**MNIST CNN TRAINED MODEL DESCRIPTION**

MODEL.SUMMARY:

Model: "sequential\_1"

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Layer (type) Output Shape Param #

=================================================================

conv2d\_4 (Conv2D) (None, 28, 28, 32) 320

max\_pooling2d\_4 (MaxPoolin (None, 14, 14, 32) 0

g2D)

conv2d\_5 (Conv2D) (None, 14, 14, 32) 9248

max\_pooling2d\_5 (MaxPoolin (None, 7, 7, 32) 0

g2D)

conv2d\_6 (Conv2D) (None, 7, 7, 64) 18496

max\_pooling2d\_6 (MaxPoolin (None, 3, 3, 64) 0

g2D)

conv2d\_7 (Conv2D) (None, 3, 3, 64) 36928

max\_pooling2d\_7 (MaxPoolin (None, 1, 1, 64) 0

g2D)

flatten\_1 (Flatten) (None, 64) 0

dense\_2 (Dense) (None, 100) 6500

dense\_3 (Dense) (None, 10) 1010

=================================================================

Total params: 72502 (283.21 KB)

Trainable params: 72502 (283.21 KB)

Non-trainable params: 0 (0.00 Byte)

plot\_model

A diagram of a number of data

Description automatically generated with medium confidence

**ANSLYSIS 1:**

a) This is a Convolutional Neural Network (CNN) model. It's defined using the Sequential API in Keras. The model consists of the following layers:

1. Conv2D layer with 32 filters and a kernel size of (3, 3).

2. MaxPooling2D layer with a pool size of (2, 2).

3. Conv2D layer with 32 filters and a kernel size of (3, 3).

4. MaxPooling2D layer with a pool size of (2, 2).

5. Conv2D layer with 64 filters and a kernel size of (3, 3).

6. MaxPooling2D layer with a pool size of (2, 2).

7. Conv2D layer with 64 filters and a kernel size of (3, 3).

8. MaxPooling2D layer with a pool size of (2, 2).

9. Flatten layer to flatten the output from the previous layer.

10. Dense layer with 100 units.

11. Dense layer with 10 units (output layer).

b) There is one input layer (the first Conv2D layer) and one output layer (the last Dense layer).

c) There are a total of 11 layers in this model.

d) The effect of each layer is as follows:

- Conv2D layers with ReLU activation function: These layers perform convolution operations on the input images to extract features. The ReLU activation introduces non-linearity to the model.

- MaxPooling2D layers: These layers reduce the spatial dimensions of the input data, helping to decrease the computational load and reduce overfitting.

- Flatten layer: Flattens the multi-dimensional output to a one-dimensional array, preparing it for the fully connected layers.

- Dense layers: Fully connected layers that perform classification based on the features extracted by the convolutional layers.

In summary, the convolutional layers extract hierarchical features from the input images, the pooling layers reduce spatial dimensions, and the dense layers perform the final classification. This architecture is commonly used for image classification tasks.

**ANSLYSIS 2:**

a) This is a Convolutional Neural Network (CNN) model represented in Keras using the Sequential API. It consists of the following layers:

1. Conv2D layer with 32 filters and a kernel size of (3, 3).

2. MaxPooling2D layer with a pool size of (2, 2).

3. Conv2D layer with 32 filters and a kernel size of (3, 3).

4. MaxPooling2D layer with a pool size of (2, 2).

5. Conv2D layer with 64 filters and a kernel size of (3, 3).

6. MaxPooling2D layer with a pool size of (2, 2).

7. Conv2D layer with 64 filters and a kernel size of (3, 3).

8. MaxPooling2D layer with a pool size of (2, 2).

9. Flatten layer.

10. Dense layer with 100 units.

11. Dense layer with 10 units (output layer).

b) The input layer is not explicitly mentioned in the summary, but based on the architecture, the input layer would be the first Conv2D layer, which takes input data with the shape (28, 28, 1) (assuming it's designed for processing grayscale images). The output layer is the last Dense layer with 10 units.

c) There are a total of 11 layers in the model.

d) The effects of these layers are as follows:

- Convolutional layers (Conv2D): These layers apply convolutional operations to the input data, extracting features using filters or kernels. The number of filters determines the depth of the output volume.

