

Handwritten Character Recognition Using Convolutional Neural Network

Chooi Shir Ley¹, Aimi Syammi Binti Ab Ghafar^{1*}

¹Department of Electrical Engineering Technology, Faculty of Engineering Technology, Universiti Tun Hussein Onn Malaysia, 84600 Pagoh, Johor, MALAYSIA

*Corresponding Author Designation

DOI: <https://doi.org/10.30880/peat.2021.02.01.058>

Received 13 January 2021; Accepted 01 March 2021; Available online 25 June 2021

Abstract: This paper applies a machine learning technique for handwritten character recognition. Handwritten recognition can recognize the handwritten character in different styles efficiently on the computer or any electronic device by obtaining the input from a touch screen, electronic pen, scanner, images, and paper documents and convert them into digital form. It has various authorities such as reading advice for bank cheques, recognizing letters and digits from form applications. The existing method from machine learning like Convolutional Neural Network (CNN) based handwritten character recognition utilized to solve this problem. The paper's objective is to develop a model base on CNN that can identify the handwritten character from the EMNIST with high accuracy. The NIST dataset consists of English alphabets and digits that use to train the neural network. NIST datasets are downloaded from the Kaggle. This paper aims to develop an algorithm based on a convolution neural network for handwritten character recognition. The objectives of this study are to improve the previous model by implementing an optimization technique that can increase the handwritten character recognition accuracy. The analyze the performance of the proposed algorithm with the test data set would discuss in this paper. It illustrates the use of convolution neural networks for generating a system that can recognize handwritten characters. In the created model, each character is described by binary values that act as input to a pure feature extraction system. The output is fed to the neural network model. CNN can apply to complete classifying words directly and character segmentation task. CNN also acts as a classifier to train the EMNIST dataset. This project used Python with TensorFlow libraries to develop a simulation of federated learning algorithms on the model. The project has made use of OpenCV to perform image processing and predicted image. The regularization parameters such as Dropout to prevent overfitting. The proposed work enhances the recognition rate of characters with an accuracy of 88.70 %.

Keywords: Convolutional Neural Network, EMNIST Dataset, Python, TensorFlow, Handwritten Recognition

1. Introduction

Handwriting character recognition is one of the interesting research realms in artificial intelligence.[1] Handwriting recognition is assumed to obtain and identify characters in handwritten documents, images, touch-screen devices, and other sources and convert them into digital form. [2] A handwriting recognition system is a tool for identifying the alphabet and number from an image. It converts the image into a machine-encoded form by applying the technique of machine learning. Machine learning is the ability to learn one of the distinctive attributes of intelligent behavior. Machine learning is a data analytics technique that teaches computers to do what comes naturally to humans. Machine learning algorithms use computational methods to “learn” information spontaneously from data without relying on a predetermined equation. The algorithms adaptively improve their achievement as the number of samples available for learning increases.

The handwriting recognition system can change cursive scripts or documents into digital characters. These digital characters can read with a computer to reduce the retrieval process. It can also make the handling of such papers more manageable.[3] This system is a new transformation way of handwriting nowadays. Using handwritten character recognition can save the resource. It would eventually lead to decreases in the manual paperwork by inserting the input text into the machine using the electronic notepad.[3] Handwriting character recognition provides the interface between humans and computers. It can improve communication between the human and machine.

There are many applications, such as postal addresses and bank cheques would need to recognize handwriting. The data concerned in this region are unconstrained. Each word can write by a different person in various styles [1] There is a strong influence on the recognition process in the source of information on the system. The handwriting recognition process can conduct over data that did not allow any other useless information than the handwritten words themselves. At most, if the character belongs to a text, semantic knowledge would introduce to the system. This project would develop a handwriting recognition system that will help recognize the letter with a high accuracy rate. This system would aid the recognizing process with minimum time and space complexity. The handwriting recognition system design is based on the biological neural network from the human brain. This system enables humans and animals to detect and model non-linear and complicated relations. It means that the handwriting recognition system can develop from the convolutional neural network (CNN). The human visual system is fundamentally committed whenever individuals read Handwriting characters, letters, words, or digits. It seems manageable whenever one is reading handwriting, but it is not as simple as people believe. Although everything is unconscious, humans still can realize what they observe based on what their brains think. A human may not recognize how challenging it is to determine to handwrite. The convolutional neural networks method is the best way to develop systems for recognizing handwriting. Neural networks help to simulate how the human brain works when reading handwritten in a more abridged form. It provides machines to coordinate and even outperform human abilities at reading handwritten.

The recognition handwriting methods include image pre-processing, segmentation, classification, feature extraction, convolution neural network. Feature extraction is the process to improves the recognition rate and misclassification. The researcher used a character extraction and edge detection algorithm for training the neural network to classify and recognize the handwritten characters. It approached the problem using convolutional neural network (CNN) because they provide better accuracy over such tasks. Using the convolution neural network to recognize the handwriting can improve the system’s performance accuracy and reduce complications. The input data coordinates a pattern that the neural network has commemorated, then the neural network attempt to determine the model accuracy

1.1 Problem Statement

The enormous amount of information important for several activity domains still store in the form of a handwritten document. For example, document forms, letters, historical documents, and exam papers. The general problem faced by the handwritten documents is hard to distinguish every character successfully. It is because the writing style of a very individual varies. Maintaining such paper documents is a very tiresome and time-consuming task. Also, after many years these handwritten paper documents might get distorted. Therefore, to solve these and identify the handwriting, all the handwritten form paperwork needs to key manually by the human on the computer for a long time ago. Until the technology is properly implemented, people have to rely on their own hands to write a text that can lead to errors. The text documents are difficult to store efficiently and access physical data. The handwritten documents need to be scanned by electronic devices to convert them into digital text and stored on the computer. Then, the handwriting recognition system explores the latest techniques that would enhance recognition accuracy. The possibility of converting handwritten data into digital text can lead people to interact more naturally with the computer. It can be training the handwriting. It can ensure the characters written are correct and reduce the error by using electronic devices. Make a handwriting recognition application for an electronic device such as a mobile phone that can recognize handwriting by applying a machine learning concept.

1.2 Project Objective

The objective of this paper is to design an authority system for handwriting character recognition using a convolutional neural network. The goals include:

- To develop a model based on Convolutional Neural Network for handwritten character recognition.
- To improve the previous model by implementing optimization technique to increase the accuracy of handwritten character recognition.
- To analyze the performance of the proposed algorithm with test data set.

