International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 08 Issue: 04 | Apr 2021 www.irjet.net p-ISSN: 2395-0072

# Handwritten Digit Recognizer using Deep Neural Network

**G. Sri Nikesh[1], T. Amruth[2], B. Ajay Reddy[3], Kota Rajashekhar[4], N. Jaya Surya[5]**

*[6]Professor Nahida Nazir (guide), Lovely Professional University, India*

---

**ABSTRACT:** *The paper will describe the best approach to get more than 90% accuracy in the field of Handwritten Digit Recognizer (HDR). There have been plenty of research done in the field of HDR but still it is an open problem as we are still lacking in getting the best accuracy. In this paper, the offline handwritten digit recognition will be done using the algorithm Deep Neural Network (DNN) using Convolutional Neural Network (ConvNet). The purpose is to develop the software with a very high accuracy rate and with minimal time and space complexity and also optimal.*

***Keywords— Handwritten Character Recognition, Deep Neural Network, Convolutional Neural Network***

## 1. INTRODUCTION

Handwriting recognition is a computer function that can receive and interpret easy-to-understand handwriting input from various concrete sources. Pictures of manually written content can be perceived "offline" on paper by optical examining (OCR) or intelligent word recognition then again, the motion of the pen tip can for example be recorded "online" via the surface of a pc screen with a pen. This is a simple job since there are usually more hints.

The problem remains albeit the variable of language is kept constant, by categorizing handwriting by similar languages. each piece of writing is different, though there are some similarities which will be classified together. The task for a knowledge scientist, during this case, is to group the digits involved in writing an identical language in categories that might indicate related groups of an equivalent numbers or characters. The optimization of this process is of utmost importance, together wants to acknowledge handwriting because it is written. The classifiers that are utilized in these cases vary in their methods and recognition speeds, alongside the accuracy of recognition.

Neural networks (NN) have proved useful during a vast range of fields, classification being one among the foremost basic. NNs show excellent performance for the classification of images by extracting features from images and making predictions supported those features. many various sorts of NN scans be used for classification, i.e., convolutional neural networks (CNN), recurrent neural networks (RNN), etc. Each type has its architecture, but some properties remain an equivalent. As each layer consists of neurons, each neuron is assigned weight and activation functions. First, we'll check out the architecture of every neural network.

Handwriting recognition plays an important role in storing and retrieving input and handwriting information. A large number of historical papers exist in physical form. This includes genealogical information, old family records, written manuscripts, personal diaries, and many other pieces of the shared past. Continuous verification of this information could damage the original document and the actual data. Handwriting recognition enables translation that converts this information into an easy-to-read electronic format.

Most modern tablet computers and hybrid notebooks have an active pen and display that users can use to write and draw on the screen. It is very useful for users to recognize handwritten text, understand its meaning, and save it as digital text that can be used by computers and word processing applications. This is the first step towards realizing handwritten number recognition. Benchmark The benchmark model is selected from one of the competing Kaggle Digit Recognizer cores and used for comparison. Benchmark accuracy rating of the model is 0.872.

## 2. LITERATURE REVIEW

Since 2009, the repetitive neural network and deep feedforward neural network developed by the Jürgen Schmidhuber research group at the AI Lab IDSIA in Switzerland have won a few worldwide writing contests. Bidirectional and multidimensional long term short term memory (LSTM), particularly Alex graves. Circa 2009, International Conference on Document Analysis and Recognition won three competitions in the field of handwriting recognition without first having to speak three different languages (French, Arabic, and Persian). A new GPU-based feedforward network deep learning method developed by Dan Ciresan of IDSIA and his colleagues won the ICDAR 2011 offline Chinese handwriting recognition competition. In the famous MNIST problem with handwritten digits by Yann LeCun and colleagues at NYU, the neural network is also the first artificial pattern recognition engine to achieve competitive human performance.

