# CSC 479/679 Final Project: Deep Convolutional Network.

Due date: May 1st, Wed end of the day. No late assignment is accepted.

You will present your work in the final exam class.

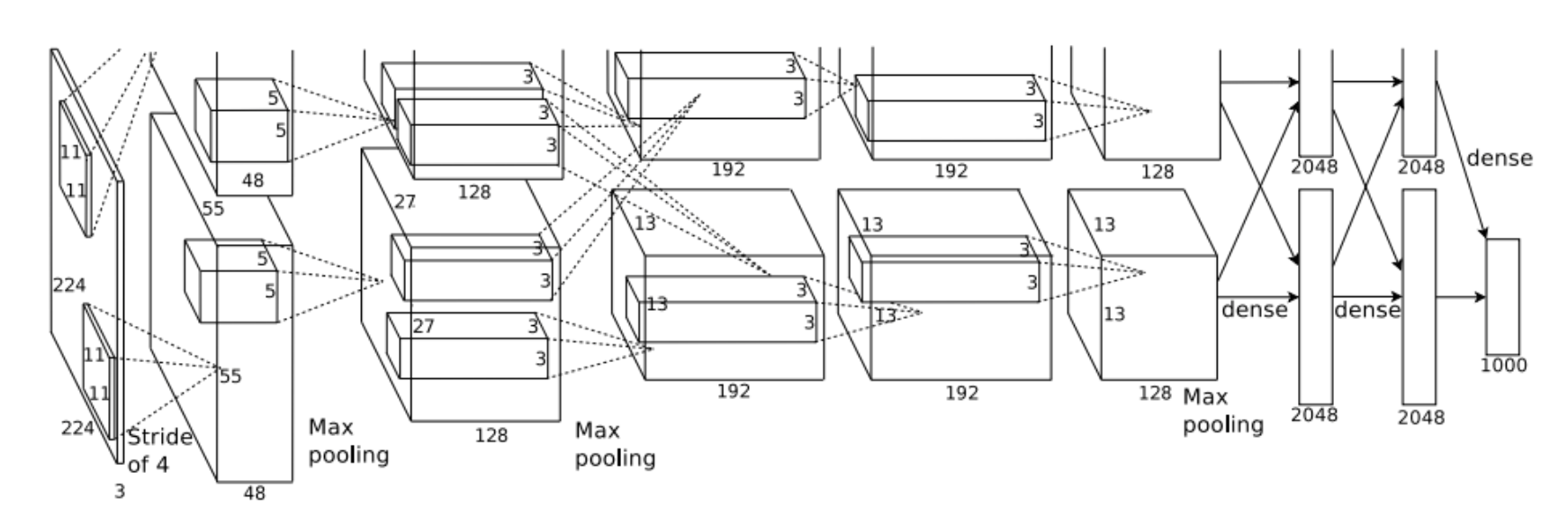
Totals points: 100pts + bonus (up to 20 points).

Teamwork: maximum of 2

Each team only needs to submit one project and needs to clarify the role of team members.

**Overview:**

In this project, we will fine tune a pre-trained ConvNet (AlexNet) for image classification and visualizing the neural network. This project is rather open-end (given the short duration). But it is expected that you are getting yourself familiar with tensorflow and the basic algorithm of ConvNet. Some part of the project is adapted from CSC5670 from Cornell Tech. However, they used Caffee as deep-learning library.



For this project, we need to install [Tensorflow](https://www.tensorflow.org/install/).

The assignment is contained in an IPython Notebook; see below.

[1] [Krizhevsky et al, "ImageNet Classification with Deep Convolutional Neural Networks", NIPS 2012](https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf)

Below is a tutorial of using pre-trained AlexNet to classify images.

<https://github.com/kratzert/finetune_alexnet_with_tensorflow/blob/master/validate_alexnet_on_imagenet.ipynb>

**Coding TODO**

**Here are requires steps to get full marks:**

1. Download or clone a repository that contains a pre-trained Alexnet. For example, the following repo contains a pretrained alex net.

<https://github.com/kratzert/finetune_alexnet_with_tensorflow>

This requires you install Python 3 and tensorflow. Feel free to use any other repository.

You need to fine tune the pre-trained AlexNet and get familiar with the whole entire code repository.

1. Download the dataset (dog versus food) from blackboard. Classify Dogs vs Food.
2. Visualize AlexNet Structure (e.g. visualize filters of the convolutional layer).

For example, this Github is a good resource.

