COMPSCI 682 Neural Networks: A Modern Introduction

In this assignment you will practice writing backpropagation code, and training Neural Networks and Convolutional Neural Networks. The goals of this assignment are as follows:

- understand **Neural Networks** and how they are arranged in layered architectures
- understand and be able to implement (vectorized) **backpropagation**
- implement various **update rules** used to optimize Neural Networks
- implement **Batch Normalization** and **Layer Normalization** for training deep networks
- implement **Dropout** to regularize networks
- understand the architecture of **Convolutional Neural Networks** and get practice with training these models on data
- gain experience with a major deep learning framework, such as **TensorFlow** or **PyTorch**.

Setup

Get the code as a zip file here.

You can follow the setup instructions here. If you would like to use good CPU/GPU resources, you can use Google Cloud.

Download data:

Once you have the starter code, you will need to download the CIFAR-10 dataset. Run the following from the `assignment2` directory:

```
cd cs682/datasets
./get_datasets.sh
```

Compile the Cython extension: Convolutional Neural Networks require a very efficient implementation. We have implemented of the functionality using Cython; you will need to compile the Cython extension before you can run the code. From the `assignment2/cs682` directory, run the following command:

```
python setup.py build_ext --inplace
```

NOTE: Check this page if you are using windows and having the “unable to find vcvarsall.bat” error.

Start Jupyter:

After you have the CIFAR-10 data, you should start the Jupyter notebook server from the `assignment2` directory, with the `jupyter notebook` command.

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

Some Notes

NOTE 1: This year, the `assignment2` 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 `assignment2` directory (instead of `jupyter notebook` above) to launch your Jupyter 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`.

Q1: Fully-connected Neural Network (20 points)

The Jupyter notebook `FullyConnectedNets.ipynb` will introduce you to our modular layer design, and then use those layers to implement fully-connected networks of arbitrary depth. To optimize these models you will implement several popular update rules.

Q2: Batch Normalization (30 points)

In the Jupyter notebook `BatchNormalization.ipynb` you will implement batch normalization, and use it to train deep fully-connected networks.

Q3: Dropout (10 points)

The Jupyter notebook `Dropout.ipynb` will help you implement Dropout and explore its effects on model generalization.

Q4: Convolutional Networks (30 points)

In the Jupyter Notebook `ConvolutionalNetworks.ipynb` you will implement several new layers that are commonly used in convolutional networks.

Q5: PyTorch / TensorFlow on CIFAR-10 (10 points)

For this last part, you will be working in either TensorFlow or PyTorch, two popular and powerful deep learning frameworks. **You only need to complete ONE of these two notebooks.** You do NOT need to do both, and we will *not* be awarding extra credit to those who do.

Open up either `PyTorch.ipynb` or `TensorFlow.ipynb`. There, you will learn how the framework works, culminating in training a convolutional network of your own design on CIFAR-10 to get the best performance you can.

NOTE: The PyTorch notebook requires PyTorch version >=0.4 (up to 1.2 as of 09/26/2019). You can install this version of PyTorch using conda or pip by following the instructions here: http://pytorch.org/

Submitting your work

There are ***two*** steps to submitting 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 to Gradescope. To do this, run the provided `collectSubmission.sh` script, which will produce a file called `assignment2.zip`.

---

compsci682-fa19
compsci682fall19@gmail.com