Homework 2: Convolutional Neural Networks and Recurrent Neural Networks

CSCI-GA 2572 Deep Learning

Spring 2022

The goal of homework 2 is to get you to work with convolutional neural networks and recurrent neural networks.

In the theoretical part, you will work on figuring out how backpropagation works in these networks. In part 2, we will implement and train them.

In part 1, you should submit all your answers in a pdf file. As before, we recommend using LaTeX.

For part 2, you will implement some neural networks by adding your code to the provided ipynb file.

As before, please use numerator layout.

The due date of homework 2 is 5:00pm 03/04. Submit the following files in a zip file `your_net_id.zip` through NYU Brightspace:

- `hw2_theory.pdf`
- `hw2_cnn.ipynb`
- `hw2_rnn.ipynb`
- `08-seq_classification.ipynb`

The following behaviors will result in penalty of your final score:

1. 10% penalty for submitting your file without using the correct naming format (including naming the zip file, PDF file or python file wrong, adding extra files in the zip folder, like the testing scripts in your zip file).
2. 20% penalty for late submission within the first 24 hours after the deadline. We will not accept any late submission after the first 24 hours.
3. 20% penalty for code submission that cannot be executed following the steps we mentioned.

1 Theory (50pt)

1.1 Convolutional Neural Netoworks (20 pts)

(a) (2 pts) Given an input image of dimension 10×11, what will be output dimension after applying a convolution with 5×5 kernel, stride of 2, and no padding?

(b) (3 pts) Given an input of dimension $C \times H \times W$, what will be the dimension of the output of a convolutional layer with kernel of size $K \times K$, padding P, stride S, dilation D, and F filters. Assume that $H \geq K$, $W \geq K$.

(c) (15 pts) In this section, we are going to work with 1-dimensional convolutions. Discrete convolution of 1-dimensional input $x[n]$ and kernel $k[n]$ is defined as follows:

$$s[n] = (x * k)[n] = \sum_m x[n-m]k[m]$$

However, in machine learning convolution is usually implemented as cross-correlation, which is defined as follows:

$$s[n] = (x * k)[n] = \sum_m x[n+m]k[m]$$

Note the difference in signs, which will get the network to learn an "flipped" kernel. In general it doesn't change much, but it's important to keep it in mind. In convolutional neural networks, the kernel $k[n]$ is usually 0 everywhere, except a few values near 0: $\forall_{|n|>M} k[n] = 0$. Then, the formula becomes:

$$s[n] = (x * k)[n] = \sum_{m=-M}^{M} x[n+m]k[m]$$

Let's consider an input $x[n]$, $x : \{1,2,3,4,5,6,7\} \to \mathbb{R}^2$ of dimension 7, with 2 channels, and a convolutional layer f_W with one filter, with kernel size 5, stride of 2, no dilation, and no padding. The only parameters of the convolutional layer is the weight W, $W \in \mathbb{R}^{1\times 2\times 5}$, there's no bias and no non-linearity.

(i) (2 pts) What is the dimension of the output $f_W(x)$? Provide an expression for the value of elements of the convolutional layer output $f_W(x)$. Example answer format here and in the following sub-problems: $f_W(x) \in \mathbb{R}^{42\times 42\times 42}$, $f_W(x)[i,j,k] = 42$.

(ii) (4 pts) What is the dimension of $\frac{\partial f_W(x)}{\partial W}$? Provide an expression for the values of the derivative $\frac{\partial f_W(x)}{\partial W}$.

(iii) (4 pts) What is the dimension of $\frac{\partial f_W(x)}{\partial x}$? Provide an expression for the values of the derivative $\frac{\partial f_W(x)}{\partial x}$.

(iv) (5 pts) Now, suppose you are given the gradient of the loss ℓ w.r.t. the output of the convolutional layer $f_W(x)$, *i.e.* $\frac{\partial \ell}{\partial f_W(x)}$. What is the dimension of $\frac{\partial \ell}{\partial W}$? Provide an expression for $\frac{\partial \ell}{\partial W}$. Explain similarities and differences of this expression and expression in (i).

