Keshav Gupta

2021101018

For training the baseline model, I used a convolutional neural network (CNN).

This is the baseline model class I used. It consists of three convolutional blocks, each followed by standard max pooling and a non-linear ReLU activation. After these layers, I applied average pooling and added a linear classifier to predict the logits for 37 classes. Since the dataset contained only four digits, the maximum sum was 36, and the minimum was 0, resulting in a total of 37 classes.

```
class SumMNISTModel(nn.Module):
    def __init__(self, config):
        super().__init__()
        self.config = config
        # self.backbone = ResNetEncoder(config.backbone)
        self.conv1 = nn.Conv2d(1, 32, kernel_size=3, padding=1)
        self.conv2 = nn.Conv2d(32, 64, kernel_size=3, padding=1)
        self.conv3 = nn.Conv2d(64, 128, kernel_size=3, padding=1)
        self.linear = nn.Linear(128, 37)

    def forward(self, x):
        x = F.relu(F.max_pool2d(self.conv1(x), kernel_size=2))
        x = F.relu(F.max_pool2d(self.conv2(x), kernel_size=2))
        x = F.max_pool2d(self.conv3(x), kernel_size=2)
        x = x.mean(-1).mean(-1)
        logits = self.linear(x)

        return logits
```

I split the training data into 2 parts and used 20000 samples for training, and 10000 samples for validating. The model takes time to converge and the val loss suddenly starts to decrease after a certain point.

The hyperparameters used are

learning_rate: 0.001
batch_size: 2048
devices: [0, 1, 2, 3]

The optimizer used is Adam. The effective batch size used is 8192.

I finally got accuracy of **24.49%** on baseline CNN.

Validation acc curve ->