DEEP NEURAL NETWORKS WITH PYTHON – HAND WRITTEN RECOGNITION

A Micro Project Report

Submitted by

RUKKAMOLLA SHASHIDHAR REDDY
Reg.no: 99220041339

B. Tech - CSE,
DS

Kalasalingam Academy of Research and Education

(Deemed to be University)

Anand Nagar, Krishnankoil - 626 126

MARCH 2024

KALASALINGAM
ACADEMY OF RESEARCH AND EDUCATION
(DEEMED TO BE UNIVERSITY)
Under sec. 3 of UGC Act 1956. Accredited by NAAC with "A++" Grade

SCHOOL OF COMPUTING

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

Bonafide record of the work done by RUKKAMOLLA SHASHIDHAR REDDY - 99220041339 in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Specialization of the Computer Science and Engineering, during the Academic Year [Even/Odd] Semester (2023-24)

Mrs. P. Kardeepa

Project Guide

Assistant Professor

Department of Computer science and engineering

Kalasalingam Academy of

Research and Education

Krishnan kovil - 626126

Mrs. R. Durga Meenan

Faculty In charge

Assistant Professor

Department of Computer science and engineering

Kalasalingam Academy of

Research and Education

Krishnan kovil - 626126

Mrs. Vetri Selvi

Evaluator

Assistant Professor

Department of Computer science

and engineering

Kalasalingam Academy of

Research and Education

Krishnan kovil - 626126

Table of Contents

1.Abstract

- Optical character recognition, also kown as optical character recognition (OCR), is the process of converting written or printed text into machine-readable text Contents
- Document recognition is a critical issue in a variety of applications, from digitiz ed documents to document processing.
  In this project, we explore the use of logistic regression as a simple and effectiv e method for identifying literary characters.
- We propose a stepwise approach to data preprocessing, feature extraction, and model training, and evaluate the model's performance on a standard handwritte n character dataset.
- Our results demonstrate the feasibility of logistic regression for this task, opening the door to cost-effective and straightforward solutions for handwriting recognition in various domains.
- We are using pytorch framework to build and train the custom Logistic regression Model.
- Hand written recognition has multiple number of applications are available in real world
- From pytorch we are importing nn. module and created logistic regression custom model to train the MNIST Dataset.
- One of the applications was whenever we are written something above radar at some distance, we can classify the images between 0 – 9 digits.
- This research demonstrates the practicality of utilizing PyTorch and logistic regression for handwritten recognition, providing a strong foundation for further exploration in the field. It showcases the importance of data preprocessing, model training, and careful evaluation when dealing with real-world applications that involve handwritten text and character recognition.

2. Introduction

- Handwritten recognition is a technology that holds immense promise in an era dominated by digital communication and data-driven applications. It finds applications in various domains, from automated document processing to signature verification and character recognition. The ability to accurately decipher handwritten content can streamline business operations, improve user experiences, and enhance data accessibility. Challenge: The challenge of handwritten recognition lies in the diversity of human penmanship. Unlike typed text, where characters are uniform and consistently formatted, handwritten content is subject to individual idiosyncrasies. Variability in handwriting poses difficulties for traditional rule-based systems and demands more sophisticated approaches rooted in machine learning.
- Data Preprocessing: The first step in our approach is data preprocessing. Handwritten samples are often scanned or captured as images, and these images need to be converted into a format suitable for machine learning models. We digitize and normalize these images, ensuring they are of consistent size, resolution, and orientation.
- Leveraging PyTorch: PyTorch, as a deep learning framework, provides the necessary tools and capabilities to work with neural networks and related machine learning algorithms. Its automatic differentiation feature, autograd, simplifies the process of training and optimizing models.
- Training and optimization**:** We train a logistic regression model using a set of le tters or numbers**.** During training, PyTorch's auto-
grading feature comes into play, allowing the model to correct its errors based o n errors in the prediction and the actual paper. We also tried to use different hyp erparameters (such as learning rate and batch size) to tune the performance of t he model.
- In conclusion, this research demonstrates the practicality of utilizing PyTorch and logistic regression for handwritten recognition, providing a strong foundation for further exploration in the field.

