ECE 449 Machine Learning (ZJU-UIUC Institute, SP21)
Assignment 2 Solution (Conceptual Part)

Problem 1

Imagine that you are working on a big and complicated Convolutional Neural Network(CNN) with many convolutional building blocks shown on the right. The convolutional building block has three layers: a convolution layer with kernel $K \in \mathbb{R}^{k\times k}$, stride=1, and no padding, a sigmoid activation layer(with sigmoid function $f(x) = \frac{1}{1+e^{-x}}$), and a 2×2 average pooling layer. The building block input is $X \in \mathbb{R}^{(2n+k-1)\times(2m+k-1)}$, the output from convolution layer is $C \in \mathbb{R}^{2n\times 2m}$, the output from activation layer is $S \in \mathbb{R}^{2n\times 2m}$, and the building block output from the 2×2 average pooling layer is $P \in \mathbb{R}^{n\times m}$.

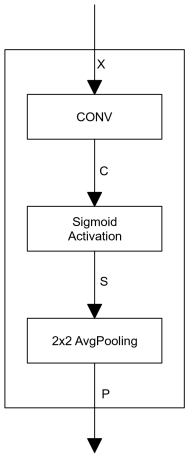

Figure 1: Convolutional Building Block For Problem 1

Given $X = \begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,2m+k-1} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,2m+k-1} \\ \vdots & \vdots & \ddots & \vdots \\ x_{2n+k-1,1} & x_{2n+k-1,2} & \cdots & x_{2n+k-1,2m+k-1} \end{bmatrix}$, $S = \begin{bmatrix} s_{1,1} & s_{1,2} & \cdots & s_{1,2m} \\ s_{2,1} & s_{2,2} & \cdots & s_{2,2m} \\ \vdots & \vdots & \ddots & \vdots \\ s_{2n,1} & s_{2n,2} & \cdots & s_{2n,2m} \end{bmatrix}$,

and $\frac{\partial loss}{\partial P} = \begin{bmatrix} g_{1,1} & g_{1,2} & \cdots & g_{1,m} \\ g_{2,1} & g_{2,2} & \cdots & g_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ g_{n,1} & g_{n,2} & \cdots & g_{n,m} \end{bmatrix}$, write down the expressions for $\frac{\partial loss}{\partial S}$, $\frac{\partial loss}{\partial C}$, and $\frac{\partial loss}{\partial K}$, using $g_{i,j}$ for $1 \le i \le n$, $1 \le j \le m$; $s_{d,e}$ for $1 \le d \le 2n$, $1 \le e \le 2m$; and $x_{a,b}$ for $1 \le a \le (2n+k-1)$, $1 \le b \le (2m+k-1)$. [8 pts]

Problem 1 Solution

$$\frac{\partial loss}{\partial S} = \begin{bmatrix} \frac{g_{1,1}}{4} & \frac{g_{1,1}}{4} & \frac{g_{1,2}}{4} & \frac{g_{1,2}}{4} & \cdots & \frac{g_{1,m}}{4} & \frac{g_{1,m}}{4} \\ \frac{g_{1,1}}{4} & \frac{g_{1,1}}{4} & \frac{g_{1,2}}{4} & \frac{g_{1,2}}{4} & \cdots & \frac{g_{1,m}}{4} & \frac{g_{1,m}}{4} \\ \frac{g_{2,1}}{4} & \frac{g_{2,1}}{4} & \frac{g_{2,2}}{4} & \frac{g_{2,2}}{4} & \cdots & \frac{g_{2,m}}{4} & \frac{g_{2,m}}{4} \\ \frac{g_{2,1}}{4} & \frac{g_{2,1}}{4} & \frac{g_{2,2}}{4} & \frac{g_{2,2}}{4} & \cdots & \frac{g_{2,m}}{4} & \frac{g_{2,m}}{4} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \frac{g_{n,1}}{4} & \frac{g_{n,1}}{4} & \frac{g_{n,2}}{4} & \frac{g_{n,2}}{4} & \cdots & \frac{g_{n,m}}{4} & \frac{g_{n,m}}{4} \\ \frac{g_{n,1}}{4} & \frac{g_{n,1}}{4} & \frac{g_{n,2}}{4} & \frac{g_{n,2}}{4} & \cdots & \frac{g_{n,m}}{4} & \frac{g_{n,m}}{4} \end{bmatrix}$$

