10.10.2022

Department of Computer Engineering
Prof. Dr. Hazım Kemal EKENEL

BLG 506E COMPUTER VISION
ASSIGNMENT 3
Two Layer Neural Networks

0. For this assignment and the others you will be given Stanford University CS231n course (http://cs231n.stanford.edu/) assignments. As stated in CS231n, you should be good at *Python*. Please have a look at *Python/NumPy/IPython* tutorials at http://cs231n.github.io/ .
Also we recommend you to have a *Linux OS* either locally (on your machine) or virtually (on your machine or Cloud services). Similarly, it is recommended to build a *Python* environment preferably by one of the methods below.
* *Anaconda* (https://www.anaconda.com/)
* *Miniconda* (https://docs.conda.io/en/latest/miniconda.html),
* *virtualenv* (https://virtualenv.pypa.io/en/latest/)
Check setup instructions page of CS231n (http://cs231n.github.io/setup-instructions/)

All works must be your own!
In your submission (.zip), provide all source files (.py, .ipynb etc.) that you used.
You should have comments in your code.
You are responsible to write a report that includes motivations behind your decisions (eg: hyperparameter tuning) related to experiments and your observations/thoughts about the results.
The report should be in .pdf, .txt or .doc(x) format.

1. Download assignment 1 from Stanford's CS231n:
http://cs231n.github.io/assignments2019/assignment1/
You will be following two_layer_net.ipynb notebook.

2. Implement the first part of TwoLayerNet.loss in cs231n/classifiers/neural_net.py where you perform forward pass using weights and biases and compute the scores. Then compare your scores within the cell.

3. In the same function, compute the softmax loss with regularization. Then compare to correct loss within the cell.

4. Complete the rest of the function where you compute the gradients with respect to W1, W2, b1 and b2. Then check the relative errors within the cell.

5. Implement the TwoLayerNet.train function. You will create random minibatches and update network parameters within the training loop.

6. Now implement the TwoLayerNet.predict function. You should run a forward pass and obtain predicted labels using scores.

7. Congrats, you have AI awoken! First test it on toy data. Then load CIFAR10 dataset, and unleash it!

8. Wait, not a good accuracy for an AI, right? Diagnose it by plotting loss and accuracy trends, and visualize the weights.

9. Some things prevent your model learning the inputs. Release your AI by tuning hyperparameters on the validation set to acquire better accuracy. Explain what you have done clearly. In real world people do this for months but luckily we have a small dataset.

10. Visualize the weights of your final model and get accuracy on the test set. Answer the inline question.

For any question or discussion, you can e-mail to saritas21@itü.edu.tr or you can (before informing is preferable) and come to the SiMiT Lab (office no: 4105).

For Colab users, you can insert the code below (do not forget changing the “FOLDERNAME”) to the beginning of the knn.ipynp file to mount Google Drive;

```
# This mounts your Google Drive to the Colab VM.
from google.colab import drive
drive.mount('/content/drive', force_remount=True)

# Enter the foldername in your Drive where you have saved the unzipped
# assignment folder, e.g. 'cs231n/assignments/assignment1/'
FOLDERNAME = None
assert FOLDERNAME is not None, "[!] Enter the foldername."

# Now that we've mounted your Drive, this ensures that
# the Python interpreter of the Colab VM can load
# python files from within it.
import sys
sys.path.append('/content/drive/My Drive/{}'.format(FOLDERNAME))

# This downloads the CIFAR-10 dataset to your Drive
# if it doesn't already exist.
%cd drive/My\ Drive/$FOLDERNAME/cs231n/datasets/
!bash get_datasets.sh
%cd /content/drive/My\ Drive/$FOLDERNAME
```