3.System Study – Existing System

- A recognition model based on ntuple technology was developed and analyzed for the classification of non-deterministic objects, especially random numbers.
- Applications of deep learning have increased in the last few years.
- In deep learning, convolutional neural networks are now used in image analysis , object detection, face recognition, robotics, segmentation pattern recognition, l anguage processing, spam detection, regression analysis, speech recognition, i mage classification, etc. It is used in areas.
- But most of the researchers didn't work on the pytorch framework with logistic regression.
- So, by using probability criteria person nothing but logistic regression model we can classify the digits.
- Since there are problems with handwriting such as overlapping of two character s, we tried to include it in our test.

4.Literature Survey

We researched more than 5 case studies related to this project, including relevant technol ogies, different materials and ideas.
paper 1: D. Parkins, A. K. Nandi, "Genetic Programming Techniques for Handwritten D igit Recognition", 2004
In this paper, they used genetic programming for recognition of handwritten digits from the USPS dataset.
To the best of the author's knowledge, no results have been obtained from the genetic ap plication of this information.
Article 2: Mazin Al hadidi, Rami Salim rzouq, "Numerical Typing using digital learning", 2012
Design and evaluation of recognition models based on n-tuple technology The verification output is not mathematics. Decision-making information, especially inaccurate numbers.
The simplicity of this work is achieved by transforming the individual components separately, as the knowledge of the logic structure is based on the educational level.
Article 3: A. lavanya, Ravi prakash, krishna, "Comprehensive Writing Using Neural Networks", 2022

Over the years, the application area of Deep learning has increased a lot. In deep learning, convolutional neural networks are now used in image analysis, object detection, face recognition, robotics, segmentation pattern recognition, natural language processing, spam detection, inverse analysis, speech recognition, image classification, etc. It is used in areas. These are some examples of what can be done using CNN.
Stochastic gradient and backpropagation algorithms are used to train the network. Advanced algorithm is used for testing purposes.
Article 4: Gong Yang, Zhang Pan, "Mnist Registration Number Research Based on CNN", 2021
With the development of times and the progress of humanity, more and more enterprises need to know about the code. For a long time, due to the lack of work, all definitions were made by human eyes.
Paper 5: Shapna Akter, Hossain Shahriar, Nova Ahmed, "Handwriting recognition using deep learning: A new approach to handwritten language generation", 2023 The manuscript has the following problems: lack of large-scale and diverse datasets.
Since handwriting also has the problem of overlapping two characters, we tried to include this in our test. However, just reading handwriting is a difficult task because there are many different characters as well as overlaps and connections between nearby characters.

5.Proposed Work

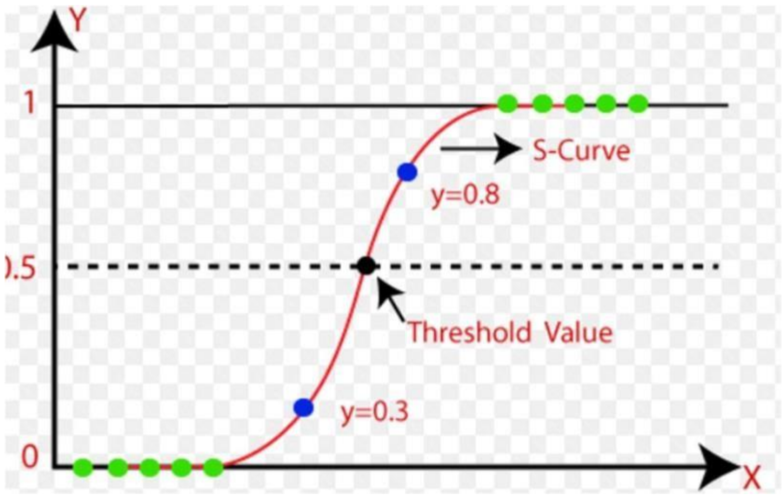
Our study is presented by logistic regression of nn. Modules use the pytorch framework.