CNN is taking part in a vital function in numerous areas Take, for example, image processing. It does have an impact on various domains. CNN is used for defect identification and ordering, in nanotechnology, such as semiconductor aggregation [18]. Digit recognition by

hand has made great strides. a trouble of pastime amongst experts in the field. This topic is the focus of extensive number of publications and documents that are being distributed these days. It has been demonstrated in study that deep learning algorithms such as multi-layer, In comparison to the most commonly used classification models like SVM and KNN, CNN using Keras with Theano and TensorFlow offers the highest precision. and RFC. Due to its most elevated precision, Convolutional Neural Network (CNN) is being utilized for an enormous scope in photo arrangement, video investigation, and so on Numerous specialists are attempting to interpret sentiment in a statement, often in the processing of natural language and the identification of sentiment, CNN is used, by shifting various boundaries [19]. Scientists are dealing with this issue to diminish the error rate however much as could reasonably be expected, in one test using 3-NN, trained and tested on MNIST, the error in recognition of writing using hand was 1.19 percent. [20].

It is being utilized in convalescing sentences in a picture. A few analysts are attempting to concoct new techniques to stay away from the drawbacks of the usual convolutional layers. Ncfm (No combination of function maps) is a tool which can be used to improve MNIST dataset execution [21]. It has a precision of 99.81 percent and is used for large-scale data, in general. With each passing year, new CNN frameworks emerge with numerous sorts of examination, Scientists are making a decent attempt to reduce the number of errors. Error costs are calculated using MNIST datasets and CIFAR [22]. CNN is often used to clean up blurry images. As a result, another model has proposed the usage of the MNIST dataset. This strategy arrives at a precision 98 percent, with losses ranging from 0.1 percent to 8.5 percent [23]. 98 percent, with losses ranging from 0.1 percent to 8.5 percent [23]. CNN's traffic sign discernment approach has been suggested in Germany. It suggested a quicker in general execution with 99.65% precision [24]. loss characteristic was once planned, this is particularly true for diaphanous 1D and 2D CNN. In this case, the precisions had been 93% and 91% separately [20, 25].

# 3. METHOD

## 3.1. Data collection

The MNIST data base [4] of written by hand digits There are sixty thousand samples in the training collection and ten thousand samples in the training set. It is just a subsection of NIST's larger collection. The digits are centred on a fixed-size picture that has already been size-normalized. The original black and white (bi-level) pics from NIST have been dimension in a 20x20 pixel container, they were standardized to a stable form while keeping their aspect ratio intact. As a consequence of the anti-aliasing strategy employed by the normalisation algorithm, the following photos contain grey levels. The images were placed in a 28x28 photograph by computing the pixels' centre of mass and converting the image, to work this point in the middle of the photograph.

## 3.2. Loading the dataset

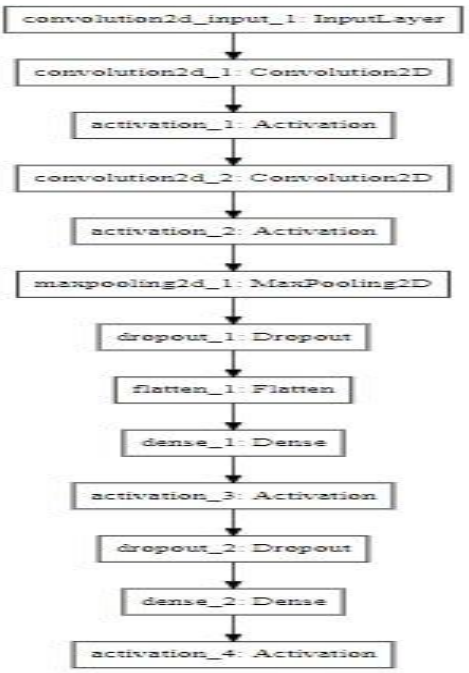
Then initiate this project by importing the required python libraries and variables. The dataset then will be downloaded and loaded as training and testing set. using Keras, a high-level neural networks python library

## 3.3. Pre-processing

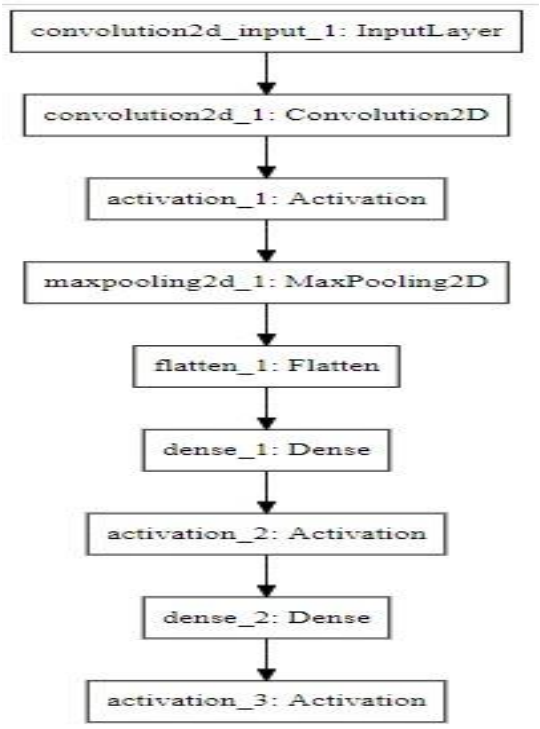
Then Visualize an image of an integer using matplotlib. As Keras can use both TensorFlow or Theano. While they each signify photo in distinctive format. Theano uses the format (no. of color channels, no. of rows, no. of cols) and TensorFlow uses (no. of rows, no. of cols, no. of color channels). A wrapper is created which is needed when switching backends. The MNIST dataset contains grayscale images where the color channel value of keras varies from 0 to 255. In order to reduce the computational load and training difficulty, map the values from 0 - 255 to 0 - 1 by dividing each pixel values (x_train, x_test) by 255. Then target labels (y_train,y_test) are in the form of numerical integers(0-9),convert them to binary form in order for the neural network to perform mapping from input to output correctly and efficiently.

## 3.4. Using the DNN model

Then build the model of Deep Neural Network consequently compile and visualize the model.



**Refined model**




and this is the initial model and layers above will be discussed in the next section.

## 3.5. Training and Testing

Then use the variables to train the model with pixel-values of the handwritten image (X_train) as features and its respective label(Y_train) as target. Then test the DNN Model built above with mnist test dataset(X_test,Y_test). By which we can attain the accuracy of the model. Factors involving compilation of model are cross entropy as loss function, SGD as optimizer and accuracy as metrics.

## 3.6. Refinement

If the accuracy is not high enough, (initial model is measured to be 95.82%). Then alter the architecture of the network. This is the how the refinement of the model is performed, two more convolutional layer and one more extra dense (fully connected) layer are added, two dropout layer is added to ensure that overfitting does not happen, Using Adam optimizer instead of SGD (Stochastic Gradient Descent) to update the weights efficiently.

## 3.7. Model Testing

On Real Data Then compile and visualize this model too (Picture is above) then he will train and test the DNN model (as done in the previous step). Consequently, the accuracy score for refined model is measured to be 0.9906 i.e., 99.06%. This is the main part, which is testing this refined model again, but with real data. By loading images of a handwritten digit and convert it to the required format (Format the images in order to match with MNIST Dataset i.e., 28x28 pixel grayscale image) and then predict what digit is written in it. That conversion to required format is little critical. It will be done using OpenCV.

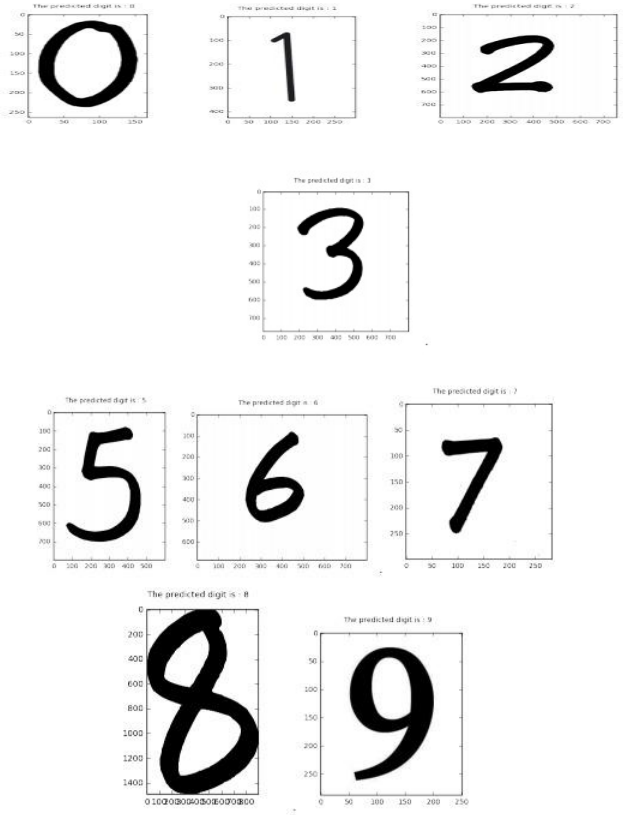
## 3.8. Final Evaluation

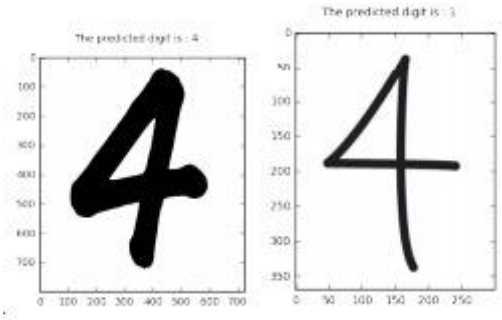
The final architecture and hyperparameters is chosen because they performed best out of the other models tried. The robustness of the final model is verified by using different images of digits other than MNIST dataset as mentioned above, the following observations are based on the results of the test

· The final model has predicted the digits of 10 out 11 images correctly.

· This model had problem in predicting digits that have thin stokes like the digit 4 with thin stroke in the test cases.

The observations of test confirms that our model is reliable enough to perform well on real data.





This was the error due to the type of stoke, and the above results are some of the successful predictions

## 4. TECHNIQUES EXECUTED

Given that the problem is a supervised learning problem and more specifically a classification the problem, there are a lot of algorithms available for training a classifier to learn from the data. The algorithm chosen for this project is Deep Neural Network (DNN) using CNN

**4.1 Convolutional neural network** (CNN, or ConvNet)**:** may be a feed-forward artificial neural network during which the availability design between its neurons is enlivened by the organization of the animal visual cortex [5]. Since it models animal visual perception, it can be applied to visual recognition tasks such as handwritten digit recognition from images. The below are the common types of layers to build ConvNet or DNN architectures: 

- **Convolutional Layer -** The entry to a convolutional layer is a m x m x r photo, where m is the photo's height and width, and r is the channel quantity, for example, r equals 3 in a photo of type RGB. The convolutional layer will have k filters of size n x n x q, where n is less than the photo's dimension and q can be either comparable in light of the fact that the channel total, r or smaller, which varies by kernel. The filter capacity affects the structure which is already closely associated, in which all of them are combined with the photo to develop k feature vectors each having the capacity of m-n+1. [6]

- **ReLU Layer -** An activation function of type elementwise is applied by The Rectified Linear Unit, like max (0, x) thresholding at zero.
- Pooling Layer - play out a down sampling activity along the spatial measurements (width, height)
- Fully-Connected or Dense Layer - computes scores (between 0-9 digits). Likewise with conventional Neural Networks and in light of the fact that the name infers, every neuron during this layer will be associated with all or any of the numbers within the previous volume. [7]
- Dropout is a neural net overfitting mitigation strategy. The main concept is to lose modules at an irregularity (along with their connections) [8]
- Flatten - Flattens the input. Does not affect the batch size. Consider the example below.[9]

If the convolutional layer the output shape is (None,64,32,32). Now when flatten is applied to the layer, the output shape is flattened by multiplying 64*32*32 = 655

### 4.2. SoftMax

SoftMax layer [16] is typically the final output layer in a neural network that performs multiclass classification (for example: object recognition). the Softmax classifier produces a more logical result and moreover has a probabilistic elucidation, something we will explore momentarily. The feature visual mapping stays consistent in the Softmax classifier; however, we now perceive these values as un - normalized log probabilities for all the classes and substitute the hinge loss with a cross-entropy error that has a technique given below:

$$L_i = -\log\left(\frac{e^{f_i}}{\sum_j e^{f_j}}\right) \text{ or equivalently } L_i = -f_{yi} + \log\sum_j e^{f_j}$$

where (Fj) signifies the j-th variable of the array of class scores f. The complete loss for the dataset is indeed the mean of (Li) across all training sets plus a regularisation function R(W)

The formula is $f_j(z) = \frac{e^{z_j}}{\sum_k e^{z_k}}$ called **the SoftMax.**

**4.3. The cross-entropy** (loss function) [16] between a estimated and a true distribution (q&p) is defined as:

$$H(p,q) = -\sum_x p(x)\log\, q(x)$$

**4.4. Stochastic Gradient Descend (SGD):** Optimizer in Gradient Descent (GD) optimization, we compute the cost gradient based on the complete training set; hence, we sometimes also call it batch GD. In case of very large datasets, using GD can be quite costly since we are only taking a single step for one pass over the training set -- thus, the bigger the preparation set, the slower our algorithm refreshes the loads and the more it might take until it meets to the global expense minimum (note that the SSE cost function is convex). In

Stochastic Gradient Descent (SGD; sometimes also referred to as iterative or on-line GD), we don't accumulate the weight updates as we see below for GD:

- for one or more epochs:
  - for each weight $j$
    - $w_j := w + \Delta w_j$, where: $\Delta w_j = \eta \sum_i (\text{target}^{(i)} - \text{output}^{(i)}) x_j^{(i)}$

Instead, we update the weights after each training sample:

- for one or more epochs, or until approx. cost minimum is reached:
  - for training sample $i$:
    - for each weight $j$
      - $w_j := w + \Delta w_j$, where: $\Delta w_j = \eta(\text{target}^{(i)} - \text{output}^{(i)}) x_j^{(i)}$

Here, the **term "stochastic"** comes from the very fact that the gradient supported one training sample may be a "stochastic approximation" of the "true" cost gradient. Due to its stochastic nature, the trail towards the worldwide cost minimum isn't "direct" as in GD, .but may go "zig-zag" if we are visualizing the cost surface in a 2D space. [17]

### 4.5. Adaptive Moment Estimation Optimizer (Adam)

Adam [15] figures versatile learning rates for every boundary. As well as putting away a dramatically decaying avg of past squared gradients (vt) like Adadelta and RMSprop, Adam likewise keeps a dramatically decaying normal of past gradients (mt), almost like momentum:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t$$
$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$$

The name of the strategy is derived from projections of the primary moment (the mean) (mt) and the consequent 2nd (the uncentered variance) (vt) of the gradients independently. Since (mt) and (vt) are instituted as vectors of 0's, the makers of Adam note that they are skewed to 0, specifically during the primary time steps and principally the decay rates are low, i.e., Beta1 and Beta2 are almost one.

By measuring bias-corrected 1st & 2nd moments, they are able to limit these biases.

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t}$$
$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t}$$

:

As we've seen in adadelta and RMSprop, which generates the adam update law below, at a specific instance are commonly accustomed to renew parameters.

