Homework 5 – Segmentation with U-Net – Due Monday 12/9 at 11:59 PM

For this homework, I am again going to allow you to work in groups of 2. Only one submission is required per group. Be sure to include both group member's names clearly in your submission. If you would like to work alone you may.

By now you should have a firm understanding of what it means to do a segmentation task using a CNN. I have provided you with a segmentation dataset. This dataset is taken from my index metacarpal segmentation project. Data can be found in the class folder under *Homework_5_files/data/*. You should take some time to explore the data to get a good idea of how it is formatted and what it looks like. I have written two templates for you to use for this assignment. One is for the cross-entropy model, one is for the dice metric model. Your U-Net implementation should be the same for both of them. These templates include a dataset class definition, a dice metric loss definition, and a very rough outline for training. You should be able to use this for your segmentation task. **No data augmentation is required for this homework**.

For your project you must write U-Net as it is originally designed by the authors. Please follow the description given in class. If you have any confusion, it may be useful to consult the original paper (can be found in the lecture slides). I have also included some tips on getting started within the U-Net lecture.

I would first like you to **implement dice loss.** Please use the version of dice loss I have provided instead of trying to make your own or finding one online (most of the online versions are incorrect and it can be very difficult to write your own correct version). When using the dice loss, your network should have output depth of exactly 1. After you have a successful network developed using this loss (including hyperparameter tuning), **please record your results in terms of your loss on training, validation, and testing set within your README.**

I would then like you to **implement pixel wise cross entropy loss**. When using cross entropy loss your network should have output depth of exactly 2. After you have a successful network developed using this loss (including hyperparameter tuning), please **record your results in terms of your loss on training, validation, and testing set within your README.**

For both runs of your network (one with pixel wise cross entropy loss and one with dice loss), I want you to measure your model effectiveness using the dice metric. You can compute the dice metric by doing 1 minus the forward pass of your dice loss function (regardless of whether or not you are using dice loss to train). **For the dice metric to perform correctly, the last layer in your U-Net implementation must be a sigmoid activation layer**. **Wherever you record your loss in your README, please also include the dice metric as the measure of your model effectiveness. Submit your README and all code files as a zip file either through the cp-assignment script or on blackboard.**

BONUS – For this homework I am giving the opportunity to earn bonus credit. Nothing described below is required to get full credit on the homework. To earn bonus credit on this assignment, simply improve the network so that it performs better. (You will still need to include results from the original network). In class, we talked about numerous ways to improve on the original U-Net network. You can also try to come up with your own improvements. This could include anything from data augmentation to network architecture changes to post processing. Tuning traditional model hyperparameters does not count for bonus credit as that should be done as part of the original assignment. The more improvements you make, the more bonus credit you can get. However, they **must actually be improvements**. Changes to the network which do not result in an increase in effectiveness (or a decrease in the number of epochs necessary to reach maximum effectiveness) will not count towards bonus credit. If you are pursuing bonus credit, in your code **you should have one network definition that is the original network.** Then you should create **a second network definition that is your improved network.** This is so we can ensure you wrote the original network correctly. **Please detail any architecture changes you made and their corresponding improvements in loss and effectiveness in your readme.**