Team ID: PNT2022TMID24646

A Novel Method For Handwritten Digit Recognition System

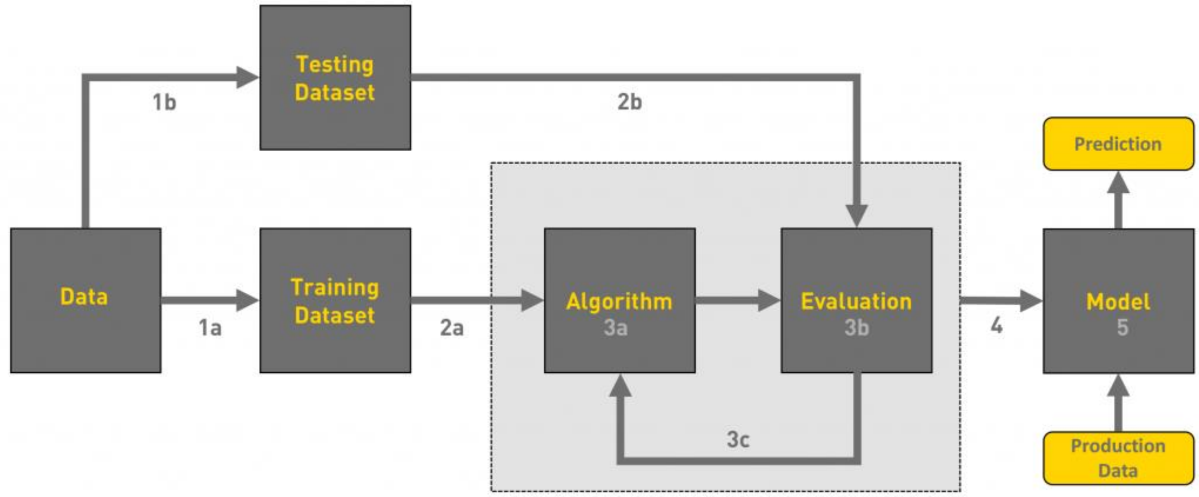
Solution Architecture:

Project Description:

Handwritten digit recognition is the ability of a computer to recognize the human handwritten digits from different sources like images, papers, touch screens, etc, and classify them into 10 predefined classes (0-9). Digit recognition has many applications like number plate recognition, postal mail sorting, bank check processing, etc.

The MNIST data collection, which contains 70000 handwritten digits, is frequently utilized for this recognition method. In order to train these photos and create a deep learning model, we use artificial neural networks. A web application is developed that allows users to upload pictures of handwritten numbers. The model examines this image and the detected result is returned to the UI.

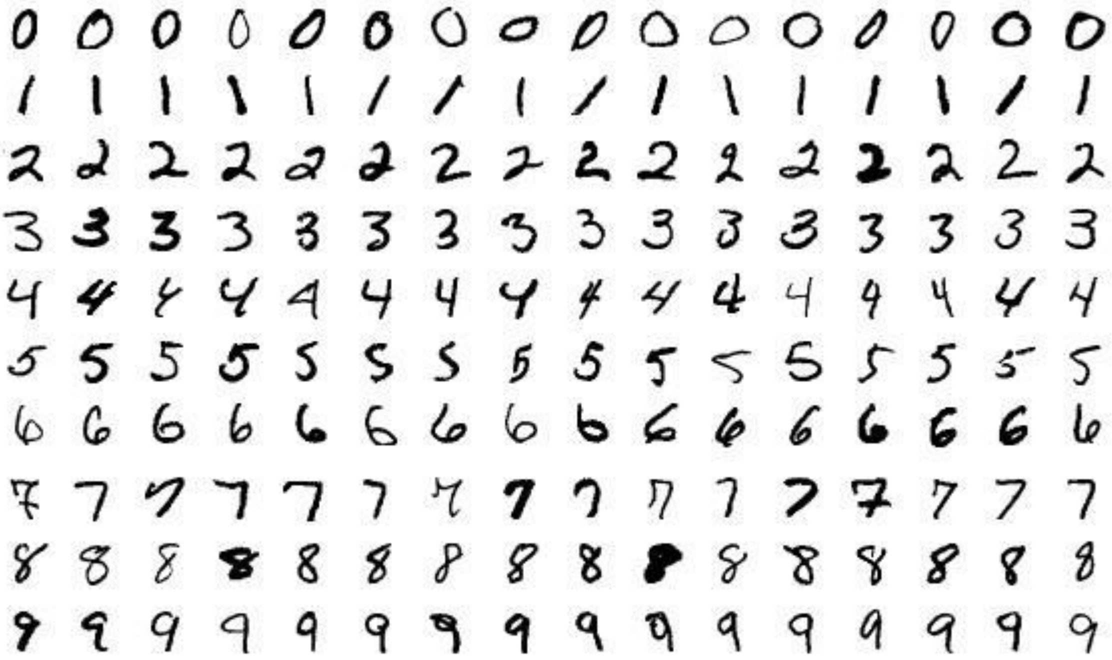
Technical Architecture:



