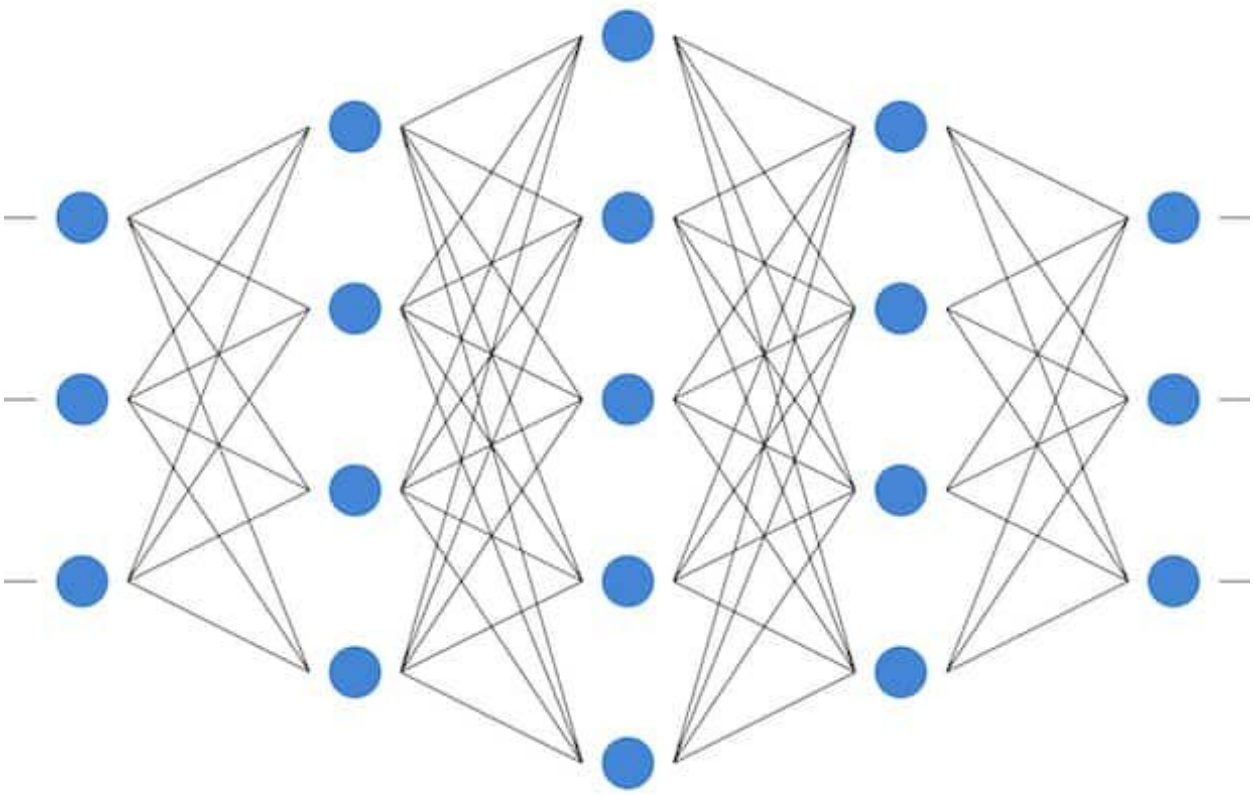
Visual Recognition

Assignment-4

IMT2016118, Shubham Gupta



Part 1:

★ Convolutional Neural Networks

Although programming frameworks make convolutions easy to use, they remain one of the hardest concepts to understand in Deep Learning. A convolution layer transforms an input volume into an output volume of different size.

- **Convolution2D**
    - Zero Padding
        - Zero-padding adds zeros around the border of an image
        - It allows me to use a CONV layer without necessarily shrinking the height and width of the volumes. Which is important for building deeper networks since otherwise the height/width would shrink as you go to deeper layers. An important special case is the "same" convolution, in which the height/width is exactly preserved after one layer.
    - Convolve window
        - Took an input volume
        - Applies a filter at every position of the input
        - Outputs another volume (usually of different size)
    - Convolution forward
        - In the forward pass, I took many filters and convolve them on the input. Each 'convolution' have given me a 2D matrix output. And I stack these outputs to get a 3D volume:
        - Formulas relating the output shape of the convolution to the input shape is:

$$n_H = \lfloor \frac{n_{H_{prev}} - f + 2 \times pad}{stride} \rfloor + 1$$

$$n_W = \lfloor \frac{n_{W_{prev}} - f + 2 \times pad}{stride} \rfloor + 1$$

$$n_C = \text{number of filters used in the convolution}$$

- **Pooling2D**

  The pooling (POOL) layer reduces the height and width of the input. It helps reduce computation, as well as helps make feature detectors more invariant to its position in the input.

  - Pooling forward
    - The formula which I used to bind the output shape of the pooling to the input shape is:

$$n_H = \lfloor \frac{n_{H_{prev}} - f}{stride} \rfloor + 1$$

$$n_W = \lfloor \frac{n_{W_{prev}} - f}{stride} \rfloor + 1$$

$$n_C = n_{C_{prev}}$$

    - Max-pooling layer: **MaxPooling2D** is used which slides an (f, f) window over the input and stores the max value of the window in the output.
    - Polling layer used hyperparameters such as window size f and no parameter for backpropagation to train

Original Image:

Output image using keras:

Original Image:

Output image after implementing Convulution2D, Pooling2D, Fully Connected layer from scratch:

Result:

Because we have used the same weight in both the implementation, one without using keras and another using Convolution2D, Pooling2D, Fully Connected Layer so the output image is supposed to be coming same and that is coming also as we can see the above images.

Part 2:

Followed steps:

- First, a random array is generated of shape (m, 4, 4, 8) with (mean = 0) and (Standard Deviation = 1).
- Random array gets mapped to random MNIST images.
- A decoder is trained using the random array as input and MNIST images as a label.
- Model is used to predict the test data.
- Output images stored as “decoded_imgs”.
- Original image and the reconstructed image gets displayed.

Observation:

- The output image is noisy and incorrect because the random image is mapped to random MNIST images which will lead to almost zero uniqueness for particular labelled images.