Published Online: 30 August 2023

Citation: Yılmaz S. and Selvi İH.. (2023). Price Prediction Using Web Scraping and Machine Learning Algorithms in the Used Car Market. *Sakarya University Journal of Computer and Information Sciences.* 6 (2) https://doi.org/10.35377/saucis...1309103

ABSTRACT

The development of technology increases data traffic and data size day by day. Therefore, it has become very important to collect and interpret data. This study, it is aimed to analyze the car sales data collected using web scraping techniques by using machine learning algorithms and to create a price estimation model. The data needed for analysis was collected using Selenium and BeautifulSoup and prepared for analysis by applying various data preprocessing steps. Lasso regression and PCA analysis were used for feature selection and size reduction, and the GridSearchCV method was used for hyperparameter tuning. The results were evaluated with machine learning algorithms.

Random Forest, K-Nearest Neighbor, Gradient Boost, AdaBoost, Support Vector and XGBoost regression algorithms were used in the analysis. The obtained analysis results were evaluated together with Mean Square Error (MSE), Root Mean Square Error (RMSE) and Coefficient of Determination (R-square). When the results for data set 1 were examined, the model that gave the best results was XGBoost Regression with 0.973 R^2, 0.026 MSE and 0.161 RMSE values. When the results for data set 2 were examined, the model that gave the best results was K-Nearest Neighbor Regression with 0.978 R^2, 0.021 MSE and 0.145 RMSE values.

Keywords: Web scrapping, machine learning, price prediction

1. Introduction

With the developing technology, the data size and traffic are increasing daily. Therefore, data analysis is very important in many different sectors and functions today. When data analysis is done correctly, it provides significant benefits in almost every field. Many websites contain large amounts of data. Obtaining this information with statistical summaries is important for individuals and businesses [1]. It enables businesses to achieve better results in decision-making. Accordingly, it provides important benefits such as competitive advantage, customer satisfaction, operational efficiency, and better management of risks.

However, data that needs to be collected and interpreted correctly only confuses. Therefore, data collection is as important as analysis. Most of the time, data cannot be copied directly. So, skills are needed to take as little time and effort as possible into analyzing data. The information is placed so that it is not easy to download; even if it is easy to download, it will be difficult to process. Web scraping is included in data science at this point [2]. Of course, data can also be collected manually from web pages. However, this process takes much time, and complex site structures may not allow information to be copied from sites so easily. For these reasons, the web scraping method has emerged [3].

With web scraping, collecting large amounts of data is automated, and the data is made more useful to the user. But in most cases, web scraping is a complex process. Websites come in different shapes and formats. This is why web scrapers vary in

functionality and features. Many websites now provide users with an API (Application Programming Interface) to access structured data stores that can be downloaded and used. There is no point in scraping if the website provides access to the API. One of the most important current uses of web scraping is for businesses to track the pricing activities of their competitors. Pricing can be created across an entire site in relatively short timescales and with minimal manual effort [4].

Accurate car price estimation requires expert knowledge because price often depends on distinguishing features and factors. To accurately estimate the price of a car, it has been shown that the knowledge of experts in this field is required, as well as many dimensions that must differ from the technical characteristics of the car, such as horsepower, engine capacity or transmission [5].

Asghar et al. [6] suggest using different ways to estimate sales prices based on the value of the tools used in marketing. The applications they recommend support both the buyer and the seller in buying and selling second-hand vehicles, can predict the best prices for themselves, and make a good personal and commercial determination. The proposed modeling performances reflect that their work includes effective and efficient methods and strategies.

Pandey et al. [7], in their study, say that machine learning algorithms are good solutions that can be used to solve the problems encountered in such areas. According to Pandey et al., machine learning algorithms and techniques can offer a good solution to such problems as the issue of used car price estimation.

Chen et al. [8] have put forward a comprehensive study by drawing attention to big data and usage areas in their studies. They emphasize in their writings that as the size of the data used increases, its importance also increases. Their work demonstrates how multiple linear regression and random forest techniques work with different complexities in the modeling part. As a result, it is argued in the study that the only purpose is to realize the most suitable model while dealing with a used vehicle price estimation.

Identifying needs is important in web scraping. First, which data will be collected from which website should be determined. Different tools and libraries are available for web scraping. Libraries like BeautifulSoup and Scrapy in Python are popular and frequently used methods for web scraping. By doing research, the most suitable method for web scraping can be determined.

Appropriate methods should be determined for the analysis of the obtained data set. These methods vary depending on the dataset's characteristics and the intended output. Machine learning, statistical analysis, data mining or other analytical methods can be used.

2. Materials and Methods

This study aimed to collect data from the website using web scraping techniques and to analyze the collected data with machine learning algorithms and make price estimations. The data used in the study were collected with the help of codes written in Python programming language from a website where used vehicles are sold and saved in a database.

Python is one of the most common programming languages in the data domain. It is preferred because it is easy to understand. Also, this programming language is flexible and scalable. Due to the many advantages of using this programming language, it has also found its way to writing codes and programs for web scraping. Web scraping is a relatively modern trend for developers, but the Python programming language is well suited for performing web scraping tasks. Python has a huge library of tools for developing algorithms to collect data using web scraping.

PyCharm is an IDE for Python with the toolset for effective Python programming. IDE (Integrated development environment) is an environment that helps users write code faster and more efficiently. It allows editing, compiling, or interpreting text, debugging and more. The IDE allows the development speed to be increased. Jupyter Notebook, on the other hand, is a server-client application where codes can be run, notes can be taken, and desired edits can be made in notepad format on a web browser. Thanks to its block logic, it is highly preferred because it allows observing the outputs by making changes in the desired parts of the code.

Selenium and BeautifulSoup were used in this study's web scraping phase for data collection. Selenium library is generally used to automate web applications for testing purposes. However, it is for more than this. It also allows data collection via the browser. The BeautifulSoup library is a library created for manipulating HTML files. It allows us to obtain the desired parts by parsing the HTML codes in the relevant source. Numpy, Pandas, Scikit-learn, and Mathplotlib libraries were mainly used in the preprocessing and analysis stages of the data.

After the data obtained by the web scraping method were converted into the appropriate format, they were written on the database in the form of tables. During these processes, pymysql and sqlalchemy libraries were used. With these libraries, the relevant table in the database can be accessed with Python and various select, insert, update, and delete operations can be performed.

The study was completed by following the steps below.

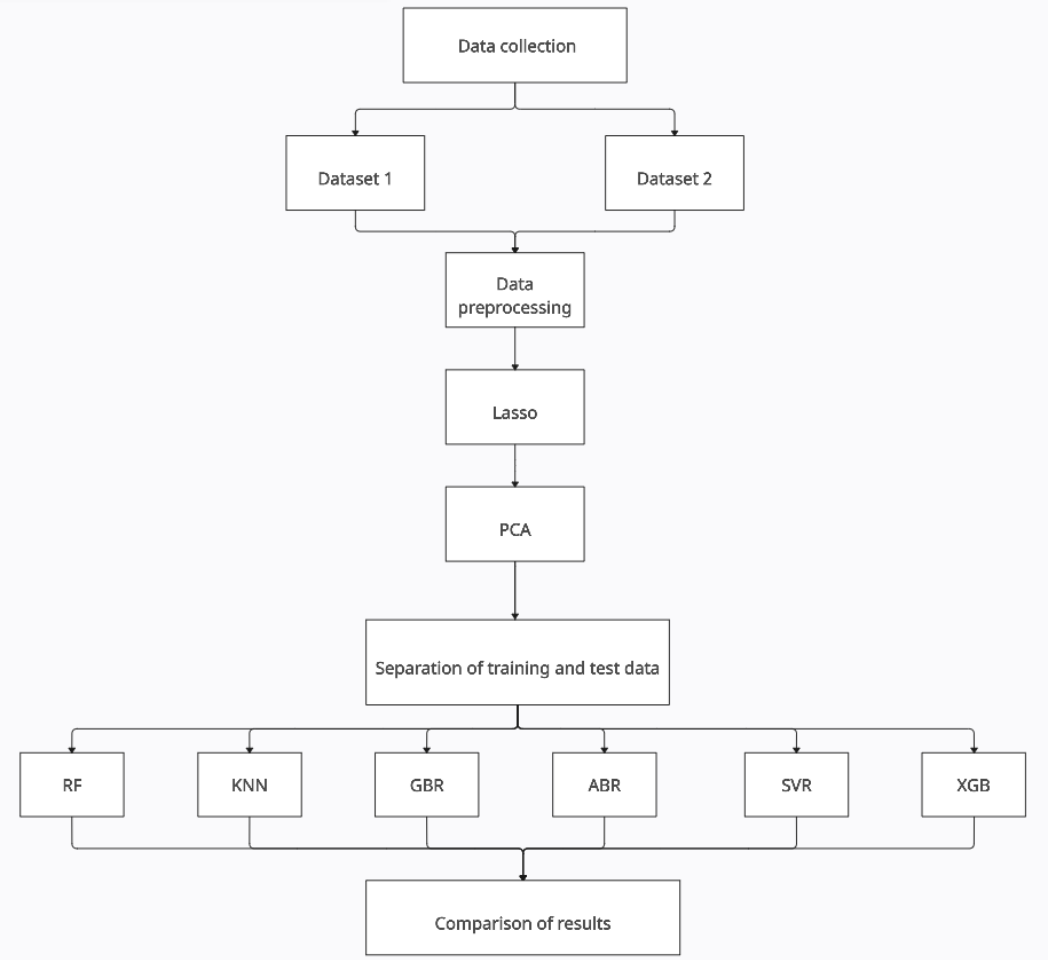


Figure 1 Prediction Method

2.1 Data collection

First, it was planned to collect follow-up data for car price estimation from a used car sales website: price, brand, model, series, type, from information, fuel, gear, engine power, and engine capacity. In addition, the car damage status, an important criterion in the used car market, was also wanted to be analyzed, and damage information was obtained from the template showing the car damage status on the site. The representative image is shown in Figure 2.