Solution:

Mnist Dataset Description:

The MNIST database (Modified National Institute of Standards and Technology database) is a large collection of handwritten digits. It has a training set of 60,000 examples, and a test set of 10,000 examples. It is a subset of a larger NIST Special Database 3 (digits written by employees of the United States Census Bureau) and Special Database 1 (digits written by high school students) which contain monochrome images of handwritten digits. The digits have been size-normalized and centered in a fixed-size image. The original black and white (bilevel) images from NIST were size normalized to fit in a 20x20 pixel box while preserving their aspect ratio. The resulting images contain grey levels as a result of the anti-aliasing technique used by the normalization algorithm. the images were centered in a 28x28 image by computing the center of mass of the pixels, and translating the image so as to position this point at the center of the 28x28 field.



Procedure:

- ❖ Import the Libraries
- ❖ Downloading and loading the Dataset
- ❖ Preprocess the data
- ❖ Create the model
- ❖ Train the Model
- ❖ Evaluate the Model
- ❖ Improve the model accuracy
- ❖ Create GUI to predict digits

Approach:

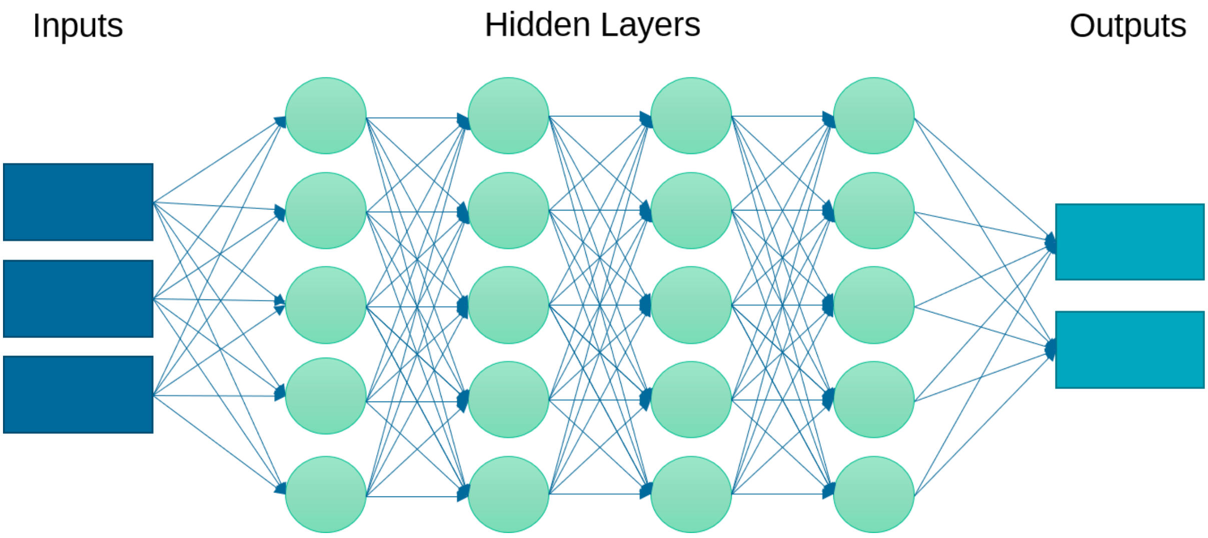
This project will be approached utilizing a three-layered neural network.