$\frac{\partial loss}{\partial S}_{d,e} = \frac{1}{4} g_{\lceil \frac{d}{2} \rceil, \lceil \frac{e}{2} \rceil}$, for $1 \leq d \leq 2n$, $1 \leq e \leq 2m$.

$$\frac{\partial loss}{\partial C} = \begin{bmatrix} \frac{g_{1,1}}{4} \times s_{1,1}(1-s_{1,1}) & \frac{g_{1,1}}{4} \times s_{1,2}(1-s_{1,2}) & \frac{g_{1,2}}{4} \times s_{1,3}(1-s_{1,3}) & \frac{g_{1,2}}{4} \times s_{1,4}(1-s_{1,4}) & \cdots & \frac{g_{1,m}}{4} \times s_{1,2m}(1-s_{1,2m}) \\ \frac{g_{1,1}}{4} \times s_{2,1}(1-s_{2,1}) & \frac{g_{1,1}}{4} \times s_{2,2}(1-s_{2,2}) & \frac{g_{1,2}}{4} \times s_{2,3}(1-s_{2,3}) & \frac{g_{1,2}}{4} \times s_{2,4}(1-s_{2,4}) & \cdots & \frac{g_{1,m}}{4} \times s_{2,2m}(1-s_{2,2m}) \\ \frac{g_{2,1}}{4} \times s_{3,1}(1-s_{3,1}) & \frac{g_{2,1}}{4} \times s_{3,2}(1-s_{3,2}) & \frac{g_{2,2}}{4} \times s_{3,3}(1-s_{3,3}) & \frac{g_{2,2}}{4} \times s_{3,4}(1-s_{3,4}) & \cdots & \frac{g_{2,m}}{4} \times s_{3,2m}(1-s_{3,2m}) \\ \frac{g_{2,1}}{4} \times s_{4,1}(1-s_{4,1}) & \frac{g_{2,1}}{4} \times s_{4,2}(1-s_{4,2}) & \frac{g_{2,2}}{4} \times s_{4,3}(1-s_{4,3}) & \frac{g_{2,2}}{4} \times s_{4,4}(1-s_{4,4}) & \cdots & \frac{g_{2,m}}{4} \times s_{4,2m}(1-s_{4,2m}) \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \frac{g_{n,1}}{4} \times s_{2n,1}(1-s_{2n,1}) & \frac{g_{n,1}}{4} s_{2n,2} \times (1-s_{2n,2}) & \frac{g_{n,2}}{4} \times s_{2n,3}(1-s_{2n,3}) & \frac{g_{n,2}}{4} \times s_{2n,4}(1-s_{2n,4}) & \cdots & \frac{g_{n,m}}{4} \times s_{2n,2m}(1-s_{2n,2m}) \end{bmatrix}$$

$\frac{\partial loss}{\partial C}_{d,e} = \frac{1}{4} g_{\lceil \frac{d}{2} \rceil, \lceil \frac{e}{2} \rceil} \times s_{d,e}(1-s_{d,e})$, for $1 \leq d \leq 2n$, $1 \leq e \leq 2m$.

$$\frac{\partial loss}{\partial K} = \begin{bmatrix} \sum_{i=1}^{2n}\sum_{j=1}^{2m} f(i,j)x_{i,j} & \sum_{i=1}^{2n}\sum_{j=1}^{2m} f(i,j)x_{i,j+1} & \cdots & \sum_{i=1}^{2n}\sum_{j=1}^{2m} f(i,j)x_{i,j+k-1} \\ \sum_{i=1}^{2n}\sum_{j=1}^{2m} f(i,j)x_{i+1,j} & \sum_{i=1}^{2n}\sum_{j=1}^{2m} f(i,j)x_{i+1,j+1} & \cdots & \sum_{i=1}^{2n}\sum_{j=1}^{2m} f(i,j)x_{i+1,j+k-1} \\ \vdots & \vdots & \ddots & \vdots \\ \sum_{i=1}^{2n}\sum_{j=1}^{2m} f(i,j)x_{i+k-1,j} & \sum_{i=1}^{2n}\sum_{j=1}^{2m} f(i,j)x_{i+k-1,j+1} & \cdots & \sum_{i=1}^{2n}\sum_{j=1}^{2m} f(i,j)x_{i+k-1,j+k-1} \end{bmatrix}$$

