e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:04/Issue:06/June-2022 Impact Factor- 6.752 www.irjmets.com

HAND WRITTEN DIGIT CLASSIFICATION USING CNN

Anusha R*1, Dr. T Vijaya Kumar M M*2

*1Dept. Of MCA, Bangalore Institute Of Technology, Bengaluru, India.

*2Dept. Of MCA, HOD, Dept. Of MCA, Bangalore Institute Of Technology, Bengaluru, India.

ABSTRACT

Deep Learning is a subset of Machine Learning in AI that has network capable of learning unsupervised from data that is unstructured or unlabelled. Now a days Deep learning can be employed to wide range of fields like medical, engineering etc. Deep Learning is known as deep neural network or deep neural learning. In deep learning CNN (Convolutional Neural Network) is a centre of spectacular advanced that mixes ANN and up to date deep learning. CNN is extensively used in pattern recognition, video analysis, natural language processing, spam detection, image classification and hand written digit classification. We are demonstration this influence by applying CNN with different hidden layer to MNIST data set (Modified National Institute of Standards and Technology). We also observe the variations of accuracies of the network for various numbers of hidden layers and epochs. We use Adam Optimizer of The Stochastic Gradient Descent Algorithm and Loss Function i.e. Back Propagation to train the data set and Forward Algorithm for testing the data set.

Keywords: Convolutional Neural Network (CNN), Handwritten Digit Recognition, MNIST Data Set, Hidden Layer And Epochs, Adam Optimizer, The Stochastic Gradient Descent And Back Propagation Algorithm, Optimizer Loss Function.

I. INTRODUCTION

Deep Convolutional Neural Network achieves human level accuracies in many different fields. CNN is used for visual imagery analysis like pattern recognition, sentence classification, speech and face recognition, hand written digit recognition etc. CNN was inspired by the Mammalian Visual System which is a biological model. CNN is an extend form of ANN (Artificial Neural Network) implemented to nonlinear transformation of the input and create output. ANN compress of input layer, output layer and some hidden layer between them. CNN was designed by Yann LeCun in 1998.

The first CNN was built in 1988 called as LeNet. LeNet was used for character recognition tasks like reading zip codes and digits. Some well-known CNN architectures are GoogleLeNet (22 layers), Alex Net(8layer), VGG (16-19 layer). CNN has seven layer and which are of three dimensions, with input layer, output layer and some hidden layer between them. CNN is the most approval tool for hand written digit classification because it uses a smaller number of hidden layers. CNN is must faster than ANN has is three dimensions each neuron in a layer is fully connected to a small localize region of previous layer. We are designing CNN to classify the had written digit of the MNIST data set. This data set consists of number of hand written digit of different people in the form of images. We can use various data set like, CENPARMI, CEDAR, MNIST etc. The images in the data set are of 28×28 pixels. The CNN take the input as an image and perform the CNN architecture with several hidden layers. The output will be in the form of numbers (0-9). Finally, the accuracies are calculated to the training data and testing data using algorithms comparing with different number

II. LITERATURE SURVEY

CNN plays important role in many fields like image recognition, speech recognition, text categorization etc. CNN had become a default tool for hand written digit classification. This is the one of the interesting topics for all the research people nowadays. We can also see large number of paper and article based on this topic and algorithm. Deep neural network algorithm like CNN gives the highest accuracy using TensorFlow and keras. The default dataset is MNIST for hand written digit classification, which contains 70,000 images of 28×28 pixel written by different people

PREVIOUS CONTRIBUTIONS

Md. Abu Bakr Siddique had published a paper on hand written digit classification using CNN in python with TensorFlow in the year 2018. The author of the article describes the CNN with several hidden layers and used the Stochastic Gradient Descent, Backpropagation algorithm for training for data set MNIST. He uses minimal of

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:04/Issue:06/June-2022 Impact Factor- 6.752 www.irjmets.com

15 epochs to get the highest accuracy of 99% with the minimal loss. CNN model was of 7 layers i.e., input, Conv1, pool1, Conv2, pool2 with 2 dropouts. He had also plotted the graphs for different epochs using the matplotlib plot and scatter function.