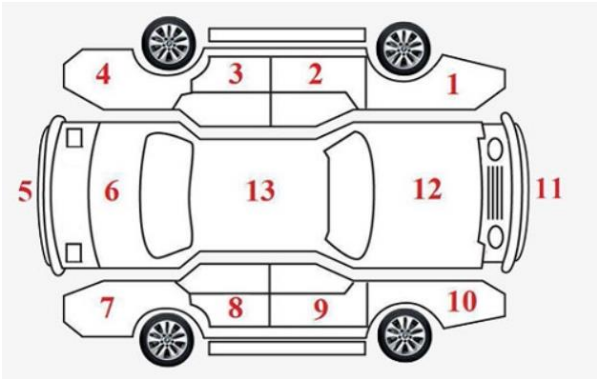


Figure 2 Representative image [9]

This information was added to the dataset in the following order: right rear fender, rear hood, left rear fender, right rear door, right front door, roof, left rear door, left front door, right front fender, engine hood, left front fender, front bumper, rear bumper. While these fields are being filled, 0 values are assigned if the relevant part of the car is original, 1 if it is painted, 2 if it is changed, and 3 if it is not specified.

In the web scraping process, the URLs of the requested products were first retrieved from the target website via Selenium. The source codes of the pages whose URLs were taken were collected, and the desired parts were parsed using BeautifulSoup.

The data collected with Numpy and Pandas were manipulated, converted into the appropriate format, and tabulated. The collected data was written to the database and used Pymysql and Sqlalchmy for database connection.

Within the scope of the study, two data sets with different data numbers were obtained to make comparisons during the analysis. Both datasets consist of 23 features. While collecting the data, the dates between 10.02.2023 and 31.03.2023 were taken as a basis. Dataset 1 contains 5557 rows of data, while Dataset 2 contains 11688.

2.2 Data preprocessing

The collected data were analyzed, and initially, some null fields in the columns reflecting the damage status were filled with the default (unspecified=3) value. Then, categorical data were converted into numerical data using the encoder method to be ready for analysis.

The price variable was set as the target variable. Lasso regression was used to determine which of the remaining 22 features would be used in the analysis. Thanks to Lasso, the coefficient of unimportant variables was reduced to zero.

After applying Lasso, 11 features were selected in the Dataset 1 result, and 9 were selected in the Dataset 2 result.

Dataset features before applying Lasso: brand, model, series, type, from who information, fuel, gear, engine power, engine capacity, right rear fender, rear hood, left rear fender, right rear door, right front door, roof, left rear door, left front door, right front fender, engine hood, left front fender, front bumper, and rear bumper.

After applying Lasso, dataset 1 features brand, model, series, from who information, fuel, gear, engine power, engine capacity, rear hood, roof, and front bumper.

Dataset 2 features after applying Lasso: brand, model, type, from who information, fuel, gear, engine power, roof, and rear bumper.

Lasso regression is a technique used in feature selection and model parameter estimation. Unlike traditional regression analysis, it encourages model parameters to approach zero. Thanks to this feature, it can reduce overfitting problems and make the model simpler and interpretable.

Afterward, PCA analysis for size reduction was applied, and the variables included were determined. As a result of the PCA analysis, the number of features was reduced to 7 for both data sets.

Principal Component Analysis (PCA) is a statistical technique for dimension reduction in multivariate datasets. PCA is used to analyze the relationships between the variables in the data set and to create new variables that best explain these relationships.

Table 1 Dataset sizes after Lasso and PCA

	Lasso		PCA	
	Number of Samples	Number of Features	Number of Samples	Number of Features
Dataset 1	5557	11	5557	7
Dataset 2	11688	9	11688	7

2.3 Separation of training and test data

Training and test data were separated for use in machine learning algorithms. K-Fold cross-validation was used for this process. Cross-validation is a healthier method than the train-test split approach because it allows both testing and training to be performed on all data; for example, when the data is separated as 20% test and 80% training with a train-test split, some deviations and errors may occur because it is not known how the data is distributed here.

In the K-Fold cross-validation method, the dataset is divided into k different subsets. It is ensured that each piece is used as both training and test data. In this way, the errors that may occur due to its distribution are minimized. In this study, analyses were carried out by taking the k value of 10.

2.4 Machine learning algorithms

In the study, 5 different machine learning algorithms were used. Dataset 1, with a low amount of data and Dataset 2, with a higher amount of data, were run with each of these algorithms. GridSearchCV was used to determine the best parameters in the algorithms.

Random forest is one of the best-performing and widely used machine learning algorithms. Random forest regression generates multiple decision trees and combines the outputs of all these decision trees to arrive at a single result. It was first

introduced in 1984 [11]. Decision nodes and leaf nodes form decision trees. With the test function, decision nodes evaluate the samples and forward them to the relevant branches according to the results. RFR is a bagging community-based model that combines bagging benefits. Thus, it improves forecasting performance [12].

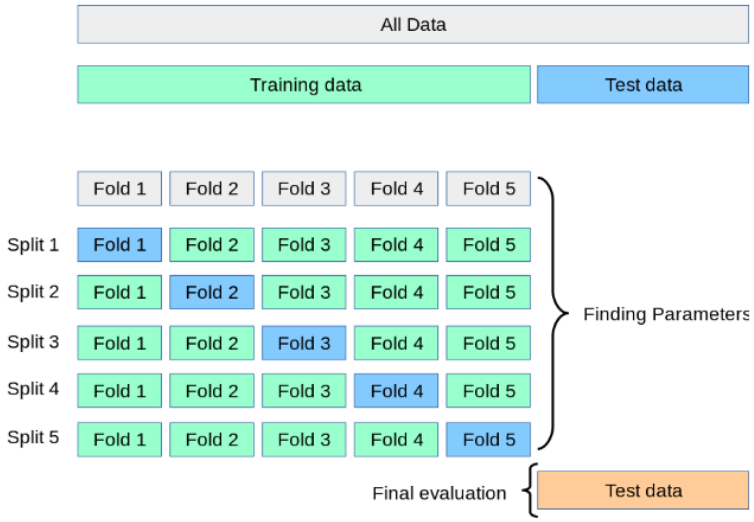


Figure 3 K-Fold working principle [10]

The K-Nearest Neighbors (KNN) algorithm is a machine learning algorithm for classification and regression problems. It is one of the supervised learning methods. It is trained with labeled data and used to classify or predict new samples. The KNN algorithm finds the k nearest neighbors to the sample and estimates by taking the neighbors' average. Other similarity criteria, usually Euclidean, can also be used to identify nearest neighbors [13].

Gradient Boosting Regression (GBR) is a machine learning technique that combines the predictions of multiple weak decision trees to create a strong prediction model. It is a powerful algorithm used for both regression and classification problems. In the gradient boosting method, weak decision trees are trained sequentially, and each next decision tree is trained to correct the errors of the previous decision trees [14]. The algorithm starts with an initial estimate, usually chosen as the mean value of the target variable for regression problems. It builds up a series of models in succession that minimizes the errors of the previous models.

Adaboost regression is an ensemble learning algorithm used in machine learning [15]. AB (Adaptive Boosting) aims to create a strong learner by bringing together weak learners. It is usually implemented using a set of decision trees. It advances by reinforcing relatively weak models that made erroneous predictions in previous steps. While it reduces the weights of the decision tree points with correct predictions, it increases the weights of those with incorrect predictions. This creates a combination of weak models whose weights are adjusted at each step and aims to obtain a strong model with the best predictive ability [16].

Support Vector Regression (SVR) is a version of support vector machines (SVM) adapted to regression problems. Support Vector Regression divides data points with a hyperplane and makes predictions using support vectors on this hyperplane [17]. It aims to minimize the regression error by classifying the data points into hyperplanes as much as possible [18]. Support Vector Regression can work effectively on low-dimensional or high-dimensional datasets. It is especially preferred because it resists outliers and shows good generalization ability in various data distributions.

XGBoost is a gradient-boosting-based learning algorithm. Under the Gradient Boosting framework, it develops extensible parallel classification and regression trees (CARTs) that can quickly and accurately solve data science problems. This approach combines weak learners (usually decision trees) and creates a strong prediction model [19]. XGBoost's high performance, strong predictive ability, scalability, resistance to outliers, feature importance scores, automatic intersection processing, and flexibility stand out.

3. Conclusion and Discussion

Analyzes were carried out using the materials and methods in Chapter 2. As a result of the analyses made with machine learning algorithms, the following outputs were obtained. The table includes R-square scores, MSE, and RMSE values.

Table 2 Results of analysis with Dataset 1 and Dataset 2

		RFR	KNN	GBR	ABR	SVR	XGB
Dataset 1	R^2	0.889	0.854	0.970	0.676	0.811	**0.973**
	MSE	0.110	0.145	0.029	0.323	0.188	**0.026**
	RMSE	0.333	0.382	0.172	0.568	0.434	**0.161**
Dataset 2	R^2	0.960	**0.978**	0.967	0.658	0.820	0.973
	MSE	0.039	**0.021**	0.032	0.341	0.179	0.026
	RMSE	0.197	**0.145**	0.179	0.584	0.423	0.164

When the results were examined, it was seen that increasing the amount of data for RF, KNN, and SVR models affected the results positively; the scores increased, and the errors decreased. For other models, it is seen that the performance decreases when the amount of data increases. The best results are shown in bold in Table 2. A learning curve graph was used to show the performance in the analysis results. A learning curve is a graph that shows how the performance of a model changes with the number of training samples. As the number of training samples increases, the model's performance on training and validation sets is examined.

