Code_Illustration.md

Derivative Manipulation for General Example Weighting

Downloading Link

https://www.dropbox.com/sh/tn0yj9jlx03oamx/AACPpChxNN2C-bVs4jrmiX45a?dl=0

To Visualise the Repository/Directory Tree Structure

```
tree
```

Dependencies

The core functions are implemented in the caffe framework. We use matlab interfaces matcaffe for data preparation.

- CaffeMex_v2
- MATLAB 2017b

Setup

- Install dependencies on Ubuntu 16.04

```
sudo apt-get install libprotobuf-dev libleveldb-dev libsnappy-dev libopencv-dev libhdf5-serial-dev protobuf-compil
sudo apt-get install --no-install-recommends libboost-all-dev
sudo apt-get install libopenblas-dev
sudo apt-get install python-dev
sudo apt-get install libgflags-dev libgoogle-glog-dev liblmdb-dev
```

- Install MATLAB 2017b

  Download and Run the install binary file

```
./install
```

- Compile Caffe and matlab interface

  Note you may need to change some paths in Makefile.config according your system environment and MATLAB path

```
cd CaffeMex_CCE_sumW
make -j8  && make matcaffe
cd ../CaffeMex_UnifiedWeight_V01
make -j8  && make matcaffe

cd ../CaffeMex_GCE
make -j8  && make matcaffe
cd ../CaffeMex_GCE_sumW
make -j8  && make matcaffe

cd ../CaffeMex_MAE_sumW
```

```
make -j8  && make matcaffe
cd ../CaffeMex_MAE_V00
make -j8  && make matcaffe

cd ../CaffeMex_MSE
make -j8  && make matcaffe
cd ../CaffeMex_MSE_sumW
make -j8  && make matcaffe
```

Usage

Examples for reproducing our results on CIFAR-100 are given.

- Data preparation for CIFAR-100
  - Generating test data:

```
cd CIFAR100_Data_Toolkit
matlab -nodisplay -nosplash -nodesktop -r "run('test_data_preparation.m');exit;" | tail -n +11
```

  - Generating training data:

```
% Symmetric noise rate: 0.0, 0.2, 0.4, 0.6
matlab -nodisplay -nosplash -nodesktop -r "run('train_data_preparationV2_noise_0_0.m');exit;" | tail -n +11

matlab -nodisplay -nosplash -nodesktop -r "run('train_data_preparationV2_noise_0_2.m');exit;" | tail -n +11

matlab -nodisplay -nosplash -nodesktop -r "run('train_data_preparationV2_noise_0_4.m');exit;" | tail -n +11

matlab -nodisplay -nosplash -nodesktop -r "run('train_data_preparationV2_noise_0_6.m');exit;" | tail -n +11
```

  - Copy data

```
cd ..
echo CIFAR100_ResNet44*/pre_pro_process | xargs -n 1 cp CIFAR100_Data_Toolkit/TestImageDataCell.mat

echo CIFAR100_ResNet44*/pre_pro_process | xargs -n 1 cp CIFAR100_Data_Toolkit/TrainImageDataCell0.0.mat

echo CIFAR100_ResNet44*/pre_pro_process | xargs -n 1 cp CIFAR100_Data_Toolkit/TrainImageDataCell0.2.mat

echo CIFAR100_ResNet44*/pre_pro_process | xargs -n 1 cp CIFAR100_Data_Toolkit/TrainImageDataCell0.4.mat

echo CIFAR100_ResNet44*/pre_pro_process | xargs -n 1 cp CIFAR100_Data_Toolkit/TrainImageDataCell0.6.mat
```

- Train & Test

  Run the training and testing scripts in the training folder of a specific setting defined by its corresponding prototxt folder.

  For example,

```
cd CIFAR100_ResNet44_V03/train_Res44_CCE_0.0

matlab -nodisplay -nosplash -nodesktop -r "run('train.m');exit;" | tail -n +11

matlab -nodisplay -nosplash -nodesktop -r "run('test.m');exit;" | tail -n +11
```

Our trained results

- Our trained results are stored in corresponding folders. For example, in Folder **CIFAR100_ResNet44_V03/train_Res44_CCE_0.0**, there are:

    - accuracy.txt
    - accuracy_curve.png

- Without changing the random seed (123), you are supposed to obtain exactly the same results.

Acknowledgements

Our implementation benefits from:

- Caffe library: https://caffe.berkeleyvision.org/
- CaffeMex_v2 library: https://github.com/sciencefans/CaffeMex_v2/tree/9bab8d2aaa2dbc448fd7123c98d225c680b066e4