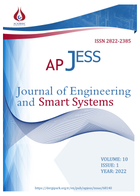


Research Paper
Academic Platform Journal of Engineering and Smart Systems (APJESS) 10(1), 9-18, 2022
ISSN: 2822-2385
Received: 19-Jun-2021 Accepted: 11-Oct-2021
https://doi.org/10.21541/apjess.1060753

Handwritten Digit Recognition With Machine Learning Algorithms

*[1]Kübra Gülgün Demirkaya, [2]Ünal Çavuşoğlu
[1]Sakarya University, Computer Engineering kubra.demirkaya@ogr.sakarya.edu.tr,
[2]Sakarya University, Software Engineering, unalc@sakarya.edu.tr,

Abstract

Nowadays, the scope of machine learning and deep learning studies is increasing day by day. Handwriting recognition is one of the examples in daily life for this field of work. Data storage in digital media is a method that almost everyone is using nowadays. At the same time, it has become a necessity for people to store their notes in digital media and even take notes directly in the digital environment. As a solution to this need, applications have been developed that can recognize numbers, characters, and even text from handwriting using machine learning and deep learning algorithms. Moreover, these applications can recognize numbers, characters, and text from handwriting and convert them into visual characters. This project, investigated the performance comparison of machine learning algorithms commonly used in handwriting recognition applications and which of them are more efficient. As a result of the study, the accuracy was 98.66% with artificial neural network, 99.45% with convolutional neural network, 97.05% with K-NN, 83.57% with Naive Bayes, 97.71% with support vector machine and 88.34% with decision tree. This study also developed a handwriting recognition system for numbers similar to these mentioned applications. A desktop application interface was developed for end users to show the instant performance of some of these algorithms and allow them to experience the handwriting recognition system.

Keywords: Machine Learning, Handwriting Recognition, Deep Learning, MNIST

1. INTRODUCTION

Today, the applications of artificial intelligence and data science are advancing very rapidly and are now used in almost all fields. Most of these applications include machine learning and deep learning. One of the application areas is handwriting recognition as an evolving field. Handwriting recognition enables communication between machines and humans and is a field that aims to facilitate this communication. A lot of information is now stored in digital media. In daily life, almost everyone has started storing their data and notes in digital media and using electronic diaries. They tend to take notes in the digital environment using the keyboard, touch screen, and smart pens of smartphones and tablets that they always have with them. Handwriting recognition systems are needed more and more every day for the following reasons: For data to be stored in digital media with recognition once handwritten, for data previously written to be transferred to digital media as optical characters. With handwriting recognition systems beginning to evolve out of this need, today's technology has moved away from keyboards to touch systems, and writing in the digital environment has begun to be done with handwriting instead of the small keyboards of touch screens. In addition, handwritten text documents, lettering on signs, etc. have begun to be transferred to digital in sectoral areas. Handwriting recognition systems are constantly evolving and being integrated into intelligent systems. Many electronic tablets and digital agendas are now offered to users with handwriting recognition system software and smart pens. At the same time, in addition to everyday use, these systems are also being used in areas such as education, health, and security, which are evolving every day. In banks, handwriting recognition systems are used in areas such as reading check amounts and forms, and reading and sorting incoming mail addresses [1]. In addition, studies are being conducted daily for the systems to be used by children in smart boards and tablets, mainly to be used in education and to support teaching.

Handwriting recognition is the computer recognition of handwritten letters, numbers and characters. This process, which is very simple for a human, is difficult for computers. In other words, making sense of lines, symbols, and their combined shapes at the word level is difficult for computers. Handwriting features such as the presence of characters that are different in many languages, the fact that each person has different handwriting, and the presence of combined handwriting make it difficult for computer systems to recognize handwriting. This topic is not yet fully developed and it is an area of limited efficiency [2]. When this technology is developed, which is mainly used in tablet computers and for which there are already examples, it will be possible to store and organize any handwritten information in a digital environment without using a keyboard. Handwriting recognition technologies can be

[*] Corresponding Author

studied in two different groups. One is interactive (online) and the other non-interactive (offline) systems. Interactive systems are systems that recognize and classify by following the movements of the writing tool as it writes the handwriting. They are usually used in devices with a touch surface such as tablets, phones, smartboards, etc. Every movement during writing is controlled by the device. This is an important factor in increasing the accuracy rate. The disadvantage of these systems is that they have to give instant results and they have to be fast running systems as they have to keep up with the writing speed [2]. Non-interactive systems are an attempt to recognize the information previously written on paper by digitizing it using methods such as photography and scanning. These systems are not expected to produce instantaneous output, so there are ways to work more slowly. However, since handwriting may not be very smooth, especially in old handwritten documents, the accuracy rates in the recognition process are low. To increase this rate, extensive pre-processing on the photo is required first. However, as a first step, the text should be divided into sections. Lines, sentences, words and characters should be divided according to their size, then preprocessing steps should be applied to each divided part before classification [1]. The advantage of this method is that it can be used for all kinds of documents that have existed for years and should be transferred to electronic media.

In this study, detailed information about the different machine learning and deep learning algorithms used in the field of handwriting recognition was given and experiments were conducted to measure their performance in this field, discover the most efficient parameters and compare the success rates on the MNIST dataset. The results obtained at the end of the experiments were compared with the results of similar studies in this field. In addition, using some of these algorithms, a system for recognizing digits from non-interactive handwriting, similar to handwriting recognition applications, was developed and an example usage method suitable for everyday use was demonstrated. A desktop application was developed for the end users of the application. In the second part of this study, similar studies on the topic in the literature are explained. In the third section, the dataset used, the development environment, the method and the evaluation criteria are explained in detail. In the fourth section, the details of the research conducted are explained. Detailed information is given about the design of each model created and the algorithm used in the model. In the fifth section, the interface application created is explained. In the sixth chapter, the results obtained at the end of the experiments are explained and the comparison of the success rates of the models in this study with similar studies are included. In the seventh chapter, the findings and evaluations obtained from the study are included in this direction.