According to Table 2 and graphics, it was determined that the Adaboost regression model gave worse results in both data sets.

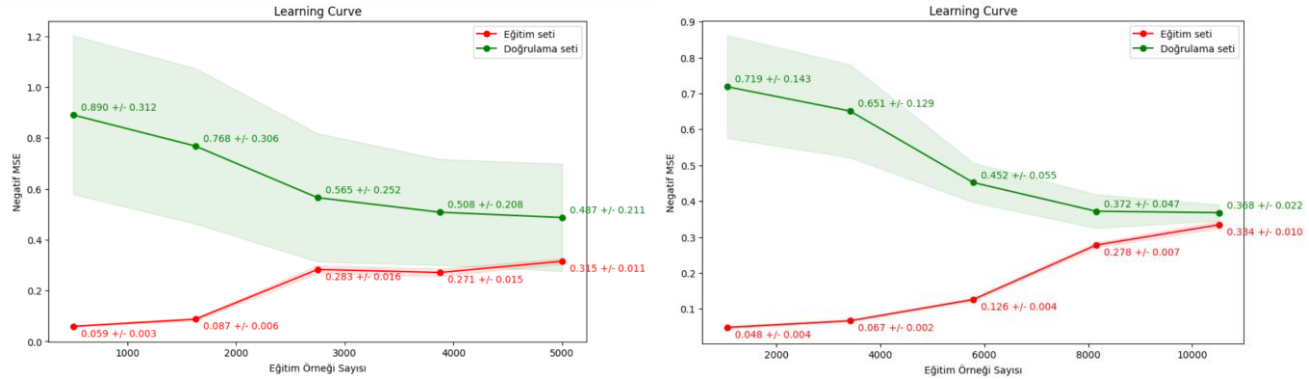


Figure 4 ABR Learning Curve graph (Dataset 1 and Dataset 2)

In the graphs, it is seen that the errors in the training set and test set are high. The high standard deviation in the test set errors indicates that the model's predictions are quite different from each other, and the reliability of the predictions is low. This indicates that the model may perform worse on some data points, and its predictions must be evaluated more carefully. As a result, error values and high standard deviation, according to the graph, show that the model's generalization ability could be stronger, and there may be an overfitting problem. Increasing the data size does not improve the results.

When the results of the other models are examined, it is seen that the results of the GBR and XGB models based on Gradient Boosting for Dataset 1 containing 5557 rows of data are better than the remaining models. When the two models are compared, it has been determined that the XGBoost model gives better results.

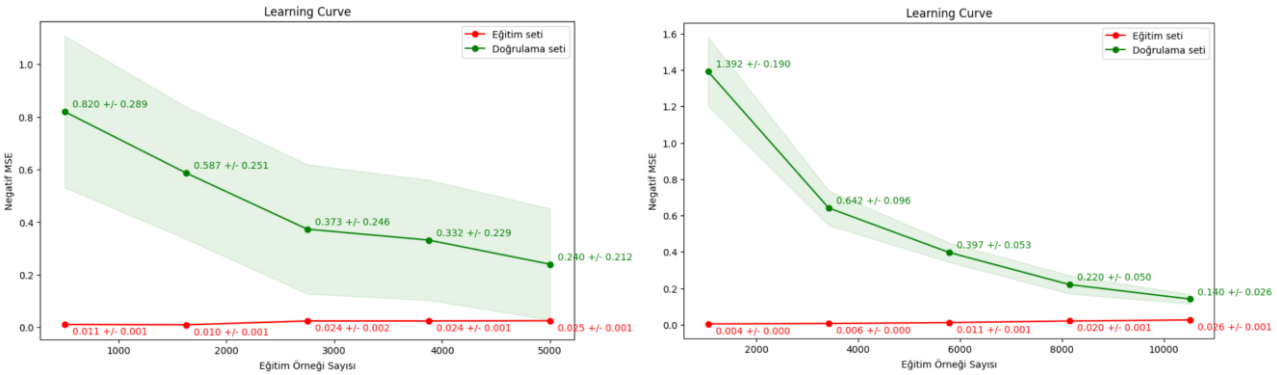


Figure 5 XGB Learning Curve graph (Dataset 1 and Dataset 2)

The training set has a low initial error value and standard deviation. This indicates that the model has learned the training data well and has a low error. However, the validation set initially appears to have a higher error value and standard deviation. As the data size increases, the error value in the validation set and the standard deviation decreases. A decrease in the error

value indicates that the model's predictions are more accurate and better fitted to the data. In summary, while the model performs well on the training set, it initially performs poorly on the validation set. However, as the data size increases, the model's generalization ability improves, and the error in the validation set decreases.

When the results for Dataset 2, which contains 11688 rows of data, are examined, it is seen that the KNN and XGB models have the best results. When the two models are compared, the results of KNN were found to be better than XGB.

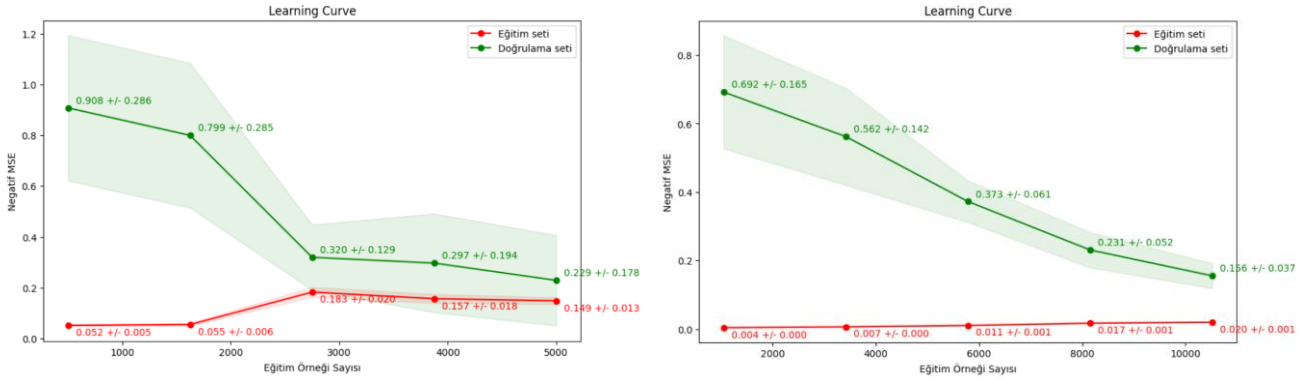


Figure 6 KNN Learning Curve graph (Dataset 1 and Dataset 2)

The error values obtained in the training set are quite low, indicating that the model performs well on the training data. The standard deviation values are also quite low, indicating that the model has low sensitivity to the training data and that the predictions are consistent. The error values obtained in the test set are higher than those in the training set. The standard deviation value is higher, indicating that the model's performance on the test data is more variable and the predictions have a wider error range. As a result, it is seen that the overall performance of the model improves with the increase in the amount of data. The error and standard deviation decrease on the validation set indicates that the model generalizes better with more data.

The learning curve graphics of other models are as follows. When the graphs are examined, it is seen that increasing the amount of data in RFR and SVR models improves the results. In the GBR model, on the other hand, it is seen that the performance has decreased slightly. However, for all models, it is seen that the performances in the validation set increase and the standard deviations decrease.

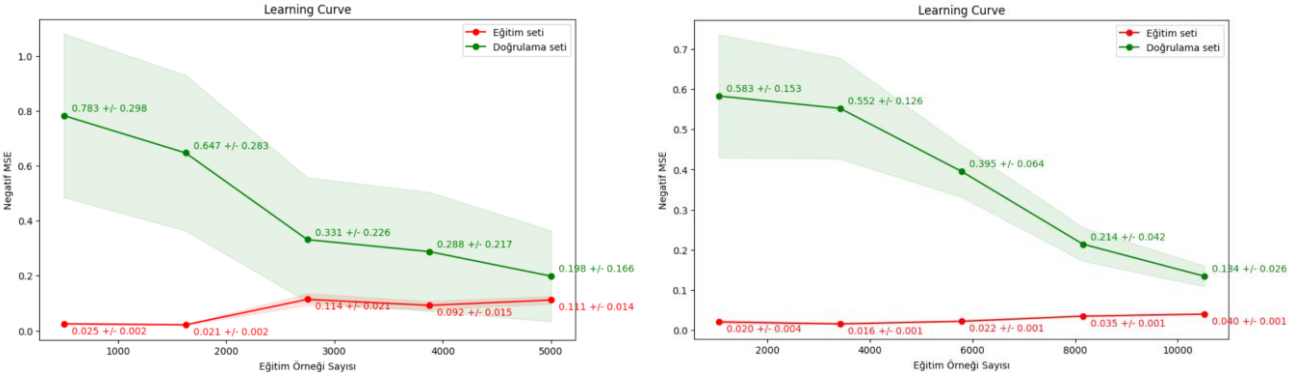


Figure 7 RFR Learning Curve graph (Dataset 1 and Dataset 2)