Where $f(i,j) = \frac{g_{\lceil \frac{i}{2} \rceil \lceil \frac{j}{2} \rceil}}{4} \times s_{i,j} \times (1-s_{i,j})$.

$\frac{\partial loss}{\partial K}_{d,e} = \sum_{i=1}^{2n}\sum_{j=1}^{2m} \frac{g_{\lceil \frac{i}{2} \rceil \lceil \frac{j}{2} \rceil}}{4} s_{i,j}(1-s_{i,j}) x_{i+d-1,j+e-1}$, for $1 \leq d \leq k$ and $1 \leq e \leq k$.

Problem 2

Problem 2.1

Please state at least two advantages of one-stage models compared with multi-stage models in object detection tasks. [4 pts]

Problem 2.2

Please state briefly why Non-maximum suppression(NMS) is generally needed for bounding-box based approaches in object detection tasks. [4 pts]

Problem 2 Solution

Problem 2.1

1. One-stage model training is a single stage pipline, it is more convenient compared with multi-stage model training, which might be a multi-stage pipleline.

2. One-stage model often runs faster than multi-stage model in model inferencing tasks, and it also often consumes fewer compuational resources.

Problem 2.2

Non-maximum suppression(NMS) is a convenient method which filters multiple bounding box predictions whose centers are close and whose areas overlaps largely, and only output the bounding-box that are the best matching results. Without NMS our bounding-box based model may output many bounding boxes that have close center positions and large overlapped areas, which are inefficient for drawing clean and clear conclusions on object detection tasks on input images. We can "clean up" our model inferece results by using NMS.

The above solutions are just examples, multiple answers could be accepted.

Problem 3

Imagine we have a Convolutional Neural Network with its structure shown below:

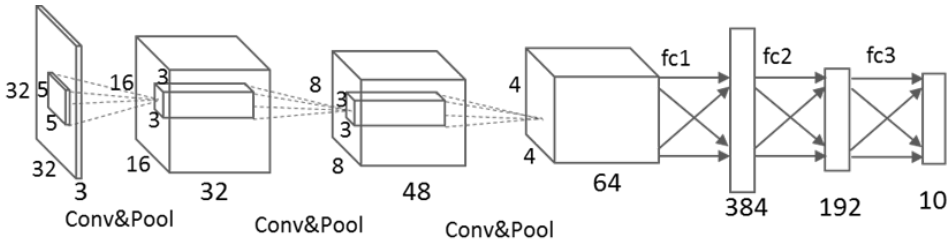

Figure 2: CNN For Problem 3

Please calculate the total number of parameters (including all weights and all biases, suppose biases are used wherever possible) of this CNN. [8 pts]

Problem 3 Solution

$(5\times5\times3+1)\times32+(3\times3\times32+1)\times48+(3\times3\times48+1)\times64+(4\times4\times64+1)\times384+(384+1)\times192+(192+1)\times10 = 513466$

Problem 4

Suppose we have a neural network model with a softmax layer shown as below:

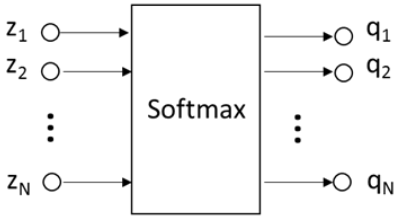

Figure 3: Softmax layer For Problem 4

Where $q_i = \frac{e^{z_i}}{\sum_{j=1}^{N} e^{z_j}}$ for $1 \leq i \leq N$.

