Assignment 6

24788 - Introduction to Deep Learning (Spring 2023)

Out Date: 2023/2/24 (Fri)
Due Date: 2023/3/04 (Sat) @ 11:59 pm EST

All exercises should be submitted to Gradescope.

You can refer to Python3 tutorial, Numpy documentation and PyTorch documentation while working on this assignment. Please follow the programming template provided in the assignment. Please use Pizza for any questions on the assignment. You should submit the assignment as:

Submission file structure

```
andrewID-HW6
        p1
            andrewID_p1_report.pdf
        p2
            andrewID_p2.ipynb
            andrewID_p2_report.pdf
```

- Submit the zip file in Homework 6 Programming section of Gradescope
- Submit the PDF of Python notebook in Homework 6 section of Gradescope

Please make sure you run all the cells and your submission clearly shows all the outputs with visible cell numbers.

Theory Exercise (50 points)

PROBLEM 1

Long Short-term Memory (LSTM) (50 points)

Long Short Term Memory networks (LSTM) are a special kind of RNN, capable of learning long-term dependencies. They were first introduced by Hochreiter Schmidhuber (1997), and were refined and popularized by many people in following work. LSTMs are explicitly designed to solve the long-term dependency problem. Remembering information for long periods of time is practically their default behavior, not something they struggle to learn. Fig. 1 shows the structure of LSTM cell.

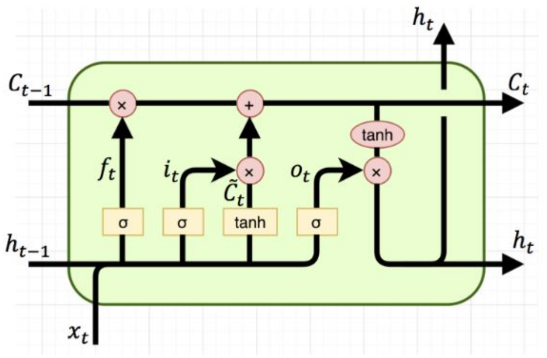


Fig. 1: LSTM

The symbols and expressions of each components are given below, where ft; ct; ot and ht represent forget gate, cell state, output gate and hidden state respectively. Here, $\odot$ represents element-wise multiplication.

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f)$$
$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i)$$
$$\tilde{c}_t = \text{tanh}(W_c x_t + U_c h_{t-1} + b_c)$$
$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t$$
$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o)$$
$$h_t = o_t \odot \text{tanh}(c_t)$$

Here σ and tanh are Sigmoid and Hyperbolic Tangent functions given as:

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$
$$\text{tanh}\,(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

Let's define the parameters as follows:

$$W_f = \begin{bmatrix}1 & 2\end{bmatrix}, U_f = \begin{bmatrix}0.5\end{bmatrix}, b_f = [0.2]$$
$$W_i = \begin{bmatrix}-1 & 0\end{bmatrix}, U_i = \begin{bmatrix}2\end{bmatrix}, b_i = [-0.1]$$
$$W_c = \begin{bmatrix}1 & 2\end{bmatrix}, U_c = \begin{bmatrix}1.5\end{bmatrix}, b_c = [0.5]$$
$$W_o = \begin{bmatrix}3 & 0\end{bmatrix}, U_o = \begin{bmatrix}-1\end{bmatrix}, b_o = [0.8]$$
$$x_1 = \begin{bmatrix}1 \\ 0\end{bmatrix}, \text{with label: } y_1 = 0.5$$
$$x_2 = \begin{bmatrix}0.5 \\ -1\end{bmatrix}, \text{with label: } y_2 = 0.8$$

The True and False problems refer to Fig. 1, For each question EXPLAIN WHY you chose your answer

a) (True/False) If xt is the 0 vector, then $h_t = h_{t-1}$- [8 points]

b) (True/False) (True/False) If f_t is very small or zero, then error will not be back-propagated to earlier time steps. - [8 points]

c) (True/False) The entries of f_t, i_t, o_t are non-negative. - [8 points]

d) (True/False) f_t, i_t, o_t can be viewed as probability distributions. (i.e., their entries are non-negative and their entries sum to 1.) - [8 points]

Problems e, f, and g refer the structure of the LSTM model shown in Fig. 2

e) What is the dimension of f_t, i_t, o_t, h_t [8 points]

