Introduction to Machine Learning

Lab 5: Convolutional Neural Network

Student ID: 111060013 Name: 劉祐廷

1. **Explain why ReLU is typically preferred over Sigmoid as the activation function in the convolutional block?**
   ReLU converts negative values to zero, reducing the computational load of the model and effectively acting as a drop node mechanism. When used in hidden layers, it introduces some randomness, helping to mitigate the vanishing gradient problem. On the other hand, the sigmoid function maps values to a range between 0 and 1, making it less suitable as an activation function for hidden layers.

2. **Describe how you design the CNN architecture and your findings in choosing parameters such as filter_size and pool_size for each layer?**
   I designed a convolutional neural network with three convolutional layers, using channel sizes of [1, 16, 32, 64]. Each convolutional layer is followed by a ReLU activation layer and a max-pooling layer with a pool size of 2. After the convolutional and pooling operations, the output is flattened and passed through three dense layers to produce the final output. During testing, I observed that applying max-pooling too many times or using a pool size that is too large makes the loss difficult to converge. Therefore, I settled on a pool size of 2. Additionally, I noticed that using too few dense layers also leads to difficulties in loss convergence. However, I did not observe significant differences when changing the filter size in the convolutional layers. Based on articles I've read, using an odd filter size is often recommended because it allows the filter to have a central position, aligning better with the feature map.

3. **Calculate and compare the number of learnable parameters between the CNN model and the NN model you designed for binary classification in Lab4. For simplicity, omit the bias parameters and calculate only the weights.**
   In lab 4, my layer node list is [784, 1568, 784, 196, 4]. There are 784 * 1,568 + 1,568 * 784 + 784 * 196 + 196 * 4 = **2,613,072** learnable parameters. In lab 5, my convolution layer node list is [1, 16, 32, 64] and filter size is 2, which has 1 * 16 * 2 * 2 + 16 * 32 * 2 * 2 + 32 * 64 * 2 * 2 = 10304 learnable parameters. And my dense layer node list is [1024, 256, 64, 1], which has 1024 * 256 + 256 * 64 + 64 * 1 = 278,592. There are **288,896** learnable parameters in total, which is much less than the number of learnable parameters in lab 4, saving lots of computing efforts.