Visual Learning and Recognition 16-826

Assignment 3 - TensorFlow
Simple Sequence Generation

DUE DATE: April 21st 2017

Introduction

In this two part assignment we will experiment with generating sequences via recurrent neural networks (RNNs) and also answer some fundamental questions in deep neural networks. For the first part, a detailed introduction into recurrent neural networks can be found here:

https://deeplearning4j.org/lstm.html

We will be implementing this in TensorFlow. Tutorial slides from class can be found here:

https://docs.google.com/presentation/d/1EBMGREMyzrPV0CUUfg7ycM-LKTYilqDfZVK162swyao/edit?usp=sharing

More information about RNNs with tensorflow can be found here:

https://www.tensorflow.org/tutorials/recurrent

This assignment is meant to be smaller and easier than the previous two assignments. You will be implementing a simple text generating RNN model. Essentially given training text data, you will train a model to predict the next character given an input character and the state of the RNN model.

In the second part of the assignment, you will be answering some questions about neural networks in general. You will not have to write any code for PART-2.

Getting Started

Starter code for this assignment can be downloaded from https://www.dropbox.com/s/kdy8jigptdvfoff/assignment3.tar.gz?dl=0

```
cd /home/${USER}
wget https://www.dropbox.com/s/kdy8jigptdvfoff/assignment3.tar.gz
tar -xvf assignment3.tar.gz
```

Source the bash to use my tensorflow installation. Qsub drivers automatically source this bashrc.

PART-1 (60 points)

Most of the changes you have to make are in the reader.py and train_rnn.py scripts. reader.py has functionality to read a text file and prepare batches of data. train_rnn.py has functionality to train the RNN models. A detailed list of things you will need to complete are as follows:

1. Complete oneHot() method of textDataLoader in reader.py
2. Complete convertHot() method of textDataLoader in reader.py
3. Complete calc_cost() method of SequenceLabelling in train_rnn.py
4. Complete calc_optim() method of SequenceLabelling in train_rnn.py
5. Complete _weight_and_bias() static method of SequenceLabelling in train_rnn.py
6. Question: What is the functionality of dynamic_rnn?
7. Question: What is the difference between the testing and training graphs?
8. Question: Why do we have a warm string?
9. Question: What does the RNN learn initially (first 100 iterations)? What does it learn in the end? Why?
10. Submit: logs, code.

PART-2 (40 points)

1. Why does using a regression loss for surface normal estimation lead to blurry results?
2. What is Batch Normalization and why is it useful? How does Batch Normalization vary for training and testing phases?
3. In a two stream network, when should you have shared weights and when separate?
4. You are training a two stream network for RGBD data (one stream for RGB and the other for D). Your dataset consists of 10,000 images for recognition. How would you initialize the network? How will you avoid overfitting in the network?
5. How would you convert a regression problem to a classification problem?
6. In a multi label image classification what loss would you use?
7. Why are CNNs used more for computer vision tasks than other tasks?
8. How do you find the best hyperparameters like learning rate, momentum, type of optimizer etc. for a new task at hand?

Submission Instructions

Prepare a pdf with the answers to the PART-1 and PART-2 questions. Prepare a folder with all the code you have written PART-1 question (10). Submit your responses here: https://goo.gl/forms/TReASFkUGqYGUSEc2

Note that you can only submit **ONE** response.