1.3 Project Scope

There are different techniques of handwritten recognition which include:

- EMNIST database-set of handwritten characters and digits derived from the NIST Special Database 19
- Pre-processing
- Open CV
- Convolutional Neural Network
- Python Programming
- Training and testing of the network
- Use TensorFlow to develop simulation of federated learning algorithms on their models and data and use Kaggle to download the EMNIST dataset and run the program of python.

2. Literature Review

2.1 Machine learning

Machine learning is one of the fastest-growing areas of computer science. [4 Machine learning is a subfield of computer science developed from the pattern recognition knowledge and computational learning theory in artificial intelligence. [5] Machine learning examines the development and research of algorithms that can learn and perform data prediction. Such algorithms perform data-driven predictions or decision-making by building models from example inputs.[5] Using machine learning algorithms can repeatedly learn to improve data, describe data and predict results. Because the algorithm absorbs training data, it can generate a more accurate model based on the data. A machine learning

model is the output generated when data is using to train a machine-learning algorithm.[6] The system has a defined model with some parameters. Learning computer program execution to use training data or experience to optimize the model parameters. [7] These models are predictive. The model also uses to make predictions in the future, and it can represent information obtained from data or both. Machine learning uses these statistical theories to build mathematical models because the core task is to make inferences based on samples.

Machine learning and computational statistics are related, and they often stand out from computational statistics. Machine learning also uses in computational tasks where explicit algorithms cannot be designed and programmed.[8]

Machine learning can help humans find explications to many problems in concept, handwriting recognition, and robotics. Let take the example of recognizing faces. Recognizing face is a duty that does effortlessly. Although the posture, expression, hairstyle, and light are different. People still get to know family and friends by looking at their faces, actions, or photos every day. However, humans do it unintentionally and cannot explain how they arrange it. It is because they are not able to interpret their competence. Humans know that a face image is not only a random collection of pixels. It also includes a random collection of pixels; a face has structure. It is symmetrical. The eyes, nose, and mouth locate in specific locations. Every person's face is a pattern composed of a particular combination. By analyzing the person's facial image sample, the machine learning algorithm can capture the unique model. Then, recognizing the image by checking the model in a given image. It is an example of pattern recognition. [9]

2.2 EMNIST Dataset

The EMNIST data set is a set of handwritten characters and numbers from the NIST Special Database 19, which can convert into a 28×28 pixel image format and data set structure that directly matches the data set. The image structure is directly matched to the data set used. The training set has a total of 697932 sample images, and the testing set has a total of 116,323 uppercase and lowercase letters and numbers between 0-9, which can map to their corresponding classes. The testing set and training set appear in the list as a list. Each item in the external list represents an image. The internal list represents the intensity values of 784 pixels, with a range of 0-225. Each item in the external list represents an image. The internal list represents the intensity value of 784 pixels, and the range is 0-225. The test image and the train image have a white foreground and a black background. Both the test image and the training image are flipped horizontally and rotated 90 degrees clockwise. The Y train and Y test are arrays containing numbers in the range of 0 to 61 because the ten numbers from 0-9 to 26 consist of uppercase and lowercase letters. Their total is 62 [10]

The dataset provides two file formats. Both versions of the data set contain the same data and fully apply for convenience. The first data set is provided in Matlab format, available through Matlab and Python. There are six different splits provided in this dataset. These are EMNIST by class, EMNIST ByMerge, EMNIST Balanced, EMNIST Letters, EMNIST Digits, and EMNIST MNIST.

A summary of the dataset is provided. The dataset includes EMNIST Byclass, EMNIST ByMerge, EMNIST Balanced, EMNIST Letters, EMNIST Digits, and EMNIST MNIST. The EMNIST Balanced dataset would use in this project. The EMNIST Balanced dataset mean to address the balance issues in the class and ByMerge dataset. It reduces misclassification errors due to capital and lowercase letters and has an equal number of samples per class. This dataset is most applicable among the other datasets. The Balanced dataset has a total 131600 datasets. It has a 112800-training set and a 18800-testing set.[10]



Figure 1: EMNIST dataset

Figure 1 shows that the images of the alphabet and digits have a label of number. The label number represents the class of the dataset. The balanced dataset has a total of 47 number classes. 47 number class contains the characters from 0 to 9, uppercase letters, and lowercase letters.

2.3 Convolutional Neural Network

A convolutional neural network is a particular multilayer neural network. CNN is the most current neural network and commonly used for analyzing visual data. It is also widely used in natural language processing, image and video recognition, a recommendation system, and a pattern recognition system.[29] A convolutional neural network is a biologically inspired neural network and very good at image recognition. A neural network is a system of interconnected artificial neurons that can exchange information with each other. Its connection has a uniform numeric weight during training. Therefore, when a well-trained network uses images or patterns for recognition, it will respond accurately. Convolutional nerves have the same function as brain neural networks. Convolutional neural networks are composed of neurons with learnable weights and biases. Each neuron receives several inputs, performing weighted summation on them, passing them to activate the function, then the output response. The network consists of a multilayer feature detecting "neuron" composition. Each layer has many neurons that respond to multiple input tissues from previous layers. As shown in Figure 2, CNN will build layers so that the first layer can detect a set of original models in the input. The second layer detects model patterns, and the third layer detects models of those patterns.

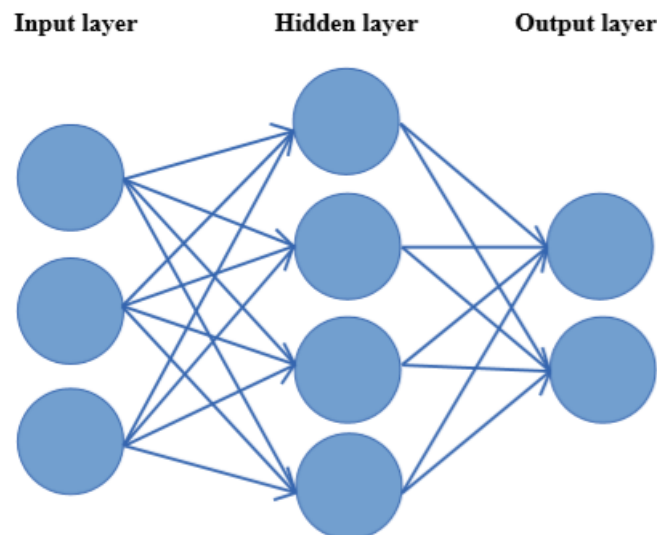


Figure 2: Neural Network

A “labeled” dataset of inputs and the input patterns marked with their dedicated output response would display the training process. The training process is the general method to determine the mediate weight and final feature neurons. Figure 3 demonstrates the training process at a block level.

