UNIVERSITY OF NEBRASKA OMAHA

Computer Science

CSCI8590-001: Fundamentals Of Deep Learning

By: Dr. ***Xin Zhong***

<u>Assignment: 2</u>

Submitted By: Zerin Shaima Meem

NUID: 77548102

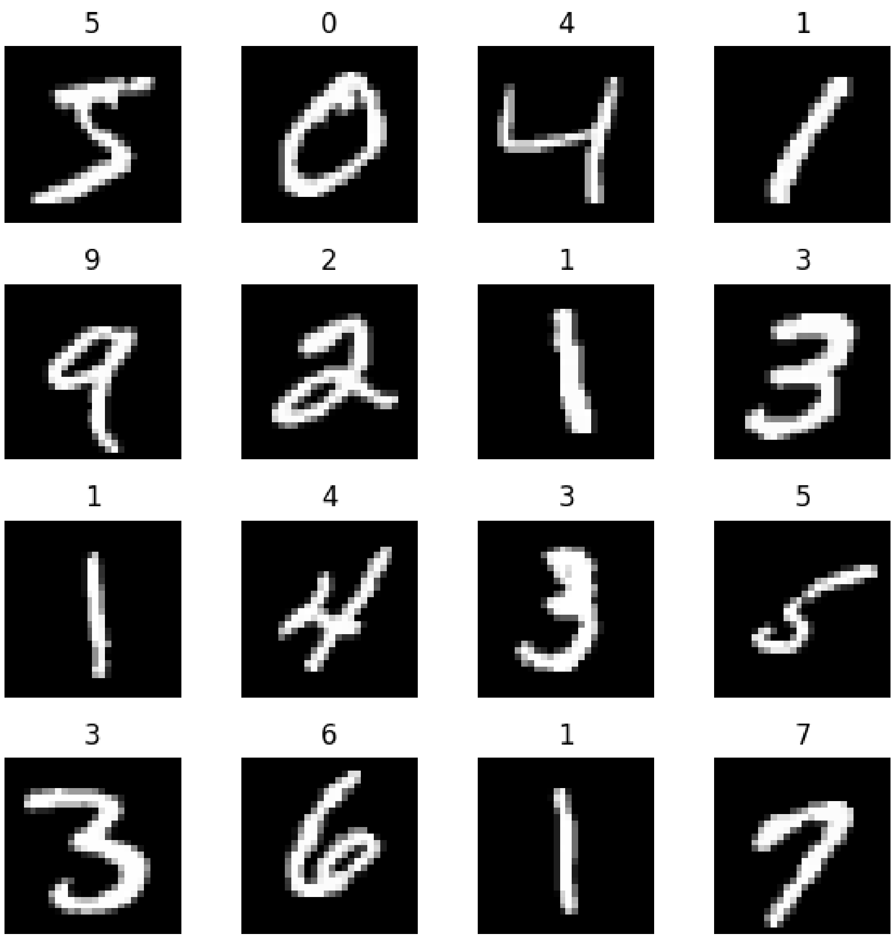
University of Nebraska Omaha

Runnable Code: [Google Colab Link]

Required Output:

- Displayed images

```
Train samples: 60000
Test samples: 10000
Original image shape: (28, 28)
Final image shape: (28, 28, 1)
```



