Handwritten digits classification

using MNIST dataset

**CSEL-393 Computer Vision and Image Processing**

**Group Members:**

Ali Raza Khan (2018-UET-NML-CS-23)

Muhammad Musawar Baig (2018-UET-NML-CS-03)

## 

# 

# **Submitted to: Dr. Sadaqat ur Rehman**

**Dated: 20th December 2021**

# **Acknowledgement**

First and foremost, praises and thanks to Allah Almighty, for His showers of blessings throughout our project to complete it successfully.

We would like to express our deep and sincere gratitude to our teacher, Dr. Sadaqat ur Rehman, Namal University, Pakistan, for giving us the chance to do this project and providing invaluable guidance throughout this project.

Finally, Our thanks go to all the individuals who have supported us to complete this project work directly or indirectly.

Ali & Musawar

# 

# 

# **1 Abstract**

In this report, we present a Convolution Neural Network (CNN) to tackle the problem of recognition of human handwritten digits from 0 to 9, inclusively. The CNN proposed here is experimented on the famous MNIST dataset which is extensively used by students, teachers, and researchers for learning and research purposes.

Our dataset is preprocessed which is why our CNN is achieving quite a high accuracy...

/Not Complete–Do add one or two lines here describing what we have built and what our system is doing/ Are we testing our system on only preprocessed data or we are also testing it on raw/unseen data and what are the results? Some of it could also be added in the next “Introduction” section

# **2 Introduction**

Image classification is a renowned deep learning application of computer vision. Image classification focuses on separating the pixels of a picture according to the classes they belong to. The MNIST's ("Modified National Institute of Standards and Technology") dataset is one of the popular datasets of computer vision. Since its release in 1999, this classic dataset of handwritten images has served as the basis for benchmarking classification algorithms. With the passage of time, As new machine learning (ML) techniques emerge, MNIST remains a dependable resource for researchers/students and learners alike [1].

In our project, We have used an updated version of the original MNIST. Our goal was to correctly classify digits from a dataset of tens of thousands of handwritten images. We have used Python as a programming language, Keras for Neural Networks, OpenCV for Computer Vision tasks, and TensorFlow for different machine learning tasks. Our training dataset comprised 60,000 images in the form of jpg files with ten target classes. 42,000 images were used in the training set and 28,000 images in the test set. This division of test and training data mirrors the split used in the kaggle’s Digit Recognizer competition [2]

Images included in the dataset are small square 28×28 pixel grayscale images of handwritten single digits between 0 and 9.

The task is to classify a given image of a handwritten digits into one of 10 classes representing integer values from 0 to 9, inclusively. As the images present in this dataset are of relatively low resolution, we were able to train and test our models without facing any significant technical issues.

We have done different experiments to find the best model for our dataset/problem. Our designed model gives 100% accuracy on handwritten digits (MNIST). We have used batch normalization techniques to make the learning process faster which also results in a converging learning behavior and takes less number of epochs to learn.

# **3 Methodology**

The code that we have developed will mainly focus on identifying handwritten digits from 0 to 9 from unseen segmented pictures of handwritten digits. Input images are pre-processed to make them of a certain fixed size. The input image also gets converted to a gray level image in the pre-processing stage before it is fed into the trained model for prediction. The intensity level of grayscale images varies from 0 to 255. We are using built-in functions to pre-process images.

For the training and testing stages, We are using a preprocessed MNIST dataset. All images in the MNIST have already normalized to 28x28 the data set itself is publicly available. The MNIST data set is a huge one. It has become a standard data set for testing various algorithms in the domain of Computer Vision. Mainly our solution is based on Convolutional Neural Networks (CNN). CNNs are best fit for classification problems like ours. We have used tensorflow for all the machine learning related tasks, Computer Vision library for CV related tasks and last but not least Numerical Python library to deal with numbers, as we all know when it comes to computers it’s just numbers and nothing else. Our ML model is inspired from models like VGG16 and AlexNet, famous for their ability to classify objects.

