CSE 635 - NLP and Text Mining
Individual Project - Sentiment Analysis using Different RNN Architectures
Project Release Date - 9th March, 2020
Due - 23rd March, 2020 (Noon)

Introduction

In this project, we'll be building an RNN-based learning model to detect sentiment (i.e. detect if a sentence is positive or negative) using PyTorch and TorchText. This will be done on movie reviews, using the IMDb dataset.

We will use:
- pre-trained word embeddings
- 3 different RNN architectures
- bidirectional RNN
- multi-layer RNN
- regularization
- optimizer

After completing this project, you are able to understand:

- How sentiment analysis is performed on a given text document and where can it fail?
- How to train RNN based models for NLP task using real data?
- What is the difference between each RNN architecture and why their performance varies so much?
- How to use PyTorch to build deep learning models?

You can download the Python notebooks from here:
https://buffalo.box.com/s/md9el7ncbboxjs3zaiysfjuv3hzn29yd

This project is divided into two phases:

Phase 1: Read the notebook and implement different RNN architectures for Sentiment Analysis.
Phase 2: In-class quiz after completion of this project.

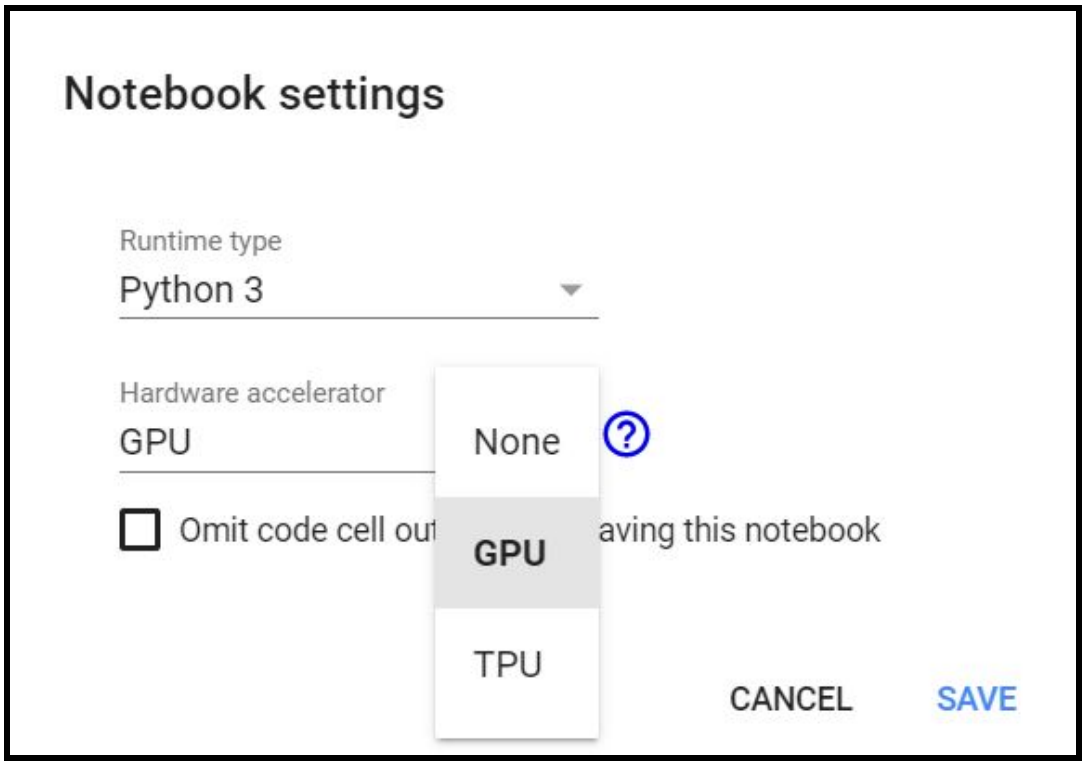
*****Warning*****
This project involves heavy tensor computations and it is required that you work with GPU at the back-end. GPU will help run each architecture within 20-30 minutes (working on CPU might take a few hours). For this project, you will be using Google Colab, which provides free GPU, to implement each architecture. See the installation instructions below to setup Google Colab.

Installation Instructions

Google Colab:

Google Colab provides free GPU access. Go to this link, sign in and upload the notebooks by going to File - > Upload Notebook.

Once the notebook is uploaded, enable GPU by going to Runtime -> Change Runtime Type -> GPU. (Note: Keep the "Omit code cell output while saving this notebook" **unchecked.**)



Phase 1 - Implement RNN Class

In each notebook, you will find a section called TO-DO where you need to write code in the given class to build a specific RNN model. You will find hints in the comments as well as in the

text description above this section that will help you in writing code. You can also refer to official documentation for reference. Some good resources are following:
Welcome to PyTorch Tutorials — PyTorch Tutorials 1.4.0 documentation
Torch.nn and torch.optim
PyTorch documentation — PyTorch master documentation

Phase 2 - Quiz

After completion of this project, we will have a quiz in the class based on the RNN implementation and sentiment analysis task. All students are required to be present in the class for this quiz.

Submission

Deadline: Monday 23rd March, 2020; 12 PM (Noon)
The grading script will trigger exactly at 12:01 PM. No late submission will be entertained, even if it's late by a few seconds.

We would use the CSE submit script for submission and post the submission instructions on Piazza 1 week before the deadline. You need to use this submit script to submit your code for each model's class in a python file (with .py extension) along with the trained models (.pt files). The code to generate .pt files and pointer to their location are already added into your notebook. Read the description in your notebook carefully to find these files. For python files, after writing the code in TO-DO section, copy the code in a text file and save it with .py extension. (Refer lecture video of 9th March.)

Note that naming convention of each file should be as given below (**you must strictly follow the naming convention otherwise you will lose points**):

1. rnn_model.pt
2. birnn_model.pt
3. bilstm_model.pt
4. rnn_model.py
5. birnn_model.py
6. bilstm_model.py

We will run 10 test sentences against the trained models to analyze sentiment of each sentence across all 3 architectures.

Grading

Total weightage of this project is 20 points. If you have submitted all the trained models and if we are able to run our test set against your models, you will receive full points otherwise you will lose points. A log file will be generated based on the testing and it will be emailed to you once we have graded all the students.

Things to keep in mind

1. **Running out of GPU:** If you try to run all three notebooks simultaneously on the GPU, you might run out of it. In that case, you will have to start all over again by doing factory reset. (Runtime -> Factory Reset Runtime.)
2. **Implementation runtime:** Downloading dataset, building vocabulary and training takes some time. Based on my implementation, each of these tasks took 5-10 minutes.
3. **Connection loss with GPU:** When working with Google Colab, you may run into connection loss with GPU. This can happen because (1) session expired (every 12 hours), (2) laptop/computer is on sleep or (3) inactivity on Google Colab platform. Reconnecting means you will have to start all over again from downloading the dataset and building vocabulary.
4. **DO NOT CHANGE PARAMETER VALUES:** You are free to play around with the parameters of the given notebooks, however, for grading, we require all of you to work with the same parameters as given in the original notebooks. Do not submit notebooks/models where you have updated any of the parameters.
5. **Academic Integrity:** This is not a group project and each student is required to run the code and implement models individually. Note that academic integrity is a serious issue and, if violated, there can be serious actions taken by the department.