2. RELATED STUDIES

In the work of Mohd Razif Shamsuddin and his colleagues who analyzed machine learning models for the MNIST dataset; They obtained 2 different versions of the MNIST dataset, grayscale and binary (black and white), with data preprocessing. They also performed the model training with these 2 different datasets. In the grayscale dataset; The accuracy rate was 99.4% with the CNN model, 94% with the random forest model, and 94% in the Extremely Randomized Trees model. In the binary data set; The accuracy rate was 90.1% with the CNN model, 91% with the Random Forest model, and 92% with the Extremely Randomized Trees model. As a result of these experiments, they showed the effect and importance of proper selection of data preprocessing step and methods in machine learning on success rate [3]. Abien Fred M. Agarap created an architecture for machine learning by combining convolutional neural networks and support vector machines for image classification problems. In addition to the model where he used the ReLu function as the activation function in the CNN structure and then added a support vector machine, he created another model where the same CNN structure determined the activation function of the output layer as a softmax function. 2 models named "CNN-Softmax" and "CNN-SVM"; trained with MNIST and Fashion-MNIST datasets and compared the results. In the study, 60000 training data and 10000 test data from both datasets were used. With the structure of "CNN-Softmax", 99.23% success rate in MNIST dataset and 91.86% success rate in FashionMNIST dataset; With the structure of "CNN-SVM", it achieved 99.04% success rate in MNIST dataset and 90.72% in Fashion-MNIST dataset [4].

Mine Altınay Günler Pirim argued that the output weights in the hidden layers in the structure of neural network used in handwriting recognition can be used to extract the feature vectors of the image. For this purpose, it used MNIST and USPS datasets in its experiments with MATLAB; using the artificial neural network, the support vector machine and the Euclidean distance classifier algorithm as the classifier, the success rates were measured. Using the structure developed in the study, it achieved success rates of 99.64% in the support vector machine model with the MNIST dataset, 98.01% in the Euclidean distance classifier model, and 98.56% in the artificial neural network model. With the USPS dataset, it achieved 97.47% success rates in the support vector machine model, 94.52% in the Euclidean distance classifier model, and 94.18% in the artificial neural network model [5]. Aoudou Salouhou, tested and compared deep learning algorithms for image classification and handwriting recognition using Fashion-MNIST, MNIST, CIFAR-10 and Arabic datasets in his study. To classify Arabic characters, the Arabic dataset created by Ahmed and his friends was used [6]. He used Deep Neural Network, Convolutional Neural Network and recursive neural network structures from Deep Learning algorithms. In his experiments with the MNIST dataset, the deep neural network model provided 99.53% accuracy rate, the convolutional neural network model provided 99.88% accuracy rate, and the iterative neural network model provided 99.05% accuracy rate. In his experiments with the Arabic dataset; The deep neural network model provided 96.48% accuracy, the convolutional network model provided 99.00% and the recursive neural network model provided 96.94% accuracy [7].

In her studies on handwriting recognition with machine learning algorithms, Rabia Karakaya conducted tests with support vector machine, decision tree, random forest, artificial neural network, K-nearest neighbor algorithm and

K-mean algorithm using MNIST dataset. She conducted her work using Scikit Learn library and tools. In the test results of the models trained with MNIST dataset using all 60000 data, she achieved 90% accuracy rate by using polynomial kernel function in support vector machine model. It also achieved 87% accuracy in the decision tree model, 97% in the random forest model, and 97% in the artificial neural network model. In the K-Nearest Neighbor algorithm model, it achieved 96% accuracy in 865.932 seconds of test time and 98% accuracy in the K-Mean algorithm model [8]. In their work, Shubham Sanjay Mor and his colleagues developed a system and an Android application to recognize handwritten characters and numbers using the EMNIST dataset and the CNN structure. At the end of the experiments they conducted in the CNN model they created, they used the "Adamax" function as an optimization parameter with which they achieved the highest performance. The model they created recognizes 62 handwritten characters and has an accuracy rate of 87.1% [9].

3. MATERIAL AND METHOD

3.1. MNIST Dataset

The dataset used for training and testing this system is the MNIST (Modified National Institute of Standards and Technology) dataset. It is a dataset created to process and make sense of the image. It consists of images of handwritten numbers, each of which is 28x28 pixels in size. It contains 60,000 training (train) and 10,000 verification (test) images as classification [10]. It is a commonly preferred dataset in studies in similar domains.

3.2. Libraries

The software libraries used in this study are TensorFlow, Scikit Learn, Keras and Numpy libraries. In the following sections of this study, some mathematical values are used to make comparisons in performance evaluations and analysis. These mathematical values defined in the mentioned libraries and their definitions are as follows:

Confusion Matrix: Confusion Matrix or error matrix is a performance measurement method for machine learning classification algorithms. It gives information about the accuracy of the predictions. It is a table that contains 4 combinations of predictions and actual values [11]. With the help of this matrix, we can calculate values like precision, recall, support, accuracy, specificity, and F1 score.

Accuracy: It is a value that indicates the accuracy rate. It is the ratio of predictions classified as correct to all predictions.

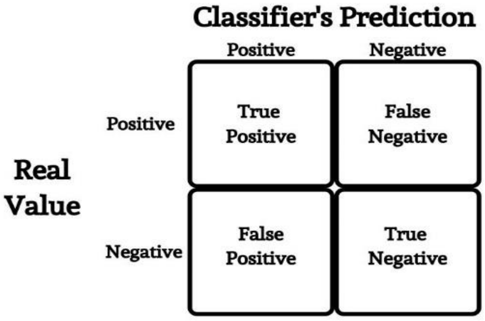


Figure 1. Confusion Matrix

Precision: It is the value that indicates how much of the positively predicted data was predicted correctly. The higher this ratio is, the more accurate the predictions were [12]. It is calculated as;

$$\mathrm{TP}/(\mathrm{TP} + \mathrm{FP}) \tag{1}$$

Recall: It is the ability of the classifier to find all positive samples. It is a metric that shows how much of the data that should be positively predicted is positively predicted [12]. It is calculated as;

$$\mathrm{TP}/(\mathrm{TP} + \mathrm{FN}) \tag{2}$$

F1 Score: The F1 score can be interpreted as a weighted harmonic average of the Precision and Recall scores. It is a measure of the level of performance exhibited by the model. It is often used to compare models.