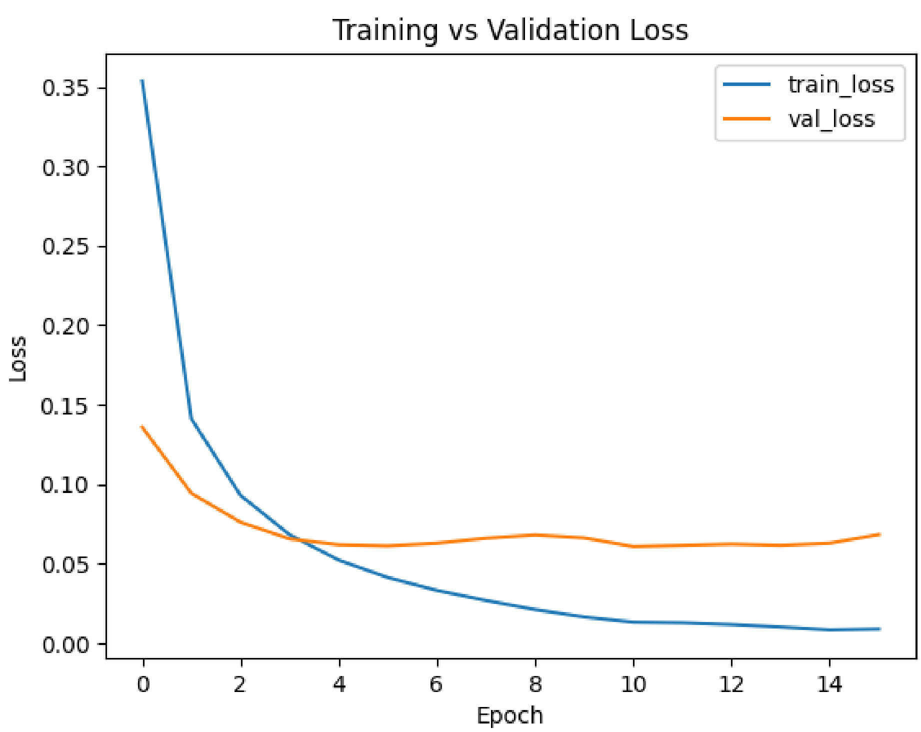
- <u>Model Summary</u>

Model: "LeNet5"

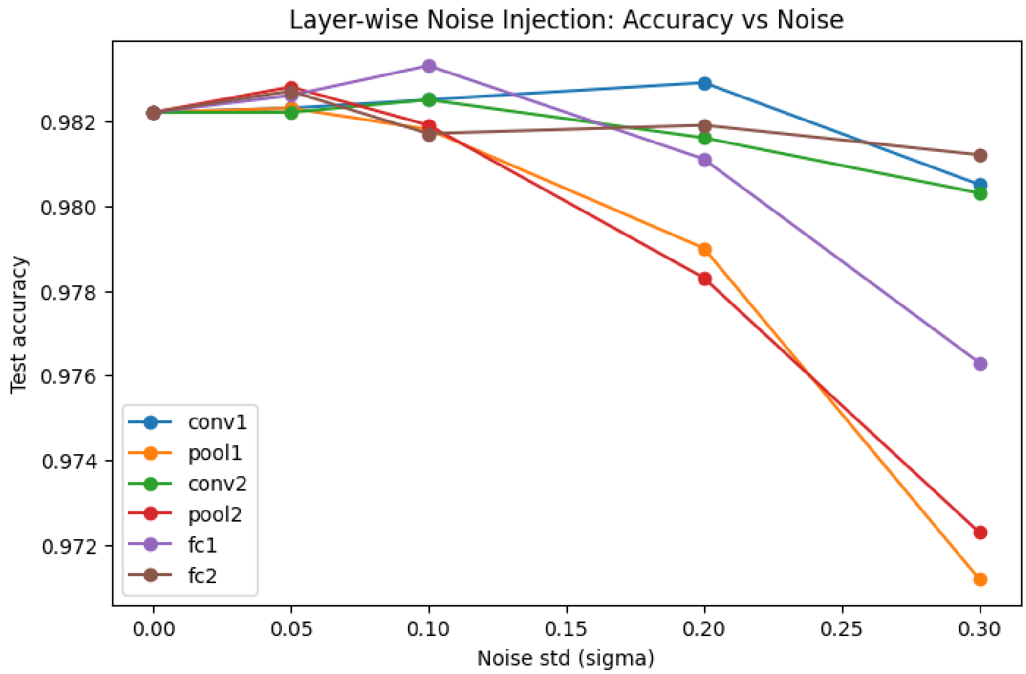
Layer (type)	**Output Shape**	**Param #**
input (InputLayer)	(None, 28, 28, 1)	0
conv1 (Conv2D)	(None, 28, 28, 6)	156
pool1 (AveragePooling2D)	(None, 14, 14, 6)	0
conv2 (Conv2D)	(None, 10, 10, 16)	2,416
pool2 (AveragePooling2D)	(None, 5, 5, 16)	0
flatten (Flatten)	(None, 400)	0
fc1 (Dense)	(None, 120)	48,120
fc2 (Dense)	(None, 84)	10,164
output (Dense)	(None, 10)	850

