IITM-CS7015 : Deep Learning Given on: March 8, 2:30pm

Programming Assignment #3 Due on : March 23, 11:55pm IST

- The aim of this assignment is to train and test a Convolutional Neural Network for image classification on the Fashion-MNIST dataset.
- We strongly recommend that you work on this assignment in teams of 2. Both the members of the team are expected to work together (and not divide the work) since this assignment is designed with a learning outcome in view.
- Collaborations and discussions with others are strictly prohibited.
- You must use Tensorflow library (Python) for your implementation. If you are using any other languages, please contact the TAs before you proceed.
- All the models will be tested on same environment on Kaggle. So, you must submit all your results on test data on Kaggle.
- You have to turn in the well documented code along with a detailed report of the results of the experiment electronically in Moodle. Typeset your report in Latex. Reports which are not written using Latex will not be accepted.
- The report should be precise and concise. Unnecessary verbosity will be penalized.
- You have to check the Moodle discussion forum regularly for updates regarding the assignment.

1 Task

1.1 Instructions

- Download the Fashion-MNIST dataset (refer to PA1 for link to the same). The images were flatterned from 28-by-28 to 784. You must convert it back to 28-by-28 using *numpy.reshape* before procedding.
- You will use Tensorflow to train the CNN.
- For the purpose of plot generation and report, you will train a convolutional neural network (where all filters are of size 3×3) with the following structure:
  (a) **CONV1**: convolutional layer with 3 inputs (RGB), 64 outputs (filter size is thus $64 \times 3 \times 3 \times 3$)
  (b) **POOL1**: 2×2 max-pooling layer
  (c) **CONV2**: convolutional layer with 64 outputs, 128 outputs (filter size $128 \times 64 \times 3 \times 3$)

(d) **POOL2**: 2×2 max-pooling layer
(e) **CONV3**: convolutional layer with 128 inputs, 256 outputs
(f) **CONV4**: convolutional layer with 256 inputs, 256 outputs
(g) **POOL3**: 2×2 max-pooling layer
(h) **FC1**: fully connected layer with 256 inputs, 1024 outputs
(i) **FC2**: fully connected layer with 1024 inputs, 1024 outputs
(j) **SOFTMAX**: softmax layer for classification: 1024 inputs, 10 outputs

- For all convolution layers, stride S = 1, padding P = 1
- All layers, except for the pooling layers and for the last (softmax-)layer should use ReLU-nonlinearities.
- You are allowed to use data augmentation. For example, for every image in your training data you can create images which are horizontally/vertically flipped versions of these images and add them to your training data. You can think of other creative ways of generating additional training data from the given training data.
- Use the same split for train, valid and test as given for PA1.
- Use batch-normalization on the last layer activations (immediately before computing the softmax) when training the network.
- It will be necessary to experiment with learning rate and different parameter initializations (Xavier, He etc.) to find settings that are stable and yield good solutions.
- Use early stopping using the validation set with a patience of 5 epochs.
- Your code should support the following options:
  - --lr (initial learning rate η for gradient descent based algorithms)
  - --batch_size (the batch size to be used - valid values are 1 and multiples of 5)
  - --init (the initialization method to be used - 1 for Xavier, 2 for He)
  - --save_dir (the directory in which the pickled model should be saved - by model we mean all the weights and biases of the network)

  You should use the `argparse` module in python for parsing these parameters.
- The task is to get an accuracy of 96% on the test data.
- For obtaining the best accuracy, you are allowed to use up to 5 Convolutional Layers (with any filter size, any stride and any padding), 3 Fully Connected Layers (any number of neurons), any non-linearity, any number of Pooling Layers in your model and any optimizer.

1.2 Kaggle Submission

It is suggested that both the team members make separate account on Kaggle. You can form teams for this assignment on the Kaggle assignment page. With this both the team members will be able to make submissions.

You must submit the test_submission.csv files on Kaggle assignment page for evaluation. The evaluation on Kaggle will be done in 2 steps. Till March 20, you are allowed to make 5 submissions on the portal everyday. These will be evaluated on 30% of the test data. Till March 20, your 1st evaluation scores will be visible on Public leaderboard. On March 20, you can select 2 of your best submissions to get evaluated in the second step. By default, your best 2 submissions will be taken for evaluation in the second step. After the deadline, your scores in the second step of evaluation will be visible on the Private leaderboard.

1.3 Report

Prepare a report containing the following:

- Configuration and training details of your best performing model.
- A plot of the learning curve showing iterations on the x-axis and negative log likelihood over labels on the y-axis. Make a single plot showing both the training loss and the validation loss.
- The performance on the test data of the model that performs best on the validation data.
- The parameter setting which gave you the best results.
- Write down the dimensions of the input and output at each layer (for example, the input to CONV1 layer is $3 \times 32 \times 32$)
- Exactly how many parameters does your network have? How many of these are in the fully connected layers and how many are in the convolutional layers?
- Exactly how many "neurons" does your network have? How many of these are in the fully connected layers and how many are in the convolutional layers?
- What was the effect of using batch normalization ?
- Plot all the 64 layer 1 filters in an 8×8 grid. Do you observe any interesting patterns?
- Apply guided back propagation on any 10 neurons in the CONV5 layer and plot the images which excite this neuron. The idea again is to discover interesting patterns which excite some neurons.
- (Extra credits) Refer to (`http://www.evolvingai.org/fooling`) and try to find different ways in which you can fool the network. One way is to randomly change some pixels in the image such that the class is changed. Report the plot of accuracy v/s number of pixels changed on the test set.

1.4 Submission Instructions:

You need to submit the source code for the assignment. Your code should include one file called train.py which should be runnable using the following command:

```
python train.py --lr 0.01 --batch_size 20 --init 1 --save_dir <some_dir>
```

All other supporting files used for generating plots, etc. should also be placed in the zip file. You need a single tar.gz file containing the following:

```
- train.py
- run.sh (containing the best hyperparameters setting)
- any other python scripts that you have written
- report (in Latex) of the results of your experiments
- Kaggle_subs/ (folder containing all the Kaggle submissions)
```

The tar.gz should be named as RollNo_backprop.tar.gz.