CSC411: Assignment #2

Due on Monday, February 18, 2018

Chawla Dhruv, Lokman Sabrina

February 7, 2018

Part 1

Dataset Description

The dataset consists of a set of images with each image representing a digit from 0 to 9.

Each image is 28x28 and has a black background with the digit handwritten in white.

Some of the images are straightforward to analyze and decipher while some are more tricky to ascertain what the handwriting is trying to represent (consider the 7th image from the left of the digit 5).

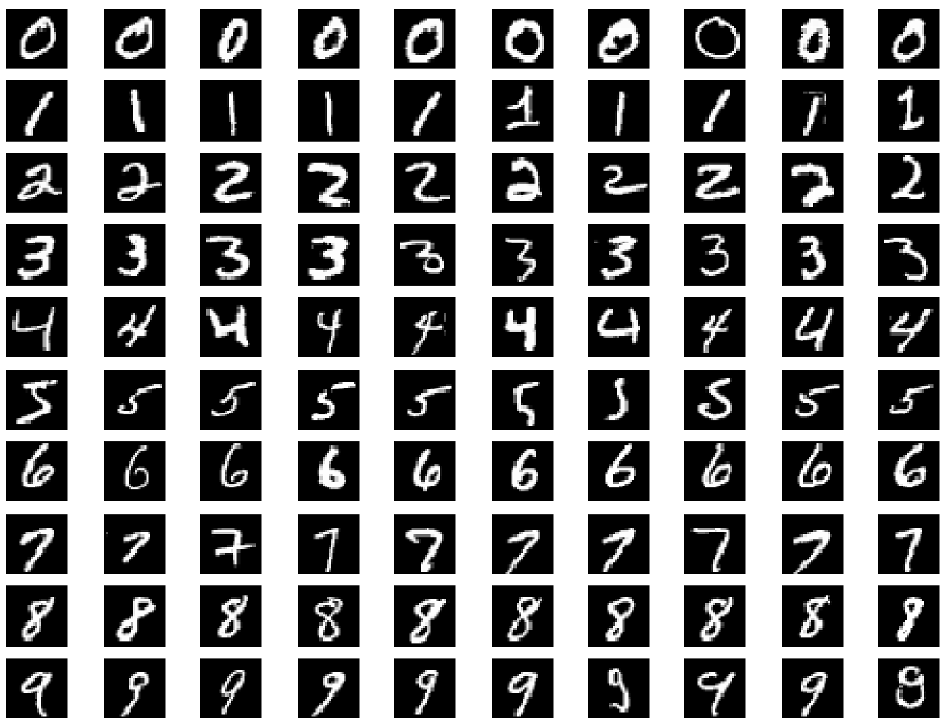

Figure 1: Images from the MNIST dataset

Part 2

Compute Simple Neural Network

The function for computing dimple neural network (no hidden layers) is in `mnist_handout.py` and is reproduced here.

```python
def compute_simple_network(x, W, b):
    '''Compute a simple network (with no hidden layers)
    '''
    o = np.dot(W, x) + b
    return softmax(o)
```

An example of the function in action is seen below (and in `part2()` of `digits.py`).

```
>>> x = np.random.rand(784, 1)
>>> W = np.random.rand(10, 784)
>>> b = np.random.rand(10, 1)
>>> compute_simple_network(x, W, b).shape
(10, 1)
```

`compute_simple_network(x, W, b)` returns the output layer of the neural network.

Part 3

Cost Function: Sum of negative log-probabilities of all training cases

Part 3(a)

Compute $\frac{\partial C}{\partial w_{ij}}$ (gradient of cost function with respect to a single weight)

For one training case, the cost function is (from slide 6 of One-Hot Encoding Lecture):

$$C = -\sum_{j} y_j log p_j \tag{1}$$

For M training examples, the cost functions gets modified to:

$$C = -\sum_{m=1}^{M} \sum_{j} y_j log p_j \tag{2}$$

We also know that (from slide 7 of One-Hot Encoding Lecture),

$$p_i = \frac{e^{o_i}}{\sum_j e^{o_j}} \tag{3}$$

and

$$\frac{\partial p_i}{\partial o_i} = p_i(1 - p_i) \tag{4}$$

This implies,

$$\frac{\partial C}{\partial o_i} = \sum_{j} \frac{\partial C}{\partial p_j} \frac{\partial p_j}{\partial o_i} = \sum_{j} \tag{5}$$

Part 4

Part 5

Part 6

Part 7

Part 8

Part 9

Part 10