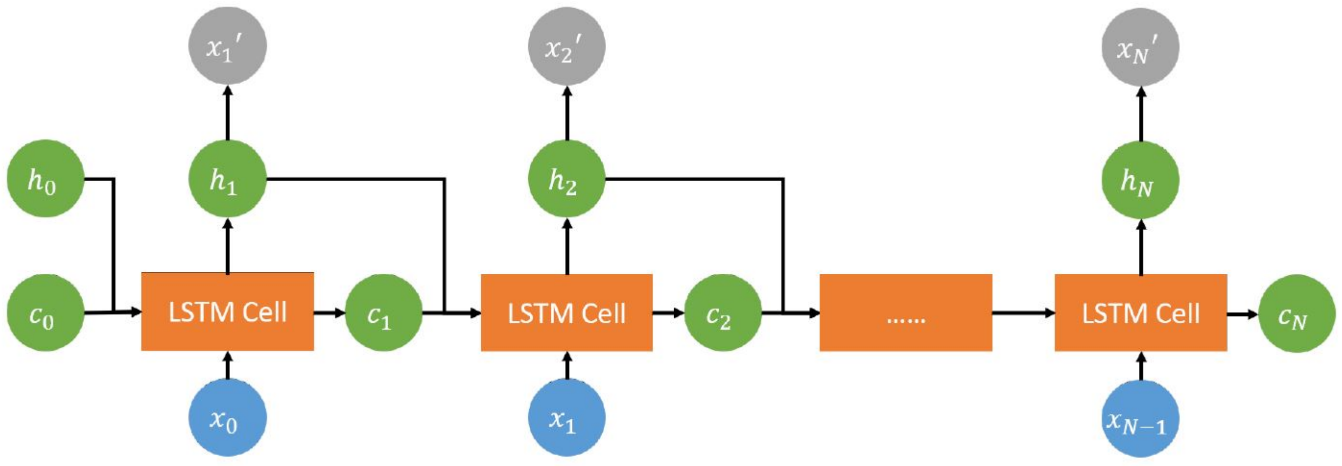
f) Assume the initial h_0, c_0 are both zero vectors. Calculate the output of the model h_1; h_2 [5 points]

g) Assume the problem we are dealing with is a regression problem. Calculate the MSE loss between current prediction and the ground truth label. [5 points]

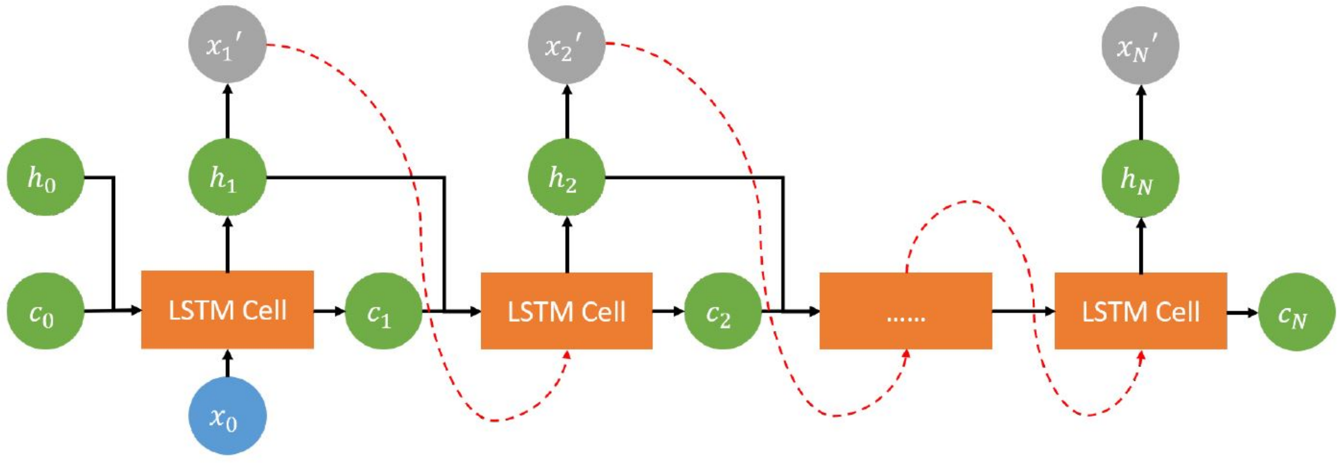
Coding Exercise (50 points)

PROBLEM 2

LSTM in Fluid (50 points)



(a) LSTM training



(b) LSTM testing

Fig. 2: LSTM feed-forward pass

Hagen-Poiseuille flow, first studied experimentally by G. Hagen in 1839 and J. L. Poiseuille in 1940, is flow through a straight circular pipe. We assume that the flow is incompressible, pressure-driven, and one-dimensional along the axis of the pipe with a no-slip condition at the pipe walls. At steady-state, the velocity profile is a parabola. If we assume that the fluid is initially at rest and then set in motion by a pressure difference at the ends of the pipe, then there is a period of time where the velocity profiles are developing until they reach the final steady-state parabolic form. These developing velocity profiles can be represented as a time series. Eq. 1 is the simplified form of Navier-Stokes for this time-dependent part of Hagen-Poiseuille flow. Here, G represents the axial pressure gradient, ρ is the fluid density, v is the fluid viscosity, and ν represents the fluid velocity along the axis of the pipe. Eq. 2 describes the solution to Eq. 1.

