In this assignment you will implement recurrent networks, and apply them to image captioning on Microsoft COCO. You will also explore methods for visualizing the features of a pretrained model on ImageNet, and also this model to implement Style Transfer. Finally, you will train a Generative Adversarial Network to generate images that look like a training dataset!

The goals of this assignment are as follows:

- Understand the architecture of *recurrent neural networks (RNNs)* and how they operate on sequences by sharing weights over time
- Understand and implement both Vanilla RNNs and Long-Short Term Memory (LSTM) networks.
- Understand how to combine convolutional neural nets and recurrent nets to implement an image captioning system
- Explore various applications of image gradients, including saliency maps, fooling images, class visualizations.
- Understand and implement techniques for image style transfer.
- Understand how to train and implement a Generative Adversarial Network (GAN) to produce images that resemble samples from a dataset.

Setup

Get the code as a zip file here. You should be able to use your setup from assignment 2.

Download data:

Once you have the starter code, you will need to download the COCO captioning data, pretrained SqueezeNet model (TensorFlow-only), and a few ImageNet validation images. Run the following from the `assignment3` directory:

```
cd cs231n/datasets
./get_assignment3_data.sh
```

Some Notes

NOTE 1: This year, the `assignment3` code has been tested to be compatible with python version `3.7` (it may work with other versions of `3.x`, but we won't be officially supporting them). You will need to make sure that during your virtual environment setup that the correct version of `python` is used. You can confirm your python version by (1) activating your virtualenv and (2) running `which python`.

NOTE 2: Please make sure that the submitted notebooks have been run and saved, and the cell outputs are visible on your pdfs. In addition, please **do not use the Web AFS interface** to retrieve your pdfs, and rely on **scp commands** directly, as there is a known Web AFS caching bug University IT is looking at that causes AFS files to not be properly updated with their most current version.

You can do Questions 3, 4, and 5 in TensorFlow or PyTorch. There are two versions of each of these notebooks, one for TensorFlow and one for PyTorch. No extra credit will be awarded if you do a question in both TensorFlow and PyTorch.

Q1: Image Captioning with Vanilla RNNs (25 points)

The Jupyter notebook `RNN_Captioning.ipynb` will walk you through the implementation of an image captioning system on MS-COCO using vanilla recurrent networks.

Q2: Image Captioning with LSTMs (30 points)

The Jupyter notebook `LSTM_Captioning.ipynb` will walk you through the implementation of Long-Short Term Memory (LSTM) RNNs, and apply them to image captioning on MS-COCO.

Q3: Network Visualization: Saliency maps, Class Visualization, and Fooling Images (15 points)

The Jupyter notebooks `NetworkVisualization-TensorFlow.ipynb` / `NetworkVisualization-PyTorch.ipynb` will introduce the pretrained SqueezeNet model, compute gradients with respect to images, and use them to produce saliency maps and fooling images. Please complete only one of the notebooks (TensorFlow or PyTorch). No extra credit will be awardeded if you complete both notebooks.

Q4: Style Transfer (15 points)

In the Jupyter notebooks `StyleTransfer-TensorFlow.ipynb` / `StyleTransfer-PyTorch.ipynb` you will learn how to create images with the content of one image but the style of another. Please complete only one of the notebooks (TensorFlow or PyTorch). No extra credit will be awardeded if you complete both notebooks.

Q5: Generative Adversarial Networks (15 points)

In the Jupyter notebooks `GANS-TensorFlow.ipynb` / `GANS-PyTorch.ipynb` you will learn how to generate images that match a training dataset, and use these models to improve classifier performance when training on a large amount of unlabeled data and a small amount of labeled data. Please complete only one of the notebooks (TensorFlow or PyTorch). No extra credit will be awarded if you complete both notebooks.

Submitting your work

Important: *Please make sure that the submitted notebooks have been run and saved, and the cell outputs are visible on your pdfs.* In addition, please *do not use the Web AFS interface* to retrieve your pdfs, and rely on scp directly.

There are ***two*** steps to submitting your assignment:

1. Run the provided `collectSubmission_*.sh` script in the `assignment3` directory, depending on which version (TensorFlow/PyTorch) you intend to submit.

You will be prompted for your SunetID (e.g. `jdoe`) and will need to provide your Stanford password. This script will generate a zip file of your code, submit your source code to Stanford AFS, and generate a pdf `a3.pdf` in a `cs231n-2019-assignment3/` folder in your AFS home directory.

If your submission for this step was successful, you should see a display message

```
### Code submitted at [TIME], [N] submission attempts remaining. ###
```

2. Download the generated `a3.pdf` from AFS, then submit the pdf to Gradescope. Again, do NOT use Web AFS to retrieve this file, and instead use the following scp command.

```
# replace DEST_PATH with where you want the pdf to be downloaded to.
scp YOUR_SUNET@myth.stanford.edu:cs231n-2019-assignment3/a3.pdf DEST_PATH
```

cs231n
cs231n
karpathy@cs.stanford.edu