IITM-CS7015 : Deep Learning — Given on: March 12, 11:55pm

Programming Assignment #2 — Due on : March 26, 11:55pm IST

- The aim of this assignment is to train and test a Convolutional Neural Network for image classification on the subset of Tiny ImageNet 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.
- This assignment is going to be time-consuming. **PLEASE START EARLY**. We can not increase the daily submission limit due to technical reasons.
- It will be better if you use **Tensorflow** library (Python) for your implementation (Many of the TAs are comfortable in Tensorflow, for other languages, it will be difficult to get all your doubts cleared). If you are using any other languages, please contact the TAs before you proceed.
- Note that you **Should Not** use any **Higher Level APIs** like keras, estimators, *etc.*. In other words, any API which will support model.train() and model.predict() function is prohibited. You should implement these functions yourself.
- 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 report with **Detailed Observations** and inferences of the results electronically in Moodle.
- Typeset your report in Latex code provided (attached). It is necessary to fill '**Checklist**' in the attached sample report. Reports which are not written using Latex will not be accepted.
- The report should be precise and concise. Unnecessary verbosity, (like **Theory about CNNs**) will be penalized.
- You **MUST** run *sanity_check.py* script (attached) on your submission zip file.
- You can find the evaluation pattern (marks distribution) at the last page.
- *You have to check the Moodle discussion forum regularly for updates regarding the assignment.*
- Please ensure that only **ONE** team member submits on Moodle.
- You can refer to this tutorial to get started on CNNs (Good starting point).

1 Task

1.1 Instructions

- Download the Tiny ImageNet (subset) dataset from Kaggle link provided.

  Given an input image of shape 64×64×3 from the dataset, the network will be trained to classify the image into 1 of 20[1] classes.

- You will use Tensorflow to train the CNN.

- For the purpose of plot generation and report, you will train a convolutional neural network with the following structure:

  (a) **Conv1**: convolutional layer with 3 inputs (RGB), 32 outputs (filter size is thus $32 \times 3 \times 5 \times 5$)

  (b) **Conv2**: convolutional layer with 32 inputs , 32 outputs (filter size $32 \times 32 \times 5 \times 5$)

  (c) **Pool1**: 2×2 max-pooling layer

  (d) **Conv3**: convolutional layer with 32 inputs, 64 outputs (filter size $64 \times 32 \times 3 \times 3$)

  (e) **Conv4**: convolutional layer with 64 inputs, 64 outputs (filter size $64 \times 64 \times 3 \times 3$)

  (f) **Pool2**: 2×2 max-pooling layer

  (g) **Conv5**: convolutional layer with 64 inputs, 64 outputs (filter size $64 \times 64 \times 3 \times 3$)

  (h) **Conv6**: convolutional layer with 64 inputs, 128 outputs (filter size $128 \times 64 \times 3 \times 3$)

  (i) **Pool3**: 2×2 max-pooling layer

  (j) **FC1**: fully connected layer with 6272 inputs, 256 outputs (i.e, number of neurons is 256)

  (k) **SOFTMAX**: softmax layer for classification: 256 inputs, 20 outputs

- All layers, except for the pooling layers and for the last (softmax-) layer should use ReLU-nonlinearities. We can view the chain of network as:

  Image → Conv1(F5-32, note that 5 is the filter size here and number of filters is 32) → Conv2(F5-32) → Pool1 → Conv3(F3-64) → Conv4(F3-64) → Pool2 → Conv5(F3-64) → Conv6(F3-128) → Pool3 → FC1(256) → SOFTMAX(20)

- For all the convolution layers, stride S = 1

[1] we have randomly sampled 20 classes from the original 200

- For all the convolution layers padding P = 1 (same) except for last convolution layer **Conv6** with P = 0 (valid)
- For obtaining the best accuracy, you are allowed to use up to 6 Convolutional Layers (with any filter size, any stride and any padding), upto 2 Fully Connected Layers (any number of neurons), any non-linearity, any number of Pooling Layers in your model and any optimizer.
- 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 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 model should be saved - by model we mean all the weights and biases of the network. You should use Tensorflow Saver)
  - --epochs (the number of epochs for which model is trained. Epoch: one iteration over training data)
  - --dataAugment (data augmentation is used or not. 1 for yes and 0 for no)
  - --train (training data file to load)
  - --val (validation data file to load)
  - --test (test data file to load)

  You should use the `argparse` module in python for parsing these parameters.
- The task is to get an accuracy of 50% on the test data.

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 26, you are allowed to make 5 submissions on the portal everyday. These will be evaluated on 30% of the test data. On March 26, 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 observations and inferences for 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 $3 \times 64 \times 64$)
- 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 32 layer-1 (Conv1) filters in an 4×8 grid. Do you observe any interesting patterns?
- Apply guided back propagation on any 10 neurons in the Conv6 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 5 --init 1 --save_dir <some_dir>
--epochs 5 --dataAugment 0 --train train.csv --val valid.csv --test test.csv
```

All other supporting files used for generating plots, etc. should also be placed in the zip file. You need a single folder (RollNoTeamMemberA_RollNoTeamMemberB_PA2, *e.g.* CS15D201_CS14B042_PA2) containing the following:

- `train.py`
- `run.sh (containing the best hyperparameters setting)`
- `any other python scripts that you have written`
- `'report.pdf' (in Latex) of the results of your experiments`
- `Kaggle_subs/ (folder containing ONLY THE BEST TWO Kaggle submissions). The two filenames in this folder should be 'firstbest.csv' and 'secondbest.csv'.`

The zip should be named as RollNoTeamMemberA_RollNoTeamMemberB_PA2.zip, (*e.g.* CS15D201_CS14B042_PA2.zip). Note that zip file and folder name **SHOULD BE SAME**.

Please run sanity_check.py by passing your zip file as command line argument (*e.g.* python sanity_check.py CS15D201_CS14B042_PA2.zip). Only, if this doesn't throws any errors, submit your assignment.

1.5 Marks Distribution:

Criteria	Marks
Kaggle Score Public	10
Kaggle Score Private	15
Plot of training loss and validation loss curves	5
Answers to the Qs asked	6
Plot of all the 32 layer 1 filters	10
Guided backpropagation	10
hyperparamater tuning for (learning rate, init, batch_size)	12 (4 marks each)
Batch Normalization	7
Initialization: He, Xavier	5
Early Stopping	5
Viva on learnings from the assignment	15
Data Augmentation (Extra)	5
Fooling Network (Extra)	10
Additional creative idea (Extra)	10

Table 1: Marks Distribution: Total marks is 100 and extra credits is 25.