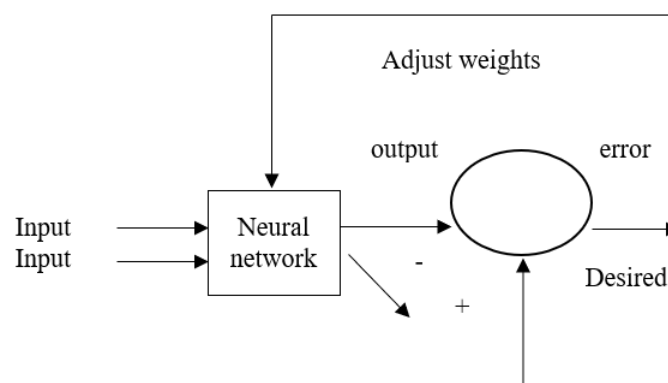


Figure 3: Training of Neural Network

CNN consists of one or more convolutional layers, usually with a sub-sampling layer. These layers are one or more fully connected layers that follow. The layers would connect the same as in standard neural networks. In a CNN, convolution layers act as a feature extractor. But they are not designed by hand. The right convolution filter kernel focuses on the training process is determined. Convolutional layers can excerpt data features as they limit the receiving field of the hidden layer to confined. CNN can use in various domains includes pattern and image recognition and video analysis. There are many reasons convolutional neural network becomes more and more important for the handwritten recognition system. [12]

In the conventional models used for image recognition, the feature extractor is a manual design. The layer fully connects for determining the classification in the training process. CNN improved network architecture may save computational complexity and memory requirements required. At the same time, it provides better enforcement for the application. CNN generally consists of three layers. CNN structure is including convolutional layers, sub-sampling layers, fully connected layers, and layers using SoftMax. These layers can implant into another layer like the SoftMax layer. Each layer would connect to the preceding layer. The softmax layer improves the accuracy of detecting the image. The creases applied to the CNN are not always the same as needed. The difference is that the language that

can recognize handwriting will influence the number of layers. The number of layers would use in the future. When another layer is interposed CNN, handwriting recognition accuracy will be higher than that softmax inserted layer CNN in the model.

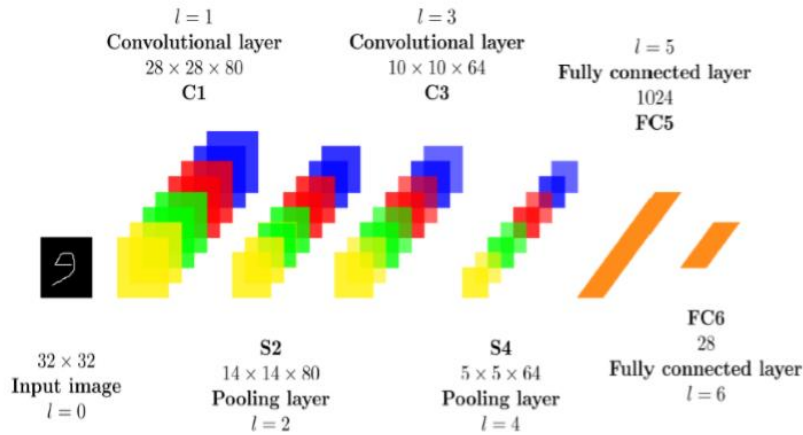


Figure 4: The example of CNN architecture

2.4 Model Explanation

A simple sequence convolutional neural network converts one activation into another through a differentiable function. Figure 4 shows the architecture of the proposed CNN model. The training model procedure in CNN would pre-process the image, like, resizing the image and normalization before the data feed to the model. The LeNet-5 structure consists of two layers pooled and convolution, convolution, and flat layer. Then, two fully connected layers and finally softmax classifier. Network architecture established insensitivity to local conversion. The same features in different parts of the input are detected. Different features on the graph at the same position in the output unit of the feature vector can regard as the same region. Due to weight sharing, neurons in successive layers extract more and more complex features. It will reduce the number of idle parameters in the system. CNN produces an output vector in each layer. Each dimension detection in the output vector is a feature from a different part of the feature map of the previous layer.

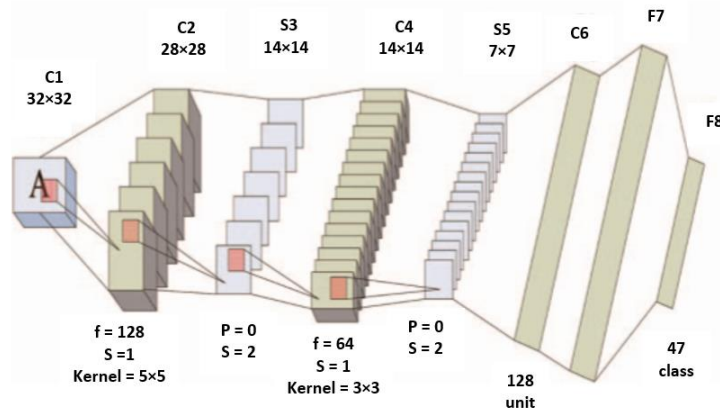


Figure 5: The Basic architecture of LeNet-5 Handwritten English Alphabet

Layer S1 (input layer) is an image with a size of 32x32. The C2 layer is activated by a convolutional layer with ReLu (rectified linear unit). It is the first convolutional layer of the CNN architecture. The image as a pre-layer of size $n \times n = 32 \times 32$ is input. Convolution filter size ($f \times f$) of 5×5 , the padding (p) is 0 (around all sides of the image), stride (s) is equal to 1, the number of filter 6. After the convolution operation, the resulting output is feature map 6 @ 28×28 , where 6 is the number of feature

maps. It is also equal to the number of filters used, and 28 comes from the formula $\left(\frac{n+2p-f}{s} = 1\right) = \frac{(32+2 \times 0 - 5)}{1} + 1 = 28$. Then, complete ReLU activation in each function diagram. Compared with SIGMOID, ReLU is the first choice because Sigmoid is a slow learning function. The objective of using the convolutional layer double is to extract more traits to improve model accuracy.

Layer S3 is a sub-sampling layer. The upper layer acquires the size of the input layer from 6 @ 28 × 28. The combined size is 2×2; padding is 0, and the stride is equal to 2. After performing this operation, the max-pooling would feature an independent acquisition feature map upper layer. So it has the same number of elements in the feature map, 14 and from the same formula. $\left(\frac{n+2p-f}{s} = 1\right) = \frac{(28+2 \times 0 - 2)}{2} + 1 = 14$. The resulting image size will reduce to 14×14×6.