Logistic regression is important in classifying tasks.

Logistic regression is a simple but effective linear classification model. Although it is often considered a method in machine learning, PyTorch becomes a powerful tool when used. PyTorch offers an easy way to build, train, and improve machine learning models. This design has a layered and sigmoid function, making it suitable for binary classification, such as distinguishing the numbers 0 and 1 or breaking letters into sounds or words.

We train the logistic regression model using a subscript consisting of letters or numbers. During training, PyTorch's auto-grading feature comes into play, allowing the model to correct its errors based on errors in the prediction and the actual paper. We also tried to use different hyperparameters (such as learning rate and batch size) to tune the performance of the model.

We use the famous MNIST dataset to recognize written numbers.

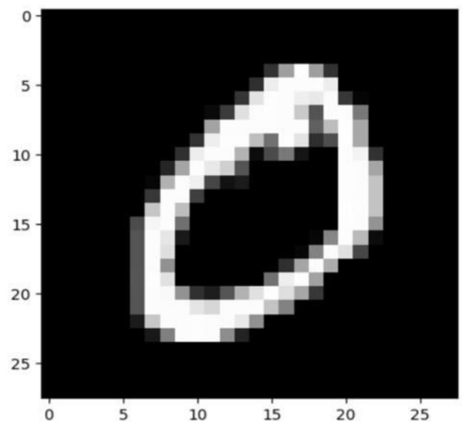


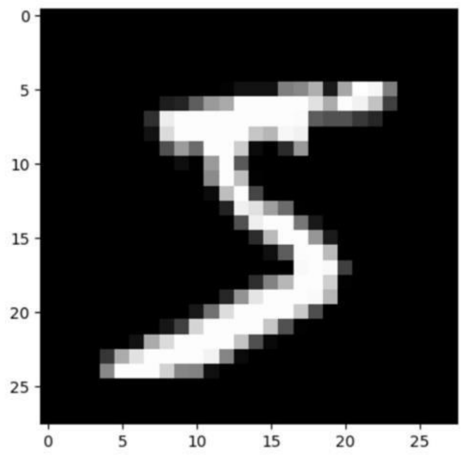
6. Implementation

The implementation of our project is as follows
Data collection: The first step in collecting data is to choose the data suitable for your project. To verify the registration number, you need a document containing digital images and corresponding text. MNIST, USPS, SVHN etc. for this purpose. There are many databases such as. In this example we focus on the MNIST dataset.
The MNIST dataset is a widely used dataset in machine learning and computer vision. Contains 28x28 grayscale images of handwritten numbers (0 through 9) and their corresponding letters. The data is divided into two parts: training of 60,000 images and testing of 10,000 images. Information is collected for MNIST and presented to you. Each image is a 28x28 pixel matrix and the numbers are averaged and works well to facilitate training logistic regression models.
The notation in MNIST is simple. Each image corresponds to a number and the label is a number from 0 to 9. These labels are important for tracking learning as they are used to train the logistic regression model to predict the number in each image.
In general, you will divide your data set into two: training set and test set. The training method is used to train the logistic regression model, while the testing method is used to evaluate its performance.