Solution:

(a)

$C_{in} = 1$
$H_{in} = 10$
$W_{in} = 11$
$stride = 2$

$$H_{\text{out}} = \left\lfloor \frac{H_{\text{in}} + 2 \times \text{ padding}[0] - \text{ dilation}[0] \times (\text{ kernel}_{\text{ size}}[0] - 1) - 1}{\text{stride}[0]} + 1 \right\rfloor \tag{1}$$

$$W_{\text{out}} = \left\lfloor \frac{W_{\text{in}} + 2 \times \text{ padding}[1] - \text{ dilation}[1] \times (\text{ kernel}_{\text{ size}}[1] - 1) - 1}{\text{stride}[1]} + 1 \right\rfloor \tag{2}$$

$H_{out} = (H_{in} - K + 2P)/S + 1 = (10 - 5)/2 + 1 = 3$
$W_{out} = (W_{in} - K + 2P)/S + 1 = (11 - 5)/2 + 1 = 4$
3 x 4

(b)

$$H_{\text{out}} = \left\lfloor \frac{H_{\text{in}} + 2 \times \text{ padding}[0] - \text{ dilation}[0] \times (\text{ kernel}_{\text{ size}}[0] - 1) - 1}{\text{stride}[0]} + 1 \right\rfloor \tag{3}$$

$$= \lfloor \frac{H + 2P - D \times (K-1)S + 1}{2} + 1 \rfloor \tag{4}$$

$$W_{\text{out}} = \left\lfloor \frac{W_{\text{in}} + 2 \times \text{ padding}[1] - \text{ dilation}[1] \times (\text{ kernel}_{\text{ size}}[1] - 1) - 1}{\text{stride}[1]} + 1 \right\rfloor \tag{5}$$

$$= \lfloor \frac{W + 2P - D \times (K-1)S + 1}{2} + 1 \rfloor \tag{6}$$

Output dimension is

$$C_{\text{out}} \times H_{\text{out}} \times W_{\text{out}}$$

If $C_{out} = C_{in}$, then the output dimension is

$$C \times H_{\text{out}} \times W_{\text{out}}$$

(c)

(i)

The dimension of output is

$$2 \times 2$$

$f_W(x) \in \mathbb{R}^{2\times 2}$,

$$f_W[i,j] = \sum_{m=-2}^{2} x[i+m]W[j,m+2], \tag{7}$$

where $i \in \{3,5\}, j \in \{1,2\}$

$$f_W(x) = \begin{bmatrix} \sum_{m=-2}^{2} x[3+m]W[1,m+2] & \sum_{m=-2}^{2} x[3+m]W[2,m+2] \\ \sum_{m=-2}^{2} x[5+m]W[1,m+2] & \sum_{m=-2}^{2} x[5+m]W[2,m+2] \end{bmatrix} \tag{8}$$

This is a column matrix, the first column is the first channel, the second column is the second channel.

W[1, j, m] is the j-th channel of the m-th weight.

(ii)

The dimension of $\frac{\partial f_W(x)[i][j]}{\partial W}$ is the same as the dimension of W, the $f_W(x)$ is a tensor, W is a matrix, when we calculate the derivative of tensor to vector, we need to calculate each element in the tensor to vector derivative

$$\frac{\partial f_W(x)}{\partial W} \in \mathbb{R}^{(2*2)\times(2*5)} \tag{9}$$

$$\frac{\partial f_W(x)[i,j]}{\partial W[1]} = \big[x[i-2], x[i-1], x[i], x[i+1], x[i+2]\big] \tag{10}$$

$$\frac{\partial f_W(x)[i,j]}{\partial W[2]} = \big[x[i-2], x[i-1], x[i], x[i+1], x[i+2]\big] \tag{11}$$

$$\frac{\partial f_W(x)[i]}{\partial W} \in \mathbb{R}^{2\times 2\times 5} \tag{12}$$

Thus, the whole derivative is a tensor of shape $\frac{\partial f_W(x)}{\partial W} \in \mathcal{R}^{(2\times 2)\times(2\times 5)}$.

while the first dimension is element index in channel , the second dimension is the channel j, and the third dimension is the j-th weight channel, the fourth dimension is the derivative for the i-th item in the j-th weightc channel.

