Rahul Ghosh **Assignment#4**
Student ID: 5476965 CSci 5561

CONVOLUTIONAL NEURAL NETWORK

Methods

Single-layer Linear Perceptron

A single-layer linear perceptron is used as a model to classify the images. Euclidean distance between the prediction and true label is used as the objective function and Stochastic gradient descent is used as an optimizer to train the model.
Optimizer Parameters: learning_rate = 0.01 decay_rate = 1
The results are shown in Figure 1.

Single-layer Perceptron

A single-layer perceptron is used as a model to classify the images. Cross entropy loss between prediction and true label is used as the objective function and Stochastic gradient descent is used as an optimizer to train the model.
Optimizer Parameters: learning_rate = 0.6 decay_rate = 0.9
The results are shown in Figure 2.

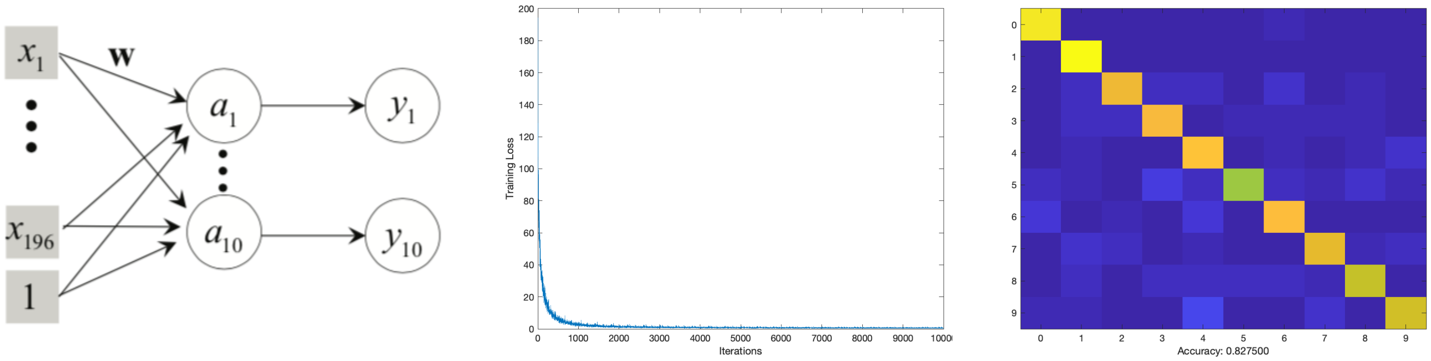
Multi-layer Perceptron

A multi-layer perceptron with 1 hidden layer of 30 units is used as a model to classify the images. Cross entropy loss between prediction and true label is used as the objective function and Stochastic gradient descent is used as an optimizer to train the model.
Optimizer Parameters: learning_rate = 0.6 decay_rate = 0.99
The results are shown in Figure 3.

Convolutional Neural Network

A convolutional neural network with 1 convolutional layer followed by ReLU and max-pooling is used. Cross entropy loss between prediction and true label is used as the objective function and Stochastic gradient descent is used as an optimizer to train the model.
Optimizer Parameters: learning_rate = 0.7 decay_rate = 0.9
The results are shown in Figure 4.