Layer C4 is the second convolution layer with ReLU activation function. This layer gets the input of size 14×14×6 from the previous layer. The filter size is 3×3, whereas padding is 0, the stride is equal 1. In this layer, only 10 out of 16 feature maps are connected to 6 feature maps of previous layer 16@14×14. The connection wat between layer S3 and layer S4 take much importance for the feature formation.

Layer S5 is the second subsampling layer. This layer gets the input of size 16@14×14 from the previous layer. The subsampling layer is 2×2 and stride of 2. Padding is equal to 0. After thus subsampling operation, the output will be reduced to 16@7×7

Layer C6 is the flattened convolutional layer with 128 feature maps of size 1×1. Each of the 128 units in C6 is connected to all the nodes (16@7×7) in the fifth layer. It has the ReLU activation function.

Layer F7 is a fully connected layer that contains 128 units and fully connects to layer C6. It will remove some unwanted characteristics of neurons. These neurons make the model larger and increase training time. It also helps avoid overfitting.

Finally, the output layer by the layer of F8 is a Euclidean RBF unit and is fully connected to the layer F7. Having a final layer 47 neurons, and 0-9 correspond to the uppercase and lowercase alphabet letters A-Z. It generates a vector of size 47. The 47 classes vector in each number corresponds to a category score. It has a softmax activation function for final output. Activate neurons are similar, but this time SOFTMAX use as the activation function. SOFTMAX is the same logical classification function as the SIGMOID function. Like the SIGMOID function, it uses to calculate the probability of different categories. The only difference is that SOFTMAX is used for multi-class classification, while SIGMOID use for binary assortment.

3. Model Approach

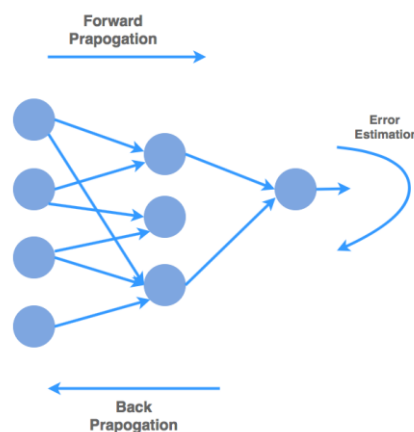


Figure 6: Forward Propagation and Backward Propagation of Neural Network

The first phase of forwarding propagation happens when the neural network is exposing to the training data. Its data across and shift the entire neural network prediction (marked). In other words, the input data sent over the neural network. So that all the neurons convert applied information received from the upper layer neurons. When the data is calculating that transforms all layers and all of its neurons, it will reach the last layer of the label prediction input example. The loss function uses to estimate loss or error. It can also use to compare and measure whether the prediction results are related to the correct output. The weight of neuron interconnection will gradually adjust until a better prediction obtain when training the model. Once calculate the loss of the model, this information would propagate backward. Hence, the process calls backpropagation. Loss information propagates to all the neurons in the hidden layer that contribute instantly to the output starting from the output layer. However, based on every neuron's contribution to the original data, compared with the hidden neurons, the hidden layer neurons only receive a part of the loss signal. This process would repeat layer by layer until all network neurons have accepted a loss signal describing their relevant enrichment to the total loss.

3.1 Back-propagation

The backpropagation was designed by extending the Widrow-Hoff learning rule to multilayer networks and nonlinear differential transfer functions. The input vector and the corresponding target vector are used to train the network until it can roughly determine its purpose. It can be a particular input vector and output vector associated with the input vector you define or classify. A network with bias, sigmoid layer, and linear output layer can determine roughly any function with a finite number of discontinuities.[13]

3.2 Loss Function – Cross Entropy

The loss function is a method of assessing the ability of the model data set model. If the prediction deviates from the actual target value, the loss function will output a higher number. Otherwise, it will output a smaller number. Because the problem is multi-class classification, and therefore will be used as a cross-entropy loss function. If the prediction model labels the input category of the actual target low probability, high value is output. The equation is described:[13]

The equation is described:

$$\text{loss}(x, \text{class}) = -\log\left(\frac{\exp(x[\text{class}])}{\sum_j \exp(x[j])}\right) = -x[\text{class}] + \log\left(\sum_j \exp(x[j])\right) \quad \text{Eq. 1}$$

3.3 Euclidean Distance

Euclidean distance describes the correct distance between two positions in n-dimensional space. In supervised learning, it can use to estimate the similarity between two representations with n features.[14]

$$d_{12} = \sqrt{\sum_{j=1}^n (x_{1j} - x_{2j})^2} \quad \text{Eq. 2}$$

Where d_{12} refers to the similarity between first and second samples.

3.4 Average Variance Similarity

In the statistical description, variance is used to calculate the difference between each sample and the mean of all sample. It defined as:[14]

$$\sigma^2 = \frac{\sum_{i=1}^m (x_i - \mu)^2}{m} \quad \text{Eq. 3}$$

Where σ^2 is variance x_i is the i^{th} sample, μ is mean of all sample, m is the number of samples,

3.5 Model Parameterization

It can increase the number of cycles, add more neurons in a layer or add more layers. However, an increase in accuracy can also increase the execution time of the learning process during training. One can use the summary() technique to check whether the number of parameters has increased. The execution time is significantly higher, even decreasing the number of epochs. The accuracy reaches 94.00 % when using this model. If the system continues to increase to 20 epochs, the accuracy rate may attain 96.00 %.[13]

3.6 Epochs

Epoch tells the developers in the process of training of all training data passed through the neural network. Real evidence is to increase the number of periods unto the accuracy indicator with validation data starts to decrease, even if training data accuracy increases continue. It happens when the developer detects potential overfitting. Using batch size can improve computational efficiency. Its value will depend on the hardware available to the user. The epoch is the prosperity of the network forward and backward. Generally, it is propitious to set its value higher. If they are satisfied with the trained neural network in a particular state (selected period) convergence, it is possible to reduce the time.[13]

3.7 Batch Size

The training data can be divided into small batches to pass the network. In Keras, batch_size is a parameter indicating the size of these batches. The size of these batches would update with the fit() method in the training iteration. The optimal size will depend on the memory capacity of the computer used for the calculation. Use batch size to improve calculation efficiency. Its value will depend on the hardware available to the user.[13]

3.8 Learning Rate

The gradient vector has direction and magnitude. Gradient descent algorithm of the gradient multiplied by a scalar (called the learning rate), sometimes referred to as steps, to determine the next point. The learning rate is a very sensitive parameter that can promote model convergence. Unless someone calls more powerful technologies (such as batch standardization), or to find the best value will be an empirical process [13]

3.9 Adaptive Moment Estimation Optimizer

Adaptive moment estimation (Adam) is an optimization algorithm that can substitute the classic stochastic gradient descent process to modernize network weights based on training data. In addition to saving the last exponential decay gradient of the average exponent square. Adam and momentum also maintained an average gradient of exponential decay.[15]

3.10 Adamax

Adamax is an extension of Adam optimizer, the optimizer based on the infinite norm. The gradient vector has direction and magnitude. The gradient descent algorithm multiplies the gradient amount by a scalar (called the learning rate) (sometimes called the step size) to determine the next point.