Total params: 61,706 (241.04 KB)
Trainable params: 61,706 (241.04 KB)
Non-trainable params: 0 (0.00 B)

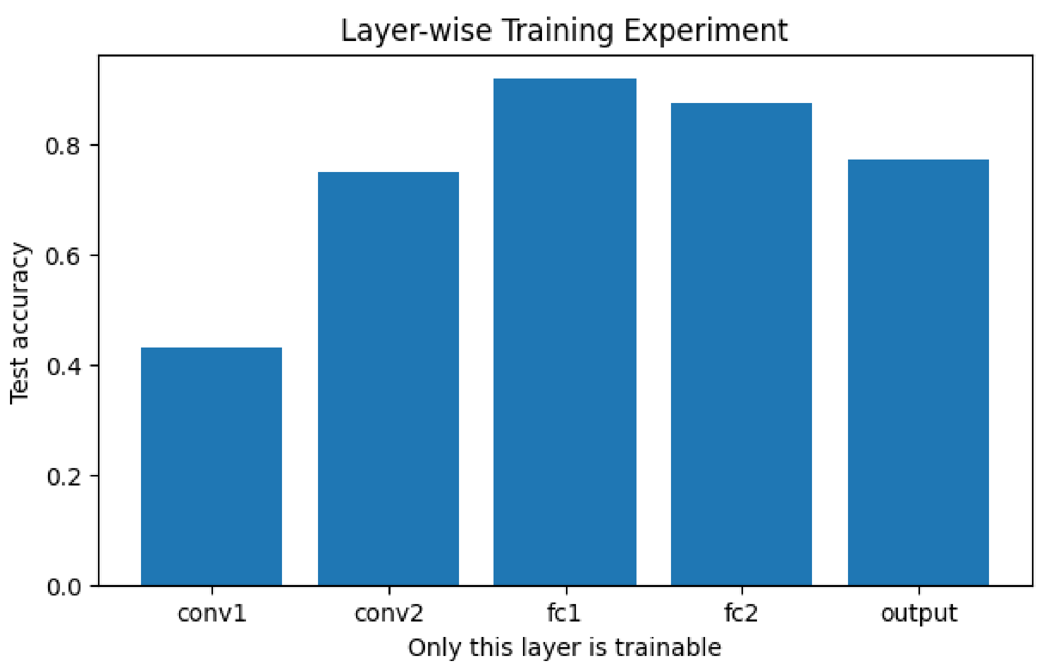
- <u>Training curves</u>



- Layer-wise Noise Injection Experiment



- Layer-wise experiment plots



Discussion:

- **Implementing LeNet-5:**
  - I implemented LeNet-5 using the Functional API, starting with convolution and pooling layers to extract image features, followed by fully connected layers for classification. The model was trained with softmax output and cross-entropy loss to correctly predict the 10 MNIST digit classes.
- **Training curves & stopping epoch:**
  - Used “sparse_categorical_crossentropy” loss function, because this is a multiclass classification problem, and all labels are in integer form.

  - The validation loss decreases steadily during the first 16 epochs, indicating effective learning. After approximately epoch 16, validation loss plateaus and begins to fluctuate slightly, while training loss continues decreasing. This suggests the model starts to overfit beyond this point. Therefore, epoch 16 was selected as a reasonable stopping point to balance learning and generalization.
- **Noise injection:**
  - After inserting noises of different level, we can see that, CNN layers are not equally sensitive to noise. Pooling layers are more affected to noise, due to compression. On the other hand, early convolution layers are robust due to distributed representations and later, the fully connected layers are sensitive to noise.
- **One-layer training:**
  - Training only one-layer shows that layers depend on each other. A single layer cannot learn a full representation without upstream/downstream support.