CSC420: Assignment #3

Saman Motamed - John Chen

University of Toronto — Due Date: October 29, 2019

Introduction

This assignment aims to get you started on practical deep learning approaches to computer vision tasks. Please read the info section below for some guidelines, requirements and suggestions about assignment 3.

Info:

- This assignment is likely to take more time than your first two assignments. **Start early!**
- It is recommended you use PyTorch or Tensorflow with a Keras back-end for this assignment.
- Include each step you take to solve the problems in your report, include your code in the report along with input and outputs where needed.
- You are to implement the architectures and functions such as U-NET and data augmentation from scratch and not use built-in models / libraries that do the job for you.
- Your final submission files should include a complete report, along with working code and saved weights of your trained network for each step. We should be able to run your model, using your weights on new input images without any error.

1 Image Segmentation

In part one of this this assignment, you will familiarize yourself with implementing an image segmentation architecture based on **This paper by Ronnenberger et. al**. You will work on the task of segmentation and improving your model's performance through different methods. You are given a set of cat images and masks. **All images should must be resized to 128 * 128 in your code.** The goals is to improve the performance of the model on the cat segmentation dataset. The challenge is the size of the dataset which is quite small. Small amount of training data is a problem in deep learning tasks and in this assignment, you will work towards practices that try to improve results of models that have access to only small datasets.

1.1 Implement U-NET - 2.5 Points

Read the paper by Ronnenberger and implement the architecture described in the paper to train for the task of medical image segmentation. You should implement two different loss functions, compare the performance of the two and rationalize why one might work better or worse than the other. (Both should be reasonable functions for the task of segmentation). Randomly divide the cat images into **(Train - 0.7)** and **(Test - 0.3) – It is VERY IMPORTANT that you keep your train and test data separate and do not train your model on your test data**. Report the performance of your trained U-NET (trained on train images) on the Test set images using **Sørensen–Dice-coefficient**. We will also test your model on cat images that are not provided to you.
Hint*: Sørensen–Dice-coefficient could be useful as one of the loss functions you will implement

1.2 Data Augmentation - 1.25 point

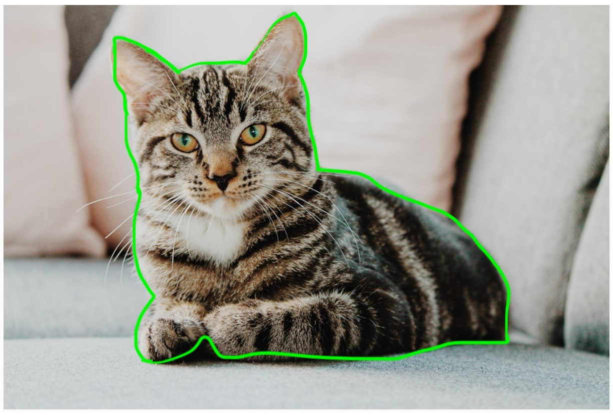
Data augmentation extends the size of your dataset by doing random flips, rotations, zooms, etc. They can improve performance of the model in exchange for training time and resource allocation. Write a data augmentation function that performs at least 4 different augmentation techniques on your input images. Train and test your U-net again on the augmented training set. Report the performance (Dice score). for images in the test set, include the image, ground truth masks and predicted masks of your model..
For all train - test instances, keep your test set the same for all steps after your first randomized split into train and test sets so your comparisons between each steps makes sense

1.3 Transfer Learning - 2.75 point

Transfer learning is one of the most practical ways of dealing with small datasets, given we have access to some other dataset. First, familiarize yourself with transfer learning. **Here** is a quick guide. In this part, you will find an online dataset of images and corresponding masks of your own choosing. First, you train your u-net on this new dataset. Then you perform the task of transfer learning to help your cat segmentation model by using what was learned from the other dataset. Explain how you perform the transfer learning and include your reasoning in your report for every step you choose to take in this part, include your code and results of the model trained using transfer learning on your test dataset. Report the performance. for images in the test set, include the image, ground truth masks and predicted masks outputted by your model.
Do not use your augmentation function in this part

1.4 Visualizing segmentation predictions - 0.5 point

Here, you have to write code that takes in the test images of the cat pictures, along with their corresponding binary masks predicted by your trained model and apply contours around the cat. The final result should look something like the example below. Include your code and 3 samples from your test set. If you have failed segmentation, include one as one of the 3 images.