4. Methodology

The handwritten character system will consist of several phases.

- Image Acquisition and Digitization
- Pre-processing
- Segmentation

- Feature extraction
- Training and Testing the model
- Classification and recognition

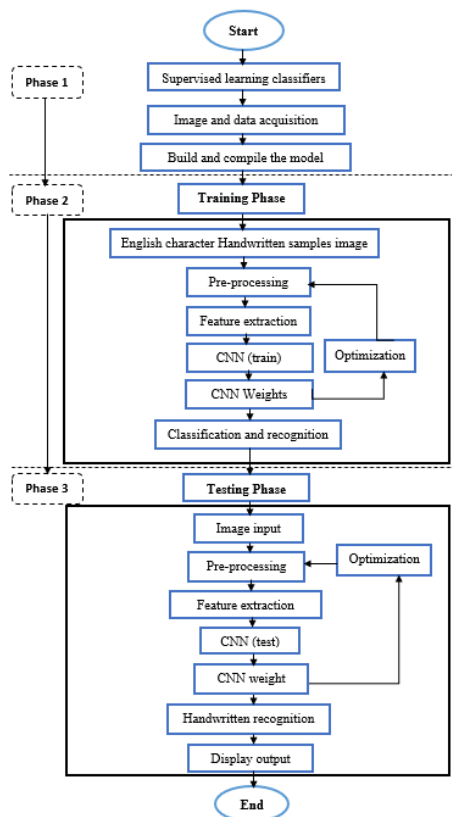


Figure 7: Flowchart of project development

4.1 Image Acquisition

The image acquisition step involves acquiring an input image containing handwriting. In this case, the image should be in a specific format, such as CSV and PNG. The data and images of handwritten character can obtain from the NIST database. EMNIST database can download from the Kaggle. The Kaggle database has stored several EMNIST dataset, such as PNG, CSV and TXT. The image for predicting the output can acquire through a digital camera, scanner, or any other suitable input device.

4.2 Pre-processing

Pre-processing of the input data acquired pretreatment. The Pre-processing improves the input data quality to make the input sample more suitable for the next stage of the recognition system. The pre-processing has the functions of noise removing, binarization, edge detection, segmentation, dilation, and fill in. Preprocessing can normalize the strokes. It can also eliminate changes that reduce accuracy. The pre-processing mainly deals with various deformation. For instance, sawtooth size, missing pen point during movement, the space around the curved and uneven.

- Noise removing: Uniform filtering and non-uniform filtering methods can use to eliminate unnecessary or unwanted patterns. Due to the influence of noise, there may be separate line segments, such as large gaps between lines. Therefore, it is significant to remove all these errors. It can make the information retrieval in the best way. There are many noises in the image. One type of additive noise is called "salt and pepper noise." The black and white dots scattered on the image usually look like salt and pepper and can be observed in almost all documents. Noise reduction techniques split into two categories, namely filtering and morphological operations.

- **Filtering:** Filtering intends to eliminate noise and reduce stray points usually caused by uneven writing surfaces and a low sampling rate of data acquisition equipment and various design and spatial frequency domain filter for filtering
- **Binarization:** Binarization is the process of converting character images into binary 0 and 1 forms. In this case, all types of characters will interpret as grayscale pictures. After converting the grayscale image into a binary matrix, all image characters will be captured vertically. Similarly, the gray image may work to reduce the overall complexity of the system.
- **Edge detection:** Edges define object boundaries, so these object boundaries are useful in phase lie recognition and segmentation. Edge detection can be improved for all exception filter and reduce the unnecessary amount of information. On the other hand, it can retain all the significant features of the image. There are many methods of performing edge detection, usually divided into gradient or Laplacian operators. Observe the highest and smallest values of the first derivative of the image, and use the gradient method to detect edges. Then, search for zero crossings in the second derivative to identify image edges.
- **Thresholding:** Color or gray images will be represented as numbers 0 or 1 to reduce storage requirements and increase processing speed. It is a binary image, meaning that every value above the selected threshold is 1 and 0. When the image is converting to grayscale, the handwritten characters are darker than their background. Threshold processing means can separate darker and lighter areas from the image.
- **Dilation and fill in:** The document may be skewing during scanning. There are several commonly used methods to detect page tilt. One is to identify the connected components and find the average angle of the centroids that connect them. Skew should eliminate as this will reduce the accuracy of the document. After that, with the help of calculating the skew angle, to horizontal the skewed line.

4.3 Segmentation

Segmentation is dividing the input text data image into several lines and then into individual characters. It removes unnecessary parts from the data image. Segmentation is subdivided into two segments, including external and internal subdivision. The external segment is sub-divided into paragraphs, lines, and words. On the other hand, internal segmentation is to segment a single character from the input text data. In this project, the technology uses words, lines, and character segmentation. It usually performs by segmenting a single character from the word picture. Besides, content is processing in a tree-like way. In the initial scheme, the line histogram would use to segment the line. Then, at each level, the characters are retrieved through a histogram technique and finally retrieved.

4.4 Feature Extraction

Feature extraction is a significant substantive data retrieved from the original data content method. The essential materialist data is an accurate and efficient embodiment of the character image. Feature extraction points to recognize the extraction of patterns that is most important for the classification. Extracted from the original data set of features to maximize character recognition rate (including minimal elements) is called feature extraction. The feature extraction process is to collect the difference. Gradient-based feature extraction is extracting features from necessary and useful data. These features are very significant for any training system that performs pattern recognition tasks. The capacities used in feature extraction include indexing and labeling, boxing and cropping, and reshaping and resizing.

- **Indexing and Labeling:** Through this process, different characters in the index image are marked. Thus, it helps in the character classification in the image and make feature extraction of character extraction.
- **Boxing and Cropping:** It is the process of creating a border around the characters represented in the image. It can make cropping of characters easier. After boxing, cut out characters to store them as input variables for recognition.

- Reshaping and Resizing: Reshape the image to change the size of the received characters to the aspired contour. Resize can reduce the character size to the minimum level.

