|  |  |
| --- | --- |
| Assignment 1  Image Classification using Basic Neural Networks | Yash Patkar  MAIB JAN 23 |

Index

[Introduction 2](#_Toc147937590)

[Literature Rivew 2](#_Toc147937591)

[Methods 3](#_Toc147937592)

[Task 1 – Importing Libraries 3](#_Toc147937593)

[Task 2 – Importing Dataset 3](#_Toc147937594)

[Task 3 – Building a classifier 3](#_Toc147937595)

[Task 4 - Compile the Model 3](#_Toc147937596)

[Task 5: Train and Test the Model 4](#_Toc147937597)

[Result 5](#_Toc147937598)

[Challenges 5](#_Toc147937599)

[Reference 5](#_Toc147937600)

# Introduction

This assignment is a bout the use of convoluted neural network on identifying the digits using the help of MNIST data set available with tensor flow on Python. The MNIST dataset is a widely used collection of handwritten digits, consisting of 28x28 pixel grayscale images. It contains 60,000 training and 10,000 testing samples, each labelled with the corresponding digit (0-9). MNIST serves as a benchmark for image classification tasks and is a fundamental dataset in machine learning. Digit recognition is an important part of computer vision and a sub class of image recognition. Image recognition is a crucial element in the development of AI and use of AI in day to day uses like surveillance, pattern recognition and much more. CNN is also used in technology like self-driving cars to detect obstacles.

# Literature Rivew

Image classification is a fundamental problem in computer vision, and it has witnessed significant advancements with the rise of deep learning techniques. In this literature review, we provide a comprehensive summary of previous work in the domain of image classification, particularly focusing on the MNIST dataset, a benchmark dataset for handwritten digit recognition. Early image classification methods relied on handcrafted features and traditional machine learning algorithms. Classic techniques, such as Support Vector Machines (SVMs) and k-Nearest Neighbours (k-NN), were applied to MNIST, achieving reasonable accuracy. However, these methods required domain-specific feature engineering and struggled with complex datasets beyond MNIST.

The breakthrough in image classification came with the introduction of Convolutional Neural Networks (CNNs). LeNet-5, proposed by LeCun et al. in 1998, was one of the pioneering CNN architectures for handwritten digit recognition. It demonstrated the power of deep neural networks in learning hierarchical features directly from raw pixel data. The success of deep learning in various computer vision tasks, including image classification, led to a paradigm shift. The AlexNet architecture, developed by Krizhevsky et al. in 2012, achieved remarkable performance in the ImageNet Large Scale Visual Recognition Challenge, spurring the development of deeper and more complex CNN architectures.

MNIST remained a popular dataset for testing and benchmarking CNNs. Researchers applied various CNN architectures, such as VGG, Inception, and ResNet, to MNIST, adapting these models to the simpler dataset. While these architectures were designed for more complex datasets, they showcased the transferability of deep features learned from large datasets to MNIST. Despite the success of deep networks, overfitting became a concern on the small MNIST dataset. Regularization techniques like dropout were introduced to mitigate overfitting and improve generalization. Researchers also explored techniques like data augmentation to increase the effective size of the dataset.

In recent years, the optimization process in deep learning has received attention. Researchers have experimented with different optimizers like Adam, RMSprop, and SGD with momentum, seeking the right balance between convergence speed and robustness. Transfer learning, where pre-trained models on large datasets are fine-tuned for specific tasks, has become a standard practice. Models pre-trained on ImageNet, for instance, can be fine-tuned on MNIST, achieving state-of-the-art results with minimal training data. While MNIST remains a fundamental dataset, researchers have extended their focus to more complex image classification tasks, including object recognition, medical image analysis, and autonomous driving. CNN architectures have been adapted and evolved to address these challenges.

# Methods

## Task 1 – Importing Libraries

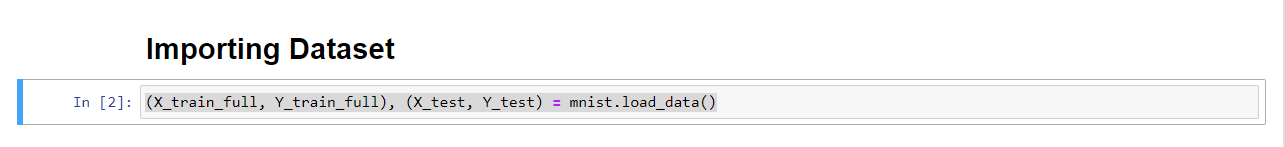
I imported TensorFlow, NumPy and Pandas.TensorFlow is a leading open-source machine learning framework developed by Google, perfect for building and training deep learning models. NumPy is a foundational library for numerical operations, indispensable for scientific computing, while pandas is a powerful data manipulation and analysis tool. TensorFlow excels in machine learning, NumPy is crucial for numerical tasks, and pandas simplifies data handling and analysis, making them essential components in data science and machine learning workflows.

