LIU, Dingdong
20583503

Programming Assignment 2

COMP5212
November 9, 2022

Report

Description of MLP and CNN Models

Model parameters are listed below:

```
CNN(
  (conv1): Conv2d(in_channels=3, out_channels=64, kernel_size=(3, 3),
  stride=(1, 1), padding=(1, 1))

  (conv2): Conv2d(in_channels=64, out_channels=128, kernel_size=(3, 3),
  stride=(2, 2), padding=(1, 1))

  (conv3): Conv2d(out_channels=128, out_channels=256, kernel_size=(3, 3),
  stride=(2, 2), padding=(1, 1))

  (conv4): Conv2d(out_channels=256, out_channels=256, kernel_size=(3, 3),
  stride=(2, 2), padding=(1, 1))

  (fc1): Linear(input_dims=4096, out_dims=1024, bias=True)
  (fc2): Linear(input_dims=1024, out_dims=1024, bias=True)
  (fc3): Linear(input_dims=1024, out_dims=10, bias=True)
)
```

Note that for CNN model, zero padding was used.

```
MLP(
  (fc1): Linear(input_dims=3072, output_dims=1024, bias=True)
  (fc2): Linear(input_dims=1024, output_dims=512, bias=True)
  (fc3): Linear(input_dims=512, output_dims=256, bias=True)
  (fc4): Linear(input_dims=256, output_dims=128, bias=True)
  (fc5): Linear(input_dims=128, output_dims=64, bias=True)
  (fc6): Linear(input_dims=64, output_dims=32, bias=True)
  (fc7): Linear(input_dims=32, output_dims=10, bias=True)
)
```

Training Loss & Final Test Accuracy of Two Models

The training loss of two models over each epochs is shown as follows in Figure 1. After 50 epochs, both model were evaluated on test set. Model parameters, test accuracy are reported as follows:

1. For MLP model,
   - Final accuracy is 61.71% on training set and 51.32% on test set.
   - The model is optimised with Adam algorithm and 0.0005 for learning rate. Batch size=64.
2. For CNN model,
   - Final accuracy is 93.52% on training set and 84.95% on test set.
   - The model is optimised with Adam algorithm and 0.0005 for learning rate. Batch size=64.

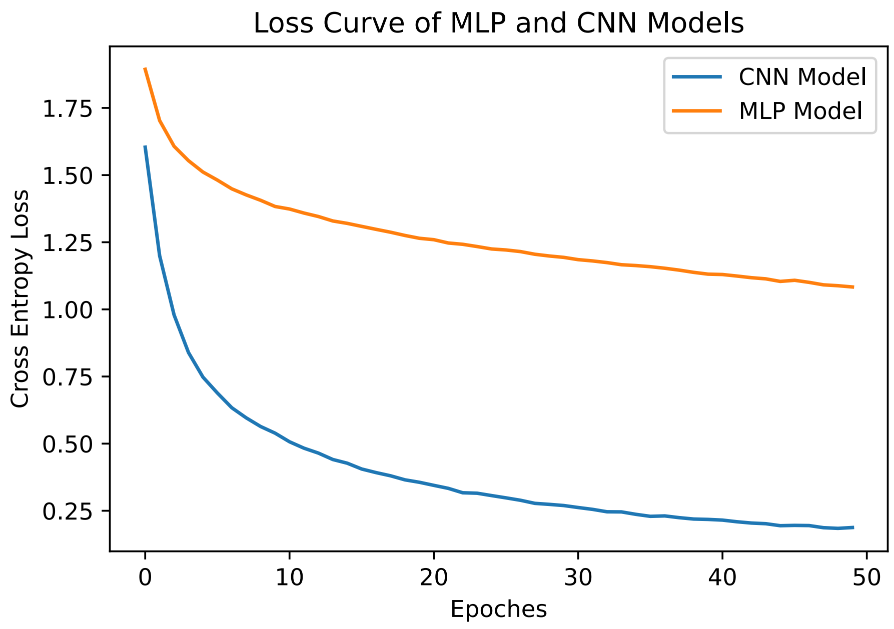

Figure 1: Training loss of MLP and CNN model over 30 epochs

Comparing MLP and CNN model

CNN model outperform MLP model in terms of test accuracy. But MLP model seems more efficient to compute. A possible reason is CNN's kernel structure can better capture the feature of images (i.e., edge features)

Model with/without Activation Function

In this section, I did an experiment on training MLP, CNN models with and without non-linear activation functions. The loss curve are plotted as following (in Figure 2):

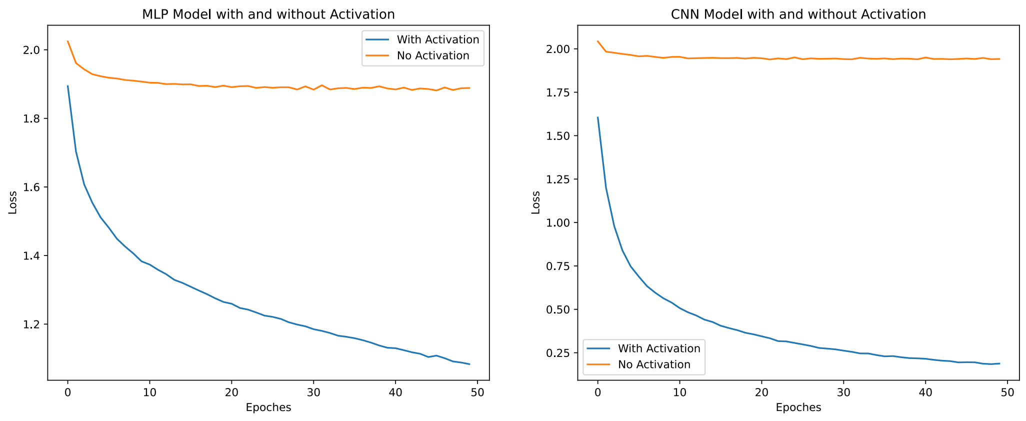

Figure 2: Training loss of MLP and CNN model with/without non-linear activation function

From the figure, we can know that without activation function, the loss stop to decrease in a early stage during training. The final accuracy without activation function cannot really catch up with the performance with a normal MLP/CNN model.

This finding probably echos the effect of non-linear activation function: combining multiple hyperplanes to construct a non-linear hypothesis.