CS231n Convolutional Neural Networks for Visual Recognition

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.

Update (5/11/18, 6:05 a.m.): The assignment zip file has been updated to include the proper optim.py and layers.py files.

You can follow the setup instructions here.

If you haven't already, you'll need to install either TensorFlow 1.7 (installation instructions here) or PyTorch 0.4 (instructions here) depending on which notebooks you decide to complete.

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
```

Start IPython:

After you have downloaded the data, you should start the IPython notebook server from the `assignment3` directory, with the `jupyter notebook` command. (See the Google Cloud Tutorial for any additional steps you may need to do for setting this up, if you are working remotely)

If you are unfamiliar with IPython, you can also refer to our IPython tutorial.

Some Notes

NOTE 1: This year, the `assignment3` code has been tested to be compatible with python version `3.6` (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: If you are working in a virtual environment on OSX, you may *potentially* encounter errors with matplotlib due to the issues described here. In our testing, it seems that this issue is no longer present with the most recent version of matplotlib, but if you do end up running into this issue you may have to use the `start_ipython_osx.sh` script from the `assignment3` directory (instead of `jupyter notebook` above) to launch your IPython notebook server. Note that you may have to modify some variables within the script to match your version of python/installation directory. The script assumes that your virtual environment is named `.env`.

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

There are *two* steps that you must complete to submit your assignment:

1. Submit a pdf of the completed iPython notebooks to Gradescope. If you are enrolled in the course, then you should have already been automatically added to the course on Gradescope.

To produce a pdf of your work, you can first convert each of the .ipynb files to HTML. To do this, simply run from your assignment directory

```
jupyter nbconvert --to html FILE.ipynb
```

for each of the notebooks, where `FILE.ipynb` is the notebook you want to convert. Then you can convert the HTML files to PDFs with your favorite web browser, and then concatenate them all together in your favorite PDF viewer/editor. Submit this final PDF on Gradescope, and be sure to tag the questions correctly!

Important: *Please make sure that the submitted notebooks have been run and the cell outputs are visible.*

2. Submit a zip file of your assignment on AFS. To do this, run the provided `collectSubmission.sh` script, which will produce a file called `assignment3.zip`. You will then need to SCP this file over to Stanford AFS using the following command (entering your Stanford password if requested):

```
# Run from the assignment directory where the zip file is located
scp assignment3.zip YOUR_SUNET@myth.stanford.edu:~/DEST_PATH
```

`YOUR_SUNET` should be replaced with your SUNetID (e.g. `jdoe`), and `DEST_PATH` should be a path to an existing directory on AFS where you want the zip file to be copied to (you may want to create a CS231N directory for convenience). Once this is done, run the following:

```
# SSH into the Stanford Myth machines
ssh YOUR_SUNET@myth.stanford.edu

# Descend into the directory where the zip file is now located
cd DEST_PATH

# Run the script to actually submit the assignment
/afs/ir/class/cs231n/submit
```

Once you run the submit script, simply follow the on-screen prompts to finish submitting the assignment on AFS. If successful, you should see a "SUBMIT SUCCESS" message output by the script.

cs231n
cs231n
karpathy@cs.stanford.edu