- MaxPooling2D layers: These layers reduce the spatial dimensions of the input volume, helping to decrease computational complexity and control overfitting by retaining the most important information.

- Flatten layer: This layer converts the multidimensional output to a one-dimensional array, preparing the data for the fully connected layers.

- Dense layers: These layers are fully connected layers that perform classification based on the features learned by the convolutional layers. The last Dense layer produces the final output with 10 units, representing the classes in a classification task.

The overall architecture of the CNN allows it to learn hierarchical features from the input images, starting with low-level features in the early layers and gradually combining them to form high-level features in the deeper layers, ultimately leading to a classification decision in the output layer.

**ANSLYSIS 3:**

a) The provided model is a Convolutional Neural Network (CNN) implemented using the Keras Sequential API. Let's break down the layers:

1. Convolutional layer (`conv2d`):

- Output shape: (None, 28, 28, 32)

- Parameters: 320 (weights) + 32 (biases) = 352

2. MaxPooling layer (`max\_pooling2d`):

- Output shape: (None, 14, 14, 32)

- No trainable parameters (it performs a down-sampling operation)

3. Convolutional layer (`conv2d\_1`):

- Output shape: (None, 14, 14, 32)

- Parameters: 9248 (weights) + 32 (biases) = 9280

4. MaxPooling layer (`max\_pooling2d\_1`):

- Output shape: (None, 7, 7, 32)

- No trainable parameters

5. Convolutional layer (`conv2d\_2`):

- Output shape: (None, 7, 7, 64)

- Parameters: 18496 (weights) + 64 (biases) = 18560

6. MaxPooling layer (`max\_pooling2d\_2`):

- Output shape: (None, 3, 3, 64)

- No trainable parameters

7. Convolutional layer (`conv2d\_3`):

- Output shape: (None, 3, 3, 64)

- Parameters: 36928 (weights) + 64 (biases) = 36992

8. MaxPooling layer (`max\_pooling2d\_3`):

- Output shape: (None, 1, 1, 64)

- No trainable parameters

9. Flatten layer (`flatten`):

- Output shape: (None, 64)

- No trainable parameters

10. Dense layer (`dense`):

- Output shape: (None, 100)

- Parameters: 6500 (weights) + 100 (biases) = 6600

11. Dense layer (`dense\_1`):

- Output shape: (None, 10)

- Parameters: 1010 (weights) + 10 (biases) = 1020

b) The model has one input layer (implicitly the first layer, which is the `conv2d` layer) and one output layer (the last layer, which is the `dense\_1` layer).

c) There are a total of 11 layers in the model.

d) The effect of each layer in the model is as follows:

- Convolutional layers (`conv2d`, `conv2d\_1`, `conv2d\_2`, `conv2d\_3`): These layers perform feature extraction by applying convolutional operations to the input data.

- MaxPooling layers (`max\_pooling2d`, `max\_pooling2d\_1`, `max\_pooling2d\_2`, `max\_pooling2d\_3`): These layers down-sample the spatial dimensions of the data, reducing computational complexity and focusing on the most important features.

- Flatten layer (`flatten`): This layer flattens the output from the previous layers into a one-dimensional vector, preparing it for the fully connected layers.

- Dense layers (`dense`, `dense\_1`): These layers are fully connected and perform classification based on the features extracted by the convolutional layers. The last dense layer produces the final output with 10 units, representing the class probabilities for a classification task.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **CNN Model** | **Run time type** | **Learning rate** | **Batch size** | **No. epochs** | **Training time** | **Accuracy (%)** |
| 1 | T4 GPU | 0.002 | 64 | 20 | 0:01:44.1289 | 98.88 |
| 2 | T4 GPU | 0.002 | 64 | 50 | 0:04:28.6597 | 99.1 |
| 3 | T4 GPU | 0.002 | 64 | 100 | 0:08:52.1190 | 99.14 |
| 4 | CPU | 0.002 | 64 | 20 | 0:20:46.4519 | 99.17 |
| 5 | CPU | 0.002 | 64 | 50 | 0:47:33.3843 | 99.2 |