Homework 3

Please upload your assignments on or before March 12, 2021.

- You are encouraged to discuss ideas with each other; but
- you **must acknowledge** your collaborator, and
- you **must compose your own** writeup and/or code independently.
- We **require** answers to theory questions to be written in LaTeX, and answers to coding questions in Python (Jupyter notebooks)
- Upload your answers in the form of a single PDF on Gradescope.

---

1. **(2 points)** *Exercises in convolution.* Suppose that the input is a 1D array (or signal; call it x). Find (i.e., design by hand) a convolutional filter that produces the following output (call it y). All you need to do is to specify the filter weights.

   a. (Approximate) derivatives:
   $$y[n] = x[n+1] - x[n-1].$$

   b. (Approximate) second derivatives:
   $$y[n] = x[n+2] - 2x[n] + x[n-2].$$

   c. (Approximate) integrals:
   $$y[n] = x[n-\Delta] + x[n-\Delta+1] + \ldots + x[n+\Delta-1] + x[n+\Delta].$$

   d. cross-correlations:
   $$y[n] = \sum_i x[i]x[i+n].$$

2. **(2 points)** 1×1 *convolution.* In the definition of a convolutional layer, if we have I input channels, J output channels, and the filter size is chosen to be $\Delta = 0$, show that the operation is equivalent to applying a regular dense layer in the *channel* domain. What is the number of trainable parameters in this layer?

3. **(2 points)** *The IoU metric.* Recall the definition of the IoU metric (or the Jaccard similarity index) for comparing bounding boxes.

   a. Using elementary properties of sets, prove that the IoU metric between any two pair of bounding boxes is always a non-negative real number in $[0, 1]$.

   b. If we represent each bounding box as a function of the top-left and bottom-right coordinates (assume all coordinates are real numbers) then argue that the IoU metric is *non-differentiable* and hence cannot be directly optimized by gradient descent.

4. **(4 points)** In this programming exercise, we will explore the performance of three different object detection networks. We will be using Detectron2, Facebook AI's object detector library; here is the repository. It will be helpful to go through the excellent tutorial here.

    a. Download the following test image (a picture of pedestrians in Central Park). We will run two different detectors on this image.

    b. First, consider the COCO Keypoint Person Detector model with a ResNet50-FPN base network, which is trained to detect human silhouettes. This can be found in the Detectron2 Model Zoo in the "COCO Keypoint" table. Use this model to detect as many silhouttes of people in the test image as you can. You may have to play around with the thresholds to optimize performance.

    c. Second, repeat the above procedure, but with the Mask R-CNN model with ResNet50-FPN backbone, available in the Model Zoo in the "COCO Instance Segmentation" table. This time, you should be able to detect both people as well as other objects in the scene. Comment on your findings.

    d. It appears that the balloons in the test image are not being properly detected in either model. This is because the COCO dataset used to train the above models does not contain balloons! Following the tutorial code above, start with the above pre-trained Mask R-CNN model and train a balloon detector using the (fine-tuning) balloon image dataset provided here. Test it on the original test image and show that you are now able to identify all the balloons.