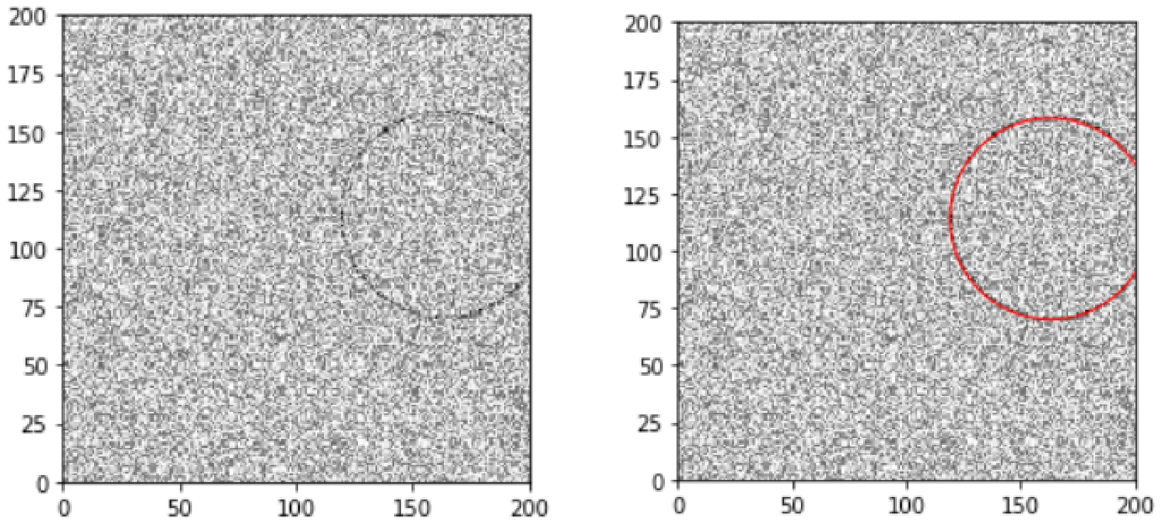
2 Bounding Box Design

This question is intended to give you end-to-end experience designing a deep learning computer vision solution for an open-ended problem. There is no one correct solution, and you are encouraged to be as creative as you like and use any (reasonable) libraries!

Given the following noisy image of a circle, we would like to find the best possible bounding polygon for the circle. The metric for evaluation will be Intersection over Union (IOU). Examine the starter code given in

`main.py`

for more information. **Note: make sure you have a clear separation between train set and test set!**



2.1 Problem definition - 0.5 point

What are the (input, output) pairs to your neural network? What is your loss function? Why have you represented the problem this way? *Comment: There are many ways to formulate the problem (some of which might be better than others). It helps to rapidly prototype and try out different implementations!*

2.2 Implementation - 2.5 points

Implement and build your network. It must use less than 200 000 training data points, and fewer than 2 million total parameters. Print out a visualization of your network, and explain your architecture. You will receive most of the mark (2.0 out of 2.5) for achieving reasonable performance (mean IOU 0.5) and explaining your architecture. A perfect mark on this section is reserved for submissions achieving significantly better results. **If you do not cleanly separate the training and test dataset you will receive 0 on this section.**

2.3 IOU Optimization - 0.5 point

Try experimenting with optimizing IOU directly (that is, use IOU as your loss function). Write briefly about what happens, and explain why these results occur.

2.4 Visualization and error analysis - 1 point

Visualize the images. Plot the predicted bounding circles over the images, along with the IOU for that prediction. Investigate where your network makes mistakes and comment.

3 Hot Dog or Not Hot Dog - 0.5 point

A friend says that they have built a binary classifier for hot dog identification using a deep convolutional neural network architecture. You give them a picture and ask them to run the network on it. The model outputs a single scalar, c, which is the result of running a final sigmoid layer on the outputs. Was the picture a hot dog or not? Explain. *Hint: it does not matter what the value of c is*