**Project Description:**

This project is focused on mastering the intricacies of deep learning through the development and optimization of a neural network tailored for the task of digit recognition. Utilizing the renowned MNIST dataset, known for its complexity yet accessibility, the project goes through the comprehensive process of building, training, evaluating, and fine-tuning a deep learning model. This hands-on experience aims to demystify the operational mechanics of deep networks, showcasing their capability to interpret and classify high-dimensional data derived from handwritten digits. This project seeks to bridge theoretical knowledge and practical application, providing a robust foundation for future projects in artificial intelligence, data analysis, and beyond, where understanding the nuances of neural network behavior is paramount.

**Task 1: Build and train a network to recognize digits**

1. **Get the MNIST digit data set**

**A number in different sizes

Description automatically generated with medium confidence**

1. **Build a network model**

A diagram of a block diagram

Description automatically generated

1. **Train the model**

**A graph with numbers and a line

Description automatically generated**

1. **Save the network to a file**

****

1. **Read the network and run it on the test set**

**A collage of numbers

Description automatically generated**

1. **Test the network on new inputs**

**A number in black squares

Description automatically generated** **A number with numbers and symbols

Description automatically generated with medium confidence**

**Task 2: Examine your network**

1. **Analyze the first layer**

A screenshot of a computer program

Description automatically generated A screenshot of a computer program

Description automatically generated

A screenshot of a filter

Description automatically generated

1. **Show the effect of the filters**

**A screenshot of a grid of numbers

Description automatically generated**

**Task 3: Transfer Learning on Greek Letters**

**A screenshot of a computer program

Description automatically generatedA group of black symbols

Description automatically generated**

**Task 4: Design your own experiment**

**Extensions**

The extensions we proceeded with are

* We created a GUI using opencv for better UX
* We created an obj file parser to insert obj models in the world
* We were able to make our system work with multiple targets
* We tested out different cameras (2 Laptop cameras, A Logitech Brio 100, Mobile Camera)
* We got our target working with an Aruco Board
* We inserted virtual objects into pre recorded videos
* We used homography to warp a target image and overlay it on both boards

**Reflection**

This project taught us a lot about the nuances of the 3D space and its relation with the 2D world. We understand the need for identifiable points between successive frames in order to consistently identify points we also learned how we can estimate the pose of the camera through the Perspective and Points algorithm. We looked at robust features and understand how they can be used to identify points. We don’t really understand how to build a coordinate system from those identified points. In the case of chessboard we could potentially identify only half the points and still get an idea of where the other points are. For robust features, we can identify points and the camera pose and maybe use those 2 points as a basis for establishing a coordinate system to represent 3D objects. We tried to explore the uber extension and came across the structure from motion problem. We understand why its an uber extension now 😊

**Acknowledgement**

<https://docs.opencv.org/4.9.0/d5/dae/tutorial_aruco_detection.html>

<https://docs.opencv.org/4.9.0/db/da9/tutorial_aruco_board_detection.html>

<https://docs.opencv.org/4.9.0/da/d13/tutorial_aruco_calibration.html>

<https://docs.opencv.org/4.9.0/d4/d94/tutorial_camera_calibration.html>

https://docs.opencv.org/4.9.0/dc/d2c/tutorial\_real\_time\_pose.html

<https://en.wikipedia.org/wiki/Rodrigues%27_rotation_formula>