Results

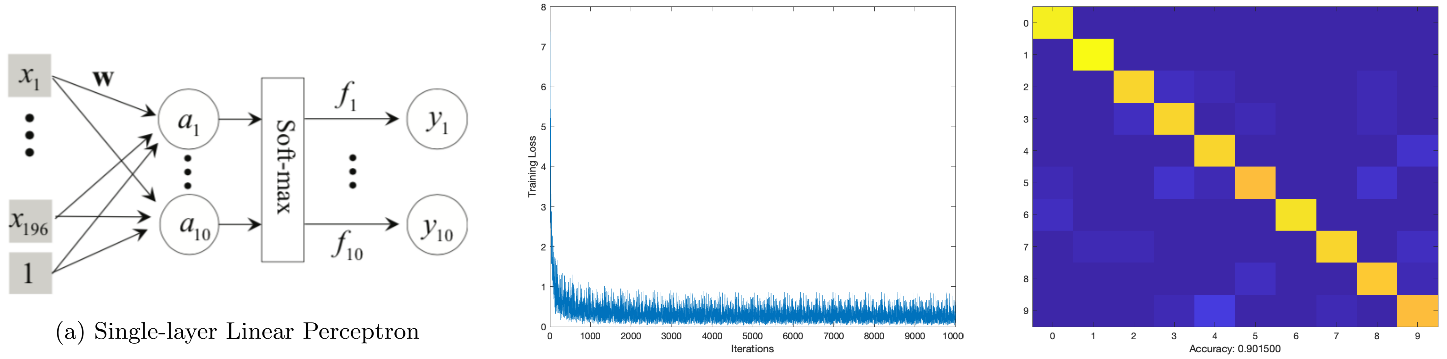


(a) Single-layer Linear Perceptron

(b) Training Loss

(c) Tiny Image

Figure 1: Single-layer Linear Perceptron

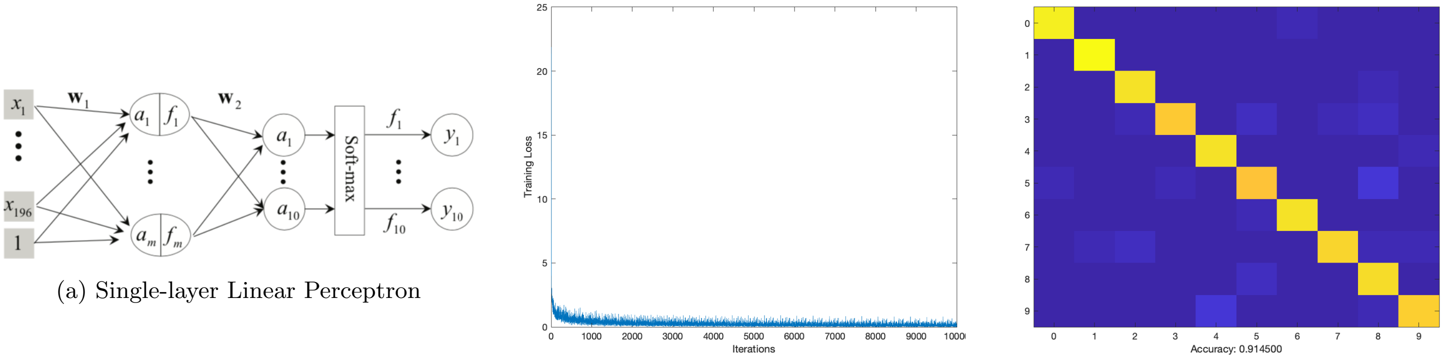


(a) Single-layer Linear Perceptron

(b) Training Loss

(c) Tiny Image

Figure 2: Single-layer Perceptron

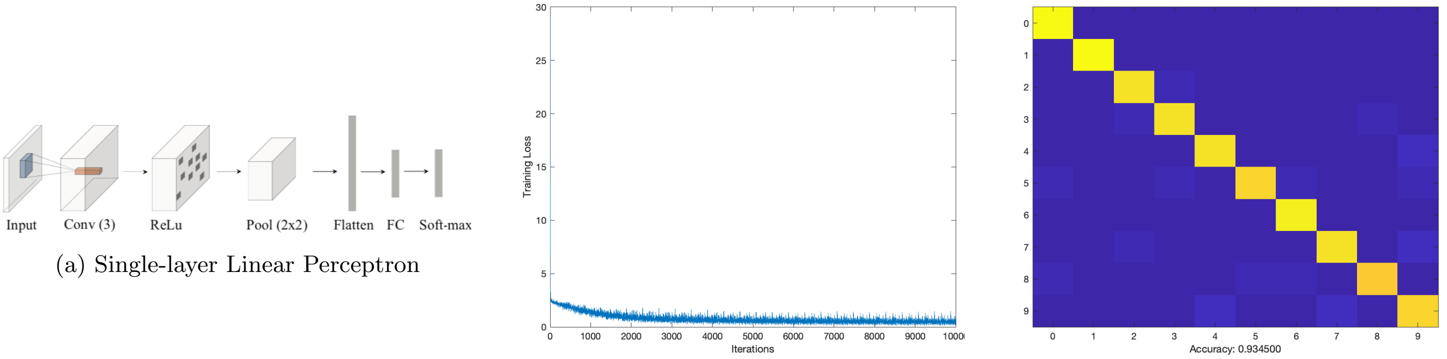


(a) Single-layer Linear Perceptron

(b) Training Loss

(c) Tiny Image

Figure 3: Multi-layer Perceptron



(a) Single-layer Linear Perceptron

(b) Training Loss

(c) Tiny Image

Figure 4: Convolutional Neural Network