Muhammad Azeem had published a paper an efficient and improved scheme for handwritten digit recognition based on convolutional neural network in the year 2019. The two primary steps are character recognition and feature extraction. The fundamental steps in character recognition are segmentation, feature extraction and classification. The CNN framework contains 7 layer which include feature extraction and feature classification. In feature extraction every layer of network collets the output from its previous layer and forward the current output to next layer. In feature classification the predicted output is generated. The author had got the highest accuracy of 97% in this article to his model to increase the accuracy and decrease the error rate large data set are recommended by the author.

III. METHODOLOGY

The Basic Model of CNN

CNN is a deep artificial neural network, like standard ANN CNN has seven layered architectures. The input layer, output layer and multiple hidden layers between them. The general hidden layers of CNN are convolution layer, pooling layer and fully connected layer.

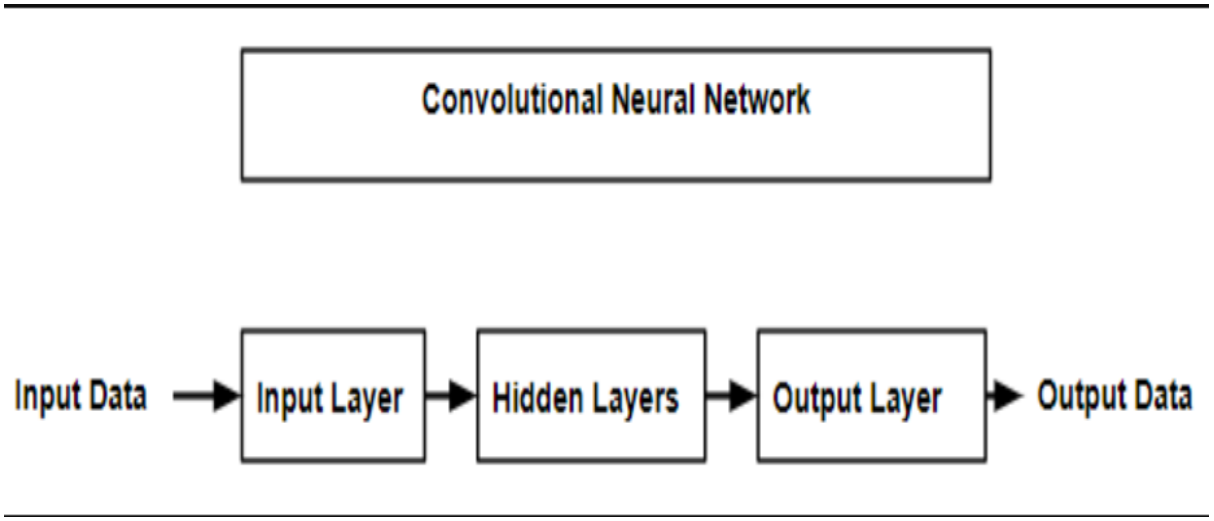


Fig 1: Block Diagram of CNN

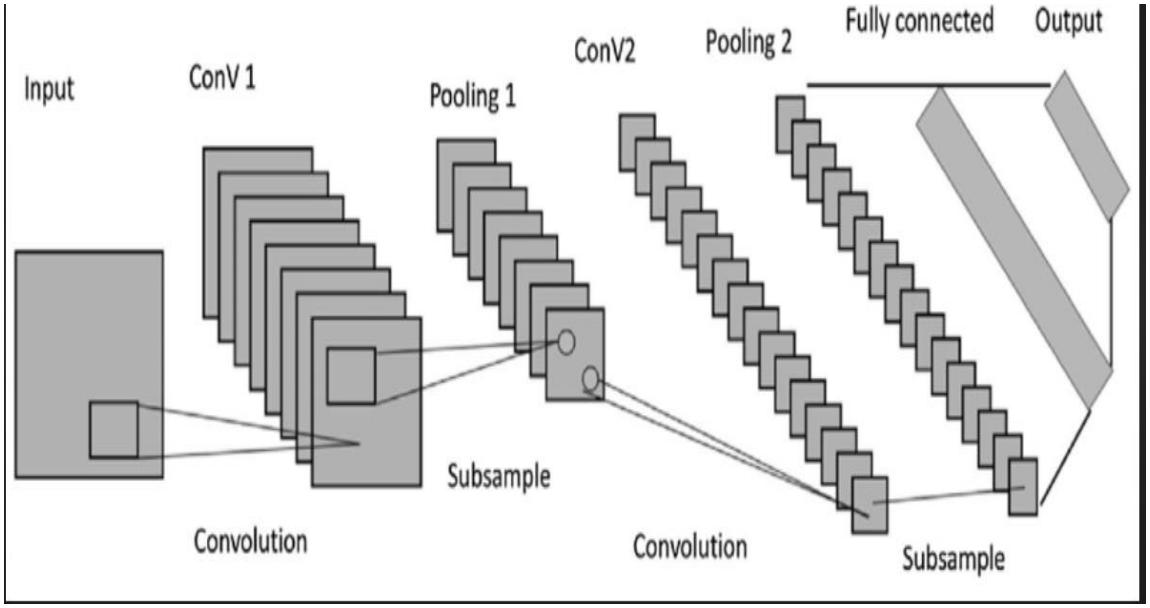


Fig 2: Typical CNN Architecture

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:04/Issue:06/June-2022 Impact Factor- 6.752 www.irjmets.com

The Typical CNN architecture is as shown above, it consists of the multiple hidden layers as shown in the above figure.