We've already generate the Hagen-Poiseuille flow dataset with different boundary conditions. We thank Nina Prakash for providing her code of generating Hagen-Poiseuille flow data. Each sample in the dataset contains

$$\frac{\partial u}{\partial t} = \frac{G}{\rho} + \nu(\frac{\partial^2 u}{\partial r^2} + \frac{1}{r}\frac{\partial u}{\partial r}) \tag{1}$$

$$u(r,t) = \frac{G}{4\mu}(R^2 - r^2) - \frac{2GR^2}{\mu}\sum_{n=1}^{\infty}\frac{J_0(\lambda_n \frac{r}{R})}{\lambda_n^3 J_1(\lambda_n)}\exp(-\lambda_n^2 \frac{\nu t}{R^2}) \tag{2}$$

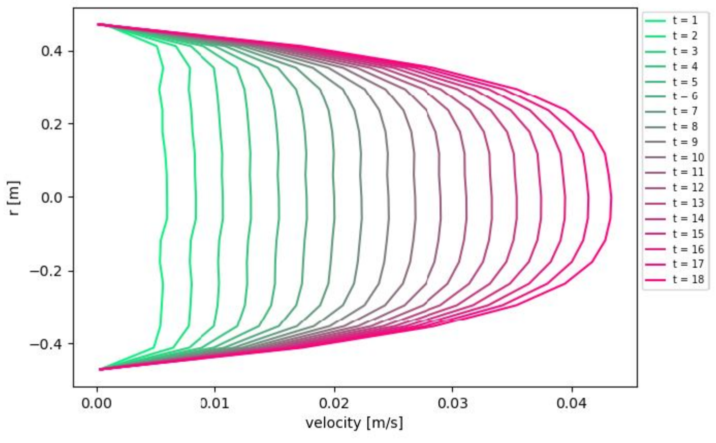
20 timesteps and velocity on 17 points. And the dataset is split into training set and testing set which contains 40500 and 10125 samples respectively. For both dataset, the first timestep represents flow conditions including diameter D axial pressure gradient G, viscosity υ and fluid density ρ. In training set, your input is of dimension (batch size , 19, 17) and ground truth us of dimension (batch size, 19, 17), meaning your model takes input of the ground truth velocity at each time step and predicts the velocity on next time step. However, for test set, the dimension of the input data is (batch size, 17) and ground truth is (batch size, 19, 17). As shown in Fig. 4, where xt is the ground truth input and x0 t is the prediction. This is because during testing, your model only takes the flow conditions (first time step) as input and predict the whole process. The dataset and dataloader is already defined in the starter codes. You can tune the batch size if that helps your training.

In this problem, you are asked to build, train and test a LSTM model via PyTorch. You are encouraged to implement from the starter code. However, you are not strictly restricted to that as long as you submit all your code and complete report. Specifically, in lstm.py file

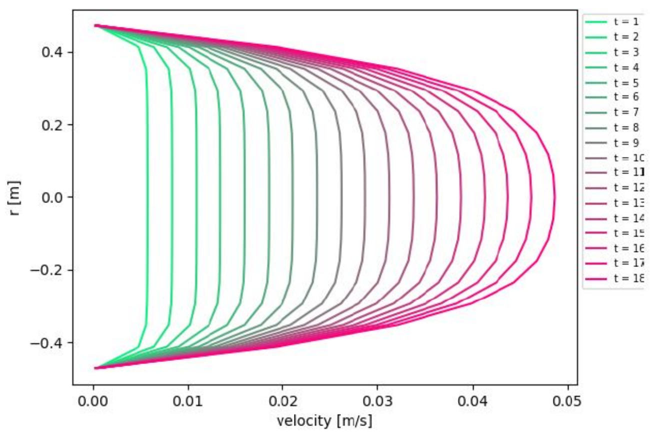
- Build your LSTM model in init function. Here we recommend nn.LSTMCell() instead of nn.LSTM() since the former is a lower-level implementation which is flexible,
- Define the feed forward pass in training in forward() function
- Define the feed forward pass in training in test() function

Also, you need to modify trainlstm.py, which is used to train and test your LSTM model

- Define you loss function and calculate loss at each training iteration
- Tune hyperparameters both in the model and optimizer to improve the performance



(a) Prediction using LSTM



(b) Ground trruth

Fig. 3: LSTM visualization

In your report, you should include:

- Structure of your model (eg. number of layers, number of neurons, etc.)

- Hyper-parameters (eg.learning rate, optimizer, learning rate decay, etc.)
- Visualization of training loss vs. epoch
- Both L1 and L2 error of trained model on test set
- Visualization of prediction of flow velocity comparing to the ground truth, as shown in Fig. 4

For implementation of LSTM in PyTorch, you can refer to this repo: `https://github.com/pytorch/examples/tree/master/time_sequence_prediction.`