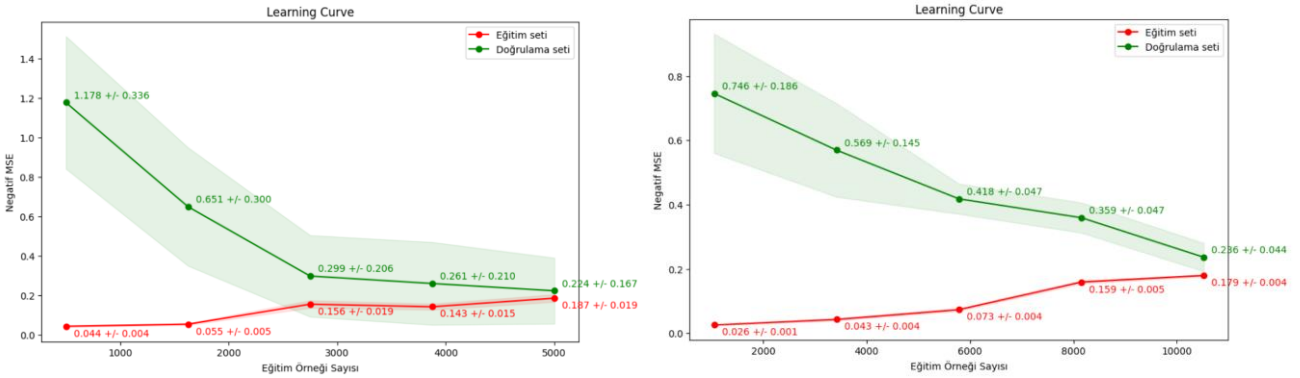


Figure 8 SVR Learning Curve graph (Dataset 1 and Dataset 2)

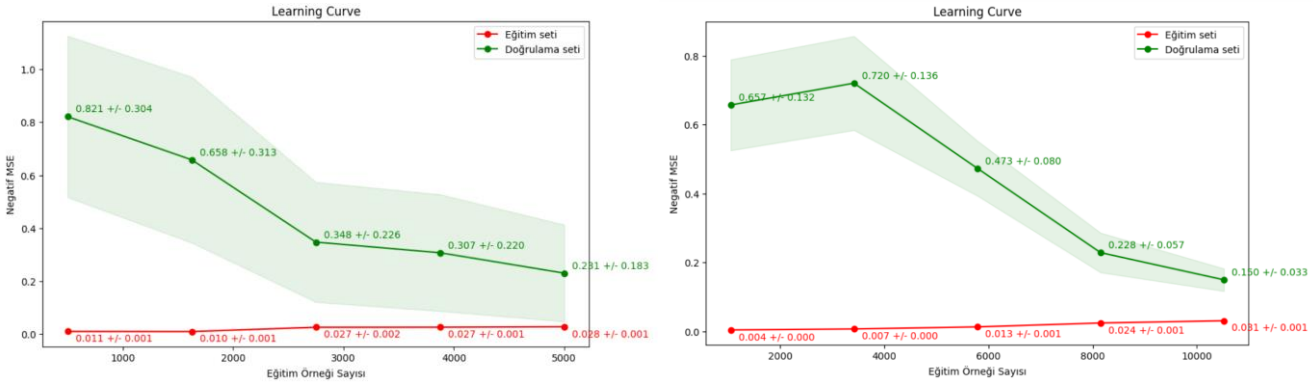


Figure 9 GBR Learning Curve graph (Dataset 1 and Dataset 2)

As a result of the study, the data collected from the website with web scraping methods were analyzed with machine learning algorithms, and the algorithms that gave the best results for different data sets in size for price estimation were determined. It has been seen that increasing the amount of data for some models has a positive effect on the results, while for some models, it has a negative effect.

Based on the study, it can be deduced that similar data from different sources can be collected by web scraping methods, the data set can be expanded, and performance predictions can be made by selecting the appropriate model after the analysis. If developed, this study can continuously extract data from websites by performing web scraping with various methods, and data can be collected, the collected data can be analyzed, and the forecast results can be shared.

References

[1] Milev, P., Conceptual Approach for Development of Web Scraping Application for Tracking Information. Economic Alternatives, 475-485, 2017.
[2] Khder, M., Web Scraping or Web Crawling: State of Art, Techniques, 73 Approaches and Application. International Journal of Advances in Soft Computing and its Applications, 2021.
[3] Banerjee, R., Website Scraping, Happiest Minds Technologies, 2014.
[4] Haddaway, N., The use of web-scraping software in searching for grey literature. Grey Journal, 11(3):186-190, 2015.
[5] Gegic, E.; Isakovic, B.; Keco, D.; Masetic, Z.; Kevric, J. Car price prediction using machine learning techniques. TEM J. 2019, 8, 113.
[6] Asghar, M., Mehmood, K., Yasin, S., & Khan, Z. M., Used Cars Price Prediction using Machine Learning with Optimal Features. Pakistan Journal of Engineering and Technology, 4(2), 113-119, 2021.
[7] Pandey, A., Rastogi, V., & Singh, S., Car's selling price prediction using random forest machine learning algorithm. In 5th International Conference on Next Generation Computing Technologies, 2020.
[8] Chen, K.-P., Liang, T.-P., Yin, S.-Y., Chang, T., Liu, Y.-C., & Yu, Y.-T., How serious is shill bidding in online auctions? evidence from eBay motors. work, 1–51, 2020.
[9] Yolcu360, Available: https://yolcu360.com/blog/oto-ekspertiz-raporunda-ne-yazar. [Accessed: 03-May-2023].
[10] Scikit-learn, Available: https://scikit-learn.org/stable/modules/cross_validation.html. [Accessed: 04-May-2023].
[11] Breiman, L., Random Forests. Machine Learning, 45(1), 5-32, 2001.
[12] Breiman, L., Bagging Predictors. Machine Learning, 24(2), 123-140, 1996.
[13] Altman, N. S. An introduction to kernel and nearest-neighbor nonparametric regression, The American Statistician, vol.46, s. 175–185, 1992.
[14] Friedman, J. H., Greedy function approximation: A gradient boosting machine. Annals of Statistics, 29(5), 1189-1232, 2001.
[15] Freund, Y. and Schapire, R. E. "Experiments with a new boosting algorithm", Icml, 96, 148-156, 1996.
[16] Schapire, R. E., Explaining adaboost. In Empirical Inference, pp. 37–52, Berlin Heidelberg., 2013.
[17] Vapnik V., The Nature of Statistical Learning Theory, 1995.
[18] Awad M. and Khanna R., Efficient Learning Machines, Apress, 2015.
[19] Chen, T., Guestrin, C., XGBoost: A scalable tree boosting system. Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 13-17-Augu, 785–794, 2016.

Conflict of Interest Notice
The authors declare that there is no conflict of interest regarding the publication of this paper.

Ethical Approval and Informed Consent
It is declared that during the preparation process of this study, scientific and ethical principles were followed, and all the studies benefited from are stated in the bibliography.

Availability of data and material
Not applicable

Plagiarism Statement
This article has been scanned by iThenticate ™.

Sakarya University Journal of Computer and Information Sciences
http://saucis.sakarya.edu.tr/

DOI: 10.35377/saucis…1339150 RESEARCH ARTICLE

Sequential and Correlated Image Hash Code Generation with Deep Reinforcement Learning

Can Yüzkollar [1]

[1]Computer Engineering Department, Faculty of Computer and Information Sciences, Sakarya University, Sakarya, Türkiye

Corresponding author:
Can Yüzkollar, Computer Engineering Department, Faculty of Computer and Information Sciences, Sakarya University, Sakarya, Türkiye
E-mail address:
can@sakarya.edu.tr

Submitted: 07 August 2023
Revision Requested: 28 August 2023
Last Revision Received: 29 August 2023
Accepted: 31 August 2023
Published Online: 31 August 2023

Citation: Yuzkollar C. (2023). Sequential and Correlated Image Hash Code Generation with Deep Reinforcement Learning. *Sakarya University Journal of Computer and Information Sciences.* 6 (2) https://doi.org/10.35377/saucis...1339150

ABSTRACT

Image hashing is an algorithm used to represent an image with a unique value. Hashing methods, which are generally developed to search for similar examples of an image, have gained a new dimension with the use of deep network structures and better results have started to be obtained with the methods. The developed deep network models generally consider hash functions independently and do not consider the correlation between them. In addition, most of the existing data-dependent hashing methods use pairwise/triplet similarity metrics that capture data relationships from a local perspective. In this study, the Central similarity metric, which can achieve better results, is adapted to the deep reinforcement learning method with sequential learning strategy, and successful results are obtained in learning binary hash codes. By considering the errors of previous hash functions in the deep reinforcement learning strategy, a new model is presented that performs interrelated and central similarity-based learning.

Keywords: Image Hashing, Reinforcement Learning, Deep Reinforcement Learning

1. Introduction

Today, millions of images are transferred to the web daily due to the increase in internet speeds and developments in web technologies. Due to this increase in the number of images on the web and the difficulties in accessing targeted images, many studies have been conducted to improve image search processes [1-8].

In visual search processes, a unique code representing an image and hash codes are usually generated using an algorithm and operations are performed on the codes. Since similar codes are generated for similar images, the similarity of the images can be easily detected. In traditional cryptographic algorithms, changes that do not perceptually distort the image (such as changing some pixel values, resizing the image, etc.) will generate different codes due to the nature of these algorithms. Therefore, the use of such algorithms is not suitable for image similarity detection. Some hashing methods use traditional hand-crafted feature extraction and deep hashing methods use deep network structures to generate hash codes. In hashing approaches, converting the input images into compact binary codes provides time and memory utilization advantages in image search processes.

Image hashing algorithms aim to represent images with short hash codes. They are frequently used for detecting similar images or content, checking for copyright violations, performing fast and efficient searches in large media libraries, creating summaries of sensitive image data to hide the original image, providing biometric recognition such as face recognition, etc. These algorithms play a crucial role in areas ranging from content similarity detection to security measures.