Input layer

The input contains the image which is in the form of pixels, that means the network contains 784 neurons as the input. The input pixels are grayscale with a value '0' for a white pixel and '1' for the black pixel. This layer sort all the input data and describes all the height, width and number of channels of the input image. After the classification of the input images, it will pass the data to the first hidden layer.

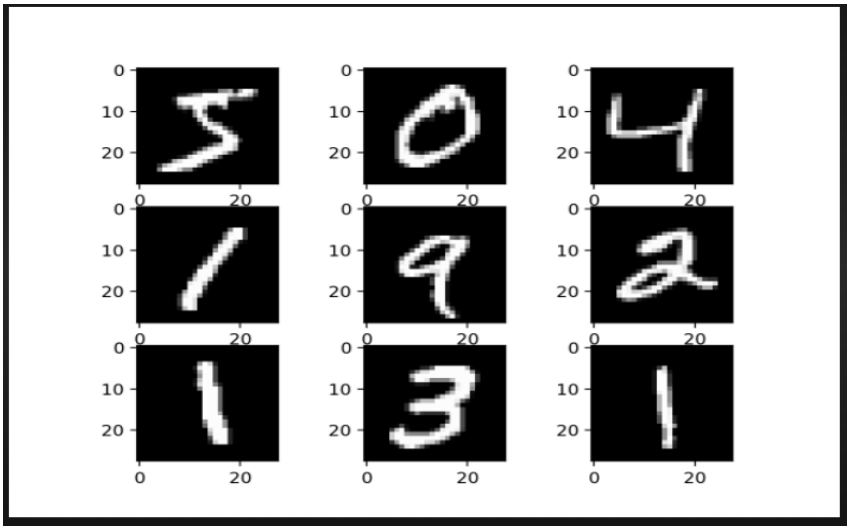


Fig 3: Input data

Convolutional Layer

The first hidden layer is convolutional layer, and it is a heart of the CNN architecture. This layer is responsible for feature extraction form the input data. The input neurons of input layer are convoluted with a filter and returns the output. Our network layer has a priori determined fixed weights for the different feature maps and serves to detect the key feature of the image. The convolution layer consists of multiple features map with learn kernels and ReLU (rectified Liner Unit). The main contribution of convolutional layer is receptive filed, Stride, dilation and padding. The concept of padding was introduced to increase the accuracy of the model, it also controls shrinking of the output of the CN layer. The size of the kernels defines the weight. To introduce the nonlinearity in CNN an activation function is applied to input data after the convolutional layer, which will improve the performance of the model.