4.5 Classification and Recognition

After feature extraction, the next step is classification. The classification process is to make decisions for a new input character like this to suit the classes or appearance. The stage of categorizing characters is recognized and assigned labels. Good performance depends on the classification and feature extraction selection. When the input image uses as the input sample for a handwritten character recognition system, all significant features will be retrieved and input into a trained classifier, such as an artificial neural network. Preparation of the input feature is compared with the stored patterns to find a matching category of the input image. It would complete with the help of the classifier. The correct label training data is needed to analyze the test samples. Then, train the convolutional neural network model in Python. Perform training with weight decay and exit on each fully connected layer.

4.6 Training and Testing model

Here is the step by step building Convolutional Neural Network:

- Step 1: Split the dataset into the training set and test set. Usually, the training set is larger than the test set. The desired output must normalize to the [0:1] range because the sigmoid function only returns values in this range. The input mode does not have to be standardized.
- Step 2: Initialize all weights (including all deviations) to a small random value in the range of [-1:+1]. It determines the starting point of the gradient descent method on the error surface, and its position may be critical to the convergence of the network.
- Step 3: Building and compile the model using the CNN
- Step 4: The first input pattern of the training set is propagated forward from the input layer to the hidden layers to the output layer where each neuron sums the weighted inputs, passes them to the nonlinear. Then, pass the weighted sum to the next Layers of neurons.
- Step 5: Calculate the difference between the actual output of each output neuron and its corresponding desired output. It is an error associated with each output neuron.
- Step 6: By using the "back-propagation learning" rule, this error is propagated back through each connection, thereby determining the amount by which each weight must change to reduce the error in the output layer.
- Step 8: Correct each weight by its weight update.
- Step 9: Present and forward the next input pattern. Repeat steps 3-7 until a particular stopping criterion reach. For example, the error drops below a predetermined value.
- Step 10: Perform the testing for testing pattern and store the weight
- Step 11: Check the updated knowledge base with all previous knowledge bases.
- Step 12: Stop with accuracy percentage.

The complete training mode displayed to the network at one time constitutes the training era. After the training phase is over, the trained network would test using a new invisible pattern from the test data set. Use the weight available during training. Then, propagate the model forward and determine the output layer error. If the performance is good enough, one can use the neural network. If it is not, you must retrain with the same model and parameters. Otherwise, some changes must make for adjusting.

5 Implementation

5.1 TensorFlow

TensorFlow is an incredible flow of data in the open-source machine learning library of the Google Brain team in 2015. It strives to simplify the scope of use, and it covers a wide range of digital and neurological issues (in different spaces). Fundamentally, TensorFlow is a low-level tool for

mathematical entanglement. Its target audience is those who realize that they are building an exploratory learning structure, interacting with it, and turning it into a running program. In most cases, it could regard as a programming framework where calculations can perform like graphics. The nodes in the figure represent mathematical activities. Its borders enclose multi-dimensional information clusters (tensors) related to it.

5.2 Python 3.7

Python is widely used for machine learning tasks. It is a high-level programming language. It only served to highlight the introduction of the code. Its language structure so that users can use fewer lines of code to express their ideas. Python is a programming language that helps users to work quickly and coordinate frameworks more effectively. Users can get many modules from the Python community to realize machine learning handwritten character recognition because Python has many module collections.

5.3 Anaconda

Anaconda is a free and open-source code for Python and R programming, used for logic diagrams such as information science, AI applications, and large-scale information preparation, with foresight. Like Anaconda Navigator, Conda Package, and Virtual Environment Director, Anaconda comes with more than 1,400 packages, so there is no need to figure out how free to import each library. Anaconda Navigator is integrated into grants graphical UI (GUI), allowing customers to schedule and manage application package Conda, channels, and conditions without occupying command-line guidance.

5.4 Jupyter Notebook

Jupyter Notebook is an open-source Web application that users can use to generate and accord documents that contain real-time code equation, visual effects, and text. Jupyter Notebook can perform the conversion, data visualization, numerical simulation, data cleansing, statistical modeling, and machine learning algorithm. There are many distributions of the python language. There are many distributions of the python language. Therefore, Jupyter Notebook can use to perform python programming.

6 Result and Discussion

The entire experiment was performed on a desktop computer with Intel® Core (TM) I7 10510U CPU @ 1.80 GHz, 2.30 GHz, and Keras with TensorFlow on the backend on a window environment.

Table 1: Parameter specification in All-Conv model

Layer	Layer Type	Output Shape	Number of Parameters
1	Conv 2D	(None, 28, 28, 128)	3328
2	Max Pooling 2D	(None, 14, 14, 128)	0
3	Conv 2D	(None, 14, 14, 64)	73792
4	Max Pooling 2D	(None, 7, 7, 64)	0
5	Flatten	(None, 3136)	0
6	Dense	(None, 128)	40153
7	Dropout	(None, 128)	0
8	Dense	(None, 47)	6063
Total Parameters: 484719			Trainable Parameters: 484719
			Non-Trainable Parameters: 0

The handwritten character recognition has been implemented using the Python 3.7 in Jupyter notebook. Epoch refers to a complete traversal through a specific data set, while batch processing is the total number of training examples in one traversal. Model accuracy increase with epoch. Training time can reduce by adopting a GPU. A model trained using our EMNIST dataset, which has been pre-processed and optimized for training. We have categorized and distributed training and test labels and

flattened the training and test arrays. So, they can insert into the model be easily. In this project, for each case, the considered data set was split into the training set and test set, respectively containing 80.00 % and 20.00 % of the data set. The training set uses to train the neural network, and the test set uses to measure its effectiveness. Initially, the model accuracy increased exponentially, and then there was a small but steady increase. At the end of the training, we achieved an accuracy of 87.86 %, which is a bit inferior to the maximum accuracy reached using the EMNIST dataset. Use the test data provided by EMNIST to evaluate the model. Our model can make predictions with 87.90 % accuracy. Table 2 shows the model accuracy result when using the Adam Optimizer.

