27.02.2020

Department of Computer Engineering
Prof. Dr. Hazım Kemal EKENEL

BLG 506E COMPUTER VISION
ASSIGNMENT 2
Linear Classifier

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 and your report (.pdf).
You should have comments in your code.
In your report, you should explain all the steps in detail.

1. Download assignment 1 from Stanford's CS231n:
http://cs231n.github.io/assignments2019/assignment1/
You will be following `svm.ipynb` and `softmax.ipynb` notebooks.

2. Download and extract CIFAR-10 dataset using `get_datasets.sh` script under `cs231n/datasets` . (If you downloaded the dataset for the first assignment, you can use symbolic links or arrange paths to save disk space.)

3. Start notebook with `jupyter-notebook` command. In `svm.ipynb`, run library/parameter setups, load CIFAR10 dataset, visualize it, split data as training-validation-test set and do preprocessing.

(If you encounter `Cannot import scipy.misc.imread` error, check if your environment have scipy 1.3.0. If so, downgrade it to scipy<=1.2.0)

4. Implement `svm_loss_naive` function in `cs231n/classifiers/linear_svm.py` . Run gradient check cell and answer inline question 1.

5. Implement `svm_loss_vectorized` function. Run gradient check cell and compare the results (loss difference and times) to naive approach.

6. Implement `LinearClassifier.train` function in `cs231n/classifiers/linear_classifier.py` . Train your svm and plot your loss. Implement `LinearClassifier.predict` function to measure the accuracy.

7. Tune hyperparameters using validation set. Determine the best ones.

8. Visualize the learned weights. (Answer inline question 2)

9. Open `softmax.ipynb` . Load data.

10. Implement `softmax_loss_naive` function in `cs231n/classifiers/softmax.py` . Run gradient check cells and answer inline question 1.

11. Implement `softmax_loss_vectorized` function. Run gradient check cell and compare the results (loss difference and times) to naive approach.

12. Train your softmax classifier and tune hyperparameters to find the best ones. Answer inline question 2.

13. Visualize the learned weights.