3.3. Data Processing

Data sets are processed through multiple phases, as with any MA study courses. Data pre processing, cutting, feature extraction and classification are the key procedures employed in the image classification and handwriting recognition applications.

Preprocessing data: Data preparation is performed ready for analysis and subsequent phase; such operations as correction, conversion, cleaning, decrease of size, standardization and noise reduction. In handwriting recognition data the data preparation processes required might be stated in the following way:

Thresholding: This is the way the image is converted into a binary picture, namely a black and white one. The basic objective is to highlight the image and identify the item with this technique. Pixels are calculated as black or White according to the supplied threshold value, depending on the type of threshold approach used [13].

Noise reduction: Refers to processes carried out to further clear the picture. The highlights can be sharpened in texts by a technique that is needed. It aims to produce a clear picture through this approach. Methods such as a medium filter, median filter, Gaussian smoothing filter and masking approach can minimize noise.

Normalization: Slope correction is an additional name for standardization, one of the image processing procedures. This procedure corrects the curvature of the text in handwritten text photographs and eliminates the skew. Histograms are used to identify and correct curvature and path, as with many other image processing procedures. In the early parts of the normalization operations, Bosinosvic and Srihari Method (BSM) is commonly used [14]. The angle between the horizontal axis in the text correction and the horizontal axis of the written text is known as the slope, and the angle between the vertical axis in the text correction and the vertical axis of the written text is known as the slant. Normalization is another term for handwriting correction using slant and slope [15].

Feature extraction: Each distinguishing feature in the photos may be designated as an attribute. Attribute information is composed of numerical data derived from the separation of features from the picture [16]. Approaches like histograms, projection-based methods, Fourier and Wavelet transforms, or defining letters as a set of basic shapes like curves and lines are utilized at this step [2]. The information collected from this stage has a direct impact on the recognition stage, and the qualities of this information have a direct impact on the recognition stage's efficiency.

Classification: During the classification phase, the attributes of the data in the picture are compared to the classes in the database to determine which class the picture belongs to. Many various approaches are utilized at this step, including template matching, neural networks, classification algorithms, statistical learning, and structural learning. The data sets employed are critical for this stage's high performance and accuracy, and they should be prepared to contain as many samples and kinds as feasible.

3.4. The Proposed Model

The implementation steps in the project are as follows:

1) The project contains necessary libraries such as Tensorflow, Keras, Scikit Learn, and Numpy.
2) The project includes the MNIST dataset from the Tensorflow-Keras library.
3) The dataset was labeled and split into training and test data.
4) The data was subjected to preprocessing processes.
5) Following the preparation processes, the data was normalized and characteristics were retrieved.
6) Algorithms to be utilized with libraries have been defined.
7) For each method, the parameters to be utilized during training were determined in the most appropriate method for the dataset. It was intended to get the best possible results.
8) The dataset was trained using the "Fit" function, and model training was completed. Scikit Learn library was widely utilized during learning in K-NN, SVM, Decision Tree, and Naive Bayes algorithms, while Tensorflow and Keras libraries were employed in neural networks.
9) The model was evaluated using the "Predict" function on a portion of the dataset that had been put aside for testing.
10) Finally, to assess the performance of the trained and tested model, the classification report and confusion matrix were generated and shown as a report using the "classification report" and "confusion matrix" functions of the Scikit Learn library's "metrics" module. Precision, recall, f1 score, support, and accuracy scores generated from the confusion matrix findings are shown in tabular form in the classification report. This table contains enough information to allow you to examine and evaluate the performance results.

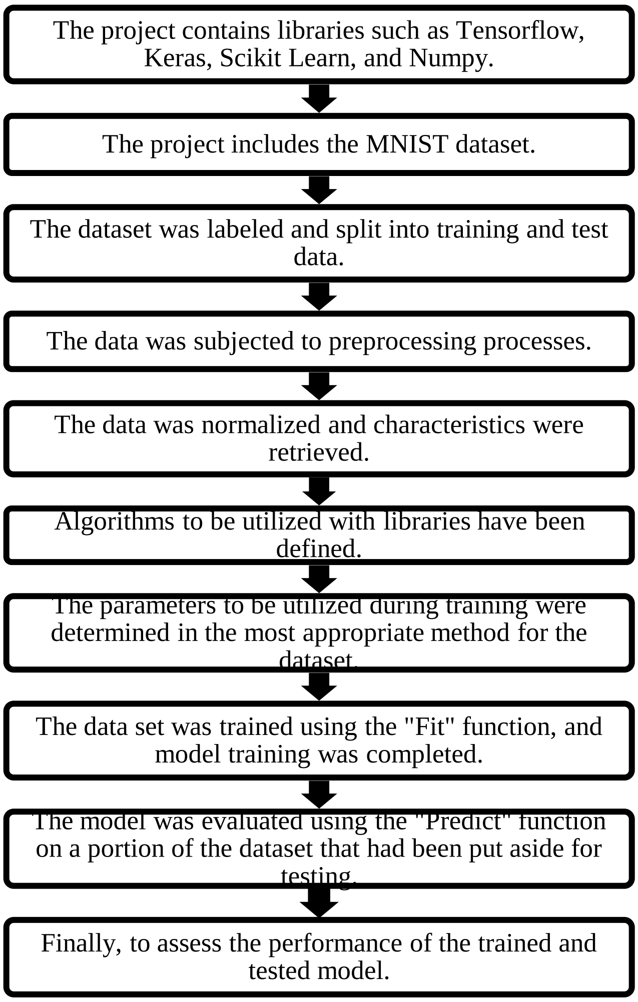

The project contains libraries such as Tensorflow, Keras, Scikit Learn, and Numpy.
The project includes the MNIST dataset.
The dataset was labeled and split into training and test data.
The data was subjected to preprocessing processes.
The data was normalized and characteristics were retrieved.
Algorithms to be utilized with libraries have been defined.
The parameters to be utilized during training were determined in the most appropriate method for the dataset.
The data set was trained using the "Fit" function, and model training was completed.
The model was evaluated using the "Predict" function on a portion of the dataset that had been put aside for testing.
Finally, to assess the performance of the trained and tested model.

