**CNN Mnist images classifier from scratch**

The program/ system needs to:

* Classify the handwritten images into 10 distinct categories
* Be able to be visualize prediction in real time
  + Visualize kernel
  + Visualize convolution actions
  + Visualize FF model’s activations
* Log all metrics from every step of the process
  + Forward pass performance metrics
  + Backpropagation metrics/ logs
  + Loss metrics
* Have customizable number of layers both for FF and CN networks
* Have customizable kernel sizes
* Support dropout for all layers
* Support rotation, shifting techniques
* The model can be saved for later inference
* Support evaluation metrics
* Shuffle input dataset
* Each layer is a method that can be called from a class

**Model architecture**

**Typical CNN for MNIST**

1. **Input Layer:** 28×28×1 grayscale image
2. **Conv Layer 1:** 3×3 kernel, ReLU activation
3. **Max Pooling 1:** 2×2
4. **Conv Layer 2:** 64 filters, 3×3 kernel, ReLU activation
5. **Max Pooling 2:** 2×2
6. **Flatten Layer:** Converts feature maps to a 1D vector
7. **Fully Connected (Dense) Layer:** 128 neurons, ReLU activation
8. **Dropout (Optional):** 025 probability to prevent overfitting
9. **Output Layer:** 10 neurons (one for each digit), softmax activation

|  |  |
| --- | --- |
| **Layer** | **Output** |
| Input (28,28,1) | (28,28,1) |
| Conv2D (3\*3, 32) | (26,26,32) |
| MaxPooling (2\*2) | (13,13,32) |
| Conv2D (3\*3, 64) | (11,11,64) |
| MaxPooling (2\*2) | (5,5,32) |
| Flatten | (1600) |
| Dense (128) | (128) |
| Dropout (0.2) | (128) |
| Dense (10, softmax) | (10) |

Sources:

<https://www.geeksforgeeks.org/applying-convolutional-neural-network-on-mnist-dataset/>