Assignment 4– CMPUT 328

Classification on CIFAR-10 Dataset

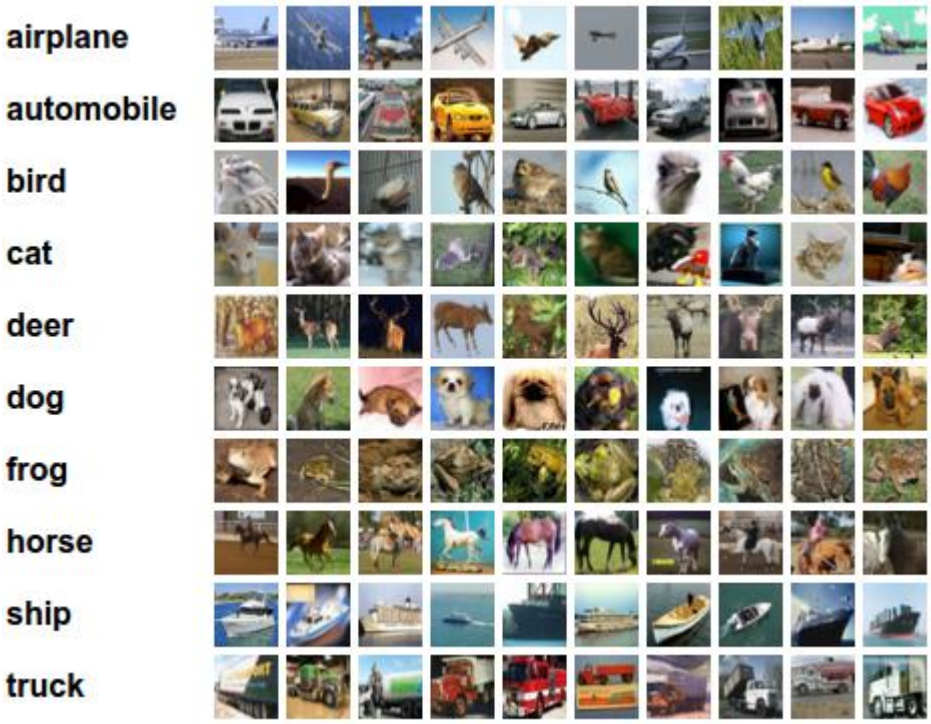
Classification on CIFAR-10 dataset [7 Marks]:

Your task in this assignment is to do classification on CIFAR-10 dataset

CIFAR-10 dataset:

Contains 32x32x3 RGB images. There are 50000 images in the train set and 10000 images in the test set. There are no validation set.

Each image in CIFAR-10 dataset belongs to one of the ten classes: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck.



For more information, see https://www.cs.toronto.edu/~kriz/cifar.html

A. Tasks:

In this assignment, you are going to implement the Convolutional Neural Network to do classification on CIFAR-10 dataset.

The network architecture in this assignment is not fixed. **You will design the network architecture yourself.**

However, there are some requirements that your network architecture must satisfy:

1. Your network must have at least 5 layers (a layer means convolution or fully connected layer) into an activation function like Relu, Softmax, Tanh or sigmoid..., (input layer, max pooling or strided convolution layer used to downsample activation map doesn't count). The number of layers in your network will be very close to the number of activation functions)

2. Must have at least 1 convolution layer

3. Must have at least 1 fully connected layer at the end

4. Must have at least 1 max pooling or strided convolution layer (i.e. Convolution with stride > 1)

If your network architecture doesn't satisfy any of the above 4 requirements, you will lose mark.

