Programming Assignment 1 Detailed ReadMe
Advanced Machine Learning

There are 2 methods which can be used to reproduce the results.

Method 1 (Common steps for MNIST/CIFAR)

Step 1: Clone the GitHub repository directly to the current working directory.

```
!git clone https://github.com/mtoneva/example_forgetting.git
```

Step 2: Navigate to the folder "example_forgetting"

```
%cd example_forgetting/
```

Step 3: Install requirements. It is important because author have used specific version of "torchvision" library. Dependencies has to be installed as same as author to reproduce the results.

```
!pip install -r requirements.txt
```

Step 4: Now train basic CNN Deep Neural Network model with the MNIST Dataset using following parameters

```
{'dataset': 'mnist', 'batch_size': 64, 'epochs': 200, 'lr': 0.01, 'momentum': 0.5, 'no_cuda': False, 'seed': 2, 'sorting_file':
'none', 'remove_n': 0, 'keep_lowest_n': 0, 'no_dropout': True, 'input_dir': 'mnist_results/', 'output_dir': 'mnist_results'}
```

Step 5: Use the pkl file saved in step 4 to calculate number of unforgettable examples. With the setting:

```
Namespace(epochs=200, input_dir='mnist_results', input_fname_args=['dataset',
'mnist', 'no_dropout', 'True', 'sorting_file', 'none', 'remove_n', '0',
'keep_lowest_n', '0'], output_dir='mnist_results', output_name='mnist_sorted')
```

```
!python order_examples_by_forgetting.py --output_dir mnist_results --output_name
mnist_sorted --input_dir mnist_results --input_fname_args dataset mnist no_dropout
True sorting_file none remove_n 0 keep_lowest_n 0
```

Method 2 (Common steps for MNIST/CIFAR)

Step 1: Load the storage (in our case we load google colab)

Step 2: Navigate to root folder

Step 3: Clone the original repository to install requirements.

Step 4: Import all dependencies.

Step 5: Decide dataset and model to use. (CIFAR10 and resnet18). Also create a data dictionary which contains all the parameters required to be set before training. It requires following inputs:
dataset
model
batch_size
epochs
learning_rate
data_augmentation
cutout
n_holes
length
no_cuda
seed
sorting_file
remove_n
keep_lowest_n
no_dropout
remove_subsample
noise_percent_labels
noise_percent_pixels
noise_std_pixels
optimizer
input_dir
output_dir

Step 6: Image pre-processing, normalize and train transofmrations.
Step 7: Download the tar file for the CIFAR10/MNIST and untar to present working directory. Further split that to trainset and testset.

Step 8: Get indices of examples that should be used for training. And assign the labels and dataset if transformed or removed some examples from the dataset.

Step 9: Introduce noise to data and flip the labels if specified.

- Step 9.1: Compute number of labels to change
- Step 9.2: Randomly choose which labels to change, get indices
- Step 9.3: Flip each of the randomly chosen labels

Step 10: Setup the model for training the dataset.

Step 11: Initialize dictionary to save statistics for every example presentation or epochs and log the best train and test accuracy so far.

Step 12 (How train method works): Train model for one epoch. It has example stats dictionary containing statistics accumulated over every presentation of example.

- Step 12.1: Get trainset indices for batch.
- Step 12.2: Get batch inputs and targets, transform them appropriately.
- Step 12.3: Forward propagation, compute loss, get predictions.
- Step 12.4: Update statistics and loss
- Step 12.5: Example classified correctly, highest incorrect class is 2nd largest output and Example misclassified, highest incorrect class is max output.
- Step 12.6: Add the statistics of the current training example to dictionary. And Save training accuracy.

Step 13 (How test method works): Get inputs and targets. Test the predictions and compute loss. Save test accuracy. Also save pkl for best model.

Step 14: Investigate learned events, unlearned events and forgettable events. Give 1 to learned event, 0 to unlearned event and for the forgettable event you give value from 0 to 1. (This is called computing forgetting statistics)

Step 15: Sort the examples based on the score assigned in previous step. If we break down, we say, sorts examples by number of forgetting counts during training. If an example was never learned, it is assigned the maximum number of forgetting counts, since multiple training runs used, sort examples by the sum of their forgetting counts over all runs. (Sort from highest forgetting to lowest forgetting)

Step 16: Steps 10 and 11 are repeated for all epochs. Compute the forgetting statistics per example for training run.

Step 17: Print the number of unforgettable examples