CMPE 258, Spring 2018

Midterm Exam #2

Due 11:59pm on Sunday, April 15th, 2018

Notes

This Midterm exam should be submitted in Canvas as a format of ipython (Jupyter) notebook.
The filename should be 'exam_2_yourFirstName_LastnName.ipynb'.
The submitted ipynb should be executable without any extra work and supposed be finished within 60 minutes.
You should not discuss how to solve the problem with other students and the work should be your own. If any portion of the code is similar to others, it will be treated as cheating.
Please do not use any library except tensorflow, tersorflow.nn. pandas, numpy, and matplotlib.pyplot.

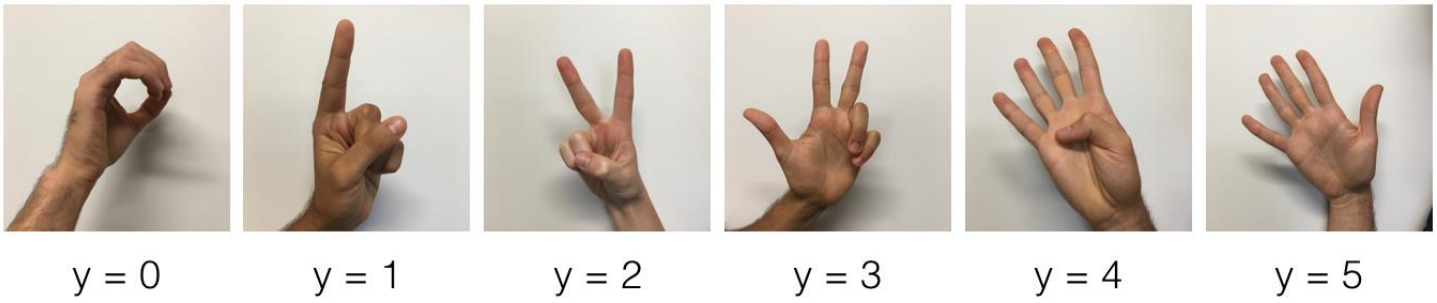
Grading policy

If the code can be executable without any error within 50 minutes, score will be assigned as following formula.
Score = accuracy of testing data + 10 (print out of CNN architecture) + 10 (plot of cost versus number of iteration)

If extra effort is needed to get reasonable result (whatever it is), 5 to 20 points for each event will be deducted.
Re-submitting is available until April 22th, but 20 points will be deducted every re-submitting after April 15th.

Dataset

Download data files (exam2_train_x.npy, exam2_train_y.npy, exam2_test_x.npy, exam2_test_y.npy) from canvas/files/exam_2.
This data is subset of SIGNS Dataset from Coursera (Deep Learning specialization).

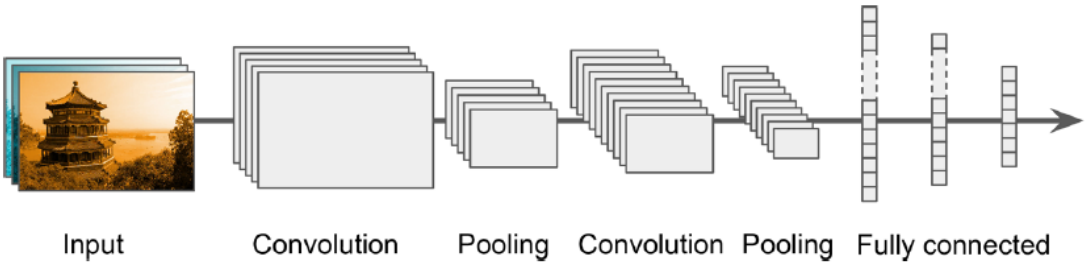


Each picture is a RGB image with 64 by 64 pixels.

Image classification using Convolution Neural Network

In Assignment_5, we used pandas and numpy to build CNN architecure.
For exam_2, we will use Tensorflow.

Please build convolution neural network model using at least 2 convolution layers, 2 pooling layers, 2 fully connected layers.



Source : Hands-on ML, Aurelien Geron

Here is one example for convolution neural network architecture.

Layer	Type	Size	Channels	Kernel size	Stride	Padding	Function
0	Input	64 x 64	3				
1	Convolution (C1)	32 x 32	8	4 x 4	2	1	ReLU
1	Pooling (P1)	28 x 28	8	5 x 5	1	0	max
2	Convolution (C2)	13 x 13	16	4 x 4	2	0	ReLU
2	Pooling (P2)	9 x 9	16	5 x 5	1	0	Avg
3	Flatten (F3)	1296					
4	Fully connected (F4)	108					ReLU
5	Fully connected (F5)	6					Sigmoid

1. (10pts) Define functions
You may need to define the following functions.
- One-hot encoding
- Create placeholders
- initialize parameters
- forward propagation with regularization
- compute cost

2. Load data

Using Jupyter notebook, load the data.

3. (10pts) Initialize parameters (Weights, bias for each layer)

Please initialize weight coefficients and bias terms for each layer.
Please make sure the size (dimension) of each Weights and bias.
Please consider optimum initialization method depending on Activation function.

4. (40pts) Build Convolution Neural Network model

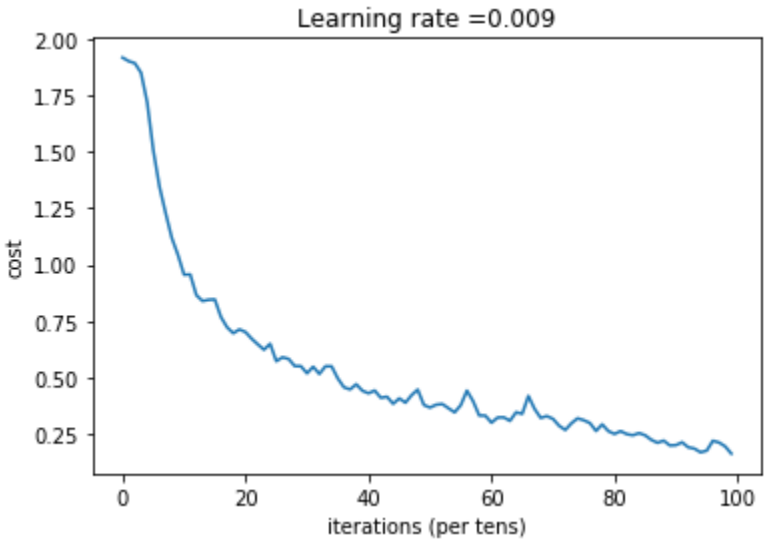
Please build your CNN model with forward propagation function.

Implement the forward_propagation function as following:
CONV2D -> Activation -> POOL -> CONV2D -> activation -> POOL -> FLATTEN -> FULLYCONNECTED -> FULLYCONNECTED

Please print (or write) your CNN architecture model as the example table (10 pts).

5. (20pts) Optimization of Convolution Neural Network model

Please optimize your model using your preferred optimization method.
Please print out cost with number of iteration as below (10 pts).



5. (20pts) Predictions

Please predict SIGNS using softmax function.
Please print out the accuracy for the prediction using training data set and testing data set.