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t + \epsilon}} \hat{m}_t$$

## 5. CONCLUSION

The accuracy of the final model is 99.19% whereas the benchmark model's accuracy is 87.2%. Our model has clearly outperformed the benchmark model. Although, in order to justify that our model has potentially solved the problem, we need to look at the results of the real time test conducted against model as shown in Fig 6. Our model can correctly predict 10 out of 11 tests. Thus, our model solved the problem with a good accuracy score and reliability in real time applications. However, using the MNIST dataset and keras library simplified the process a lot. MNIST Dataset was a good choice in deep learning, as there was lot of tutorials and guides available. using MNIST dataset which helped in understanding them easily. Also, Keras helped in kickstarting into deep learning without writing complex code which was really helpful in quick prototyping and experimenting in order to build a better model. SUGGESTED ADVANCEMENTS The final model in this project uses only simple Convolutional Neural Network architecture. There is more advanced architecture like the Deep Residual Neural Network (ResNet) [12] and VGG-16[13] which could be used to improve the accuracy a little. Also we could hyperas[14] library to fine tune the hyper-parameters and architecture of the model.

## 6. REFERENCES

[1] en.wikipedia.org/wiki/Handwriting recognition

[2] www.ehow.com/list_7353703_benefits-advantages-handwritingrecognition.html

[3] www.cse.unsw.edu.au/~billw/cs9444/crossentropy.html

[4] http://yann.lecun.com/exdb/mnist/

[5]en.wikipedia.org/wiki/Convolutional_neural_network

[6]

ufldl.stanford.edu/tutorial/supervised/ConvolutionalNeuralNetwork/

[7] http://cs231n.github.io/convolutional-networks/

[8] www.quora.com/What-is-dropout-in-deep-learning

[9] https://keras.io/layers/core/#flatten

[10] https://www.quora.com/What-are-softmax-layers

[11]www.kaggle.com/romanroibu/digit-recognizer/random-forest-digit-classifier

[12] https://arxiv.org/abs/1512.03385

[13] www.robots.ox.ac.uk/~vgg/research/very_deep/

[14] https://github.com/maxpumperla/hyperas

[15] sebastianruder.com/optimizing-gradient-descent/

[16] http://cs231n.github.io/linear-classify/

[17] https://www.quora.com/Whats-the-difference-between-gradient-descent-andstochasticgradient-descent/answer/Sebastian-Raschka-1?srid=hhRTP

[18] K. B. Lee, S. Cheon, and C. O. Kim, "A convolutional neural network for fault classification and diagnosis in semiconductor manufacturing processes," IEEE Transactions on Semiconductor Manufacturing, vol. 30, no. 2, pp. 135-142, 2017.

[19] K. G. Pasi and S. R. Naik, "Effect of parameter variations on accuracy of Convolutional Neural Network," in 2016 International Conference on Computing, Analytics and Security Trends (CAST), 2016, pp. 398-403: IEEE.

[20] M. Wu and Z. Zhang, "Handwritten digit classification using the mnist data set," Course project CSE802: Pattern Classification & Analysis, 2010.

[21] Y. Yin, J. Wu, and H. Zheng, "Ncfm: Accurate handwritten digits recognition using convolutional neural networks," in 2016 International Joint Conference on Neural Networks (IJCNN), 2016, pp. 525-531: IEEE.

[22] L. Xie, J. Wang, Z. Wei, M. Wang, and Q. Tian, "Disturblabel: Regularizing cnn on the loss layer," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 4753-4762.

[23] A. Tavanaei and A. S. Maida, "Multi-layer unsupervised learning in a spiking convolutional neural network," in 2017 International Joint Conference on Neural Networks (IJCNN), 2017, pp. 2023-2030: IEEE.

[24] J. Jin, K. Fu, and C. Zhang, "Traffic sign recognition with hinge loss trained convolutional neural networks," IEEE Transactions on Intelligent Transportation Systems, vol. 15, no. 5, pp. 1991- 2000, 2014.

[25] Y. Liu and Q. Liu, "Convolutional neural networks with large margin SoftMax loss function for cognitive load recognition," in 2017 36th Chinese Control Conference (CCC), 2017, pp. 4045- 4049: IEEE.