Name: Zichen Li Andrew ID: zichenli

Model Architecture

The architecture my model based on is **ResNet**. The building block I used is just the basic residual block. I experimented with the number of layers, the number of output channels at each layer, and the stride used in each block. Specifically, I removed the pooling layer in the beginning and decreased the number of residual blocks with stride = 2.

The following picture shows my model architecture:

```
class Network(nn.Module):

    def __init__(self, feat_dim=2):
        super(Network, self).__init__()

        self.layers = []

        self.layers.append(nn.Conv2d(in_channels=3, out_channels=128, kernel_size=3, padding=1, stride=1, bias=False))
        self.layers.append(nn.BatchNorm2d(128))
        self.layers.append(nn.ReLU(inplace=True))

        self.layers.append(IdentityBlock(in_channels=128, out_channels=128, stride=1))
        self.layers.append(IdentityBlock(in_channels=128, out_channels=128, stride=1))

        self.layers.append(IdentityBlock(in_channels=128, out_channels=256, stride=2))
        self.layers.append(IdentityBlock(in_channels=256, out_channels=256, stride=1))

        self.layers.append(IdentityBlock(in_channels=256, out_channels=512, stride=1))
        self.layers.append(IdentityBlock(in_channels=512, out_channels=512, stride=1))

        self.layers.append(IdentityBlock(in_channels=512, out_channels=1024, stride=2))
        self.layers.append(IdentityBlock(in_channels=1024, out_channels=1024, stride=1))

        self.layers = nn.Sequential(*self.layers)

        self.linear_label = nn.Linear(1024, 2300, bias=True)
```

Loss function

Cross Entropy Loss and Center Loss (weight = 0.003, feat_dim=2)

Optimizer

Adam with initial learning rate 3e-4 for cross entropy loss. **SGD** with unchanged learning rate 0.5 for center loss

Training process

I first trained the model for 5 epochs. The model seems to be stuck at a local minimum after 5 epochs. Then I trained the model one epoch at a time with a smaller learning rate (divide by 10) for Adam. My model reached accuracy of 84.4% for validation and 76.6% for test.