Homework 5

LSTM gradient, Here you will derive the backprob updates for the univariate version of the LSTM, i.e., all input, hidden state, and output are univaraiate variables. For reference, below is a LSTM unit at time t and the computations it performs:

Three types of binary gates are created

- Forget gate: $f_t = \sigma(W^{hf} h_{t-1} + W^{xf} X_t + W_0^f)$
- Memory gate: $i_t = \sigma(W^{hi} h_{t-1} + W^{xi} X_t + W_0^i)$
- Output gate: $o_t = \sigma(W^{ho} h_{t-1} + W^{xo} X_t + W_0^o)$

Information generation via gating

- Intermediate memory content generation: $\tilde{C}_t = \tanh(W^{hc} h_{t-1} + W^{xc} X_t + W_0^c)$
- Current memory content generation via gating: $C_t = f_t \otimes C_{t-1} \oplus i_t \otimes \tilde{C}_t$
- Current state generation via gating: $h_t = o_t \otimes \tanh(C_t)$
- Ouput Generation: $y_t = \sigma(W^y h_t + W_0^y)$

Given the gradient for the output y at time t to be $\bigtriangledown y_t$

All these variables are scalar

1. Drive the gradient for the three gates

To obtain the gradient of the gates, we should first know the gradient of $\bigtriangledown h_t$, $\bigtriangledown C_t$. Because the diagram showed there are next step, so we can assume that $t < T$, where T is the last step of time sequence.

Let's first consider the dimension. Since the output layer is a sigmoid function, y_t should be a scalar. h_t is also a scalar, and w^y and w_0^y should also be scalar.

1.1 Compute the gradient of h_t

Because this equation: $y_t = \sigma(z_t^y)$ and $z_t^y = W^y h_t + W_0^y$

The gradient is:

$$\begin{aligned}\bigtriangledown h_t =& \frac{\partial y_t}{\partial h_t} \bigtriangledown y_t + \frac{\partial f_{t+1}}{\partial h_t} \bigtriangledown f_{t+1} + \\ & \frac{\partial i_{t+1}}{\partial h_t} \bigtriangledown i_{t+1} + \frac{\partial o_{t+1}}{\partial h_t} \bigtriangledown o_{t+1} + \frac{\partial \tilde{C}_{t+1}}{\partial h_t} \bigtriangledown \tilde{C}_{t+1} \\ =& w^y y_t (1 - y_t) \bigtriangledown y_t + \frac{\partial f_{t+1}}{\partial h_t} \bigtriangledown f_{t+1} + \\ & \frac{\partial i_{t+1}}{\partial h_t} \bigtriangledown i_{t+1} + \frac{\partial o_{t+1}}{\partial h_t} \bigtriangledown o_{t+1} + \frac{\partial \tilde{C}_{t+1}}{\partial h_t} \bigtriangledown \tilde{C}_{t+1}\end{aligned}$$

However, it seems like we do not need to consider the next level.

$$\begin{aligned}\bigtriangledown h_t =& \frac{\partial y_t}{\partial h_t} \bigtriangledown y_t \\ =& w^y y_t (1 - y_t) \bigtriangledown y_t\end{aligned}$$

1.2 Compute the gradient of $\tilde{C}_t$

Because this equation: $C_t = o_t \otimes \tanh(C_t)$.

In this univariate condition, we can rewrite this equation as: $h_t = o_t \times \tanh(C_t)$.

To help us obtain the gradient we add an auxilliary variable $z_t^c = \frac{h_t}{o_t} = \tanh(C_t)$

The gradient is:

$$\begin{aligned}\bigtriangledown C_t =& \frac{\partial h_t}{\partial C_t} \bigtriangledown h_t + \frac{\partial C_{t+1}}{\partial C_t} \bigtriangledown C_{t+1} \\ =& \frac{\partial z_t^c}{\partial C_t} \frac{\partial h_t}{\partial z_t^c} \bigtriangledown h_t + f_t \bigtriangledown C_{t+1} \\ =& o_t(1 - (\tanh(C_t))^2) \bigtriangledown h_t + f_t \bigtriangledown C_{t+1}\end{aligned}$$

If we do not need to consider the next level.