Figure 2. The Implementation Steps

4. RESEARCH AND PERFORMANCE RESULTS

The tests carried out, the models constructed using the methods utilized, the hyper-parameter modifications, and the performance results are all detailed in depth in this section. All 70000 bits of data from the data set were used, with 60000 reserved for training and 10000 designated for testing, and the ratios of these portions were kept consistent throughout all models.

Artificial Neural Network – ANN

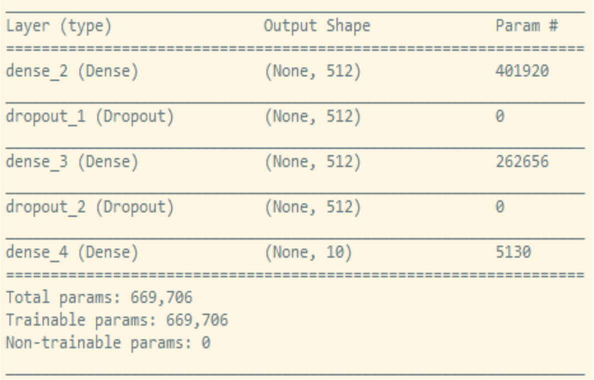
Deep learning is based on neural networks, which were formed by modeling the neuron structure of humans and adapting this neuron structure to machines. The human neuron structure was used to generate features such as brain structure, learning, and information use. To detect real-world relationships, artificial neural networks have been built. They can also conduct classification, pattern recognition, grouping, and estimate. They may process many inputs and generate results [17]. The network structure is formed by the combination of artificial neural network cells within the framework of particular rules. Layers are generated when these brain networks, or neurons, join together. Receiving inputs and sending outputs are handled by cells and layers. As a result, they are linked to the outside world [18]. This structure's layers are divided into three sections: the input layer, the concealed layer, and the output layer.

"Multiple-layer perceptron neural network (MLPNN)" refers to artificial neural networks that have one or more hidden layers in addition to the input and output layers. MLPNN structures are artificial neural network structures that are employed and represented in many artificial neural network research nowadays [19]. The study's artificial neural network is a three-layer fully linked network. In order to build the neural network, the "Sequential" model from the Keras library's "model" module was used. According to the "Sequential" paradigm, a neural network will be formed.

The "Dense" class was then added to the "layers" module to generate neural network layers. The number of neurons in the layers was calculated using these modules, and the neural network was built. Neurons in the neural network's hidden layers have activation functions [19] A neuron's activity is determined by activation functions. As a result, they are extremely important in neural networks. Different functions are selected based on where they will be utilized and how they will effect performance. The most preferred and recommended activation functions in neural network layers are "relu", "sigmoid" and "softmax" functions. In the neural networks in this study, while the "relu" activation function is used in the input layers, the "softmax" function, which is the most preferred in multiple classification problems, is used in the output layers. The "Softmax" function generates values between [0,1] and these values show the probability that each input belongs to a class. After creating the neural network structure, it was compiled with the necessary parameters.

Table 1. Parameters applied to the ANN model in this study

Parameters	Values applied in this model
Batch Size	256
Epochs	40
Dropout Rate	(0.5,0.2)
Learning Rate	0.001
Activation Function	Relu
Activation Function (output layer)	Softmax
Loss Function	Categorical cross entropy
Metric	Accuracy



```
Layer (type)                 Output Shape              Param #
=================================================================
dense_2 (Dense)              (None, 512)               401920
_________________________________________________________________
dropout_1 (Dropout)          (None, 512)               0
_________________________________________________________________
dense_3 (Dense)              (None, 512)               262656
_________________________________________________________________
dropout_2 (Dropout)          (None, 512)               0
_________________________________________________________________
dense_4 (Dense)              (None, 10)                5130
=================================================================
Total params: 669,706
Trainable params: 669,706
Non-trainable params: 0
_________________________________________________________________
```

Figure 2. Architecture of the designed ANN model

For optimization, tests were carried out with 3 different parameters. These are the "Adam", "Adadelta" and "RMSprop" parameters. Finally, the model was trained and tested. The results obtained after 40 epochs are as follows:

Table 2. Comparison of success rates of optimization parameters

Optimization Parameters	Accuracy Rate
Adadelta	%85.39
Adam	%98.66
Rmsprop	%98.50

Convolutional Neural Network – CNN

Convolutional neural networks are the form in which artificial neural networks were developed for image data training to be performed using convolution. It is a neural network with a convolutional mathematical operation in some of the layers. In these neural networks, a convolution structure is created by convolution, pooling and smoothing steps just before the ANN structure [20].

The neural network structure has been started with the "Sequential" model, as described in the previous title. "Convolution2D" function was used for convolution in the convolution layer and "Maxpooling2D" function was used in the pooling layer. These two layers can be added as many times as desired. Finally, the "Flatten" function has been used for flattening. Activation functions are used in convolution layers as in neural network structures. In this study, the "relu" function was preferred in neurons of all convolution layers. After the flattening layer, the convolution structure was connected to the ANN structure.

"Batch_size" and "epochs" values given as parameters to the "compile" function in neural networks indicate how long the training will take. The "epochs" value represents the number of rounds of the training, which determines how many times the data will be shown to the network. The "batch_size" value is a measure of how much data will be received in each epoch. By changing these values, it is possible to improve the test results and increase the accuracy rate.