We want our softmax layer to learn a mapping from $Z = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_N \end{bmatrix}$ to $P = \begin{bmatrix} p_1 \\ p_2 \\ \vdots \\ p_N \end{bmatrix}$,

where we have

$$\sum_{i=1}^{N} p_i = 1$$

.

Given the objective function

$$E = -\sum_{i=1}^{N} (p_i \times ln(q_i))$$

Please prove: $\frac{\partial E}{\partial z_x} = q_x - p_x$, for $1 \leq x \leq N$. [8 pts]

Problem 4 Solution

$$E = -\sum_{i=1}^{N} (p_i \times ln(q_i)) = -\sum_{i=1}^{N} \left(p_i \times ln \left(\frac{e^{z_i}}{\sum_{j=1}^{N} e^{z_j}} \right) \right)$$

$$= -\sum_{i=1}^{N} \left(p_i \times \left(ln\left(e^{z_i}\right) - ln\left(\sum_{j=1}^{N} e^{z_j} \right) \right) \right) = -\sum_{i=1}^{N} \left(p_i \times \left(z_i - ln\left(\sum_{j=1}^{N} e^{z_j} \right) \right) \right)$$

$$= -\sum_{i=1}^{N} \left(p_i \times z_i - p_i \times ln\left(\sum_{j=1}^{N} e^{z_j} \right) \right) = -\sum_{i=1}^{N} (p_i \times z_i) + \sum_{i=1}^{N} \left(p_i \times ln\left(\sum_{j=1}^{N} e^{z_j} \right) \right)$$

Then, for some integer $1 \leq x \leq N$ we have:

$$\frac{\partial E}{\partial z_x} = \frac{\partial}{\partial z_x} \left(-\sum_{i=1}^{N} (p_i \times z_i) \right) + \frac{\partial}{\partial z_x} \left(\sum_{i=1}^{N} \left(p_i \times ln\left(\sum_{j=1}^{N} e^{z_j} \right) \right) \right)$$

$$= \frac{\partial}{\partial z_x} \left(-\left(p_x \times z_x \right) \right) + \sum_{i=1}^{N} \left(p_i \times \frac{\partial}{\partial z_x} \left(ln\left(\sum_{j=1}^{N} e^{z_j} \right) \right) \right)$$

$$= -p_x + \sum_{i=1}^{N} \left(p_i \times \frac{\partial}{\partial z_x} \left(ln\left(\sum_{j=1}^{N} e^{z_j} \right) \right) \right)$$

Let $A = \sum_{j=1}^{N} e^{z_j}$, then we have $\frac{\partial A}{\partial z_x} = \frac{\partial}{\partial z_x} \left(\sum_{j=1}^{N} e^{z_j} \right) = \frac{\partial}{\partial z_x} e^{z_x} = e^{z_x}$.
By chain rule we have $\frac{\partial ln(A)}{\partial z_x} = \frac{1}{A} \times \frac{\partial A}{\partial z_x} = \frac{e^{z_x}}{\sum_{j=1}^{N} e^{z_j}}$.
So, we get:

$$\frac{\partial E}{\partial z_x} = -p_x + \sum_{i=1}^{N} \left(p_i \times \frac{e^{z_x}}{\sum_{j=1}^{N} e^{z_j}} \right) = -p_x + \frac{e^{z_x}}{\sum_{j=1}^{N} e^{z_j}} \times \sum_{i=1}^{N} (p_i)$$

$$= -p_x + \frac{e^{z_x}}{\sum_{j=1}^{N} e^{z_j}} = -p_x + q_x = q_x - p_x$$

Problem 5

Suppose we have a simple neural network shown as below:

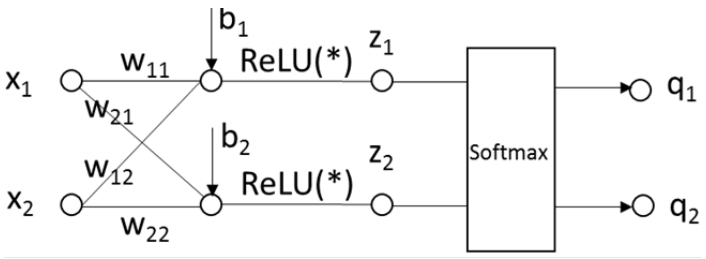

Figure 4: Neural Network For Problem 5

The initial parameters of this simple neural network model are:
$W = \begin{bmatrix} w_{1,1} & w_{1,2} \\ w_{2,1} & w_{2,2} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} =, B = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}.$