(iii)

What is the dimension of $\frac{\partial f_W(x)}{\partial x}$? Provide an expression for the values of the derivative $\frac{\partial f_W(x)}{\partial x}$.

$f_W(x)$ is a matrix, x is a vector, so the derivative is a matrix of shape $\frac{\partial f_W(x)}{\partial x} \in \mathcal{R}^{2\times 2\times 7}$.

$$\frac{\partial f_W(x)[1,1]}{\partial x} = \left[W[1,1], W[1,2], W[1,3], W[1,4], W[1,5], 0, 0\right]^T \tag{13}$$

$$\frac{\partial f_W(x)[1,2]}{\partial x} = \left[W[2,1], W[2,2], W[2,3], W[2,4], W[2,5], 0, 0\right]^T \tag{14}$$

$$\frac{\partial f_W(x)[2,1]}{\partial x} = \left[0, 0, W[1,3], W[1,4], W[1,5], W[1,6], W[1,7]\right]^T \tag{15}$$

$$\frac{\partial f_W(x)[2,2]}{\partial x} = \left[0, 0, W[2,3], W[2,4], W[2,5], W[2,6], W[2,7]\right]^T \tag{16}$$

(iv)

(5 pts) Now, suppose you are given the gradient of the loss ℓ w.r.t. the output of the convolutional layer $f_W(x)$, *i.e.* $\frac{\partial \ell}{\partial f_W(x)}$. What is the dimension of $\frac{\partial \ell}{\partial W}$? Provide an expression for $\frac{\partial \ell}{\partial W}$. Explain similarities and differences of this expression and expression in (i).

The dimension of $\frac{\partial \ell}{\partial W}$ is the same as the dimension of W, which is $\mathcal{R}^{1\times 2\times 5}$. The loss is a scalar, so the derivative is scalar by matrix, the dimensino is the same as the dimension of W.

$$\frac{\partial \ell}{\partial W} = \frac{\partial \ell}{\partial f_W(x)} \frac{\partial f_W(x)}{\partial W} \tag{17}$$

$$\frac{\partial \ell}{\partial f_W(x)} \in R^{2\times 2}$$

$$\frac{\partial f_W(x)}{\partial W} \in \mathcal{R}^{(2\times 2)\times(2\times 5)}$$

$(2\times 2)\times((2\times 2)\times(2\times 5)) = 2\times 5$

We can **sum all the path gradients** of the loss to get the gradient of the loss w.r.t. the weights.

$$\frac{\partial \ell}{\partial W[j]} = \sum_{i=1}^{2} \frac{\partial \ell}{\partial f_W(x)[i][j]} \frac{\partial f_W(x)[i][j]}{\partial W[j]} \tag{18}$$

The similarities is that the term use the result from (i).

The difference is that scalar by matrix derivative sum up all the path gradients to the the element.

1.2 Recurrent Neural Networks (20 pts)

In this section we consider a simple recurrent neural network defined as follows:

$$c[t] = \sigma(W_c x[t] + W_h h[t-1]) \tag{19}$$
$$h[t] = c[t] \odot h[t-1] + (1 - c[t]) \odot W_x x[t] \tag{20}$$

where σ is element-wise sigmoid, $x[t] \in \mathbb{R}^n$, $h[t] \in \mathbb{R}^m$, $W_c \in \mathbb{R}^{m \times n}$, $W_h \in \mathbb{R}^{m \times m}$, $W_x \in \mathbb{R}^{m \times n}$, $\odot$ is Hadamard product, $h[0] \doteq 0$.