Feature Map= input data × kernels

Pooling Layer

The pooling layer is the second hidden layer in CNN model, it's also called as data reduction layer. In this layer the shrinkage of volume of data takes place, i.e., it reduces the output information form the convolutional layer and reduces the number of parameters and computational complexity of the model, this will control the overfitting of the model. There is different type of pooling – Max pooling, Min pooling, average pooling, L2 pooling. The most commonly used pooling technique is max pooling and average pooling. In the case of max pooling the data will be converted to kernel of 2×2 dimensions.

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:04/Issue:06/June-2022 Impact Factor- 6.752 www.irjmets.com

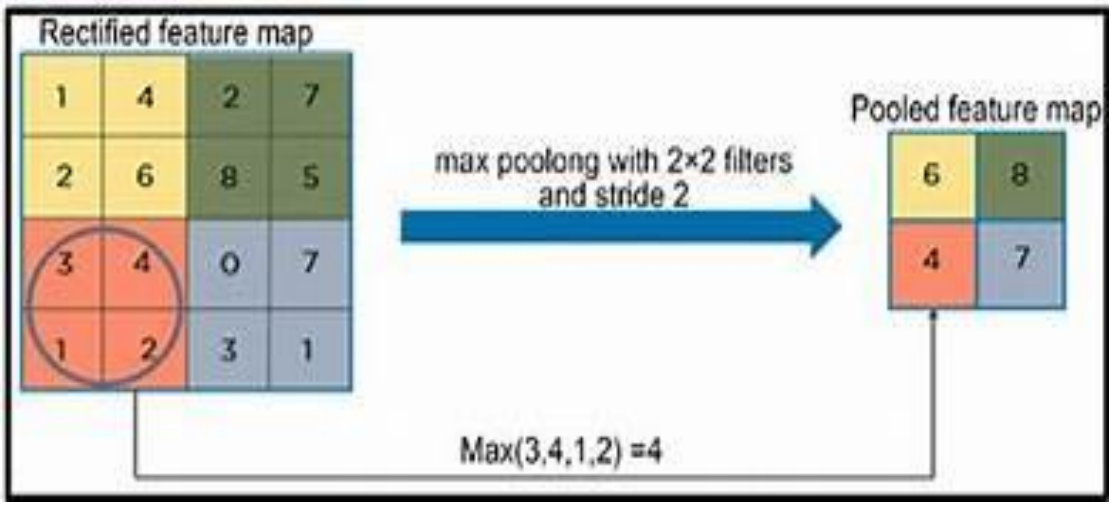


Fig 4: Max pooling

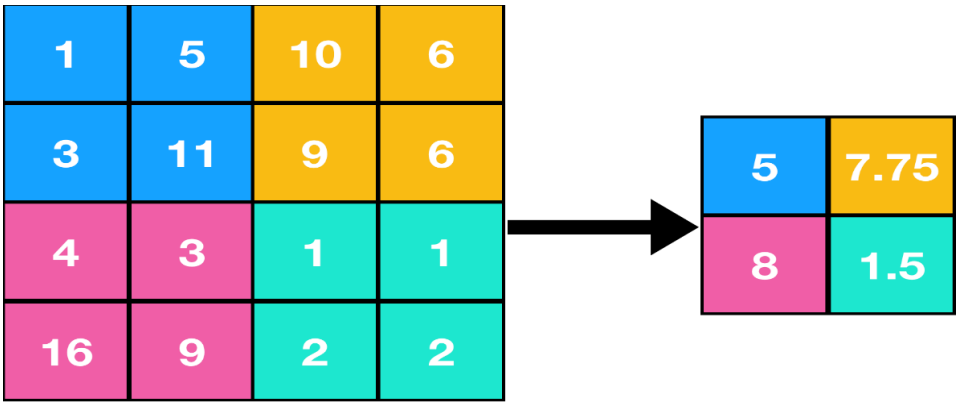


Fig 5: Average pooling

Fully Connected layer

It's the next hidden layer in CNN model. This layer consists of n number of neurons which are connected together. These neurons are predicted class numbers. The main task of this layer is to combine all the features from convolutional and pooling layer to produce a probable class store for classification of the input images. Some activation functions like

