Introduction to Neural Networks 67103

Assignment 2

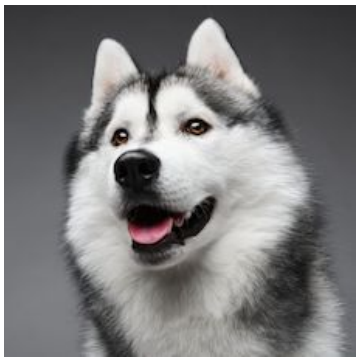
Due: 7/1/2018

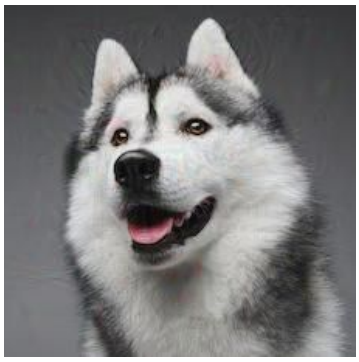
In the practical part of this assignment you will implement some techniques for visualizing and fooling CNNs, in the spirit of the techniques presented in the lecture. Your implementation and experiments should be done using TensorFlow. As a basis you should start from this simple pretrained AlexNet natural image classifier.

A. **Visualization of neurons via input optimization.** The provided AlexNet CNN has five convolution layers (some followed by local response normalization and/or pooling steps) and three fully connected layers. Implement a routine that visualizes neurons in any specified conv or fc layer by optimizing the input image, such that the selected neuron achieves a desired target activation. The optimization should consist of an additional term that acts as a regularizer, for example by keeping the norm of the input image small, similarly to the technique of Simonyan et al. 2014. More sophisticated regularization approaches, such as the ones described by Yosinski et al., are optional, but they will probably result in more interesting visualizations.

B. **Natural image statistics**. Modify the above solution to use a regularization based on natural image statistics to visualize the neurons of the last pre-softmax layer (fc8). Specifically, it is known that the Fourier power spectrum of natural images falls off as 1/w, where w is the frequency. While Tensorflow supports fast Fourier Transform, it is advised to define this term over a reduced-size version of the unknown image.

C. **Fooling a network**. Find an image which is correctly classified by AlexNet (as defined by its final output), and infers its top class with a high probability. Now, implement a routine that would modify the input image as little as possible, such that the modified image will be classified to a





completely different class. Try ones that originally received low and high probabilities. For example, the left image above was originally classified as “husky” with probability 0.59, but the right one has been modified such that the network now infers its category to be “ostrich” with a

probability above 0.98. The minimal change done to the image here should be implemented by running a few gradient descent steps (minimal change, maximal influence) until the target class receives the highest probability.

D. **CHANGED: Classification heat map**. Given an image whose top class is known (and visually obvious), zero out a small block at different locations across the image, and evaluate the class's probability for each of these modified images. Use the resulting values to construct a 2D heat map showing the effect of each region of the image over the image classification.

Theoretical Questions

1. Convolution (yet again). What would be the support (receptive field) of a neuron in the last layer of the following networks: (i) conv1 7x7, (ii) conv1 3x3, conv2 3x3, conv3 3x3, (iii) conv1 5x5, conv2 3x3, (iv) conv1 5x5, stride 2, conv2 3x3. How many filter weights are needed in cases (i) and (iii), how many multiplication operations are needed to implement these two cases? One way to approach this problem is to understand how the support of two filters grow when they are convolved with each other (i.e., use to commutative nature (I*f1)*f2 = I*(f1*f2) and then consider I to be a delta function). The stride can be explained by a zero-interpolation (adding zeros between every two weights) - explain how the support grows as function of the stride rate.

2. Deep versus shallow networks. Show how any shallow network is described by a deep network (with O(n) neurons) without assuming alpha_i = 0 as we did in class.

3. Training complexity. The gradient descent algorithm stops at points where the gradient vector vanishes (hence no further update). Assume that it takes a fixed number of operations to reach convergence, regardless of the initial weights configuration. Would that necessarily contradict the NP-completeness of the network training we've seen in class? Explan.

Submission and Grading

Submit (via moodle) a ZIP archive file named:
"ex2 _(student1name)[_student2name].zip"
The file should contain your code, and a document named "answers.pdf", which clearly describes your solutions and reports your results.

Remarks:
- The "answers.pdf" document should be a pdf, not a docx, (ONLY ONE PDF), additional documents will not be checked.
- If you are doing the assignment in pairs, only one of you should submit one zip file.
- Please write your names inside the pdf file.
- Please write some documentation explaining your code.

Grading: exercises which provide complete and correct answers and working solutions that satisfy the exercise requirements will result in a grade up to 95. A bonus of up to 5 points may be provided to

creative solutions that go beyond the requirements.