- **The input Layer:** The input layer transfers the information from our example systems to the following layer so that the latter can compute its activations.
- **The hidden Layer:** The network's nonlinear ties are provided by hidden units termed activations that make up the hidden layer. Depending on our needs, there can be a variety of concealed layers.
- **The output Layer:** The nodes in this stratum are referred to as output units. It gives us access to the neural network's final prediction, which may be used to make final predictions.

A neural network is a model of the brain's operations. It is made up of numerous layers with a variety of activations; these activations mimic the neurons in our brain. An attempt is made by a neural network to learn a set of parameters from a set of data that might aid in understanding the underlying relationships. Since neural networks are capable of adapting to changing input, the network can produce the best outcome without having to change the output criterion.

Methodology:

A neural network with one hidden layer and 100 activation units has been put into practice (excluding bias units). The features (X) and labels (Y) were retrieved after the data was loaded from a.mat file. To prevent overflow during computation, features are then scaled into a range of [0,1] by dividing by 255. 10,000 testing cases and 60,000 training examples make up the data. With the training data, feedforward is used to calculate the hypothesis, and backpropagation is then used to lower the error between the layers. To combat overfitting, the regularization parameter lambda is set to 0.1. To identify the model that fits the situation the optimizer runs for 70 times.



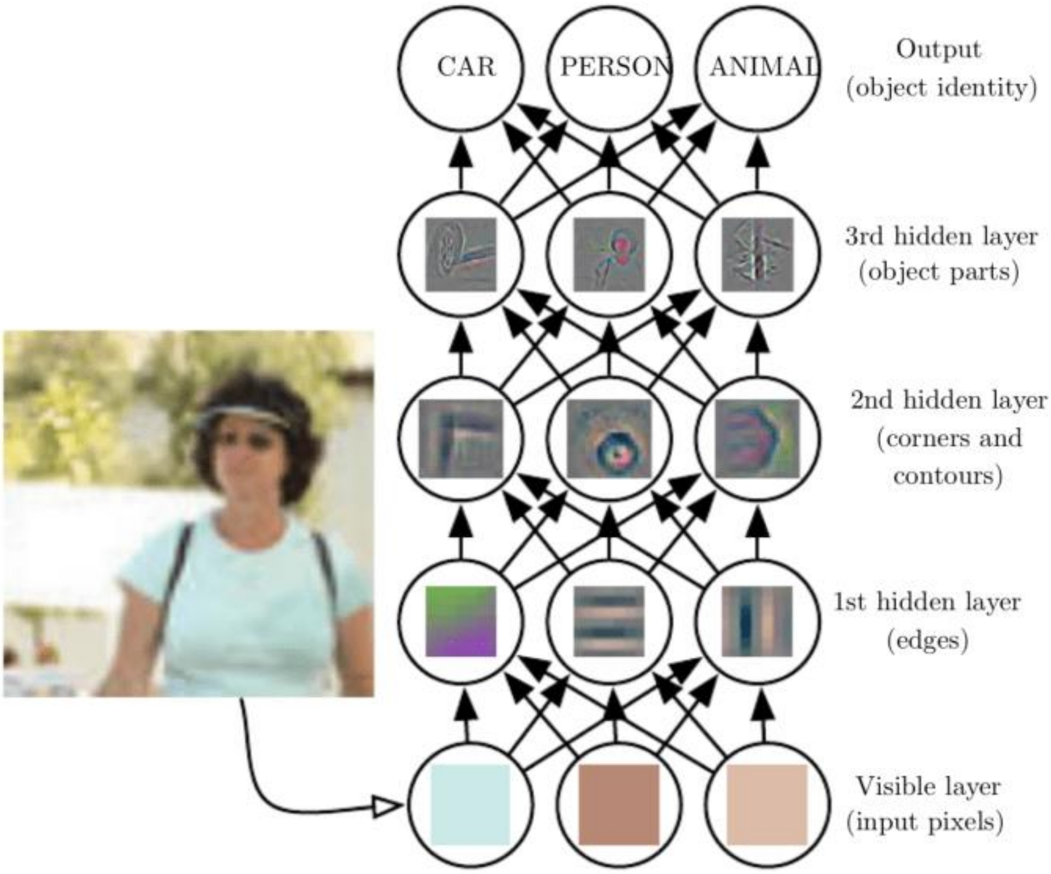
Convolutional Neural Networks :