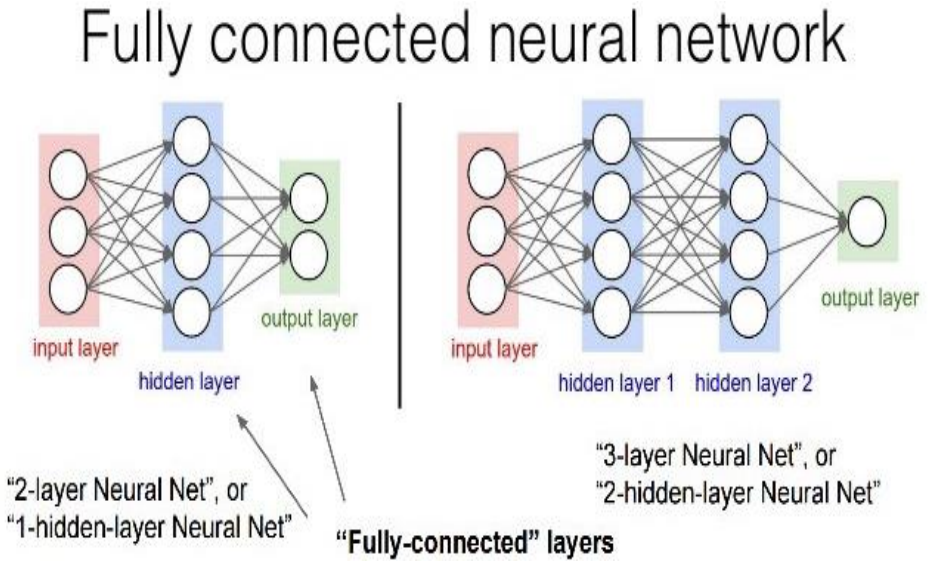


Figure 6

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:04/Issue:06/June-2022 Impact Factor- 6.752 www.irjmets.com

Rectified linear unit (ReLU)

The ReLU is an activation function, its is one of the few milestones in deep learning. While creating the CNN or any network model the ReLU is the default activation function. This ReLU helps to overcome the gradient problem, and help the model to learn better and faster. The general formula is,

f (x) = max (0, x)

it will return 0 if the input is negative, and if the input is any positive value x it will return the same vale x. Thus, we can say that the output has the range from 0 to infinity.

SoftMax Activation Function

SoftMax is an activation layer used in the output layer, we need the functions which takes whatever the values and transform into probability distribution, this activation function is used. It will work grate for classification problems, especially if you're dealing with multi-classification problem, and will report back the confident score for each class. The formula used is

$$\frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

IV. IMPLEMENTATION

MNIST Data SET

We are using MNIST data set as a input in our model, this is an standard data set which is used while dealing with the neural network which works on the input type images. This data ha the 70,000 of data set which or of images. These images are of pixels 28×28 size and grayscale. The 60,000 data out of 70,000 is used as training data and the rest of it i.e., 10,000 is used for testing the data. The data in this data set is a hand written digit of differ people, some are students, professor, scientists etc. The data set will look as shown below.

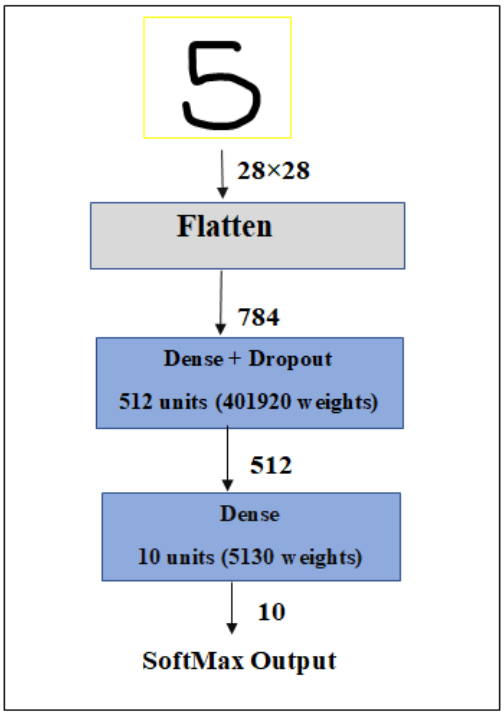
Fig 7: MNIST Data Set

Convolutional neural network model for hand written digit classification.