<https://github.com/InFoCusp/tf_cnnvis>

1. Try it out with another dataset (e.g. scene database) from here.

<https://www.cs.toronto.edu/~kriz/cifar.html>

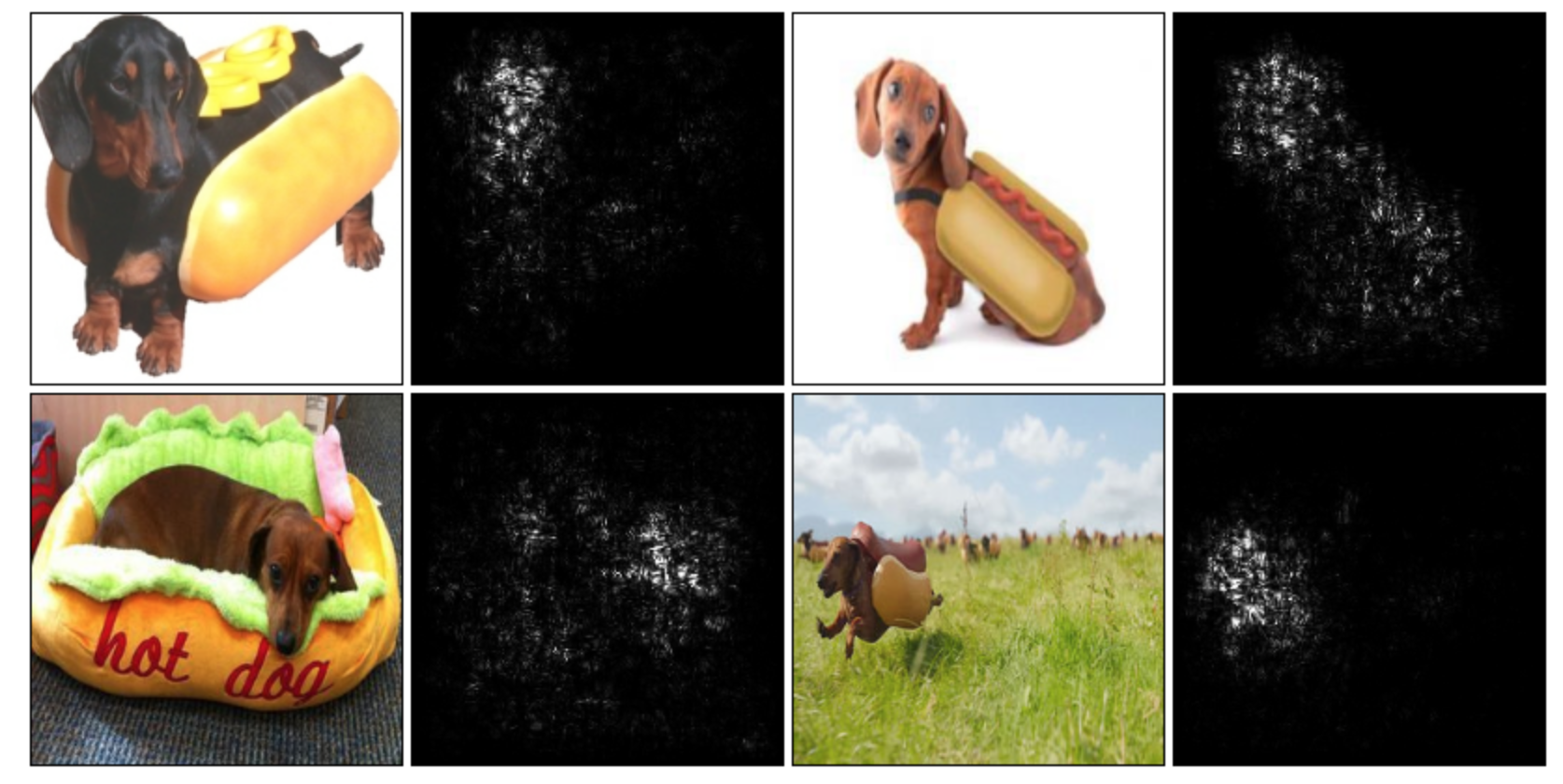
\*\*iPython notebook code

**Here are extra steps that you can gain bonus points:**

1. Visualize class saliency.

**Saliency**: we expect that pixels related to the class have a higher value. Left: Input image. Right: saliency.

The core idea behind this approach is to use the gradients at the image layer for a given image and class, to find the pixels which need to be changed the least i.e, the pixels for which the gradients have the smallest values.