In the convolutional neural network model designed in this study, the hyper-parameters were determined as follows:

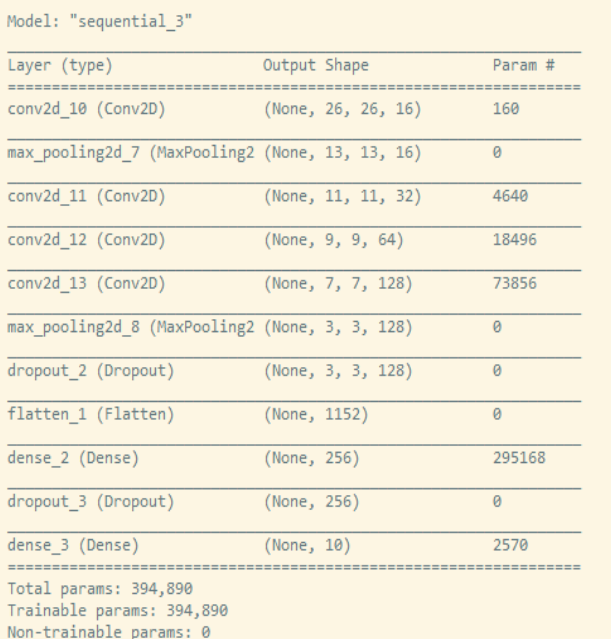
Table 3. The parameters applied to the CNN model in this study

Parameters	Values applied in this model
Batch Size	128
Epochs	30
Dropout Rate	(0.25,0.5)
Learning Rate	0.001
Activation Function	Relu
Activation Function (output layer)	Softmax
Loss Function	Categorical cross entropy
Metric	Accuracy

Optimization parameters given in the compilation section of the created CNN structure also affect the accuracy rate. In this study, 2 different trainings were made using the "adam" and "adadelta" functions and the test results were obtained. All variables except the “optimizer” parameter were kept constant. The architecture of the proposed model is shown in the report obtained with the "summary" function below.

Table 4. Comparison of accuracy rates of tested parameters

Optimization Parameters	Accuracy	Loss
Adam	0.994	0.304
Adadelta	0.864	0.548



```
Model: "sequential_3"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_10 (Conv2D)           (None, 26, 26, 16)        160
_________________________________________________________________
max_pooling2d_7 (MaxPooling2 (None, 13, 13, 16)        0
_________________________________________________________________
conv2d_11 (Conv2D)           (None, 11, 11, 32)        4640
_________________________________________________________________
conv2d_12 (Conv2D)           (None, 9, 9, 64)          18496
_________________________________________________________________
conv2d_13 (Conv2D)           (None, 7, 7, 128)         73856
_________________________________________________________________
max_pooling2d_8 (MaxPooling2 (None, 3, 3, 128)         0
_________________________________________________________________
dropout_2 (Dropout)          (None, 3, 3, 128)         0
_________________________________________________________________
flatten_1 (Flatten)          (None, 1152)              0
_________________________________________________________________
dense_2 (Dense)              (None, 256)               295168
_________________________________________________________________
dropout_3 (Dropout)          (None, 256)               0
_________________________________________________________________
dense_3 (Dense)              (None, 10)                2570
=================================================================
Total params: 394,890
Trainable params: 394,890
Non-trainable params: 0
```

Figure 4. Architecture of the designed convolutional neural network model

Naive Bayes Algorithm

Naive Bayes classification algorithm is an algorithm based on Bayes' theorem, that is, probability and using probability calculation. This algorithm works by calculating the probability of belonging to each class for each new data and classifying it according to the highest probability value [21]. Very efficient results can be achieved even with low computation time. It can work well on unbalanced datasets. Bayes' theorem equation is given below.

$$P(A/B) = (P(B/A) \times P(A))/P(B) \tag{3}$$

The "naive_bayes" module in the Scikit Learn library has been added for the Naive Bayes algorithm. There are 3 most important submodules that can be used for the Naive Bayes algorithm in the Scikit Learn library. These are: “GaussianNB”, “MultinomialNB”, “BernoulliNB”. These modules are methods that can be used in different data sets and they work by using Gaussian, Multinomial, Bernoulli distribution methods. The ones that can be used in the MNIST dataset in this study are the "GaussianNB" and "MultinomialNB" methods. As a result of the tests, a higher accuracy rate was obtained with the “MultinomialNB” method. Finally, the training and testing process was carried out, the test results are as follows:

Table 5. Accuracy rate of Naive Bayes model

Model	Accuracy
Naive Bayes (MultinomialNB)	0.835

Decision Trees

Decision trees are algorithms that are frequently used in classification problems, which divide a data set into smaller classes by putting certain decision rules and querying within the framework of these rules. It consists of three basic parts called node, branch and leaf. Each variable is defined as a node. They are easily understandable structures. It can be used for processing both categorical and numerical data. The most important disadvantage of decision trees is that the tree structure cannot be read and followed if very complex trees are produced. In addition to these, there is a possibility of over-fitting [22]. Important points in decision trees; how the division will take place, how the branching will take place, that is, how the tree structure will be created and according to which algorithm. There are many decision tree algorithms, but in order to achieve high accuracy, the most suitable algorithm for the data set and the problem should be selected [23].

In order to use decision trees, the "tree" module of the Scikit Learn library has been added to the project. For classification, the “DecisionTreeClassifier” submodule is included from this module. When creating a model with the "DecisionTreeClassifier" class, the "criterion" parameter must be specified. With this parameter, the tree forming criterion and the function that will create the division criterion of the data are determined. In this study, the “criterion” parameter was chosen as “Gini”.

Table 6. Parameters applied to the Decision Tree model in this study

Parametres	Values applied in this model
Criterion	gini
Splitter	best

Class Weight	balanced
Maximum Depth	15
Minimum Samples Split	2
Minimum Samples Leaf	1
Minimum Weight Fraction Leaf	0.0
Minimum Impurity Decrease	0.0

Finally, the training and testing process was carried out, the test results are as in Table 7:

Table 7. Accuracy rate of Decision Tree model

Model	Accuracy
Decision Tree	0.8834

Support Vector Machine – SVM

Support vector machines that can perform well on both linear and non-linear data are easy-to-implement algorithms. High success results can be obtained with support vector machines in many data sets [24].

Support vector machines are a method that creates support vectors based on existing class data and classifies new data accordingly. Support vector machines farthest removed from any two points dealt with in the dataset will allow to decide between the two classes is work to create a border [25]. Support Vector Machines are divided into two according to the linear separability and non-separability of the data set.
One of the most important points in support vector machines is kernel functions. The appropriate kernel function should be selected according to whether the data set can be separated linearly or not [26]. These kernel functions greatly affect success. The most commonly used kernel functions are "linear", "rbf", "poly" and "sigmoid". While the “linear” kernel function is used for linearly separable data sets, “rbf” is used for non-linearly separable data sets.