Traditional methods using handcrafted features could be more efficient in expressing visual content, which has a negative impact on performance in the visual search process. The high performance of deep learning methods in image classification

and object recognition has led to using these methods in hash code generation. Some deep hashing methods have been developed that utilize the feature extraction power of deep networks. These hashing methods treat all hash functions independently and do not consider the correlations of different hash functions. These studies usually utilize pairwise or triplet data similarity [9-11]. Accordingly, loss calculations are performed. It has been stated in some studies that the sequential handling of hash codes will provide error correction capability in the learning process. Another study emphasized that the central similarity metric gives more successful results than triplet and pairwise metrics [12].

Deep reinforcement learning has achieved human-like performance in various studies and has been addressed in many fields. A standard reinforcement learning model consists of an environment and an agent interacting. [13] The agent evaluates the information coming from the environment and aims to choose an action that maximizes the sum of future rewards. In hash methods, if we consider the function to obtain the hash code as the agent, the actions generated can be expressed as 0 or 1. Rewards can be obtained by using a loss function to evaluate the quality of the generated hash code information. Maximization of these rewards can be achieved through reinforcement learning. Determining the reward function in a reinforcement learning problem is one of the most important problems. In a study conducted in this direction, the reward function was considered a triplet loss, which was found to give successful results [14]. As stated in [12], the central similarity loss function has a more discriminative aspect than triplet loss and pairwise loss. Our study proposes a deep reinforcement learning method that provides the sequential decision-making process of hash functions and the central similarity reward function.

This paper proposes a deep reinforcement learning model with a sequential learning strategy that considers global data distribution, data relationships and global similarity. The proposed model includes a feature extraction network and a policy network. In the policy network, the RNN structure is used to obtain the probabilities of the binary transformation of images into binary codes sequentially. In the proposed Sequential and Correlated Image Hash Code Generation with Deep Reinforcement Learning (SCIHCGRL) method, the central similarity is suggested as a reward function instead of pairwise or triplet functions used in previous studies.

2. Related Works

Currently used hashing methods are classified into two main groups: data-dependent and data-independent. Data-dependent hashing methods can also be categorized into supervised, unsupervised, and reinforcement. Data-independent methods are based on the nearest-neighbor search method. Peng and Indyk presented Locality-Sensitive Hashing (LSH) [14, 15], where the basic idea is to summarize data and query points so that the probability of collision is much higher for points close to each other than for points far apart. The LSH method requires long codes or multiple hash tables. Lv [16] and Raginsky [17] have proposed several approaches to overcome these problems.

Data-dependent methods are generally divided into two main groups: supervised methods where data labels are used in the learning process and unsupervised methods where data labels are not considered in the learning process. Reinforcement learning methods, in which data labels are used only for evaluation purposes and the learning strategy is not supervised, can also be considered a separate class.

Studies on unsupervised methods are as follows: Spectral hashing (SH)[18], iterative quantization (ITQ)[19], topology preserving hashing (TPH)[30], locally linear hashing (LLH)[20], discrete graph hashing (DGH)[21], scalable graph hashing (SGH)[22] ordinal embedding hashing (OEH)[23], semi-paired discrete hashing (SPDH)[24], similarity adaptive discrete deep hashing (SADH)[25], multiview discrete hashing (MDH)[26], isotropic hashing (IsoH)[27]. In recent years, supervised methods have been used instead of unsupervised methods with more successful results.

To strengthen feature extraction and feature learning in supervised methods, deep learning structures have started to be used and have improved classification performance by characterizing the non-linear features of the data more effectively. Based on the convolutional neural network structure CNNH [28], a two-stage method is preferred in this study. In the first stage, approximate hash code learning was performed by preserving pairwise semantic information and in the second stage, deep hash network training was performed using the learned hash codes as labels. The independence of the deep hash network from the first stage limits the derivation of better hash codes. To overcome this problem, a network within a network (NINH) hash function and image features are presented as simultaneous inputs [29]. Furthermore, various studies use Deep Hashing methods [9,30-34].

Yuan et al. presented a new general similarity metric in [12], which offers a different approach to methods that aim to learn hash functions based on pairwise and triplet data relations. It has been stated that the use of pairwise and triplet-based functions uses only partial relationships between data pairs and may damage the identifiability of the generated hash codes, may have low efficiency in unbalanced data, low efficiency in generating profile similarity among the entire data set and high temporal complexity. It was observed that the proposed central similarity method achieved more successful results.

In this study, a deep reinforcement hashing method using the Central similarity metric is developed.

2. Proposed Method

Most image-hashing studies in the literature have been performed with supervised learning methods. In this paper, we implement a new methodology for generating image hashes based on the methodology described in [14]. This work generates image hashes with several deep networks and deep reinforcement learning.

In our proposed deep reinforcement learning-based approach, RNN network structures are chosen as an agent modeling the hash functions so that the previous hash functions' errors are considered during sequentially transferring images to binary codes. In addition, the Central Similarity metric, which can achieve more successful results, was adapted to the model and more effective results were obtained in learning binary hash codes.

Reinforcement learning involves a challenge encountered by an agent that needs to acquire behavior by engaging in trial-and-error interactions within a dynamic environment [13,35]. Within the conventional framework of deep reinforcement learning, the agent takes the current environmental state as input and produces an output action. This action influences the state of the environment, and the environment communicates with the agent using a scalar reinforcement signal referred to as a reward, which signifies the actions' effectiveness. The reinforcement learning objective is to train the agent to select actions that maximize the cumulative reward. Significant advancements have been made in various domains through deep reinforcement learning techniques.

The deep reinforcement learning we used in this study is shown in Figure 1. In this network, the agent performs actions with stochastic policies and tries to maximize the amount of future reward with a reward-punishment system for each action. In reinforcement deep learning, the policy, state, action, and reward must be determined. In this study, the feature layer output of a CNN network and the historical information of an RNN network are used for the state. For the policy and actions to be implemented, an RNN network and a linear deep network layer are nested. An internal reward system is prepared for each action taken.

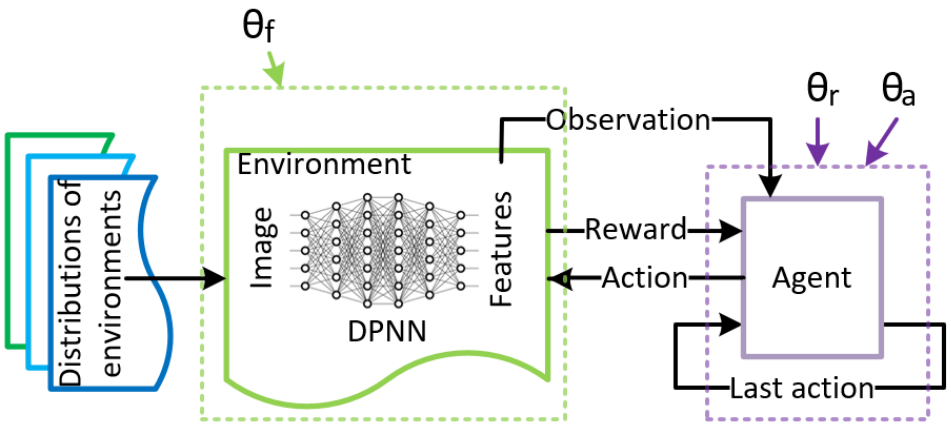


Figure 1 The deep reinforcement learning architecture for image hashing.

States:
A state (s), is a binary given in the output of Figure 2 and is defined as $s = (fv, hv)$. where fv is the image feature vector and hv is the background information of the hash codes. In a pre-trained CNN network, the feature vectors of the original images are extracted. The historical information is obtained from the policy layer.

The first 18 layers of the VGG-19 network were used as is to generate the fv in the state. The last fully connected layer output of the VGG-19 network is used as image features. The structure of this VGG-19 network is shown in Figure 3.

Actions:

There are two possible actions in a hashing problem. These actions are a=1 bit or a=0 bit. Therefore, the probability sum of the possible actions will be 1. This study, -1 is used instead of the 0-action resulting from the policy layer. The agent used generates the probability of action between -1 and 1 as the output of the policy layer.

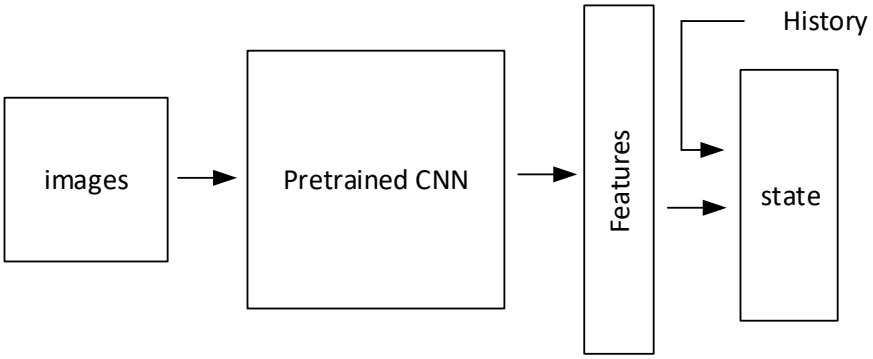


Figure 2 Generation State

$$P(a|s,\theta) = \begin{cases} 1 - policy(s,\theta) & a = -1 \\ policy(s,\theta) & a = 1 \end{cases} \quad (1)$$

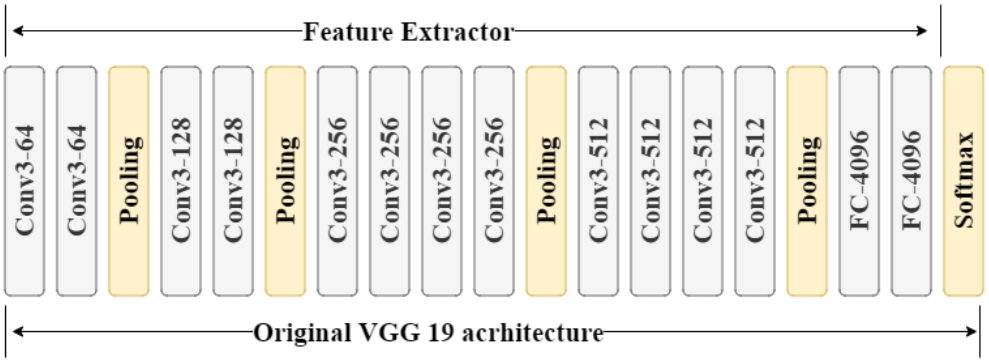


Figure 3 VGG – 19 Network Architecture

The probability distribution of the policy layer is given in Equation 1. $policy(s,\theta)$ denotes the output of the policy layer, s,θ denote the feature vector and the weights of the deep network respectively.