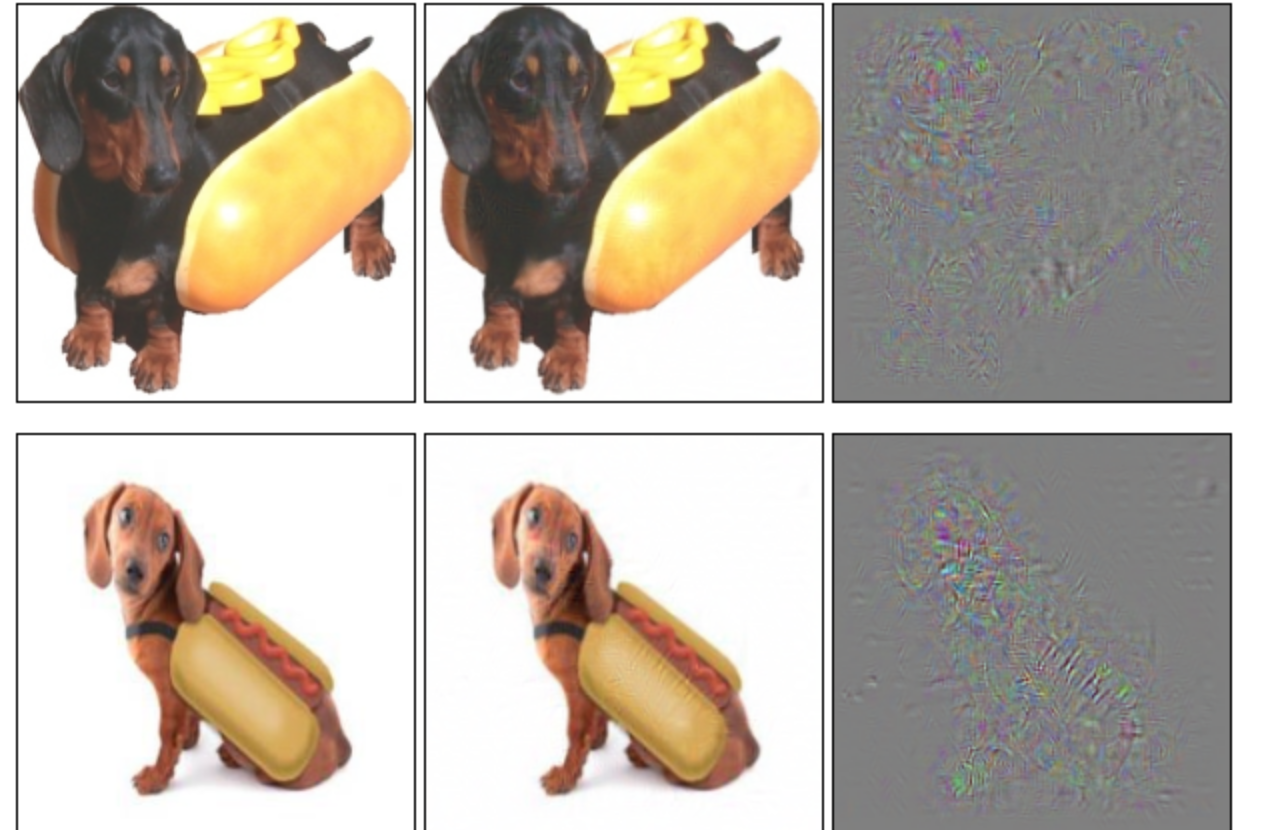
Here are resources that could be useful:

<https://github.com/artvandelay/Deep_Inside_Convolutional_Networks>

<https://github.com/raghakot/keras-vis>

1. Fool AlexNet into making wrong predictions:

These images look nearly identical, and yet AlexNet will classify each image on the middle as "snail". If you look really closely you can notice some tiny visual differences. The right image shows the difference magnified by 5x (with 0 re-centered at gray).



**Set up**

1. **Install tensor flow with Python 3.**
2. **Install** [**jupyter notebook**](https://jupyter-notebook-beginner-guide.readthedocs.io/en/latest/install.html)
3. **Make sure you have opencv3 installed**
4. **Download the codebase and dataset**
5. **Check and clone the code from pre-trained Alexnet.**

**What to hand in**

1. Write a documentation of how you have tuned AlexNet for image classification.
2. Create a .HTML file that summarize your results and discuss what you have learned from the implementation. You might discuss the paper you have read and the methods you have tried to visualize the neural network.
3. Save your notebook file and your .html, and zip up the images/code and upload to blackboard and also GITHUB.