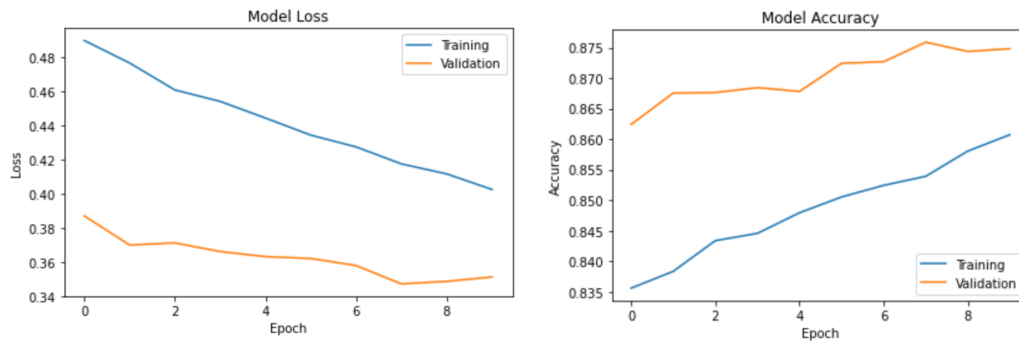


Figure 8: The Graph of Model Accuracy and Model Loss When Using SGD

Figure 8 shows the result when using the SGD optimizer instead of Adam Optimizer. The gradient descent backpropagation method with momentum and adaptive learning rate, log sigmoid transfer functions used to test the training accuracy. The learning rate was equal to 0.1, while the momentum is equal to 0.5. In helping the model find the best learning rate process, one uses gradual decay. Gradual decay is one of the most popular forms of learning rate annealing. After a certain number of training periods, the learning rate will decrease by a certain percentage (hence the term "decay"). The result shows that the model in the graph is overfitting. The validation loss is 0.3722, whereas the validation accuracy is 87.00 %. Overfitting is a common cause of an unsatisfying learning algorithm performance. The intuition of overfitting is that the network has memorized the training data well. But it cannot guarantee that it can use on invisible data, which is why the accuracy of training and verification differs. If the learning model using the input has too many complex functions, SGD may be a good fit for the training set. However, it may not be able to generalize predictions in new examples. Besides, if the classifier summarizes the hypothesis into an overly simplified form, an insufficient fitting increases the training set error and error in new prediction examples. Therefore, the SGD optimizer was not suitable for classifying handwritten characters. It is because the handwritten character recognition system had a complex function. Also, the handwritten characters had many different styles and patterns.

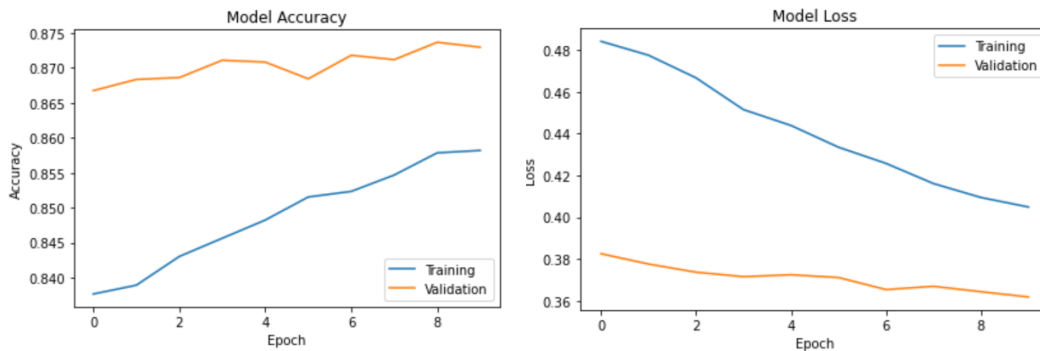


Figure 9: The Graph of Model Accuracy and Model Loss when Adamax optimizer

Figure 9 show the model accuracy and model loss results when using the Adamax optimizer function. The model looks underfitting. The underfitting concept was the same as the overfitting. The

only difference between underfitting and overfitting was, underfitting is high bias, whereas overfitting is high variance. The highest accuracy obtained was when we experiment with optimizer Adamax. The model accuracy and model loss were 88.65 % and 0.3618 when using Adamax optimizer. The accuracy provided by other optimizers is very close to that of Adamax to some extent, such as, Adam, Adadelata, Adamax, SGD, but Adamax also greatly reduces the training time. Adamax optimizer is very suitable for training large models. The optimizer uses for this project was Adam optimizer. It was because the EMNIST dataset did not have any outliers. Adamax is used for the that data that is traditionally noisy in terms of gradient updates. Adamax more appropriate when recognizing the outlier dataset.

Table 2: Summary of the experiment

Batch size	No. of epochs	Accuracy
512	10	87.63
512	15	88.13
80	15	88.29

The test accuracy is 88.15 %, indicating that the model is well-trained and can predict. The size of the training set will affect the accuracy, and the accuracy will increase as the number of data increases. The more data in the training set, the smaller the impact of training error and test error. Ultimately, they can improve model accuracy. A large number of neurons in the hidden layer helps to improve accuracy. However, this may have some upper limits, depending on the data used. Besides, a higher number of neurons in the hidden layer will significantly increase the training time. That means the higher the model accuracy, the increase the training time needed.

Table 3: Summary of the experiment 2

Database	Classes	No. Training	No. Testing	Total	Accuracy
Balanced	47	112,800	18,800	131,600	86.17
MNIST	10	60,000	10,000	70,000	97.46
Letters	37	88,800	14,800	103,600	89.96

Table 3 shows the accuracy result from the various dataset. EMNIST balanced dataset, MNIST dataset, and EMNIST dataset were the dataset that can carry out the data validation among the six datasets within the NIST Special database. The result stated that the validation accuracy from MNIST was higher than the Balanced dataset and Letters dataset. The main reason for this happening was, MNIST dataset contains only ten classes. There were only digits (0-9) inside the data. The Balanced and Letter datasets provide more image samples and output classes and even provides more complex and diverse classification tasks. Therefore, the training process was less complicated than the Balanced dataset and Letter dataset. The training time needed to validate the model was also less than the other two datasets.

By further preprocessing the data set and adding new hyperparameters to the Keras model, this accuracy can improve future research work. The smaller the loss, the better the model, unless the model overfitting to the training data. We have calculated loss on our train and test data. In the case of neural networks, the loss is usually negative log-likelihood. It is the sum of squared residuals of classification and regression respectively. Then naturally, our objective was to improve the accuracy value concerning the model's parameters. As described in the model section, our classifier contains two convolutions and two Maxpooling layers and consists of a dropout layer and a dense layer with 512 units.