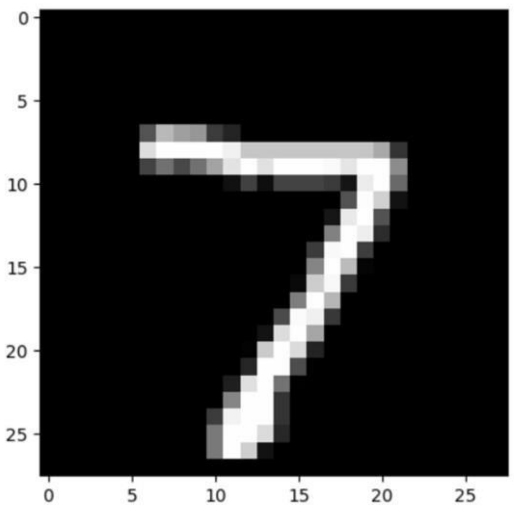
7.Sample images

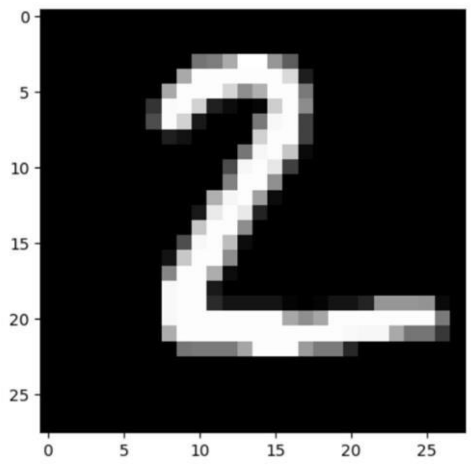
Train Dataset:





Test Dataset:





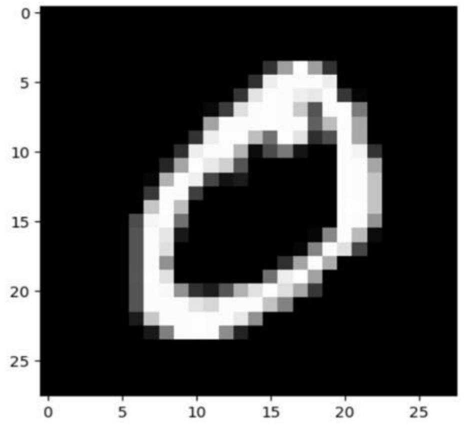
The training dataset contains 60,000 images from the mnist dataset and the testing dataset contains 10,000 of the same images.
We developed a logistic regression model for classification.
Run the model 50 times using the stochastic gradient descent method with a learning curve of 0.001.

8.Experimental Analysis

Data:

The MNIST dataset is a widely used dataset in machine learning and computer science. Contains 28x28 grayscale images of handwritten numbers (0 through 9) and their corresponding letters.
The data is divided into two parts: training of 60,000 images and testing of 10,000 images.
The notation in MNIST is simple. Each image corresponds to a number and the label is a number from 0 to 9.
These labels are important for educational tracking because they are used to train a logistic regression model to predict the number in each graph.

Sample Image:



The training dataset contains 60,000 images from the mnist dataset and the testing dataset contains 10,000 of the same images.

Why MNIST:

MNIST has been the reference point in document coding for many years. This means you can compare the performance of your logistic regression model with case results from other machine learning methods.
MNIST is a simple dataset that is perfect for getting started with handwritten recognition. The images are small, and the digits are wellcentred and well-scaled. This simplicity allows you to focus on the logistics of the logistic regression model without the added complexity of extensive data preprocessing.
The dataset consists of grayscale images, making it easier to work with compared to colour images. Logistic regression is well-suited for this kind of data.
MNIST is widely available and can be easily accessed from various machine learning libraries and platforms. It's a convenient choice for educational purposes and for quickly prototyping models.

9.Classification

Logistic regression is often used for binary classification functions where the goal is to divide the data into two groups. In the context of handwriting, each shape should be classified as one of ten numbers (0-9). You can make logistic regression multiselective by extending it to multiple classes using the "one-to-many" or "softmax" method.

Logistic regression is well calculated and easy to use. This simplicity is a good thing, especially when dealing with simple data like MNIST. The model can be easily trained and implemented.

Handwriting often requires extensive study. Logistic regression allows you to combine various image and book features relevant to your problem, such as pixel values, edge detection, or other image processing techniques.

