Indian Institute of Technology, Mandi
February - May 2019

CS671 - Deep Learning and its Applications

Assignment 4

Course Instructor : Aditya Nigam

03 May 2019

Instructions

- Plagiarism is strictly prohibited. In case of violation, a zero will be awarded for this assignment as a warning and a quick F grade if repeated later.
- Submit a README.MD file for each question as well as the full assignment which provides the instructions for running your codes in detail including the versions of programming language and all the modules that have been used.
- Students using Windows or any other OS, are requested to make sure that their code runs perfectly on Linux as mentioned in each problem. Your evaluation will be done on computers in the PC lab.
- Only submit documents that are mentioned in the submission sub-section of each problem, on **Moodle**. Other stuff has to be shown separately.
- The deadline for submission is **Friday, 28th May, 2019, 2359 HRS**. No late submissions will be entertained.
- You are required to upload your codes on **Github** as well as Moodle for all the assignments.
- You are required to make a web-page for your progress in this course. Links to your codes, videos, project pages, should be present on this web-page. Details of this will be conveyed shortly via Moodle.
- The Github and web-page requirement carry **10** marks for each assignment. So take it seriously.
- Contact Aayush Mishra or Daksh Thapar for any queries.

1 Basic RNN

1.1 Motivation

RNNs work on sequential data. Building and training them is quite different from CNNs. This problem is aimed at getting yourself familiar with RNNs. By the end of this problem, you will know how to build a simple RNN and how different it is compared to other networks like CNNs.

1.2 Problem Statement

1. Given sequential values from a **Sine** wave, you are required to predict the next value. The same has to be done for a **Triangle**[1] wave of period 2π bound between -1 and 1.

2. Given sequential terms from an Arithmetic Progression, you are required to predict the next term.

3. Given a positive integer n, you have to generate a sequence using the following generator function,

$$f(n) = \begin{cases} \frac{n}{2} & \text{if } n \equiv 0 \pmod 2 \\ 3n+1 & \text{if } n \equiv 1 \pmod 2 \end{cases} \tag{1}$$

   The sequence is then defined as,

$$a_i = \begin{cases} n & \text{if } i = 0 \\ f(a_{i-1}) & \text{if } i > 0 \end{cases} \tag{2}$$

   The sequence has to be generated until a_i becomes equal to 1. Assume the Collatz Conjecture[2] to be true.

1.3 Notes

- For all the tasks, you will need to generate your own training data.
- For Task 1,
  - For simplicity, do not assume input sequences less than 5 and more than 10 values long.
  - Generate training sequences by function values of sine and triangle waves keeping inputs to those function 10^{-3} apart. An example training sample would be X = [sin(0), sin(1e-3), sin(2e-3), sin(3e-3), sin(4e-3), sin(5e-3)], y = [sin(6e-3)].

[1] Triangle Wave, Wikipedia
[2] Collatz Conjecture, Wikipedia

  - Test data is provided in the pickle files named *sin_tests.pkl* and *triangle_tests.pkl* as a list of tuples (X, y). X is a variable length input sequence list and y is a list of 10000 next outputs.
  - Train the two models separately.

- For Task 2,
  - Generate training data and models for 5 common differences viz. 1, 2, 3, 4, 5.
  - Start each AP from 0.
  - Note that input sequences will have to be at least 2 terms long. Assume maximum input sequence length to be 10.
  - Assume maximum AP length to be 10000.
  - Test data is provided in the pickle files named *ap_[1..5].pkl* as a list of tuples (X, y). X is the variable length input AP, y is the next term in that AP.

- For Task 3,
  - Generate training data with $a_0 <= 10000$.
  - Test data is provided in *collatz_test.pkl* as a list of tuples (X, y). X is the test a_0 and y is the output sequence list ending with $..., 1, 2, 1$.

- Networks are bad with handling large numbers, therefore you might want to normalize your data or use linear activation function on the output layer. The largest numbers in test sets will be less then the largest possible numbers in the training set, in case you normalize.

1.4 Submission

- For Task 1, use the input sequences to generate outputs. Use the predicted output at each timestep to generate another input sequence. Do this for 10000 steps and plot these outputs. On the same graph, plot the actual output values with a different color. Find and report the MSE between all these sequences.

- For Task 2, generate an output file of the predicted numbers, for each test case from corresponding models.

- For Task 3, generate an output file of the predicted sequence for each test case.

- Zip the codes and result files as **1.zip**

2 Playing with RNNs

2.1 Motivation

Having learnt how to make RNNs, you should now be able to understand and reason about their working. Conducting a few experiments with trained RNN models will help you understand them better.

2.2 Experiments

1. Switch the test cases of **sine** and **Triangle** waves. See and analyze the results. Train a single model for both types of input. See and analyze results from this model.

2. Switch the test cases of the 5 models and see the results. Train a new model by picking random common differences from $[1, 5]$. Keep other constraints same as before. Test this model with the given test cases and see if your new model has learnt the generic working of an AP.

2.3 Submission

- For Task 1, plot the results on old models and on the new model, just like in Problem 1.
- For Task 2, generate output files of predicted numbers for each switched test case, and for all cases from the new model.
- Zip the codes and result files as **2.zip**

3 Text Generation

3.1 Motivation

One major field of application of RNNs is Natural Language Processing. RNNs have been exploited in tasks like Neural Translation, Sentiment Analysis and Poetry generation to name a few. This problem will help you to appreciate the strength of RNNs and get a flavour of their power yourself. You will also learn about handling text data, and how it is processed while training intelligent models.

3.2 Problem Statement

You are given a famous novel written by Jane Austen, Pride and Prejudice[3]. You have to train your RNN model to generate words after getting a seed sequence of words.

[3]Taken from Project Gutenberg.

3.3 Notes

- The novel is provided as *pap.txt*.
- Text pre-processing and filtering out irrelevant lines has to be done yourself.
- The vocabulary size is large therefore you need to use word embedding. See **Word2vec** for reference.
- Group words having frequency less than 15 together.
- Input test cases are provided as lines in *text_gen_test.txt*. Generate a paragraph of 5 sentences in a single line for each test case. Save the outputs in *text_gen_test_output.txt* starting each output line with the corresponding input provided.

3.4 Submission

- Evaluation criteria will be conveyed to you during evaluation.
- Zip the code and output file as **3.zip**.

4 General Instructions

- Try RNN, LSTM and GRU layers for these tasks.
- For each problem, the only submission required is the code and whatever result files are mentioned specifically.
- Prepare a report to show all your findings.
- Zip the three zips made for each question as group_ID.zip e.g., **13.zip**. Submit this on Moodle.