The first we have to import the libraries and load the data set, TensorFlow and Keras allow us to import and download the MNIST dataset (Modified National Institute of Standards and Technology) directly from their API. The MNIST database contains 60,000 training images, and 10,000 testing images. This dataset is a 28×28 divided in terms of pixels i.e., 0 to 27 rows and 0 to 27 columns which will be having 28×28 different features.

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:04/Issue:06/June-2022 Impact Factor- 6.752 www.irjmets.com

We cannot analyse all the data so we will pre-process the data. We will Reshaping the array to 4-dims so that it can work with the Keras API (greyscale image). Need to normalize our data as it is always required in neural network models, by dividing the RGB codes to 255. So initially will be converting data to float as we are dividing. Now we are using the Sequential Model which will help to build the model layer by layer. Convolutional 2D is the first layer of CNN model. It will extract all the features from the input image, we are using conv2D because we are dealing with the 2D input images. Next, we apply maxpooling2D to the data, Max pooling is a pooling operation that selects the maximum element from the region of the feature map covered by the filter. Thus, the output after max-pooling layer would be a feature map containing the most prominent features of the previous feature map. Next, Flattening involves transforming the entire pooled feature map matrix into a single column which is then fed to the neural network for processing. The 28×28 image pixels data is converted into 784 units of singe column data. The Flattening passes the single column data to Dense Layer. Dense Layer, A linear operation in which every input is connected to every output by a weight. It connects neurons in one layer to neurons in another layer. It is used to classify images between different category by training using ReLU activation function. So, the 784 units are converted into 512 units. A Simple Way to Prevent Neural Networks from Overfitting is by using Dropout. By removing all overfittings 512units are turned into 10 units. SoftMax Layer is the last layer of CNN. It resides at the end of FC layer, SoftMax is for multi-classification.



Now we compile and train the model by using three parameters like optimizer, loss and metrics. We will be using 'Adam' as our optimizer which is of The Stochastic Gradient Descent Algorithm. Adam is generally a good optimizer to use for many cases. The Adam optimizer adjusts the learning rate throughout training. Loss function i.e., Back Propagation that can be used to estimate the loss of the model so that the weights can be updated to reduce the loss on the next evaluation. We use fit method then training starts. With a simple evaluate function to know the accuracy.

V. RESULTS

```
In [91]: import tensorflow as tf
         import matplotlib.pyplot as plt

In [92]: dataset=tf.keras.datasets.mnist.load_data()
         (x_train, y_train), (x_test, y_test)=dataset
         x_train.shape

Out[92]: (60000, 28, 28)
```

Fig 8: Train and test data

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:04/Issue:06/June-2022 Impact Factor- 6.752 www.irjmets.com

```
In [96]: from tensorflow.keras.models import Sequential
         from tensorflow.keras.layers import Dense, Conv2D, Dropout, Flatten, MaxPooling2D
         model = Sequential()
         model.add(Conv2D(28, kernel_size=(4,4), input_shape=input_shape))
         model.add(MaxPooling2D(pool_size=(2, 2)))
         model.add(Flatten())
         model.add(Dense(128, activation=tf.nn.relu))
         model.add(Dropout(0.2))
         model.add(Dense(10, activation=tf.nn.softmax))
```

Fig 9: CNN Model

```
In [97]: model.compile(optimizer='adam',
                       loss='sparse_categorical_crossentropy',
                       metrics=['accuracy'])
         model.fit(x=x_train,y=y_train, epochs=2)

         Epoch 1/2
         1875/1875 [==============================] - 17s 9ms/step - loss: 0.1925 - accuracy: 0.9422
         Epoch 2/2
         1875/1875 [==============================] - 17s 9ms/step - loss: 0.0807 - accuracy: 0.9754

Out[97]: <keras.callbacks.History at 0x1d42d2b1070>
```