$$\begin{aligned}\bigtriangledown h_t =& \frac{\partial h_t}{\partial C_t} \bigtriangledown h_t \\ =& o_t(1 - (\tanh(C_t))^2) \bigtriangledown h_t \\ =& o_t(1 - (\tanh(C_t))^2) w^y y_t(1 - y_t) \bigtriangledown y_t\end{aligned}$$

1.3 Compute the gradient of the gates

Since $h_t = o_t \otimes \tanh(C_t) \Rightarrow h_t = o_t \times \tanh(C_t)$:

The $\bigtriangledown o_t = \frac{\partial h_t}{\partial o_t} \bigtriangledown h_t = \tanh(C_t) \bigtriangledown h_t = \tanh(C_t) w^y y_t(1 - y_t) \bigtriangledown y_t$

Since $C_t = f_t \otimes C_{t-1} \oplus i_t \otimes \tilde{C}_t \Rightarrow f_t \times C_{t-1} + i_t \times \tilde{C}_t$:

The $\bigtriangledown f_t = \frac{\partial c_t}{\partial f_t} \bigtriangledown c_t = C_{t-1} \bigtriangledown C_t = C_{t-1} o_t(1 - (\tanh(C_t))^2) w^y y_t(1 - y_t) \bigtriangledown y_t$

The $\bigtriangledown i_t = \frac{\partial c_t}{\partial i_t} \bigtriangledown c_t = \tilde{C}_t \bigtriangledown C_t = \tilde{C}_t o_t(1 - (\tanh(C_t))^2) w^y y_t(1 - y_t) \bigtriangledown y_t$

2 Derive the gradient for the weight W^{hi}

To compute the memory gate, we use: $i_t = \sigma(w_t^{hi} h_{t-1} + w_t^{xi} X_t + w_{t,0}^i)$

We can add an auxillary variable $z_t^i = w_t^{hi} h_{t-1} + w_t^{xi} X_t + w_{t,0}^i$

So the equation becomes: $i_t = \sigma(z_t^i)$

The gradient becomes:

$$\begin{aligned}
\bigtriangledown w_t^{hi} =& \frac{\partial i_t}{\partial w^{hi}} \bigtriangledown i_t \\
=& \frac{\partial z_t^i}{\partial C_t} \frac{\partial i_t}{\partial z_t^i} \bigtriangledown i_t \\
=& h_{t-1} i_t (1 - i_t) \bigtriangledown i_t \\
=& h_{t-1} i_t (1 - i_t) \tilde{C}_t o_t (1 - (\tanh(C_t))^2) w^y y_t (1 - y_t) \bigtriangledown y_t
\end{aligned}$$

Hence we need to sum over time, to caculate w^{hi}

$$\bigtriangledown w_t^h = \sum_{i=1}^{T} h_{t-1} i_t (1 - i_t) \tilde{C}_t o_t (1 - (\tanh(C_t))^2) w^y y_t (1 - y_t) \bigtriangledown y_t$$

3 Based on your answers above, explain why the gradient don't explode if the value of the forget gates are very close to 1. and the values of the memory gates are very close to 0.

if the value of the forget gates are close to 1 and memory gates are close to 0, whici means $f_t \to 1, i_t \to 0$. Then $c_t \approx c_{t+1}$, the memory cell will always be close to the initial value, which should be a small number, due to we usually initiate the network with a small value. All the h_t should also be a small value. Because $h_t = o_t \otimes \tanh(C_t)$, $o_t \in [0, 1]$ as the output of sigmoid, $\tanh(C_t) \in [-1, 1]$ as the output of tanh function. $\tilde{C}_t = \tanh(W^{hc} h_{t-1} + W^{xc} X_t + W_0^c)$ shouldn't be large if we carefully preprocessed the input. Based on all these, if we look at the $\bigtriangledown w_t^h$ equation, h_{t-1} is small, $i \to 0$, $\tilde{C}_t$ is small, $o_t \in [0, 1]$, $1 - (\tanh(C_t))^2) \in [0, 1]$ $y_t(1 - y_t) \in [0, 1/4]$, so the gradient will not explode. The same conclusion should be able to be applied to other gradients.

In [ ]: