

# DATA AUGMENTATION USING MNIST

## Introduction

Deep learning is gaining popularity with the increase in the field of data science and the amount of data produced every single day. The process of acquiring or generating larger amount of data for the deep learning models can be expensive. To cut down the cost of producing labelled data, data augmentation is a cost-effective solution. Data augmentation is a widely accepted technique which is used to artificially increase the potential of deep learning networks. It plays a crucial role in cases where the ground truth data is limited. This method introduces various types of transformations to create new training samples for the purpose of executing machine learning models. Shorten C and his teammate[2] presented a survey on the data augmentation techniques which included geometric transformations, erasing, cropping and many more. The goal of this project is to execute data augmentation techniques, initialise a convolutional neural network, execute it and obtain desired results.

## Experimental Setup

This project utilizes the MNIST dataset for the implementation of data augmentation and convolutional neural network. The MNIST dataset is a database of images of handwritten digits ranging from 0 to 9. It is a dataset which is a subset of a much larger dataset NIST. The training samples in this dataset amount up to 60,000 while there are 10,000 test samples. The digits are normalized, and the size has been initialized to 28x28. This dataset is publicly available on the link which is provided [here](#).

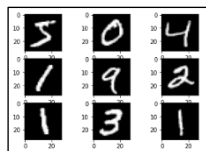


Figure 1: Random images from the MNIST

The MNIST data was partitioned into train and test portions using the following script.



Figure 2: Dataset partitioning into train and test

## Methodology

Firstly, this project was implemented on Google Colab platform as it enables the user to connect to the GPU. Libraries such as NumPy, Matplotlib, tensorflow, keras and dataset are imported.

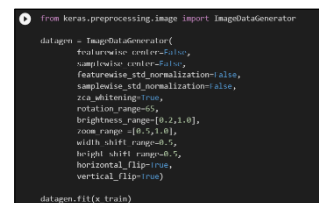


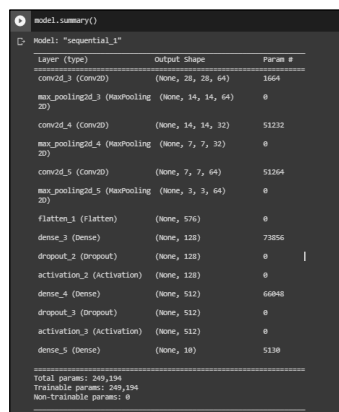
Figure 3: Data augmentation implementation

The main aim of this project was to import libraries and dataset, apply data augmentation, generate a new dataset, execute a convolution neural network, and obtain the desired results. The problem to be solved is decreased accuracy, problem of overfitting and limited amount of ground truth data. Data augmentation is a technique of generation of new training samples by applying various a list of transformations. The transformations might include random shifts, cropping, erasing, addition of noise, random flips, zooming and many more. This project implements the data augmentation parameters as shown in figure 3.

ZCA whitening is a process where the data displays decorrelated features. This parameter was set to a boolean value of “True”. The horizontal and vertical flip were set to “True” which indicated that the pixels of rows and columns were reversed. Further, the rotation angle was set to a value of 45. This shows that every image of handwritten digit was rotated at an angle of 45 degrees. Rotation is a parameter which takes the input ranging from 0 to 360

degrees. Floating point numbers can be seen in the shift parameters which implies that the images undergo a horizontal and vertical shift by 50%. The zoom and brightness parameters are specified in floating point ranges. Here, the zoom parameters were specified as [0.5,1.0]. the zoom operation enables the pixel zooming in a uniform manner throughout the image. The brightness range for this project was set to [0.2,1.0]. The aim of specifying the brightness parameter is to enable the model to be accustomed to a dataset which contains images of different brightness levels.

The above data augmentation parameters were further applied to a CNN network. The CNN network consisted of three convolutional layers of 64, 32 and 32 neurons respectively with ReLU activation function. The kernel size was set to 5. The output layer of the convolutional neural network consisted of 10 units with softmax activation function.



```

model.summary()
Model: "sequential_1"
Layer (type) Output Shape Param #
-----
conv2d_3 (Conv2D) (None, 28, 28, 64) 1664
max_pooling2d_3 (MaxPooling (None, 14, 14, 64) 0
2D)
conv2d_4 (Conv2D) (None, 14, 14, 32) 51232
max_pooling2d_4 (MaxPooling (None, 7, 7, 32) 0
2D)
conv2d_5 (Conv2D) (None, 7, 7, 64) 51264
max_pooling2d_5 (MaxPooling (None, 3, 3, 64) 0
2D)
flatten_3 (Flatten) (None, 576) 0
dense_3 (Dense) (None, 128) 73856
dropout_3 (Dropout) (None, 128) 0
activation_3 (Activation) (None, 128) 0
dense_4 (Dense) (None, 512) 66048
dropout_4 (Dropout) (None, 512) 0
activation_4 (Activation) (None, 512) 0
dense_5 (Dense) (None, 10) 5130
Total params: 249,194
Trainable params: 249,194
Non-trainable params: 0

```

Figure 4: CNN model summary

The above model was compiled with the Adam optimizer and categorical cross entropy loss function. The number of epochs and batch size were set to 100 and 128 respectively. After the successful execution of the above specified model, the accuracies are determined.

## Results

Figure 5 displays the graph of number of epochs versus loss and accuracy respectively. The graph on the shows a left hand side trade-off

between the number of epochs versus the accuracy on the MNIST dataset. The training accuracy increases from 90% and reaches a value of approximately 99%. On the right hand side, we can see the graph of number of epochs versus the loss on the MNIST dataset.

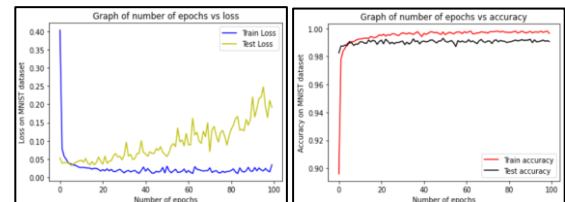


Figure 5: Graphs

The total time taken for the execution of model was approximately 900 seconds which amounts to approximately 15 minutes. Using data augmentation, model accuracy was increased to around 99%. Also, this proved to be a cost-effective solution. In addition to this, the problem of overfitting was solved.

## Conclusion

In this project of data augmentation using MNIST dataset, the CNN model was executed with an accuracy of 99% and a total execution time of 900 seconds. Through this project, one can understand various methods of data augmentation, advantages of data augmentation, more about MNIST dataset and the implementation of CNN. The future scope of this project involves the implementation of other data augmentation techniques such as cropping, random erasing and addition of noise.

## References

- [1] Class lectures and homework assignments
- [2] Shorten, Connor, and Taghi M. Khoshgoftaar. "A survey on image data augmentation for deep learning." Journal of big data 6, no. 1 (2019): 1-48.
- [3] Baldominos, Alejandro, Yago Saez, and Pedro Isasi. "A survey of handwritten character recognition with mnist and emnist." Applied Sciences 9, no. 15 (2019): 3169.