In classical deep learning-based hashing methods, the output layer's output is the number of hash lengths to be obtained. This work generates a sequential action using the past hash information. A policy layer can only output one bit of action. However, a one-bit hash code does not provide enough information to correct previous action errors.

In other words, generating a hash sequence where each bit is generated independently or generating it as a group independently reduces the correlation between the hash bits. Therefore, the previous action errors will be eliminated if the correlation between hash bits is utilized to generate hash bits. The RNN layer in our agent provides this. Instead of a one-bit hash code in our work, we used a sequential action with reduced correlation between pixels.

$$A_t^i = \{a_1, a_2, \dots, a_k\} \quad (2)$$

In short, an action in reinforcement learning is a set of actions in our study.

Rewards:

An action set is rewarded with a single reward. Each action set is a set of k-ordered hash functions. A_t^i is the i-th action set at time t. k is the number of actions and corresponds to the desired hash length. From probability theory, the action probability of a set of k ordered actions are defined as $P(A_t^i|s_t^i,\theta)$.

$$P\left(A_t^i|s_t^i,\theta\right) = \prod_{j=1}^{k} P(a_j|\hat{s}_j,\theta) \quad (3)$$

s_t^i is the i-th state at time t, a_j is an element of the action setA_t^i and $\hat{s}_j$ is the current input state entering the policy layer. The details will be discussed in the next section.

A reward function was created for the performance of each action taken. In creating this function, the concept of hash center in [12] is used. For a dataset with m classes, m hash centers are created. We try to force the hash codes generated for classes in similar groups to resemble the hash centers generated for the class.

The hash center for all classes is defined as $HC = \{c_1, c_2, \dots c_m\}$ and c_i is defined as a k-element array of {-1,1}. Equation 4.1 is a reward function for action set i at time t,

$$J_t^i(A_t^i, c_i) = 1 - BCE(A_t^i, c_i) \tag{4.1}$$

$$BCE = L(\theta) = -[A_t^i * Log(c_i) + (1 - A_t^i) * log(1 - c_i)] \tag{4.2}$$

Equation 4.2, BCE is a creation that calculates the Binary Cross Entropy between the target and input probability. In this study input is model output and target is generated with central similarity hash codes.

The pseudo-code for creating the hash center is given in Algorithm 1.

Algorithm 1 Generation of Hash Center [12]

Input : The number of hash centers m, the dimension of the Hamming space (hash codes) K.
Initialization: construct a $K \times K$ Hadamard matrix $H_K = [h_a^i]$ and construct $H_{2K} = [H_K, -H_K]^\top = [h_{2k}^i]$.
for *iteration* i, i=1 *to* m **do**
  if $m \leqslant K$ & $K = 2^n$ **then** // n is any $\mathbb{Z}^+$
    $c_i = h_a^i$;
  end
  else if $K < m \leqslant 2K$ & $K = 2^n$ **then**
    $c_i = h_{2k}^i$;
  else
    c_i[random half position] $= 1$;
    c_i[other half position] $= 0$;
  end
end
Replace all -1 with 0 in these centers;
Output: hash centers: $\mathcal{C} = \{c_1, \dots, c_m\} \subset \{0,1\}^K$.

For single-label datasets, hash centers are created as in Algorithm 1. For multi-label datasets, hash center $\{c_1, c_2, \dots c_m\}$ is created for each class. If an image is associated with two or more labels (classes), the hash center centroid of the corresponding image is used. For example, a dataset contains $\{l_1, l_2, \dots l_m\}$ labels. If an image has a label such as l_1, l_3, l_6, the centroid of these three centers is calculated by using the hash centers c_1, c_3, c_6 of these labels. In the multi-label dataset, the generated hash center centroid is used instead of c_i in the reward function.

Two sequential rewards are used to find correct hash codes. The first one, R_t^i, is the internal cluster reward for the i-th action set and is concerned with hash code accuracy at the cluster level. The second reward R^i is the global reward and is concerned with the accuracy of all hash codes of image i.

$$\begin{aligned} R_t^i &= -J_t^i(A_t^i, c_i) \\ R^i &= -J^i(A^i, c_i) \end{aligned} \tag{5}$$

A. Deep Reinforcement Learning Approach for Image Hashing

The general structure of the study in Figure 4 shows the Agent and Environment, which are the components involved in reinforcement learning. The Environment prepares image features and hash center representations and executes the reward function. The Agent contains a policy layer and executes the implementation of the policy. The state is formed with the image feature obtained from the Environment and the history information formed in the policy layer.

Image features and hash centers are obtained in Environment with VGG19 and Algorithm 1. A single RNN cell is used in the policy network. RNN cells separate themselves from normal neurons in that they have a state and thus can remember information from the past. The RNN converts image features into local states. By applying a linear layer to the local states, policy values between -1 and 1 are obtained. The basic idea behind the application of RNN is to continue the training without ignoring the past information and to generate the actions in this process.

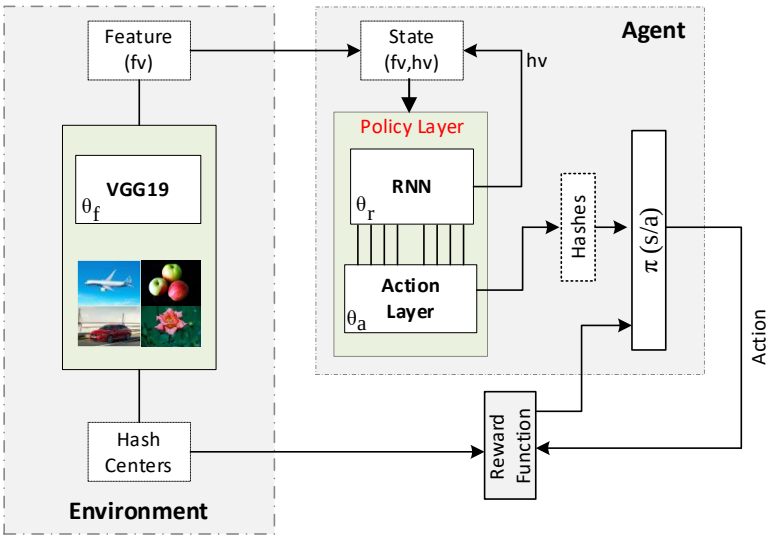


Figure 4 Proposed Method Schema

The schematic of the policy network created for generating actions as a set and reducing the correlation of actions is given in Figure 5. Here, instead of an RNN network, a single RNN cell is used to generate actions with the desired number of hashes. Background knowledge is applied in this part to eliminate previous action errors. The input of the RNN memory cell is the current input image features x_t and the previous RNN cell's h_{t-1} history information obtained from the previous step. The RNN cell generates the desired number of hashes (k) from a single RNN cell using the background information obtained from its inputs and an inner loop. A single action is obtained by applying a linear layer to the h_n obtained in each loop. At the output of the loop, the current RNN cell history h_t and k actions are generated.

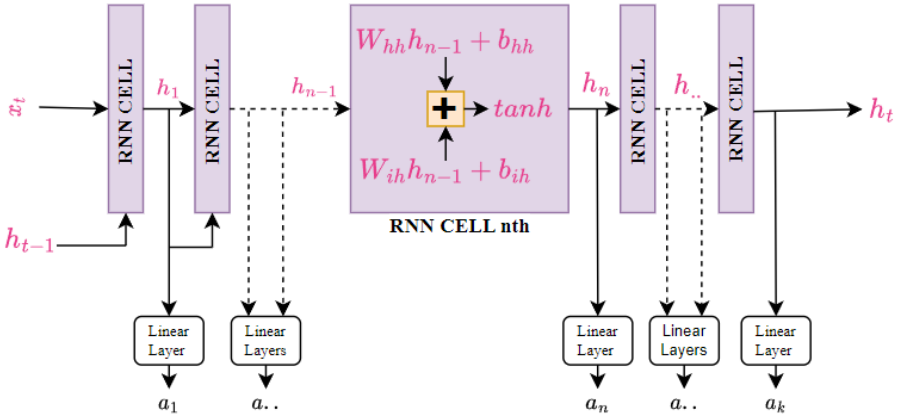


Figure 5 Diagram of the policy network where actions are generated as a cluster.

For (x_t, h_{t-1}) as a state, the RNN layer converts the inputs into outputs with the activation function applied to the cell. Equation 6 is used for the first cycle and Equation 7 for the other cycles.

$$h_n = \tanh\left(W_{ih}x_t + b_{ih} + W_{hh}h_{t-1} + b_{hh}\right) \quad (6)$$

$$h_n = \tanh\left(W_{ih}h_{n-1} + b_{ih} + W_{hh}h_{n-1} + b_{hh}\right) \quad (7)$$

The output of the policy network is the action set and the final history is $h_t = h_n$. In the equations x_t and h_t are the input and hidden vectors, where t is the t th step value. W_{ih} is the weight matrix generated for the input values x_t and h_{n-1}. W_{hh} are weight matrices generated against h_n. b_{ih} and b_{hh} are bias terms.