Fig 10: Compile the mode

```
In [98]: val_loss, val_acc =model.evaluate(x_test,y_test)

         313/313 [==============================] - 1s 4ms/step - loss: 0.0574 - accuracy: 0.9799
```

Fig 11: Evaluate the model

```
In [99]: pred=model.predict(x_test)
         pred

         313/313 [==============================] - 1s 4ms/step

Out[99]: array([[4.42566304e-08, 1.19944508e-08, 1.09397897e-05, ...,
                 9.99978304e-01, 3.26422395e-07, 6.37775838e-06],
                [4.15072435e-07, 2.26339922e-04, 9.99750197e-01, ...,
                 2.19775760e-11, 1.70566491e-05, 1.00394667e-11],
                [2.50875132e-06, 9.99828219e-01, 1.65044257e-05, ...,
                 4.74683075e-05, 2.03966374e-05, 4.32541100e-07],
                ...,
                [9.40889510e-13, 1.26895570e-08, 2.51890953e-10, ...,
                 3.14730755e-06, 2.50536186e-06, 8.42372447e-06],
                [7.75535284e-07, 1.08828635e-08, 4.64897099e-08, ...,
                 1.15600924e-08, 5.19204128e-04, 1.42168460e-07],
                [5.29613658e-07, 5.86104443e-10, 5.94477910e-07, ...,
                 1.24206990e-11, 7.84040921e-08, 9.21535914e-09]], dtype=float32)
]
```

Fig 12: Prediction Model

```
In [100]: import numpy as np

In [101]: print(np.argmax(pred[15]))

          5

In [102]: plt.imshow(x_test[15] , cmap=plt.cm.viridis)
          plt.show()
```

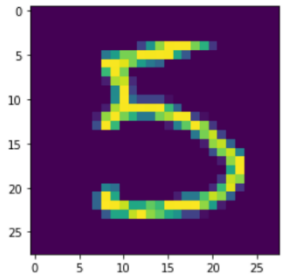


Fig 13: Prediction

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:04/Issue:06/June-2022 Impact Factor- 6.752 www.irjmets.com

VI. CONCLUSION

Hand Written Digit classification with the good accuracy and prediction is much required for this real world. Hence, it is essential that the CNN model should identify any kind of hand written digit in minimal time. CNN model is the prominent tool used for many Machines Learning models, because it is multilayer algorithm. CNN algorithm automatically extracts the important feature of the input image without the human supervision. Our model provides the accuracy of 97% to the minimal loos function of 0.0574. Prediction model is also built to check whether our model is predicting the proper output. CNN Algorithm yet again proved to be very efficient to any classification done to the dataset of images when compare to other Algorithm as ANN and SNN. Our model is only used to detect the digit form 0-9 which are written in hand. As the future extension we can find Hand Written Digit Classification of any language, Handwriting Analysis, and Image recognition etc., using the CNN algorithm.

VII. REFERENCE

[1] Recognition of Handwritten Digit using Convolutional Neural Network in Python with TensorFlow and Comparison of Performance for Various Hidden Layers
[2] https://ieeexplore.ieee.org/abstract/document/8975496/
[3] Recognition of Handwritten Digit using Convolutional Neural Network (CNN)
[4] https://computerresearch.org/index.php/computer/article/view/1823
[5] An Effective and Improved CNN-ELM Classifier for Handwritten Digits Recognition and Classification https://www.mdpi.com/864522
[6] Study and Observation of the Variations of Accuracies for Handwritten Digits Recognition with Various Hidden Layers and Epochs using Convolutional Neural Network
[7] https://ieeexplore.ieee.org/abstract/document/8628078/
[8] Handwritten Hindi Digits Recognition Using Convolutional Neural Network with RMSprop Optimization
[9] https://ieeexplore.ieee.org/abstract/document/8662969/
[10] Improved Handwritten Digit Recognition Using Convolutional Neural Networks (CNN)
[11] https://www.mdpi.com/741864
[12] Handwritten digits recognition with decision tree classification: a machine learning approach
[13] https://www.academia.edu/download/63236157/18004-37834-1-PB20200507-108150-qgc5qj.pdf