We use batch size = 1, learning rate $\alpha = 0.02$, objective function $E = -p_1 ln(q_1) - p_2 ln(q_2)$.
In the first training iteration our input sample (X, P) is: $X = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$, $P = \begin{bmatrix} p_1 \\ p_2 \end{bmatrix} = \begin{bmatrix} 0.3 \\ 0.7 \end{bmatrix}$.

If we use gradient decent formula

$$W^{'} = W - \alpha \times \frac{\partial E}{\partial W}, \quad B^{'} = B - \alpha \times \frac{\partial E}{\partial B}$$

during model training, please calculate the new parameters in our simple neural network $W^{'}$ and $B^{'}$ after the first iteration in training. [8 pts]

Problem 5 Solution

In the first training iteration we do two steps: forward propagation, and backward propagation.
Forward propagation:

$$Z = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = ReLu\left(W \times X + B\right) = ReLu\left(\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \times \begin{bmatrix} 1 \\ 1 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix}\right) = ReLu\left(\begin{bmatrix} 1 \\ -1 \end{bmatrix}\right) = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

$$Q = \begin{bmatrix} q_1 \\ q_2 \end{bmatrix} = Softmax\left(\begin{bmatrix} z_1 \\ z_2 \end{bmatrix}\right) = \begin{bmatrix} \frac{e^1}{e^0+e^1} \\ \frac{e^0}{e^0+e^1} \end{bmatrix} = \begin{bmatrix} \frac{e}{1+e} \\ \frac{1}{1+e} \end{bmatrix}$$

In fact, it is not necessary to calculate E for backward propagation, using what we have proved in problem 4, we know that $\frac{\partial E}{\partial z_x} = q_x - p_x$, for $1 \leq x \leq N$, which means:

$$\frac{\partial E}{\partial Z} = Q - P$$

Backward propagation:

$$\frac{\partial E}{\partial Z} = Q - P = \begin{bmatrix} \frac{e}{1+e} \\ \frac{1}{1+e} \end{bmatrix} - \begin{bmatrix} 0.3 \\ 0.7 \end{bmatrix} = \begin{bmatrix} \frac{e}{1+e} - 0.3 \\ \frac{1}{1+e} - 0.7 \end{bmatrix} = \begin{bmatrix} \frac{7e-3}{10(1+e)} \\ \frac{3-7e}{10(1+e)} \end{bmatrix}$$

Let

$$M = W \times X + B = \begin{bmatrix} m_1 \\ m_2 \end{bmatrix} = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

And

$$sgn(x) = \begin{cases} 0 & x < 0 \\ 1 & x > 0 \end{cases}$$

Then we have:

$$\frac{\partial E}{\partial W} = \frac{\partial E}{\partial Z} \cdot \frac{\partial Z}{\partial W} = \frac{\partial E}{\partial Z} \cdot \frac{\partial Z}{\partial M} \frac{\partial M}{\partial W} = \frac{\partial E}{\partial M} \frac{\partial M}{\partial W}$$

$$\frac{\partial E}{\partial M} = \begin{bmatrix} \frac{7e-3}{10(1+e)} \cdot sgn(m_1) \\ \frac{3-7e}{10(1+e)} \cdot sgn(m_2) \end{bmatrix} = \begin{bmatrix} \frac{7e-3}{10(1+e)} \cdot 1 \\ \frac{3-7e}{10(1+e)} \cdot 0 \end{bmatrix} = \begin{bmatrix} \frac{7e-3}{10(1+e)} \\ 0 \end{bmatrix}$$

$$\frac{\partial E}{\partial W} = \frac{\partial E}{\partial M} \frac{\partial M}{\partial W} = \frac{\partial E}{\partial M} X^T = \begin{bmatrix} \frac{7e-3}{10(1+e)} \\ 0 \end{bmatrix} \begin{bmatrix} 1 & 1 \end{bmatrix} = \begin{bmatrix} \frac{7e-3}{10(1+e)} & \frac{7e-3}{10(1+e)} \\ 0 & 0 \end{bmatrix}$$

