الجامعة الهاشمية

The Hashemite University

Faculty of Engineering

The Hashemite University

Artificial Intelligence and Machine Learning

Dr. Bassam Jamil

Prepared by:

1. Omar Ehab Abu Dayyeh (2136038)
2. Ahmad Mohammad Alhaj (2141147)
3. Muhammad Mustafa Almashayekh (2138237)

May 14, 2025

• Introduction

In this project, we created a fundamental Convolutional Neural Network (CNN) from the ground up using only NumPy. We avoided using deep learning libraries such as TensorFlow and PyTorch for a more engineered approach. The target application is the classification of 19 x 19-pixel grayscale shape images (0-9). The initialization of the model starts with the specification of four unique convolutional kernels. Each kernel is a filter of 3×3 dimensions. The purpose of the kernels is to perform edge and pattern detection on the incoming images. After convolving and max-pooling, the incoming images are reduced to a set of feature maps. Each feature map is a kind of abstract representation of the image that has been optimized for further processing.

The architecture of the neural network is as follows:

1- Image size: Each input image is convolved with 4 filters, followed by max-pooling, 2×2, resulting in a total size of $8 \times 8 \times 4 = 256$ neurons.
2- Concealed layer: A completely connected concealed layer comprising 64 neurons.
3- Output layer: Has 10 neurons; each stands for a shape from 0 to 9.

Every weight was initialized with a random value in the initialization, while the biases in the network were assigned the value of zero in the initialization. The mapping from the input to the hidden layer uses W_1 and b_1. W_2 and b_2 function to map from the hidden layer to the output layer.

This model is trained on a set of premade grayscale images. It does not make any assumption about the structure of the images. What it does is to use a basic neural network to train itself using a set of images, and afterwards to evaluate itself on a set of images that it has never seen before. The model was trained on a dataset of 15,000 grayscale images. Training was conducted over 50 epochs, and a learning rate of 0.05 was used to optimize the weights using backpropagation.

• Functions Used in the Neural Network

1- **Convolve Function:** This function applies a 2-D kernel to the image. For each 3×3 patch, it computes the element-wise multiplication and sums the result. The output is a smaller 2-D array called a feature map.

2- **Max-pooling Function:** Downsamples the feature map by a factor of 2, takes the max value from each 2×2 patch, and then reduces the dimensions while keeping the strongest features.

3- **RELU Function:** Applies RELU, which removes the non-negative values, adding non-linearity.

4- **Softmax Function:** Converts raw scores into probabilities and normalizes the output to be in the range [0,1] and the sum to be 1.

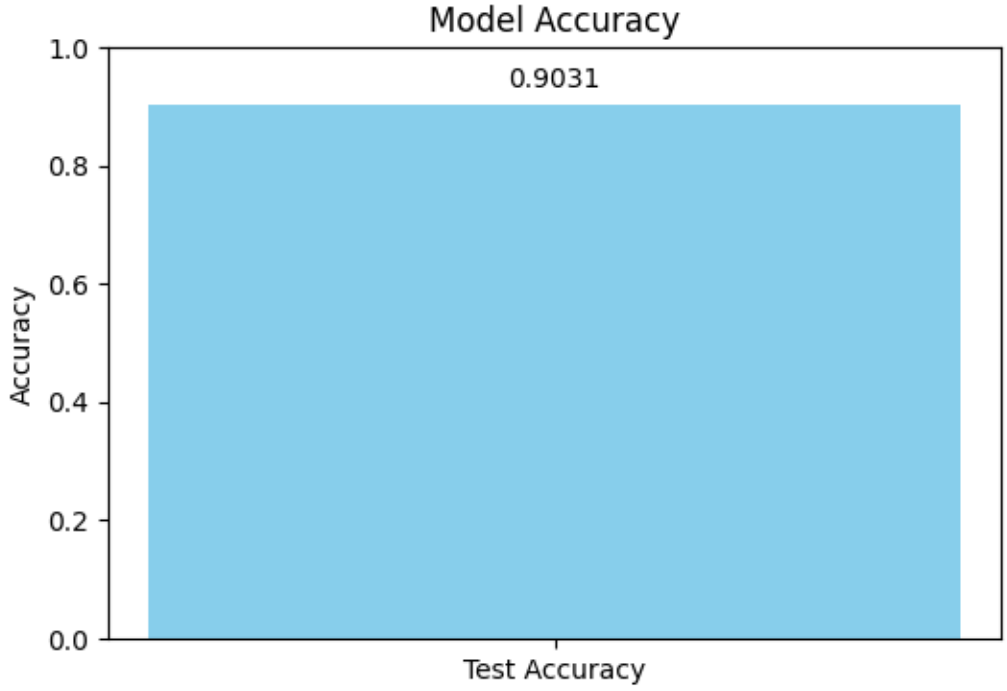
5- **Forward Pass:** This function runs the image through the network:
   a- Convolution + ReLU for each kernel.
   b- MaxPooling on each output.
   c- Flatten all pooled outputs into a 1D array.
   d- Pass through:
      ✓ Hidden layer (with ReLU).
      ✓ Output layer (with Softmax).
   e- Stores intermediate results in a cache for use in backpropagation.

6- **Back Propagation:** Performs gradient descent to update the weights:
   a- Computes gradients from loss (cross-entropy with softmax).
   b- Propagates error backward through the network
   c- Applies gradients to update weights and biases using the learning rate.

- ## Testing Accuracy:

The trained model was tested on 10,000 unseen grayscale images. The model achieved an accuracy of 90.31%, demonstrating its ability to generalize and accurately classify previously unseen data.

- ## Model Accuracy:



- **Confusion Matrix:**