Convolutional neural networks are composed of multiple layers of artificial neurons. Artificial neurons, a rough imitation of their biological counterparts, are mathematical functions that calculate the weighted sum of multiple inputs and outputs an activation value.

When you input an image in a ConvNet, each layer generates several activation functions that are passed on to the next layer.

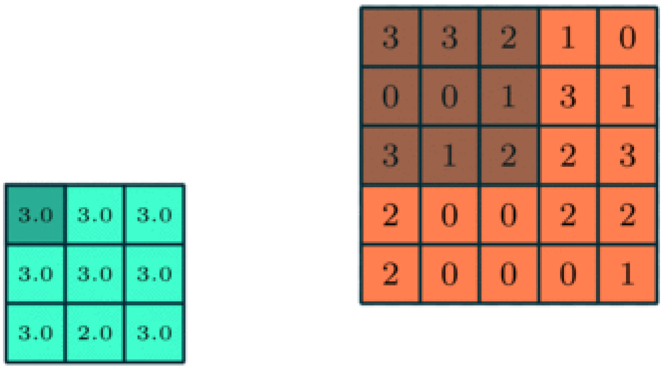
The first layer usually extracts basic features such as horizontal or diagonal edges. This output is passed on to the next layer which detects more complex features such as corners or combinational edges. As we move deeper into the network it can identify even more complex features such as objects, faces, etc.

Based on the activation map of the final convolution layer, the classification layer outputs a set of confidence scores (values between 0 and 1) that specify how likely the image is to belong to a "class." For instance, if you have a ConvNet that detects cats, dogs, and horses, the output of the final layer is the possibility that the input image contains any of those animals.



Pooling Layer:

Similar to the Convolutional Layer, the Pooling layer is responsible for reducing the spatial size of the Convolved Feature. This is to decrease the computational power required to process the data by reducing the dimensions. There are two types of pooling average pooling and max pooling. I’ve only had experience with Max Pooling so far I haven’t faced any difficulties.



3.0	3.0	3.0
3.0	3.0	3.0
3.0	2.0	3.0

3	3	2	1	0
0	0	1	3	1
3	1	2	2	3
2	0	0	2	2
2	0	0	0	1

So what we do in Max Pooling is we find the maximum value of a pixel from a portion of the image covered by the kernel. Max Pooling also performs as a Noise Suppressant. It discards the noisy activations altogether and also performs de-noising along with dimensionality reduction.

On the other hand, Average Pooling returns the average of all the values from the portion of the image covered by the Kernel. Average Pooling simply performs dimensionality reduction as a noise suppressing mechanism. Hence, we can say that Max Pooling performs a lot better than Average Pooling.

Feature Extraction:

All neurons in a feature share the same weights. In this way all neurons detect the same feature at different positions in the input image. Reduce the number of free parameters.

Tensorflow:

The TensorFlow platform helps you implement best practices for data automation, model tracking, performance monitoring, and model retraining. Using production-level tools to automate and track model training over the lifetime of a product, service, or business process is critical to success. TensorFlow provides developers of all skill levels with APIs for desktop, mobile, web, and cloud applications. To get started, refer to the sections below. We can achieve text output and sound output by scanning the number digit and converting it to PNG format using the python3 command in the terminal.