CMPE 258, Spring 2018

Assignment #5

Due 11:59pm on Sunday, March 8th, 2018
Deadline for re-submitting is 11:59pm on Sunday, March 15th, 2018

Notes

This programming assignment should be submitted in Canvas as a format of ipython notebook (assignment_5_yourFirstName_LastnName.ipynb).
You can discuss how to solve the problem with other students or search internet or other resources, but the work should be your own. If any portion of the code is similar to others, it will be treated as cheating.
The submitted ipynb should be executable without any extra work.

Please do not use any library except pandas, numpy, and matplotlib.pyplot.

Grading policy

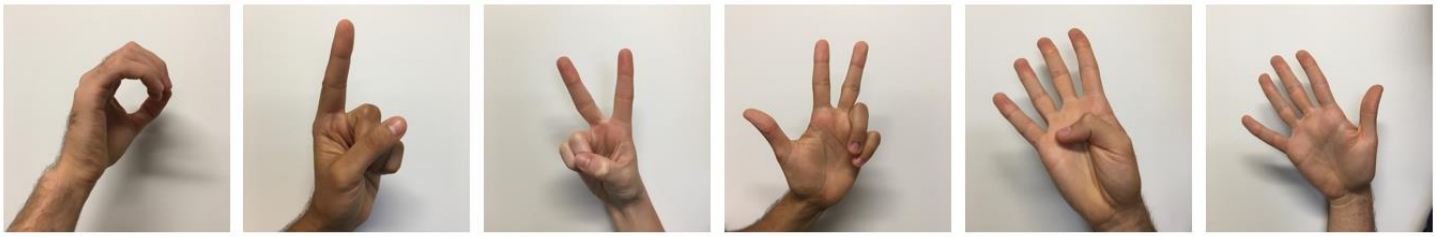
The code is supposed to be executable without any extra effort and produce reasonable result within 50 minutes.
If the code cannot be executable with any error or taking more than 50 minutes, 50 points will be assigned.
If the code can be executable without any error within 50 minutes, score will be assigned as following formula.

Score = (10 – cost) * 10

Re-submitting is available until March 15th, but 10 point will be deducted every re-submitting after March 8th.
If extra effort is needed to get reasonable result (whatever it is), 5 to 10 points will be deducted.

Dataset

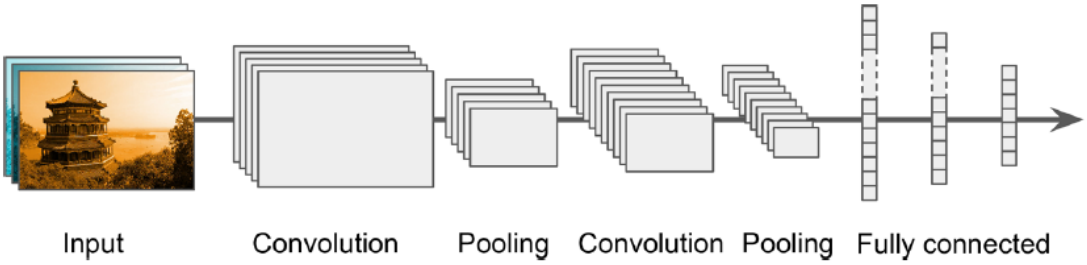
Download data files (ex5_train_x.npy and ex5_train_y.npy) from canvas/files/assignment_5.
This data is subset of SIGNS Dataset from Coursera (Deep Learning specialization).



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

Image classification using Convolution Neural Network

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 model.

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. **(70pts) Define functions**

Please define the functions which are needed for CNN architectures. The following list is a suggestion.

- One-hot encoding
- Activation forward (Relu, sigmoid)
- Compute cost
- Zero pad
- Convolution with single step
- Convolution forward (for all data)
- Pooling forward (max, average)
- Pooling backward (max, average)
- Activation backward (Relu, sigmoid)
- Convolution backward (for all data)
- Forward propagation (including all steps)
- Backward propagation (including all steps)
- Parameter updating (Gradient descent or other optimization method)

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.
You may use your trained weights and bias. In this case, please make sure to submit the trained weights and bias as one separate file (*para_yourFirstName_LastnName)*

4. (20pts) Optimization of Convolution Neural Network model

Please build your model with forward propagation procedure and backward propagation procedure.
Please print out the size (dimension) of each layer (C1, P1, C2, P2, F3, F4, F5)
Please print your CNN architecture model as the above table.
Please optimize your model using a learning rate and number of iteration.
Please print out cost with number of iteration. It may take long time to calculate. You may limit the number of iteration less than 10.