In the given policy network, the input image and the past RNN memory cell are used in the first state (first cycle) to generate actions for an image. In other cases (cycles), the image features information can be stored in the RNN hidden states and

replace past action information. This way, historical information is utilized to generate hash codes and correction of past action errors is realized. The h_t at the last stage of the cycle gives the history information for the next steps in Figure 1.

The policy layer, the linear layer, which is the fully connected layer, is defined as in Equation 8. The output of the linear layer includes a linear network and an activation function.

$$a_n(x) = \tanh\left(W_h^T h_n + b\right) \tag{8}$$

h_n is the RNN layer output, W_h^T is the policy layer weights and v is the bias term. At each step t, the output of the linear policy network is mapped between -1 and 1. With a threshold value, the map values are converted into hash codes. In this study, the threshold value will be 0 since probability values between -1 and 1 will occur. ($i = 1 \dots k$)

$$b_i(x) = \begin{cases} -1 & a_i(x) < 0 \\ 1 & a_i(x) \geq 0 \end{cases} \tag{9}$$

B. Environment vs Agent

In our study, three networks are trained. A stochastic gradient descent algorithm is used to update the parameters of the networks. The first network is used in the CNN network in the environment. The training parameter. θ_f is updated at each step. While in other deep hashing methods, the generation of image features is mostly performed once, in this work the weights of the network are updated again at each step. The objective function is $J_t^i\left(A_t^i, c_i\right)$ and the parameter θ_f is updated as in Equation 10.

$$\theta_f = \theta_f - \alpha_f \nabla_\theta J_t^i\left(A_t^i, c_i, \theta_f\right) \tag{10}$$

The goal of reinforcement learning is to find an optimal behavior strategy for the agent to obtain optimal rewards. This study aims to determine the optimization parameters for maximum reward.

$$\max G\left(\theta\right) = L_g(\theta) + R_g(\theta) \tag{11}$$

θ represents the model parameters, L_g is the reward of the action set and R_g is the reward of the objective function in the model. The model parameters are iteratively updated for the maximum reward.

In this work, hashing is considered as a dynamic Markov chain. The conditional probability distribution of the Markov chain is expressed as $p(s_{t+1}|s_t, a_t)$. $s_t \epsilon S$, $a_t \epsilon A$ and $s_{t+1} \epsilon S$; represent the state and action at time t and the next state, respectively. The objective is to learn the stochastic policy $\pi_\theta(a_t|s_t)$ conditional on the training parameter θ. A trajectory with t=1, 2,...,T

$$p_\theta(\tau) = p_\theta(s_1, a_1, s_2, a_2, \dots, s_\tau, a_\tau, s_{\tau+1}) == p(s_1) \prod_{t=1}^{T} \pi_\theta(a_t|s_t) p(s_{t+1}|s_t, a_t) \tag{12}$$

Let $g_t = G(a_t, s_{t+1})$ be the reward at time t. The objective function can find the value of θ^* that satisfies the optimal policy.

$$\theta^* = \arg max_\theta J(\theta) \tag{13}$$

Objective function $J(\theta)$,

$$J(\theta) = E_{\tau \sim p_\theta(\tau)}[\sum_{t=1}^{T} g_t] = E_{\tau \sim \pi_\theta(\tau)}[\sum_{t=1}^{T} g_t] \tag{14}$$

For the expected total reward of each set of actions taken, the Monte-Carlo Policy Gradient is used to optimize the model parameter. Gradient-based policy methods aim to model and optimize the policy directly. Policy gradient methods estimate the gradient in one iteration to maximize the expected total reward.

The policy gradient is expressed as $\nabla_\theta J(\theta) = \nabla_\theta E_{\tau\sim\pi_\theta(\tau)}[\sum_{t=1}^{T} g_t]$. In this study, Equation 15 is preferred among the many policy gradient definitions in the literature.

$$\nabla_\theta J(\theta) = E_{\tau\sim\pi_\theta(\tau)}[\nabla_\theta \log \pi_\theta(\tau)\, g(\tau)] \tag{15}$$

Expected total reward of the action set.

$$L_g(\theta) = E_{\tau\sim p_\theta(\tau)}\left[\sum_{t=1}^{T} g_t\right] == \sum_{t=1}^{T} \log[\pi_\theta(a_t|s_t)]\, g_t \tag{16}$$

s_t is the k th time in the loop in the RNN layer with the outputs of the RNN cell,

$$L_g(\theta) = \sum_{t=1}^{T}\sum_{i=1}^{k} \log\left[\mathrm{P}(a_{i,t}|\hat{s}_{i,t},\theta)\right] g_t \tag{17}$$

R_g is the global reward elated to the accuracy of all generated hash codes. Gradient descent algorithm is used to optimize the global reward. In model training we can calculate the sub-gradient for A_t^i, c_i is $\frac{\partial J_t^i\left(A_t^i, c_i\right)}{\partial\theta}$ by Equation 4.1. The hash codes generated with this training strategy are forced to be like those generated with Central Similarity.

The pseudo-code of the proposed method is roughly shown below.

Algorithm 2 Pseudo code of proposed method

```
Generate hash center of classes
While(1)
Create a history vector (hv) and assign all
zeros.
Generate image feature (fv) from CNN and
define state s=(fv,hv).
Generate actions from RNN. The input of RNN
is state (s).
Calculate the loss between the hash vector
and actions.
Calculate Backward-Propagation of Image
Adjust the learning rate every N epoch.
If loss < 0.001 break
End
```

3. Experiment Results

CIFAR10 and NUS-WIDE datasets, which are generally accepted in the literature, were used in this study. The CIFAR10 dataset contains about 60000 images of size 32 x 32, while NUS-WIDE consists of about 265000 images collected in Flicker and manually annotated with 81 concepts. For CIFAR10, we randomly select 1000 images as a query set and further randomly select 5000 images to form the training set. For NUS-WIDE datasets, we choose the 21 most prevalent concepts for our experimentation. Our query set consists of 2100 images, with 100 images per concept. Additionally, we randomly select an extra 10000 images for the training set.

The Hamming radius refers to the maximum number of bit positions by which two binary strings of equal length can differ from each other while still being considered within a specified distance from each other. In other words, it measures the extent of dissimilarity between two binary strings regarding the differing bit positions.

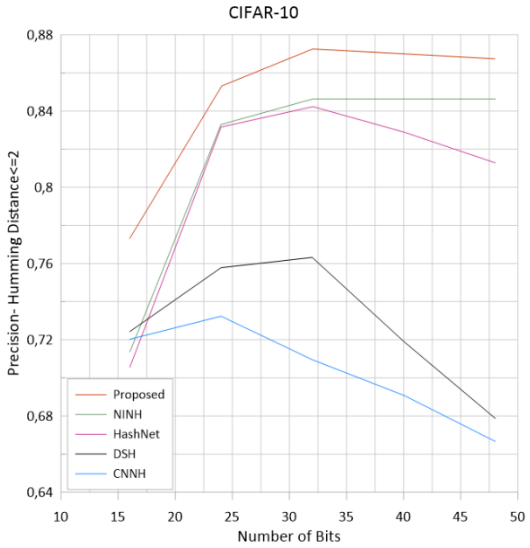


Figure 6 Precision within Hamming radius 2 using hash lookup for CIFAR Dataset

In this study, CNNH [28], DSH [31], HASHNET [9] and NINH [28], which are deep hashing approaches, were compared with the proposed method. When Figure 6 is analyzed for the images of the CIFAR10 dataset using Hamming Radius 2, the precision of the proposed method outperforms the other methods for all bit values. In addition, when the length of the hash codes increases, the number of images sharing the same Hamming code will decrease, so the image-matching process with Hamming radius 2 is unsuccessful for most algorithms. When Figure 6 is analyzed, it can be observed that the proposed method also performs better for longer hash codes.

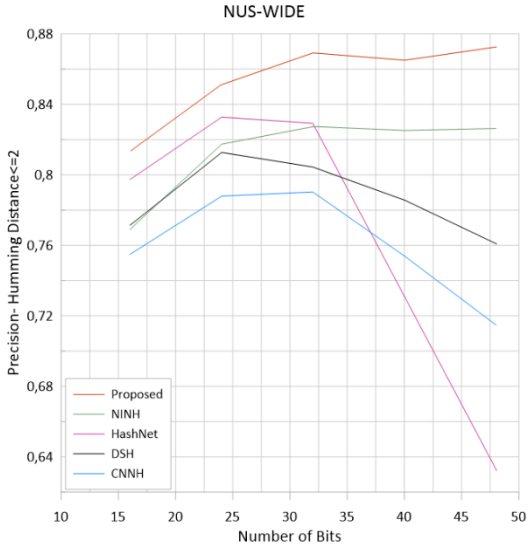


Figure 7 Precision within Hamming radius 2 using hash lookup for NUS-WIDE Dataset

In Figure 7, for NUS-WIDE, the proposed method outperforms the other methods at all hash code lengths.

