Machine Learning and Deep Networks

Assignment 2: visualization and fooling

Due: 7/12/2018

In 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 a simple pretrained AlexNet natural image classifier.

A. **Visualization of neurons via input optimization.** The provided AlexNet CNN has five convolution (conv) layers (some followed by local response normalization and/or pooling steps) and three fully connected (fc) 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. **Classification importance 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.

Submit a ZIP or RAR archive that contains your code, and a document, which clearly describes your solutions and reports your results.