(a) (5 pts) Draw a diagram for this recurrent neural network, similar to the diagram of RNN we had in class. We suggest using diagrams.net.

$$c[t] = \sigma(W_c x[t] + W_h h[t-1]) \tag{21}$$
$$h[t] = c[t] \odot h[t-1] + (1 - c[t]) \odot W_x x[t] \tag{22}$$

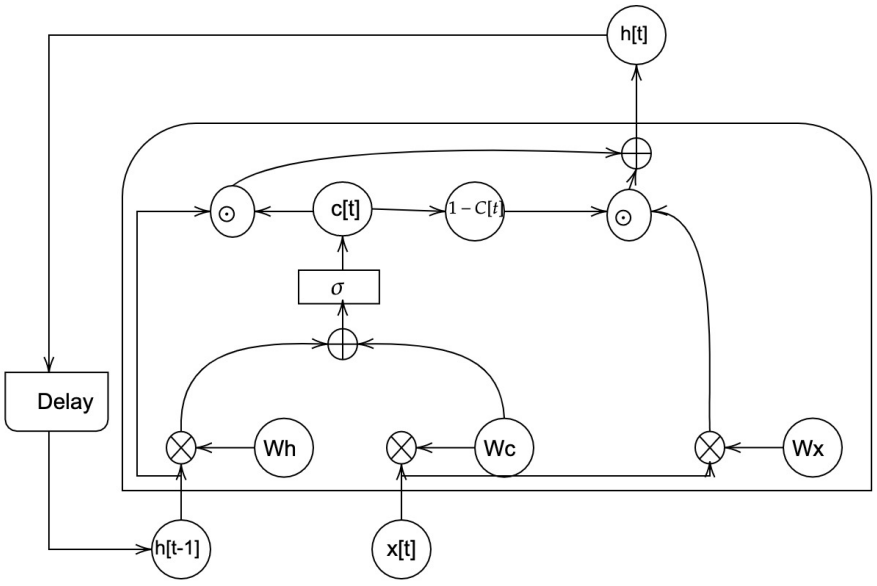
(b) (2pts) What is the dimension of $c[t]$?

(c) (10 pts) Suppose that we run the RNN to get a sequence of $h[t]$ for t from 1 to K. Assuming we know the derivative $\frac{\partial \ell}{\partial h[t]}$, provide dimension of and an expression for values of $\frac{\partial \ell}{\partial W_x}$. What are the similarities of backward pass and forward pass in this RNN?

(d) (3pts) Can this network be subject to vanishing or exploding gradients? Why?

Solution:

(a)



(b)

where σ is element-wise sigmoid, $x[t] \in \mathbb{R}^n$, $h[t] \in \mathbb{R}^m$, $W_c \in \mathbb{R}^{m \times n}$, $W_h \in \mathbb{R}^{m \times m}$, $W_x \in \mathbb{R}^{m \times n}$, $\odot$ is Hadamard product, $h[0] \doteq 0$.

$$C[t] = \sigma(W_c x[t] + W_h h[t-1]) \tag{23}$$

$$C[t] \in (\mathbb{R}^{(m \times n)(n)} + \mathbb{R}^{(m \times m)(m)}) \tag{24}$$

$$\in \mathbb{R}^m \tag{25}$$

(c)

(10 pts) Suppose that we run the RNN to get a sequence of $h[t]$ for t from 1 to K. Assuming we know the derivative $\frac{\partial \ell}{\partial h[t]}$, provide dimension of and an expression for values of $\frac{\partial \ell}{\partial W_x}$. What are the similarities of backward pass and forward pass in this RNN?

$$\frac{\partial \ell}{\partial W_x} \in \mathcal{R}^{m \times n} \tag{26}$$

$$\frac{\partial \ell}{\partial h[t]} \in \mathcal{R}^{m \times 1} \tag{27}$$

$$h[t] = c[t] \odot h[t-1] + (1 - c[t]) \odot W_x x[t] \tag{28}$$

$$\frac{\partial \ell}{\partial W_x} = \frac{\partial \ell}{\partial h[t]} \frac{\partial h[t]}{\partial W_x} \tag{29}$$

$$= (1 - c[t]) \odot (\frac{\partial \ell}{\partial h[t]}) x[t]^T \tag{30}$$

The similarities are that the forward pass is a function of the current input $x[t]$ and the previous hidden state $h[t-1]$, and the backward pass is a function of the current hidden state $h[t]$ and the next input $x[t]$.