$$W' = W - \alpha \times \frac{\partial E}{\partial W} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} - 0.02 \times \begin{bmatrix} \frac{7e-3}{10(1+e)} & \frac{7e-3}{10(1+e)} \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} \frac{503+493e}{500(1+e)} & \frac{3-7e}{500(1+e)} \\ 0 & -1 \end{bmatrix} \approx \begin{bmatrix} 0.9914 & -0.008621 \\ 0 & -1 \end{bmatrix}$$

$$\frac{\partial E}{\partial B} = \frac{\partial E}{\partial Z} \cdot \frac{\partial Z}{\partial B} = \frac{\partial E}{\partial Z} \cdot \frac{\partial Z}{\partial M} \cdot \frac{\partial M}{\partial B} = \frac{\partial E}{\partial M} \cdot \frac{\partial M}{\partial B} = \frac{\partial E}{\partial M} = \begin{bmatrix} \frac{7e-3}{10(1+e)} \\ 0 \end{bmatrix}$$

$$B' = B - \alpha \times \frac{\partial E}{\partial B} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} - 0.02 \times \begin{bmatrix} \frac{7e-3}{10(1+e)} \\ 0 \end{bmatrix} = \begin{bmatrix} \frac{3-7e}{500(1+e)} \\ 0 \end{bmatrix} \approx \begin{bmatrix} -0.008621 \\ 0 \end{bmatrix}$$

Problem 6

In recurrent neural network, define loss function:

$$L = \sum_{t=0}^{T} L_t$$

Differentiate L_t with respect to W:

$$\frac{\partial L_t}{\partial W^o} = \sum_{t=0}^{T} \frac{\partial L_t}{\partial y_t} \frac{\partial y_t}{\partial W^o} \quad \text{.....(1)}$$

$$\frac{\partial L_t}{\partial W^i} = \sum_{t=0}^{T} \sum_{k=0}^{t} \frac{\partial L_t}{\partial y_t} \frac{\partial y_t}{\partial h_t} \left(\prod_{j=k+1}^{t} \frac{\partial h_j}{\partial h_{j-1}} \right) \frac{\partial h_k}{\partial W^i} \quad \text{.....(2)}$$

$$\frac{\partial L_t}{\partial W^h} = \sum_{t=0}^{T} \sum_{k=0}^{t} \frac{\partial L_t}{\partial y_t} \frac{\partial y_t}{\partial h_t} \left(\prod_{j=k+1}^{t} \frac{\partial h_j}{\partial h_{j-1}} \right) \frac{\partial h_k}{\partial W^h} \quad \text{.....(3)}$$

Define:

$$h_t = \sigma(W^i x_t + W^h h_{t-1})$$

where σ denotes sigmoid activation function.

Based on equation (2) or (3), explain why there will be gradient exploding and gradient vanishing. [5pts]

Problem 6 Solution

Based on the formula below:

$$h_t = \sigma(W^i x_t + w_h h_{t-1})$$

$$\frac{\partial h_t}{\partial_{t-1}} = diag\left(\sigma'(W^i x_t + w_h h_{t-1})\right) w_h$$

Then we have:

$$\prod_{j=k+1}^{t} \frac{\partial h_t}{\partial_{t-1}} = (w_h)^{t-k} \prod_{i=k+1}^{t} diag\left(\sigma'(W^i x_j + w_h h_{j-1})\right)$$

Then consider the eigenvalue of w_h, if the largest eigenvalue less than 1, then $||\frac{\partial L_t}{\partial_{W^i}}||$ and $||\frac{\partial L_t}{\partial_{W^h}}||$ shrink exponentially, which is gradient vanishing.

On the contrary, if the largest eigenvalue greater than 1, then $||\frac{\partial L_t}{\partial_{W^i}}||$ and $||\frac{\partial L_t}{\partial_{W^h}}||$ increase exponentially, which is gradient exploding.