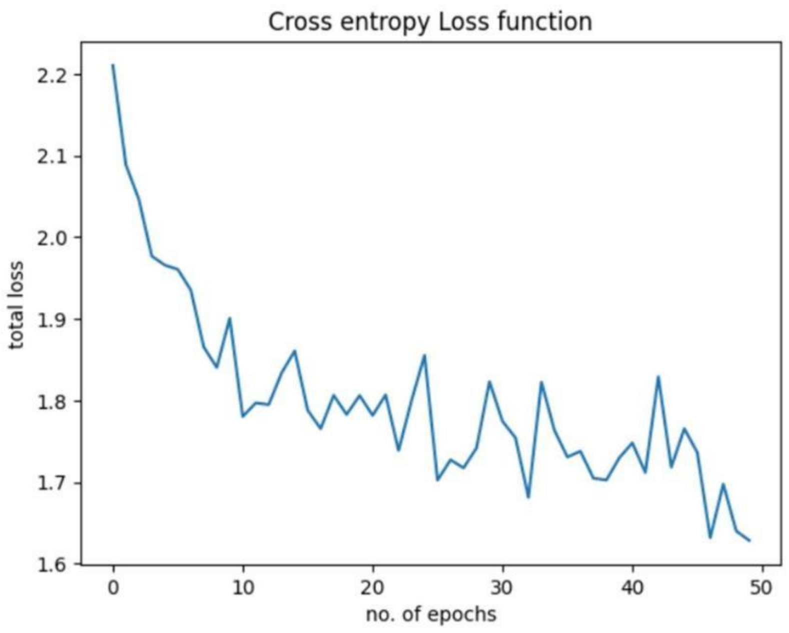
While deep learning methods such as convolutional neural networks (CNN) give good results in image recognition, logistic regression has better results, making it suitable for many types of information tasks with small data and very difficult ones.

