EE6337: Deep Learning, Spring 2019

Indian Institute of Technology Hyderabad

HW 0, 60 points. Assigned: Sunday 17.02.2019. **Due: Friday 22.02.2019 at 11:59 pm.**

In this assignment you will implement each of the components of a Convolutional Neural Network (CNN) from scratch (i.e., without using built-in functions for convolution, pooling, non-linearity, padding, striding). Your implementation must accept an image input and generate an output vector. Use *random weights* for filter kernels and fully connected layers. Specifically, implement the following:

1. A *convolution function* that accepts as input an image, a filter kernel, stride, padding and the non-linear function. The function must convolve the input image (after padding if specified) with the kernel (at the specified stride size) and generate an output activation after applying the specified non-linearity. Display the input image, the filter kernel and the output activation map. Ensure that your function can accept color images (i.e., a volume) and a corresponding kernel volume. (10)

2. A *pooling function* that accepts as input the activation map output from the convolution function and a pooling function. The function must output the appropriately pooled activation map. Display the input activation map and the pooled output. (5)

3. A *convolution layer function* that accepts as input a volume (image or activation maps), filter kernels, stride, padding and the non-linear function. The function must convolve the input volume (after padding if specified) with each of the kernels (at the specified stride size) and generate an output activation volume after applying the specified non-linearity. Display the input image, the filter kernels and the output activation maps. Verify that the output of this function does indeed have the expected size ($width \times height \times depth$) as discussed in class. (10)

4. A *pooling volume function* that accepts as input the activation map volume, the pooling function and generates a pooled output volume. Display the input and output volumes. (5)

5. A *composition of convolution layer functions* that accepts as input an image volume, number of convolution layers, kernels for each layer, strides and padding for each layer, non-linearity for each layer, and the pooling function for each layer. Display the activation map generated at each of the convolution layers. (10)

6. An *unraveling function* that accepts as input the activation map volume output by the pooling layer and generates a vector of a specified size. It is important to note that this function has a weight matrix associated with it whose size is chosen such that the input and desired output sizes are matched. (5)

7. A *multilayer perceptron (MLP) function* that accepts as input a vector, the number of hidden layers, the size of each hidden layer, the non-linear function, and the size of the output layer. This function should generate an output vector of the specified size. Generate the output with and without the *softmax* function applied to the output layer. (10)

8. Finally, connect the *composition of convolution layer functions* to the *MLP function* via the *unraveling function* to complete the forward path of a CNN. This function should generate an output vector for a given input image volume. (5)