In addition to the above requirement, to do well in this assignment, you may have to:

- Use He/Xavier initialization (**will be explained in section B**)
- Regularization

- Try different activation functions (relu, tanh, sigmoid, elu...)
- Try different optimizer (For example: Vanilla SGD, Adam, RMSProp, AdaDelta...)
- Try different loss function (For example: cross entropy, square error)

B. Explanation for the terms mentioned above:

1. Use He/Xavier initialization:

See: http://andyljones.tumblr.com/post/110998971763/an-explanation-of-xavier-initialization

In Tensorflow, you can access this kind of initialization by using https://www.tensorflow.org/api_docs/python/tf/contrib/layers/xavier_initializer

C. What to do:

You are given some code. There are 4 files in the code:

1. **cifar10**: This part provides utilities on Cifar10 dataset via the class Cifar10. You don't have to modify this code. Some example usage:

```
cifar10_train = Cifar10(batch_size=100, one_hot=True, test=False, shuffle=True)   # Load Cifar10 train set with batch size 100 and one hot label.

cifar10_test = Cifar10(batch_size=100, one_hot=False, test=True, shuffle=False)  # Load Cifar10 test set

cifar10_test_images, cifar10_test_labels = cifar10_test._images, cifar10_test._labels  # Get all images and labels of the test set.

batch_x, batch_y = cifar10_train.get_next_batch()  # get the next batch of Cifar10 train set
```

2. main: Part used as main entry point for the program. You don't need to modify this file. However, there is one thing you must notice inside this function. You can set TRAIN = False at the beginning to skip the training phase. This is useful when you saved your model and just want to do testing.

3. ops: Part contains some utilities function for your network architecture creation. For example, there is already an example batch normalization wrapper utility function there which you are recommended to use. Here you can also implement functions that create a convolution or fully connected layer. These functions should allocate all variables and compute the result of the convolution/fully connected layer automatically from the input for you.

4. net.py: You will need to implement your network architecture, training code and testing code here in this file.

def net (input, is_training)

Function that return the output of your network based on the input. is_training is a boolean placeholder that should be set to True in training phase and False in testing phase.

The architecture of the network you design inside this function must satisfy the 4 requirements mentioned in section A. For each requirement that is not satisfied, you lose 10% of your mark on this assignment.

def train()

Write your training code here. This should be similar to Assignment 1. However, in this assignment, **you should and probably have to add code to save your model after training in order to satisfy the time constraint in this assignment.** Training the model in this assignment will be quite slow. If you do both training and testing at once, your program will exceed the time limit. The best way to do this is: train your model, save your model. In test time, load your model and predict labels.

Note: This function will NOT be called during marking time!

def test()

Your testing code to generate labels for test images using your trained model should go here. During marking time the **train()** function will NOT be called. Therefore, if you don't want to split your training and testing, you must put your training code here.

COLLABORATION POLICY: This must be your own work. Do not share or look at the code of other people (whether they are inside or outside the class). Do not search for or copy the code from the Internet. You can talk to others that are in the class about solution ideas (but not so detailed that you are verbally sharing or hearing about or seeing the code). You must cite whom you talked to in the comments of your programs.

D. SUBMISSION: You need to submit one file: Submit 2 files: **net.py** and **ops.py** to eclass. **If you have a saved model that will be loaded during test time, submit that model too (IMPORTANT!)**.

DUE DATE: The due date is Friday, October 12 by 11:59 pm. The assignment is to be submitted online on eclass. For late submissions' rules please, check the course information on eclass

E. MARKING:

2 Marks Describe your program for TAs (Tuesday, Oct 9 – Thursday, Oct 11). **(Monday, Oct 8 is the holiday so the students of this lab should attend to one of the other labs)**
TAs can select five random questions based on your code, results, and algorithms. The time to present will be in your lab section in the week when this lab is due.
5 Marks Your final score depends on the correct implementation of algorithms and it must work on CIFAR10 datasets.

1. Base mark:
Your algorithm will get mark based on its accuracy on Cifar-10 dataset:
score scales linearly from 28-38% on the test set

2. Time constraint:

Your whole program (testing part) should not run for more than **200s (With Google colab on GPU)**

3. Architecture requirements:

As mentioned above, your network architecture must satisfy these requirements:

5. Your network must have at least 5 layers (a layer means convolution or fully connected layer) into an activation function like relu, softmax, tanh or sigmoid..., (input layer, output layer, max pooling or strided convolution layer used to downsample activation map doesn't count. The number of layers in your network will be very close to the number of activation functions)

6. Must have at least 1 convolution layer

7. Must have at least 1 fully connected layer at the end

8. Must have at least 1 max pooling or strided convolution layer (i.e. Convolution with stride > 1)

For every requirement that is not satisfied, you lose 10% of your mark.