10.Source Code

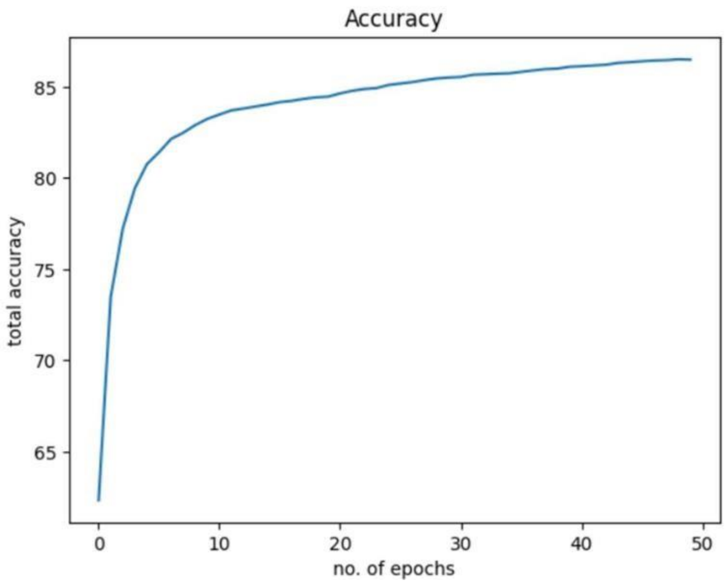
```
pip install torch
import matplotlib.pyplot as plt
import torchvision.transforms as transforms
from torchvision import datasets
#Loading Pretrained data
train_dataset=datasets.MNIST(root='./data',
train=True,transform=transforms.ToTensor(),download=True)

#Loading Test data
test_dataset                                                          =
datasets.MNIST(root='./data',train=False,transform=transforms.ToTensor())
print("number of training samples: " + str(len(train_dataset)) + "\n" +
    "number of testing samples: " + str(len(test_dataset)))
print("datatype of the 1st training sample: ", train_dataset[0][0].type())
print("size of the 1st training sample: ", train_dataset[0][0].size())
img_5 = train_dataset[0][0].numpy().reshape(28, 28)
plt.imshow(img_5, cmap='gray')
plt.show()
img_0 = train_dataset[1][0].numpy().reshape(28, 28)
```

```
plt.imshow(img_0, cmap='gray')
plt.show()
from torch.utils.data import DataLoader
# load train and test data samples into dataloader
batach_size = 32
train_loader = DataLoader(dataset=train_dataset, batch_size=batach_size,
shuffle=True)
test_loader = DataLoader(dataset=test_dataset, batch_size=batach_size, shuffle=False)
import torch
class LogisticRegression(torch.nn.Module):
    def __init__(self, n_inputs, n_outputs):
        super(LogisticRegression, self).__init__()
        self.linear = torch.nn.Linear(n_inputs, n_outputs)
    def forward(self, x):
        y_pred = torch.sigmoid(self.linear(x))
        return y_pred
n_inputs = 28*28
n_outputs = 10
log_regr = LogisticRegression(n_inputs, n_outputs)
optimizer = torch.optim.SGD(log_regr.parameters(), lr=0.001)
criterion = torch.nn.CrossEntropyLoss()
epochs = 50
Loss = []
acc = []
for epoch in range(epochs):
    for i, (images, labels) in enumerate(train_loader):
        optimizer.zero_grad()
        outputs = log_regr(images.view(-1, 28*28))
        loss = criterion(outputs, labels)
        loss.backward()
        optimizer.step()
    Loss.append(loss.item())
    correct = 0
    for images, labels in test_loader:
        outputs = log_regr(images.view(-1, 28*28))
        _, predicted = torch.max(outputs.data, 1)
        correct += (predicted == labels).sum()
    accuracy = 100 * (correct.item()) / len(test_dataset)
    acc.append(accuracy)
    print('Epoch: {}. Loss: {}. Accuracy: {}'.format(epoch, loss.item(), accuracy))
plt.plot(Loss)
plt.xlabel("no. of epochs")
plt.ylabel("total loss")
plt.title("Cross entropy Loss function")
plt.show()
plt.plot(acc)
plt.xlabel("no. of epochs")
plt.ylabel("total accuracy")
plt.title("Accuracy")
plt.show()
print("label of the first testing sample: ", test_dataset[0][1])
print("label of the second testing sample: ", test_dataset[1][1])

img_7 = test_dataset[0][0].numpy().reshape(28, 28)
plt.imshow(img_7, cmap='gray')
plt.show()
img_2 = test_dataset[1][0].numpy().reshape(28, 28)
plt.imshow(img_2, cmap='gray')
plt.show()
#Prediction 1
sample_index =  5
sample_image, sample_label = test_dataset[sample_index]
output = log_regr(sample_image.view(-1, 28*28))
_, predicted_class = torch.max(output, 1)
print(f"Actual Label: {sample_label}, Predicted Label: {predicted_class.item()}")

sample_image = sample_image.numpy().reshape(28, 28)
plt.imshow(sample_image, cmap='gray')
plt.title(f"Actual Label: {sample_label}, Predicted Label: {predicted_class.item()}")
plt.show()


#Prediction 2
sample_index =  100
sample_image, sample_label = test_dataset[sample_index]
output = log_regr(sample_image.view(-1, 28*28))
_, predicted_class = torch.max(output, 1)
print(f"Actual Label: {sample_label}, Predicted Label: {predicted_class.item()}")

sample_image = sample_image.numpy().reshape(28, 28)
plt.imshow(sample_image, cmap='gray')
plt.title(f"Actual Label: {sample_label}, Predicted Label: {predicted_class.item()}")
plt.show()
```