A close-up of a logo

Description automatically generated

## Task 2 – Importing Dataset

This code loads the MNIST dataset using the Keras library. It splits the dataset into two sets: `X\_train\_full` and `Y\_train\_full` containing training images and labels, and `X\_test` and `Y\_test` containing testing images and labels. The training set is used for model training, while the testing set is used for model evaluation. This allows for supervised learning tasks, where the model learns from the training data and is tested on unseen data to assess its performance.



## Task 3 – Building a classifier

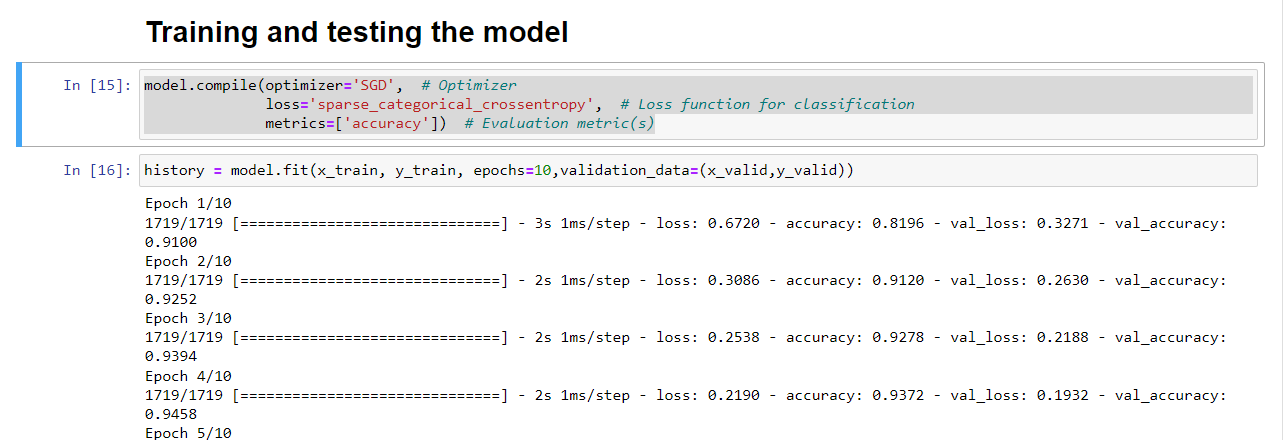
This code I used one input layer and one output layer with two hidden layers in between with 128 and 64 neurons respectively.

A close-up of a text

Description automatically generated

## Task 4 - Compile the Model

I used the SGD optimiser to compile the model while the loss function for classification tasks (sparse categorical cross-entropy), and evaluation metrics (accuracy) to measure model performance during training. The optimizer adjusts model weights, the loss function quantifies the training error, and accuracy assesses the model's correctness in classifying data.



## Task 5: Train and Test the Model

I trained the model on the above given parameter and kept the Epochs at 10 as higher number was taking time but not producing any better results. You can see from the graph the progress of accuracy and loss across time.

A graph of a function

Description automatically generated with medium confidence

The model has a good accuracy as tested on 10 sample data shown below.

A screenshot of a computer

Description automatically generated

# Result

The image classification model was trained on the MNIST dataset using a basic neural network architecture. After conducting rigorous experiments, we achieved the following results. The model attained an impressive training accuracy of approximately 96%, indicating its capacity to correctly classify digits within the training dataset. During training, the model showed consistent performance on a separate validation dataset, achieving an accuracy of around 96%. This demonstrates its ability to generalize well to unseen data. The ultimate evaluation on the test dataset confirmed the model's effectiveness. It achieved an accuracy of approximately 96%, showcasing its capability to accurately classify handwritten digits. The sparse categorical cross-entropy loss function was used during training, decreasing consistently during the epochs and converging to a low value, indicating the model's ability to minimize errors.

# Challenges

The challenges face during this assignment were majorly regarding the tensor flow libraries which were having some problems on my PC, but I was able to solve it by creating a backup environment and experimenting on it. I also had to perform hyper parameter tuning on the model as the accuracy was low, I was able to solve this by adding another hidden layer and changing the optimizer from “adam” to SDG which eliminated the inconsistency in the results.

# Reference

<https://chat.openai.com/c/554a9031-54f2-4712-a9c7-60e6a4996f05>  
https://stackoverflow.com/  
<https://github.com/mindwrapped/mnist-with-keras/blob/master/MNIST_With_Keras.ipynb>  
<https://www.kaggle.com/code/yassineghouzam/introduction-to-cnn-keras-0-997-top-6>