The Mean Average Precision (MAP) is computed as the arithmetic mean of the Average Precisions (AP) obtained for each query within a given dataset. It is a metric commonly employed to assess the performance of information retrieval systems. It provides a single value that summarizes the overall performance of a retrieval system. Higher MAP values indicate better retrieval performance. As can be seen from Table 1, the MAP value is calculated for each hash code length. This value gives us for how good our model is.

$$AP = \frac{1}{R}\sum_{i=1}^{n}\frac{i}{Ri} \times rel_i \tag{16}$$

$$MAP = \frac{1}{N}\sum_{i=1}^{n} APi$$

R represents the overall count of true positive instances, n stands for the total number of images under consideration, APi denotes the precision at point i, and rel_i functions as a relevance indicator. The relevance function, characterized as an indicator function, assumes a value of 1 when the document at rank i is considered relevant, and it takes a value of 0 otherwise.

Table 1. MAP Values of Hash Bits

Methods	**CIFAR10**			**NUS-WIDE**		
	24 bit	32 bit	48 bit	24 bit	32 bit	48 bit
NINH	0.818	0.832	0.830	0.827	0.827	0.827
HashNet	0.823	0.840	0.843	0.833	0.830	0.840
DSH	0.712	0.751	0.720	0.804	0.815	0.800
CNNH	0.692	0.667	0.623	0.784	0.790	0.740
Proposed	0.857	0.868	0.865	0.859	0.862	0.865

In this study, MAP values of the proposed method and other methods are calculated for 24, 32 and 48 bit hash codes. MAP scores are calculated for hash codes of different lengths on the CIFAR10 dataset. The proposed method outperforms the other methods for all hash bit lengths. It is seen that the proposed method has the highest average MAP value of 0.863 and consistently outperforms the other methods. Similarly, for the NUS-WIDE dataset, When the MAP values are analyzed, the proposed method performs better at all hash bit lengths.

Conclusion

In this paper, we have proposed a Sequential and Correlated Image Hash Code Generation with Deep Reinforcement Learning (SCIHCGRL). To consider, the errors of previous hash functions in the deep reinforcement learning strategy, the RNN model is used, and a new model is presented that performs central similarity-based learning involving the correlation of hash functions. Thus, since the errors of the previous hash functions are considered during the sequential transfer of the images into binary codes, more efficient results are obtained in learning binary hash codes by adapting the Central Similarity metric to the model, which can achieve more successful results. Comparisons using CIFAR-10 and NUS-WIDE and datasets show that the proposed method gives better results.

Future work can be done to improve the time complexity of the algorithm. In addition, the proposed method can also be applied to videos by considering various properties of video media (correlation between video frames etc.).

References

[1] A. Swaminathan, Y. Mao and M. Wu, "Robust and secure image hashing," in IEEE Transactions on Information Forensics and Security, vol. 1, no. 2, pp. 215-230, June 2006, doi: 10.1109/TIFS.2006.873601.
[2] J. Wang, T. Zhang, N. Sebe, H. T. Shen et al., "A survey on learning to hash," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 40, no. 4, pp. 769–790, 2018.
[3] S. Zhang, J. Li, M. Jiang, and B. Zhang, "Scalable discrete supervised multimedia hash learning with clustering," IEEE Transactions on Circuits and Systems for Video Technology (TCSVT), vol. PP, no. 99, pp. 1–1, 2017.
[4] M. Kafai, K. Eshghi, and B. Bhanu, "Discrete cosine transform locality sensitive hashes for face retrieval," IEEE Transactions on Multimedia (TMM), vol. 16, no. 4, pp. 1090–1103, 2014.
[5] P. Li, M. Wang, J. Cheng, C. Xu, and H. Lu, "Spectral hashing with semantically consistent graph for image indexing," IEEE Transactions on Multimedia (TMM), vol. 15, no. 1, pp. 141–152, 2013.
[6] S. Zhang, J. Li, M. Jiang, and B. Zhang, "Scalable discrete supervised multimedia hash learning with clustering," IEEE Transactions on Circuits and Systems for Video Technology (TCSVT), vol. PP, no. 99, pp. 1–1, 2017.
[7] V. E. Liong, J. Lu, Y.-P. Tan, and J. Zhou, "Deep video hashing," IEEE Transactions on Multimedia (TMM), 2016.
[8] K. Ding, B. Fan, C. Huo, S. Xiang, and C. Pan, "Cross-modal hashing via rank-order preserving," IEEE Transactions on Multimedia (TMM), vol. 19, no. 3, pp. 571–585, 2017.
[9] Z. Cao, M. Long, J. Wang, and S. Y. Philip. Hashnet: Deep learning to hash by continuation. In ICCV, pages 5609–5618, 2017.
[10] W.-J. Li, S. Wang, and W.-C. Kang. Feature learning based deep supervised hashing with pairwise labels. arXiv preprintrXiv:1511.03855, 2015.

[11] M. Norouzi, D. J. Fleet, and R. R. Salakhutdinov. Hamming distance metric learning. In Advances in neural information processing systems, pages 1061–1069, 2012.
[12] Yuan, L., Wang, T., Zhang, X., Tay, F. E., Jie, Z., Liu, W., & Feng, J. (2020). Central similarity quantization for efficient image and video retrieval. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 3083-3092).
[13] Y. Li Deep reinforcement learning: An overview,2017
[14] Y. Peng, J. Zhang and Z. Ye, "Deep Reinforcement Learning for Image Hashing," in IEEE Transactions on Multimedia, vol. 22, no. 8, pp. 2061-2073, Aug. 2020, doi: 10.1109/TMM.2019.2951462.
[15] P. Indyk and R. Motwani. Approximate Nearest Neighbor {Towards Removing the Curse of Dimensionality. In Proceedings of the 30th Symposium on Theory of Computing, 1998, pp. 604{613.
[16] A. Gionis, P. Indyk, R. Motwani et al., "Similarity search in high dimensions via hashing," in International Conference on Very Large Data Bases (VLDB), vol. 99, no. 6, 1999, pp. 518–529.
[17] M. Raginsky and S. Lazebnik, "Locality-sensitive binary codes from shift-invariant kernels," in Annual Conference on Neural Information Processing Systems (NIPS), 2009, pp. 1509–1517.
[18] Y. Weiss, A. Torralba, and R. Fergus, "Spectral hashing," in Annual Conference on Neural Information Processing Systems (NIPS), 2009, pp. 1753–1760.
[19] Y. Gong and S. Lazebnik, "Iterative quantization: A procrustean approach to learning binary codes," in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2011, pp. 817–824.
[20] G. Irie, Z. Li, X.-M. Wu, and S.-F. Chang, "Locally linear hashing for extracting non-linear manifolds," in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014, pp. 2115–2122.
[21] Liu, W.; Mu, C.; Kumar, S.; and Chang, S. 2014. Discrete graph hashing. In NIPS, 3419–3427.
[22] Jiang, Q.-Y., and Li, W.-J. 2015. Scalable graph hashing with feature transformation. In IJCAI, 2248–2254.
[23] Liu, H.; Ji, R.; Wu, Y.; and Liu, W. 2016b. Towards optimal binary code learning via ordinal embedding. In AAAI, 1258–1265.
[24] X. Shen et al., "Semi-paired discrete hashing: Learning latent hash codesfor semi-paired cross-viewretrieval," IEEE Trans. Cybern., vol. 47, no. 12,pp. 4275–4288, Dec. 2017.
[25] F. Shen, Y. Xu, L. Liu, Y. Yang, Z. Huang, and H. T. Shen, ``Unsupervised deep hashing with similarity-adaptive and discrete optimization," IEEE Trans. Pattern Anal. Mach. Intell., vol. 40, no. 12, pp. 30343044, Dec. 2018.
[26] X. Shen et al., "Multiview discrete hashing for scalable multimedia search," ACM Trans. Intell. Syst. Technol., vol. 9, no. 5, pp. 1–21, 2018.
[27] Kong, W., and Li, W.-J. 2012. Isotropic hashing. In NIPS,1655–1663.
[28] R. Xia, Y. Pan, H. Lai, C. Liu, and S. Yan, "Supervised hashing for image retrieval via image representation learning," in Proc. AAAI Conf. Artif.Intell., 2014, pp. 2156–2162.
[29] H. Lai,Y. Pan,Y. Liu, and S.Yan, "Simultaneous feature learning and hash coding with deep neural networks," in Proc. IEEE Conf. Comput. Vision Pattern Recognit., 2015, pp. 3270–3278.
[30] R. Zhang, L. Lin, R. Zhang, W. Zuo, and L. Zhang, "Bit-scalable deep hashing with regularized similarity learning for image retrieval and person re-identification," IEEE Trans. Image Process., vol. 24, no. 12, pp. 4766–4779, Dec. 2015.
[31] H. Liu, R. Wang, S. Shan, and X. Chen, "Deep supervised hashing for fast image retrieval," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 2064–2072.
[32] T. Yao, F. Long, T. Mei, and Y. Rui, "Deep semantic-preserving and ranking-based hashing for image retrieval," in Proc. Int. Joint Conf. Artif. Intell., 2016, pp. 3931–3937.
[33] W.-J. Li, S. Wang, and W.-C. Kang, "Feature learning based deep supervised hashing with pairwise labels," in Proc. Int. Joint Conf. Artif. Intell., 2016, pp. 1711–1717.
[34] Wu, L., Ling, H., Li, P., Chen, J., Fang, Y., & Zhou, F. (2019). Deep supervised hashing based on stable distribution. IEEE Access, 7, 36489-36499.
[35] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: A survey," Journal of artificial intelligence research, vol. 4, pp. 237–285, 1996.

Conflict of Interest Notice
The authors declare that there is no conflict of interest regarding the publication of this paper.

Ethical Approval and Informed Consent
It is declared that during the preparation process of this study, scientific and ethical principles were followed, and all the studies benefited from are stated in the bibliography.

Availability of data and material
Not applicable.

Plagiarism Statement
This article has been scanned by iThenticate ™.