# **4 Results**

We have tried multiple Convolutional Neural Network (CNN) structures, different training and testing dataset ratios and proportions. For example we have tried a CNN architecture with 3 hidden layers which gave us a good accuracy, as well as a CNN of 2 hidden layers with less accuracy. Following is the model structure which was best fit for our problem. We have achieved more than 99.2 percent accuracy with very less loss and fast learning when we used a batch normalization layer at the 2nd-hidden layer of the neural network. Below is the structure of our CNN Model.

|  |
| --- |

We have done different experiments on different settings of CNN model (Sequential). But the settings of CNN made an impact, of course we had done two experiments on the above mentioned neural network settings as well. One while using batch normalization and the other without using batch normalization layer.

We have generated some results graphs using some techniques like **Confusion matrix**, Accuracy and Loss graph plotting using **Tensorboard** and **classification report** and plotting predicted labels and the original images using subplots (matplotlib).

**Confusion matrix:** In the field of machine learning and specifically the problem of statistical classification, a confusion matrix, also known as an error matrix, is a specific table layout that allows visualization of the performance of an algorithm, typically a supervised learning one

**Classification report:** A classification report is a performance evaluation metric in machine learning. It is used to show the precision, recall, F1 Score, and support of your trained classification model.

Following are graphs and pictures for two of the experiments we performed, with and without batch normalization.

## 4.1 With/Using Batch Normalization Layer

### Confusion matrix

| **Fig: Confusion Matrix while using BatchNorm** |
| --- |

### Classification report

| **Fig: Classification Report of Model Evaluation (with using Batch Norm)** |
| --- |

### Accuracy and Loss graph

| **Fig: Accuracy and Loss per Epoch (while using BatchNorm)** |
| --- |

## 4.2 Without Using Batch Normalization

### Confusion matrix

| **Fig: Confusion Matrix while not using BatchNorm** |
| --- |

### 

### Classification report

| **Fig: Classification Report of Model Evaluation (Without using Batch Norm)** |
| --- |

### 

### Accuracy and Loss graph

| **Fig: Accuracy and Loss per Epoch (without BatchNorm)** |
| --- |

# **5 Conclusion**

We have come to the following conclusions:

1. ANNs cannot give you 100% Accuracy but in case of CNN, a good CNN model can give you a 100% accuracy
2. CNN models can overfit easily so to prevent overfitting we used multiple dropouts
3. Without batch normalization, your learning curve is never that smooth as it is in case you use batch normalization
4. Using batch normalization, you can reduce number of iterations/epochs for learning
5. Cleaning of dataset is necessary

# **6 References**

[1] MNIST as .jpg, Kaggle Digit Recognizer Competition Dataset as .jpg Image Files. <https://www.kaggle.com/scolianni/mnistasjpg>

[2] Digit Recognizer, Learn computer vision fundamentals with the famous MNIST data. <https://www.kaggle.com/c/digit-recognizer/data>

[3] https://research.google.com/colaboratory/faq.html

[4]

[5]

1. /Add other references if necessary including the references of addition material etc/

# **We need to submit report + code <-**

The above structure was given by Sir. We need to submit 5 pager report.

# **Some useful resources for you to complete the report :)**

# [A simple 2D CNN for MNIST digit recognition toward data science article](https://towardsdatascience.com/a-simple-2d-cnn-for-mnist-digit-recognition-a998dbc1e79a)

1. [Final Report of Term Project—ANN for Handwritten Digits Recognition](https://eng.ucmerced.edu/people/wwang5/misc/ann.pdf)
2. [Computer Vision](https://en.wikipedia.org/wiki/Computer_vision)

# [Applying Convolutional Neural Network on mnist dataset (GeeksforGeeks)](https://www.geeksforgeeks.org/applying-convolutional-neural-network-on-mnist-dataset/)

1. [Our ML Report](https://drive.google.com/file/d/1V6elehIWeLyvV58nRcy_3hOE9lE9XgKc/view?usp=sharing)
2. [Our CV Presentation](https://docs.google.com/presentation/d/1KD4whAJgfvPPre589V1p1I9fnL3VNuqH/edit?usp=sharing&ouid=104561688666712931496&rtpof=true&sd=true)