111062118 江佩霖

Machine Learning

Lab5 Report

I. Explain why ReLU is typically preferred over Sigmoid as the activation function in the convolutional block?

計算效率高，減少 overfitting，避免 vanishing gradient problem。

II. Describe how you design the CNN architecture and your findings in choosing parameters such as filter_size and pool_size for each layer?

1. Filter size : 通常使用 3*3 的 filter，用小的計算成本捕捉到特徵。
2. Pool size : 2*2 或 3*3 ，有效降低維度(測試後使用)。

III. 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?

```
# build the model
# ...
model = Model()
model.add(Conv(filter_size=3, input_channel=1, output_channel=8, pad=1, stride=1))
model.add(Activation("relu", None))
model.add(MaxPool(pool_size=2, stride=2))
model.add(Flatten())  # Flatten the output of the last pooling layer
model.add(Dense(2048, 1))  # Fully connected layer
model.add(Activation("sigmoid", None))
```

CNN : (3*3)*8 + 2048 = 72 + 2048 = 2110

```
### START CODE HERE ###
loss_function = "cross_entropy"
# layers_dims = [784, 128, 64, 32, 16 , 1]
# activation_fn = ["relu","relu", "relu", "relu", "sigmoid"]
layers_dims = [784, 196, 98, 50, 25, 10, 1]
activation_fn = ["relu", "relu", "relu", "relu", "relu", "sigmoid"]
learning_rate = 0.01
num_iterations = 1
num_iterations = 100
print_loss = True
print_freq = 200
decrease_freq = 100
decrease_proportion = 0.99
# You might need to use mini_batch to reduce training time in this part
# batch_size = int(X_train.shape[1]*0.05)
batch_size = 64
losses = []
```

NN : 784*196 + 196*98 + 98*50 + 50*25 + 25*10 + 10*1 = 179282