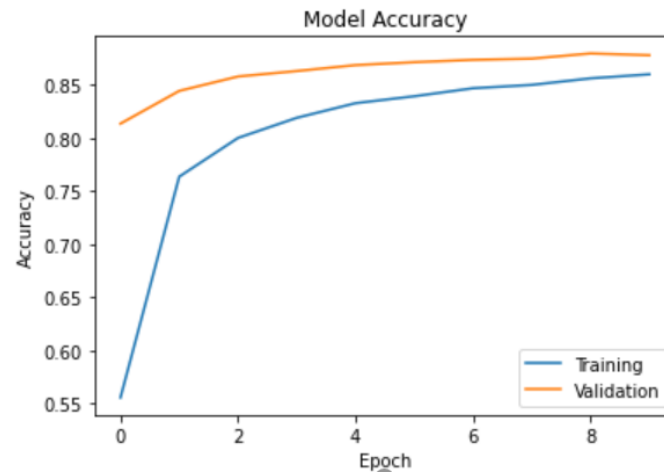








Figure 10: The Graph of Model Accuracy

The training accuracy of the handwritten character recognition model is 87.47 %. The test accuracy of the handwritten character recognition model is 87.63 %. The accuracy of training and validation increases with the increase of the times.

Table 4: Predicted Output

Character	Input Image	Output
English alphabet (uppercase)		
English alphabet (lowercase)		
Digit		

Implemented the proposed method in Python3-7, insert the input images into the CNN classifier. The selected input was uppercase letters, lowercase letters, and numbers. The CNN classifier modeled using the EMNIST data set can predict the input characters. The experimental results show in Table 4.

7. Conclusion

The project has proposed and developed a scheme for recognizing the handwritten character. The different character handwriting styles obtain from the EMNIST dataset were trained and tested in the project. The project proposed the Handwritten Character Recognition using machine learning methods had been implemented. The project developed a handwritten character recognition algorithm based on a convolutional neural network. The project had successfully improved the previous model by implementing the optimization techniques, thereby increasing the handwritten character recognition accuracy. This system successfully analyzed the performance of the algorithm and the test data set. Compared to other research methods, the method convolution neural network focuses on which classifier works better by improving the classification model accuracy by more than 88.00 %. Using Keras as the backend and Tensorflow as the software, a CNN model with more layers can give accuracy

more closely to 90.00 %. But it would improve in the future. The use of neural networks in conjunction with the best-trying layers can provide higher tolerance to noise and more accurate results. Use backpropagation to train the feedforward model of the neural network to classify and recognize characters. Facts have proved that the combination of feature extraction and normalization can produce higher character recognition accuracy.

The EMNIST dataset was acted as the input sample to train the CNN model and tested with different optimizers. Finally, the Adam optimizer decided to select for this project. It gives high accuracy not only in each period of the training data but also in the test data. The NIST data set that contains six data sets increases the challenges of using only the MNIST data set. Although the structure of the EMNIST data set is similar to that of MNIST, it provides more image samples and output categories as well as more complex and diverse classification tasks. But, if the EMNIST data set not use as the backbone for the project, it is almost impossible to achieve this accuracy. Convolution by more and more hidden layer neurons, the results can be more accurate and can eliminate the requirement to type. Character recognition is a good model problem for learning neural networks. It can provide the best way to develop more advanced deep learning methods. In the future, we plan to evolve a real-time handwritten character recognition system.

Acknowledgement

The authors would like to thank the Faculty of Engineering Technology, Universiti Tun Hussein Onn Malaysia for its support.

References

- [1] S. Attigeri, "Neural Network based Handwritten Character Recognition system," *Int. J. Eng. Comput. Sci.*, vol. 7, no. 03, pp. 23761–23768, 2018, doi: 10.18535/ijecs/v7i3.18.
- [2] P. Bojja, N. Sai, S. Teja, G. K. Pandala, and S. D. L. R. Sharma, "Handwritten Text Recognition using Machine Learning Techniques in Application of NLP," *Int. J. Innov. Technol. Explor. Eng.*, vol. 9, no. 2, pp. 1394–1397, 2019, doi: 10.35940/ijitee.a4748.129219.
- [3] A. Sharma, S. Khare, and S. Chavan, "A Review on Handwritten Character Recognition 1 1,2,3," vol. 8491, pp. 71–75, 2017.
- [4] S. Ben-david, *Understanding Machine Learning : From Theory to Algorithms*. 2014.
- [5] Y. Baştanlar and M. Özuysal, "Introduction to machine learning," *Methods Mol. Biol.*, vol. 1107, pp. 105–128, 2014, doi: 10.1007/978-1-62703-748-8_7.
- [6] S. R. Patel and J. Jha, "Handwritten character recognition using machine learning approach - A survey," *Int. Conf. Electr. Electron. Signals, Commun. Optim. EESCO 2015*, 2015, doi: 10.1109/EESCO.2015.7253978.
- [7] E. Alpaydin, *INTRODUCTION TO Machine Learning second edition*, Second edi. London, England: The MIT Press Cambridge, Massachusetts London, England, 2014.
- [8] C. Grosan and A. Abraham, *Machine Learning*, vol. 17. 2011.
- [9] P. Dönmez, "Introduction to Machine Learning, 2nd ed., by Ethem Alpaydın. Cambridge, MA: The MIT Press 2010. ISBN: 978-0-262-01243-0. \$54/£ 39.95 + 584 pages.," *Nat. Lang. Eng.*, vol. 19, no. 2, pp. 285–288, 2013, doi: 10.1017/s1351324912000290.
- [10] S. S. Mor, S. Solanki, S. Gupta, S. Dhingra, M. Jain, and R. Saxena, "Handwritten text recognition: With deep learning and android," *Int. J. Eng. Adv. Technol.*, vol. 8, no. 2, pp. 172–178, 2019.

- [11] R. Vaidya, D. Trivedi, S. Satra, and P. M. Pimpale, "Handwritten Character Recognition Using Deep-Learning," *Proc. Int. Conf. Inven. Commun. Comput. Technol. ICICCT 2018*, no. Icicct, pp. 772–775, 2018, doi: 10.1109/ICICCT.2018.8473291.
- [12] A. Yuan *et al.*, "A Review on Handwritten Character Recognition 1 1,2,3," *Int. J. Innov. Technol. Explor. Eng.*, vol. 8, no. 2, pp. 71–75, 2019, doi: 10.1053/otsm.
- [13] H. P. Indiran, "Handwritten Character Recognition using Convolutional Neural Networks in Python with Keras," *Asian J. Conver. Technol.*, vol. 5, no. 3, pp. 123–131, 2016.
- [14] J. Zou, J. Zhang, and L. Wang, "Handwritten Chinese Character Recognition by Convolutional Neural Network and Similarity Ranking," 2019, [Online]. Available: <http://arxiv.org/abs/1908.11550>.
- [15] M. A. Hossain and M. M. Ali, "Recognition of Handwritten Digit using Convolutional Neural Network (CNN)," *Glob. J. Comput. Sci. Technol.*, vol. 19, no. 2, pp. 27–33, 2019, doi: 10.34257/gjcstdvol19is2pg27.