(d)

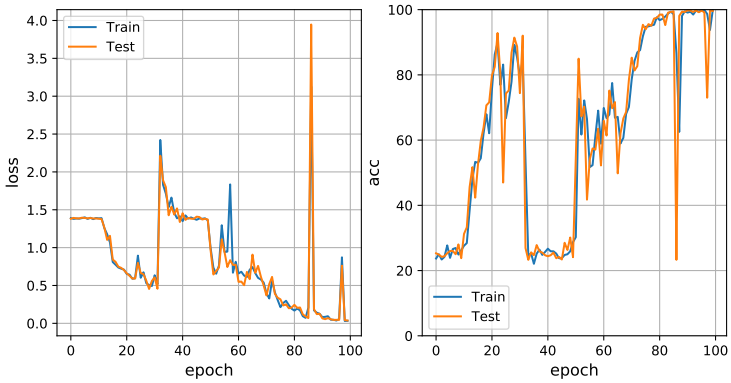
$$\frac{\partial \ell}{\partial h[t-1]} = \frac{\partial \ell}{\partial h[t]} (\frac{\partial h[t]}{\partial h[t-1]} + \frac{\partial h[t]}{\partial c[t]} \frac{\partial c[t]}{\partial h[t-1]}) \tag{31}$$

There is a path from h[t] to h[t-1] without passing through the sigmoid, which can somehow prevent the gradient vanishing problem.

But for the gradient exploding, there is no gate to regularize the gradient, there is a direct path from h[t] to h[t-1], if the gradient is too large, it will make all the steps gradients explode, the model will still suffer from the gradient exploding problem.

1.3 Debugging loss curves (10pts)

In the class on Wednesday, February 16, when working with notebook `08-seq_classification`, we saw RNN training curves. In Section 8 “Visualize LSTM”, we observed some “kinks” in the loss curve.



1. (2pts) What caused the spikes on the left?
2. (2pts) How can they be higher than the initial value of the loss?
3. (3pts) What are some ways to fix them?
4. (3pts) Explain why the loss and accuracy are at these values before training starts. You may need to check the task definition in the notebook.

Solution:

1

The network get very large gradients in the wrong direction.

2

The wrong gradients step is very large, which is larger than the cummulative gradient step from the training start.

3

Clipping the gradients to prevent large wrong gradients. Momentum can also help.

4

There are 4 sequence classes Q, R, S, U, thus the initial acc is 0.25 due to random guessing.

The loss is at the initial value before training starts.

For Cross Entory, the loss is is the expectation of the uniform distribution and the one hot distributino.

$$Loss = -1 * log(1/4) = log(4) = 1.3863 \tag{32}$$

$$\text{(e as the base)} \tag{33}$$

2 Implementation (50pts + 5pts extra credit)

There are three notebooks in the practical part:

- (25pts) Convolutional Neural Networks notebook: `hw2_cnn.ipynb`
- (20pts) Recurrent Neural Networks notebook: `hw2_rnn.ipynb`
- (5pts + 5pts extra credit) : This builds on Section 1.3 of the theoretical part.
  - (5pts) Change the model training procedure of Section 8 in `08-seq_classification` to make the training curves have no spikes. You should only change the training of the model, and not the model itself or the random seed.
  - (5pts extra credit) Visualize the gradients and weights throughout training before and after you fix the training procedure.

Plase use your NYU Google Drive account to access the notebooks. First two notebooks contain parts marked as `TODO`, where you should put your code. These notebooks are Google Colab notebooks, you should copy them to your drive, add your solutions, and then download and submit them to NYU Brightspace. The notebook from the class, if needed, can be uploaded to Colab as well.