The "SVC" (Support Vector Classification) module under the "svm" module from the Scikit Learn library for the support vector machine was included in the study. “poly”, “rbf” and “linear” kernel functions were tested with the MNIST dataset. As a result of the test, in this study, the "poly" kernel function was preferred in accordance with the MNIST data set.

The results after the training and testing processes are as in Table 8:

Table 8. Comparison of the accuracy rates of the parameters applied to the SVM model and the kernel functions tested

Kernel Function	Gamma value	C value	Accuracy Rate
poly	scale	1	%97.71
linear	scale	1	%94.04
rbf	scale	1	%97.92

K-Nearest Neighbor Algorithm – K-NN

K-NN is a supervised learning algorithm that provides learning by looking at the nearest neighbor data around the evaluated data. It is frequently preferred because it can be used in very wide areas, can provide high success results, and is easy to understand.

The two most important parameters affecting the success rate of the algorithm are the number of neighbors (k) and the distance criterion, which is the value of the distance to the neighbors [27]. The data is classified by looking at the nearest neighbors as many as the k value determined in the algorithm. The data selected as the nearest neighbor is selected according to the parameter determined as the distance criterion. The most common distance criterion; Minkowski, Euclidean and Manhattan distances are used [28].

The K-nearest neighbor algorithm is a powerful algorithm because it is simple and resistant to noisy training data. The disadvantage of this algorithm is that it needs very large memory space in large data set, as it stores all data and accounts.

Table 9. Variation of accuracy rate according to the parameters applied to the K-NN model and the k value

Number of neighbors (k)	Distance criterion	Accuracy rate
1	minkowski	%96.91
2	minkowski	%96.27
3	minkowski	%97.05
4	minkowski	%96.82

In this study, the k value, which represents the number of neighbors, was determined as 3 with the "n_neighbors" parameter. The “metric” parameter, which is the distance criterion, was determined as “minkowski”, that is, the Minkowski distance was used.

The results after training and testing are as follows:

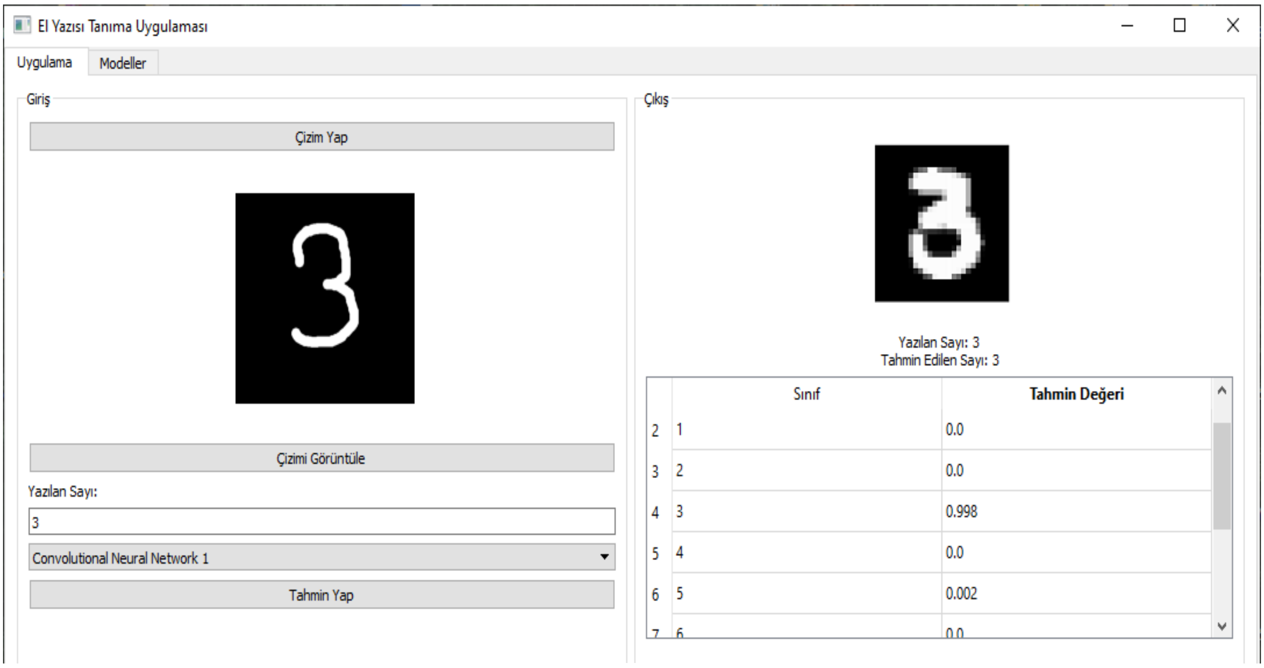
Table 10. Accuracy rate of K-NN model

Model	Number of neighbors (k)	Distance criterion	Accuracy
K-NN	3	minkowski	0.9705

5. USER INTERFACE APPLICATION

In this project, an interface application was created for the end user by using the "PyQt5" library created for the graphical user interface of the Python programming language. The application has been developed with the aim of taking a handwritten number using the mouse as input data and classifying this data through previously trained and recorded models.

In the "Modeller" tab of the application, the parameter values of the trained algorithms used in the training are listed and presented to the user's information. Users click on the Draw button, write a number in the drawing area window opened with the mouse and save the entry with the "Kaydet (Save)" button. If it is written incorrectly and it is desired to repeat the drawing process, the drawing screen is returned to its original state by clicking the "Temizle (Clear)" button.



Yazılan Sayı: 3

Convolutional Neural Network 1

Yazılan Sayı: 3
Tahmin Edilen Sayı: 3

	Sınıf	Tahmin Değeri
2	1	0.0
3	2	0.0
4	3	0.998
5	4	0.0
6	5	0.002
7	6	0.0

Figure 5. Interface implementation classification process result

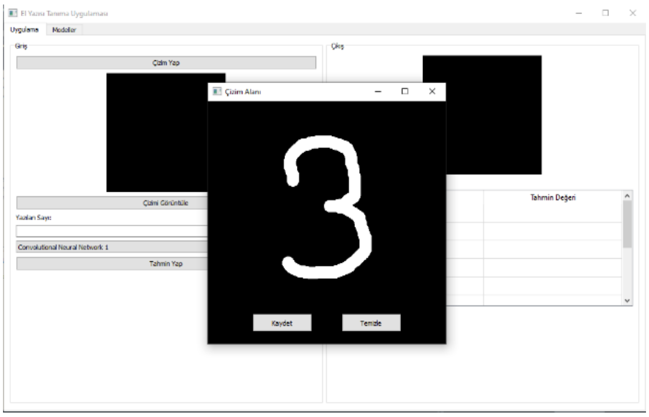


Figure 6. Interface application drawing window

Then, with the "Çizimi Görüntüle (View Drawing)" button on the main screen, the input data is displayed in the main window. In the next step, users enter the "Yazılan Sayı (Actual Number):" information in the relevant place.

The users, among the algorithms previously described in this study; details of models trained using Convolutional Neural Network, Multiple-layer perceptron neural network, Naive Bayes, Support Vector Machine, K-Nearest Neighbors and Decision Tree algorithms can be accessed in the "Modeller" tab. Then, on the main screen, they select the model they want to be used in the classification process. The classification process is started with the "Tahmin Yap (Predict)" button. After the classification process is finished, "Sınıflandırma tamamlandı (Classification is complete)." information message is received.

Then, in the "Çıkış (Output)" pane, a representative data from the MNIST dataset is displayed as the classification result in the appropriate label. In the "Tahmin Edilen Sayı (Predicted Number):" section, the label of the classification result is shown. In the lower part, the results table and prediction rates are shown.

6. EXPERIMENT RESULTS

The results obtained from the test results are shown in Table 11. According to the results of the experiment performed with the MNIST data set in handwriting recognition processes, an accuracy rate of over 80% was obtained in all models. The highest accuracy rate is 99.45% and this rate was obtained with the CNN-Adam model. After the convolutional neural network models, the ANN-Adam model provided the highest accuracy rate with 98.66%. In addition, in the same data set, with the number of 3 neighbors, the K-NN model provided 97.05% accuracy and the SVM-poly model 97.71% accuracy. High success has been achieved in these models as well.

Table 11. Accuracy rate comparison of models

Model	Accuracy Rate
ANN-RMSprop	%98.50
ANN-Adam	%98.66
CNN-Adadelta	%86.44
CNN-Adam	%99.45
K-NN	%97.05
Naive Bayes	%83.57
SVM-poly	%97.71
Decision Tree	%88.34

Neural networks, which have not been studied much due to hardware deficiencies until recently, often show higher success than other classification algorithms with today's hardware.

Algorithm and method selection in handwriting recognition systems depends on many factors such as hardware, working environment, interactive or non-interactive systems. However, considering the developing technology and hardware, the use of neural networks in this field will provide high efficiency and accuracy.

In other classification algorithms, success rates can be increased by using different hyper-parameters and applying detailed data preprocessing. These algorithms are frequently preferred in handwriting recognition, show high success and are already used in many software in this field.

Comparison of the accuracy rates of the models created in similar studies in the literature with the accuracy rates of the models created in this study are given in Tables 12, 13, 14, 15.

Table 12. Comparison of the K-NN model with similar studies

Reference (Year)	Dataset	Accuracy Rate
M. A. G. Pirim (2017)	MNIST	%98.01
M. A. G. Pirim (2017)	USPS	%94.52
R. Karakaya (2020)	MNIST	%96
This Study	MNIST	%97.05

Table 13. Comparison of ANN-Adam model with similar studies

Reference (Year)	Dataset	Accuracy Rate
M. A. G. Pirim (2017)	MNIST	%98.56
M. A. G. Pirim (2017)	USPS	%94.18
A. Salouhou (2019) – Derin Sinir Ağı	MNIST	%99.53
A. Salouhou (2019) – Yinelemeli Sinir Ağı	MNIST	%99.05
R. Karakaya (2020)	MNIST	%97
This Study	MNIST	%98.66

Table 14. Comparison of the CNN-Adam model with similar studies

Reference (Year)	Dataset	Accuracy Rate
M. R. Shamsuddin ve ark. (2019)	MNIST	%99.4
M. R. Shamsuddin ve ark. (2019)	MNIST (binary)	%90.1
A. Salouhou (2019)	MNIST	%99.88
A. F. M. Agarap (2017) - CNN-Softmax	MNIST	%99.23
S. S. Mor ve ark. (2019)	EMNIST	%87.1
This Study	MNIST	%99.45

Table 15 Comparison of the SVM-poly model with similar studies

Reference (Year)	Dataset	Accuracy Rate
M. A. G. Pirim (2017)	MNIST	%99.64
M. A. G. Pirim (2017)	USPS	%97.47
R. Karakaya (2020)	MNIST	%90
A. F. M. Agarap (2017) - CNN-SVM	MNIST	%99.04
This Study	MNIST	%97.71

In addition to these experiments, an end-user desktop interface application has been developed for the handwriting recognition system. The application was developed using the Python programming language and its interface library, the PyQt5 library. The application is designed to make handwritten digit prediction with all algorithms used in the experiments according to selection.

7. CONCLUSION AND EVALUATION

In this study, the achievements and performances of machine learning algorithms in handwriting recognition processes were examined. An interface has been developed for test studies by sampling recognition models with different algorithms. These algorithms were selected as ANN, CNN, K-NN, Naive Bayes algorithm, SVM and decision trees, all of them were discussed in detail and experiments were carried out on the MNIST dataset for each of them. During these experiments, Python programming language and accordingly; Keras and Tensorflow libraries for MNIST dataset, CNN and ANN structures; Scikit Learn library was used for test result reports of K-NN, Naive Bayes algorithm, SVM, decision trees and models. The result reports and accuracy value obtained using the Scikit Learn library metrics were used to compare the success rates of the algorithms.

In the study, a total of 6 different algorithms were used. ANN and CNN models were the most successful when the models were compared in terms of accuracy. A success rate of 98.66% was achieved with the ANN-Adam model and 99.45% with the CNN-Adam model. Experimental results have shown that; Neural networks are more successful than other algorithms studied. However, neural network architecture and selected activation functions significantly affect the performance. In traditional classification algorithms, it is observed that the K-NN and SVM models can achieve a success rate of over 97%.

As a result almost all classification algorithms were examined with the MNIST data set, unlike similar studies in the literature. Experiments were carried out with many different parameters in all of them, and it was aimed to create the most efficient combination.

REFERENCES

[1] I. S. MacKenzie and K. Tanaka-Ishii, Text entry systems: mobility, accessibility, universality. San Francisco, Calif: Morgan Kaufmann, 2007. doi: 10.1016/B978-0-12-373591-1.X5000-1.

[2] P. Duygulu, “El Yazısı Tanıma,” in Bilişim Ansiklopedisi, Papatya Yayıncılık, 2006.

[3] M. R. Shamsuddin, S. Abdul-Rahman, and A. Mohamed, “Exploratory Analysis of MNIST Handwritten Digit for Machine Learning Modelling,” Communications in Computer and Information Science, vol. 937, pp. 134–145, 2019, doi: 10.1007/978-981-13-3441-2_11.

[4] A. F. M. Agarap, “An Architecture Combining Convolutional Neural Network (CNN) and Support Vector Machine (SVM) for Image Classification,” arXiv, pp. 5–8, 2019.

[5] M. A. Günler Pirim, “Neural Network Based Feature Extraction for Handwriting Digit Recognition,” Ankara, 2017.

[6] A. El-Sawy, M. Loey, and H. El-Bakry, “Arabic Handwritten Characters Recognition using Convolutional Neural Network,” WSEAS Transactions on Computer Research, vol. 5, pp. 11–19, 2017.

[7] A. Salouhou, “Deep Learning Approaches in Handwritting Character Recognition and Image Classification,” Istanbul, 2019.

[8] R. Karakaya, “Makine Öğrenmesi Yöntemleriyle El Yazısı Tanıma,” Sakarya, 2020.

[9] S. S. Mor, S. Solanki, S. Gupta, S. Dhingra, M. Jain, and R. Saxena, “Handwritten Text Recognition: With Deep Learning and Android,” International Journal of Engineering and Advanced Technology, vol. 8, no. 2, pp. 172–178, 2019.

[10] “THE MNIST DATABASE of handwritten digits.” url: http://yann.lecun.com/exdb/mnist/ (accessed Nov. 08, 2020).

[11] J. M. Banda, R. A. Angryk, and P. C. Martens, “Steps Toward a Large-Scale Solar Image Data Analysis to Differentiate Solar Phenomena,” Solar Physics, vol. 288, no. 1, pp. 435–462, 2013, doi: 10.1007/s11207-013-0304-x.

[12] “Scikit-Learn.” url: https://scikit-learn.org/stable/modules/model_evaluation.html#precision-recall-f-measure-metrics (accessed Dec. 20, 2020).

[13] “Thresholding Process.” url: http://www.atasoyweb.net/Otsu-Esik-Belirleme-Metodu (accessed Dec. 02, 2020).

[14] A. Vinciarelli and J. Luettin, “A New Normalization Technique for Cursive Handwritten Words,” Pattern Recognition Letters, vol. 22, no. 9, pp. 1043–1050, 2001, doi: 10.1016/S0167-8655(01)00042-3.

[15] B. Yılmaz, “Design of A Mobile Device Application with Handwriting Recognition to Make Learning Easy For Students Who Have Learning Disabilities,” Istanbul, 2014.

[16] H. H. Çelik, “Recognition of Handwritten Numerals by Using Neural Network,” Istanbul, 1999.

[17] O. A. Erdem and E. Uzun, “Turkish Times New Roman, Arial, And Handwriting Characters Recognition by Neural Network,” journal of the Faculty of Engineering and Architecture of Gazi University, vol. 20, no. 1, pp. 13–19, 2005.

[18] H. A. Şahin, “Comparison of Artificial Neural Networks and Different Optimization Methods,” Samsun, 2020.

[19] E. Öztemel, Yapay Sinir Ağlari. İstanbul: Papatya Yayıncılık, 2012. [Online]. Available: http://papatyabilim.com.tr/PDF/yapay_sinir_aglari.pdf

[20] B. Ma, X. Li, Y. Xia, and Y. Zhang, “Autonomous deep learning: A genetic DCNN designer for image classification,” Neurocomputing, vol. 379, pp. 152–161, 2020, doi: 10.1016/j.neucom.2019.10.007.

[21] E. Yalçın, “Binary-Data Multi-Criteria Recommender Systems Based on Naive Bayes Classifier,” Eskişehir, 2016.

[22] M. W. Berry and M. Browne, Eds., Lecture Notes In Data Mining. World Scientific, 2006.

[23] F. Köktürk, “Comparing Classification Success of K-Nearest Neighbor, Artifical Neural Network and Decision Trees,” Zonguldak, 2012.

[24] N. Turdaliev, “Destek Vektör Makineleri ile Otel Öneri Sistemi,” 2018.

[25] C. D. Manning, P. Raghavan, and H. Schutze, Introduction to Information Retrieval. Cambridge University Press, 2008. Accessed: Feb. 10, 2021. [Online]. Available: https://nlp.stanford.edu/IR-book/information-retrieval-book.html

[26] E. Dağdeviren, “El Yazısı Rakam Tanıma İçin Destek Vektör Makinelerinin ve Yapay Sinir Ağlarının Karşılaştırması,” 2013.

[27] G. Sakkis, “A Memory-Based Approach to Anti-Spam Filtering for Mailing Lists.”

[28] E. Taşcı and A. Onan, “The Investigation of Performance Effects of K-Nearest Neighbor Algorithm Parameters on Classification,” in Xviii. Akademi Bilişim Konferansı, 2016, p. 8.