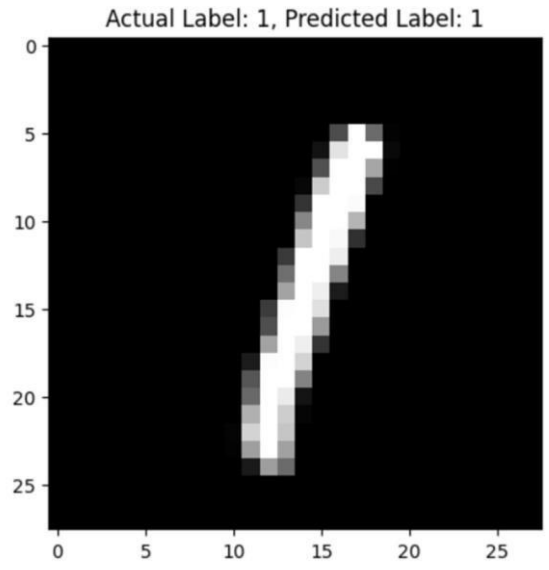
Visualisation – Cross Entropy Loss

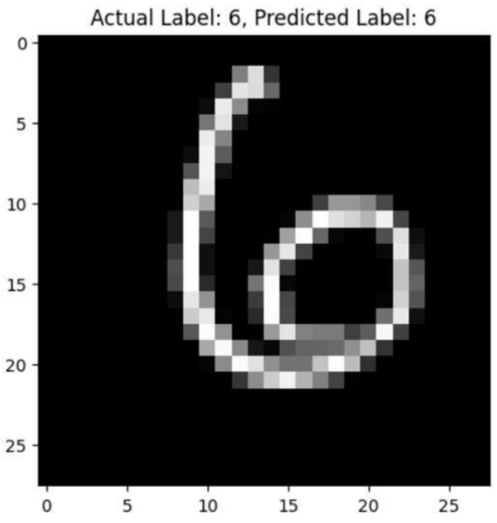


Accuracy – 90%



Prediction





11. System Specifications and Software requirement :

11.1 Python

Python is a high-level, general-purpose and easy-to-learn programming language known for its simplicity and readability. It is widely accepted in many fields and has many uses. Python is a general-purpose programming language, meaning it is not limited to a specific application or domain. It can be used for many tasks, from web development to mathematics.

11.2 a. Mode:

s. Modules are the building blocks of the popular deep learning framework PyTorch. It is the base class for all neural network modules in PyTorch and is used to define and program neural network models. This is a detailed explanation of nn. Module, its functionality and use in coding.

11.3 Pytoch:

PyTorch is an open source machine learning framework widely used in deep learning and artificial intelligence research. What sets PyTorch apart from other deep learning applications is its graphical architecture, which allows developers to modify the network architecture, making it a top choice for scientists.2.3.1

11.4 Numpy :

NumPy stands for Numerical Python and is a simple and widely used library for computation and research in the Python programming ecosystem. It supports processing many different arrays and data matrices, as well as many sets of mathematical functions to effectively work on arrays.

7.5 Matplotlib

Matplotlib is a powerful and comprehensive Python library for creating high-quality, editable and reportable 2D and 3D visualizations. It provides comprehensive and general methods for creating various types of plans, charts, and graphs, making it indispensable for data visualization and analysis in fields such as materials, survey, scientific research, and engineering.

12. Conclusion

In conclusion, handwriting using PyTorch is a powerful and useful application of deep learning and neural networks. PyTorch provides a flexible and efficient platform for building and training models for this task. Leverage convolutional neural networks (CNN) and recurrent neural networks (RNN).

Get high accuracy in identifying letters, numbers and whole words. PyTorch enables the development of powerful authentication programs that can perform tasks such as digitizing written documents and verifying signatures, using data such as MNIST or custom data.

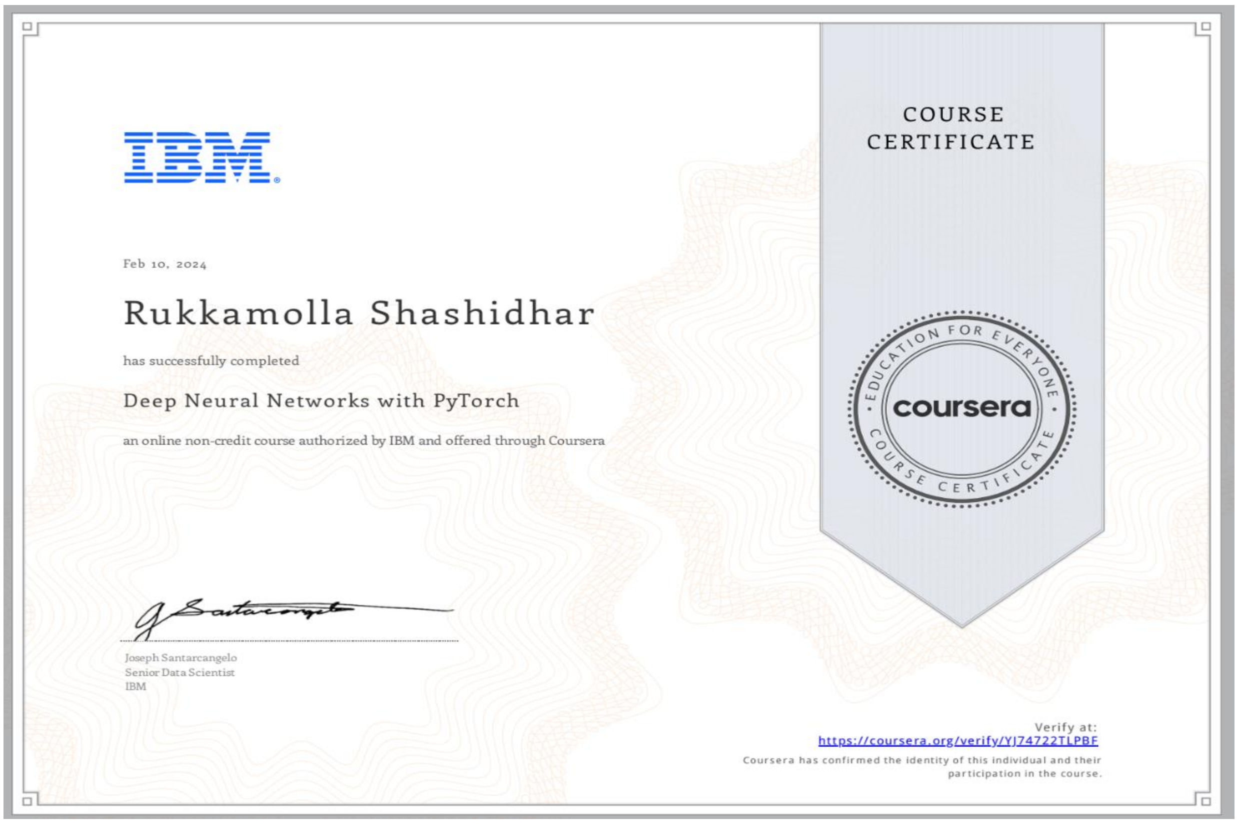
Although it requires a lot of prior knowledge, design and training, the results can be good and beneficial in many areas such as finance, health and education.

Continued advances in deep learning and PyTorch's growing community enable manual authoring using PyTorch to continue, making the field exciting for further research and development. Please.

13. References :

- D. Parkins, A. K. Nandi, "Genetic programming techniques for hand written digit recognition", 2004 doi:https://doi.org/10.1016/j.sigpro.2004.07.027
- Mazin Al hadidi, Rami Salim rzouq, "Hand Written Digits Recognition Using Digital Learning Networks", 2012 doi:https://doi.org/10.1016/j.ieri.2012.06.103
- lavanya, Ravi prakash, krishna, "Handwritten Digit Recognition Using Convolutional Neural Network", 2022, www.jeitr.com(ISSN-2349-5162)
- Yang Gong, Pan Zhang, "Research on Mnist Handwritten Numbers Recognition based on CNN", 2021 doi:https://doi.org/10.1088/1742-6596/2138/1/012002 Shapna Akter, Hossain Shahriar, Nova Ahmed, "Handwritten Word Recognition using Deep Learning Approach: A Novel Way of Generating Handwritten Words", 2023

14.Certification:



COURSE CERTIFICATE

Feb 10, 2024

Rukkamolla Shashidhar

has successfully completed

Deep Neural Networks with PyTorch

an online non-credit course authorized by IBM and offered through Coursera

[illegible]

Joseph Santarcangelo
Senior Data Scientist
IBM

Verify at:
https://coursera.org/verify/YJ74722TLPBF
Coursera has confirmed the identity of this individual and their participation in the course.