

# RNN Vanishing and Exploding Gradient

Lecture 23-23



# Vanishing Gradients Problem

**The brown and black dog, which was playing with the cat, was a german shepherd.**

**(x<sub>2</sub>)**

**(x<sub>4</sub>)**

**(x<sub>5</sub>)**

**(x<sub>14</sub>)**

**(x<sub>15</sub>)**



# Vanishing Gradients Problem

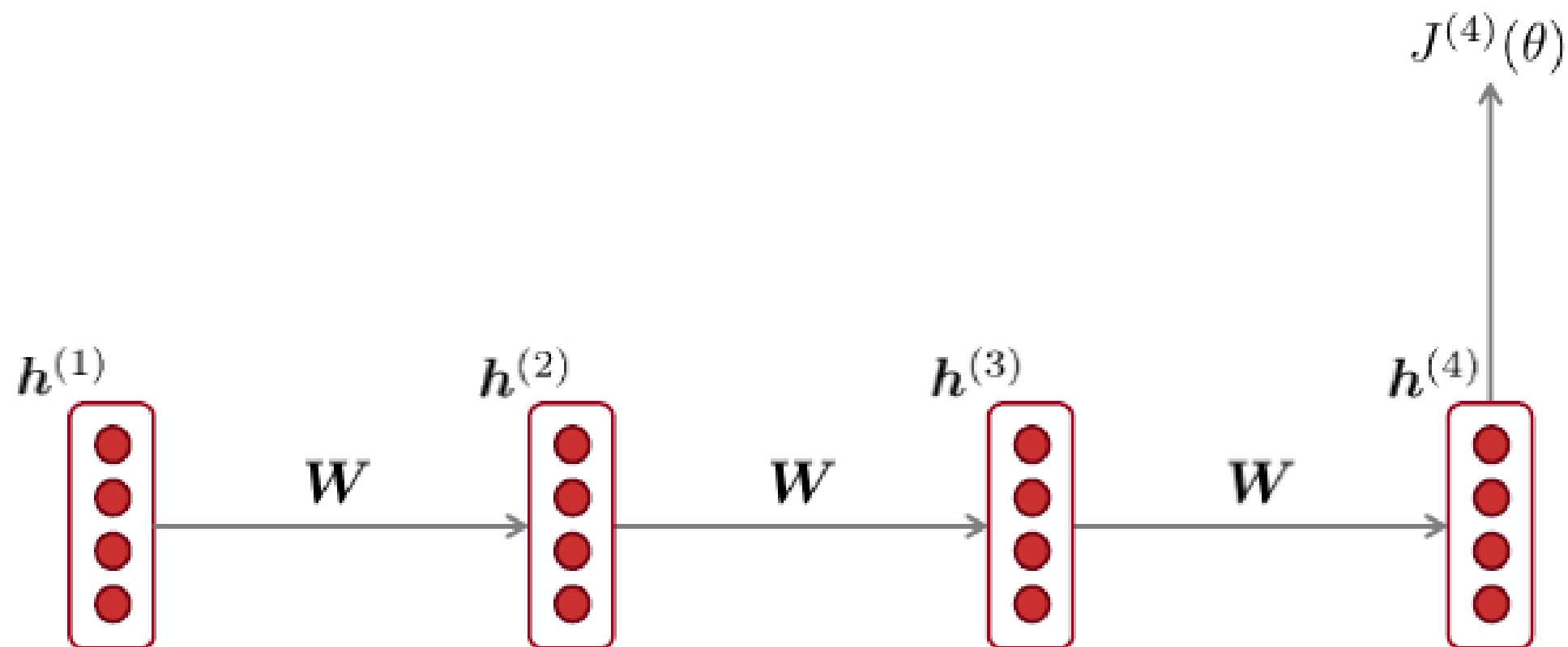
backpropagation error of the word “shepherd” (x15) back to “brown” (x2)

$$\frac{\partial L_{15}}{\partial \hat{y}_{15}} \frac{\partial \hat{y}_{15}}{\partial z_{15}} \frac{\partial z_{15}}{\partial h_{15}} \frac{\partial h_{15}}{\partial h_2} \frac{\partial h_2}{\partial W_x}$$

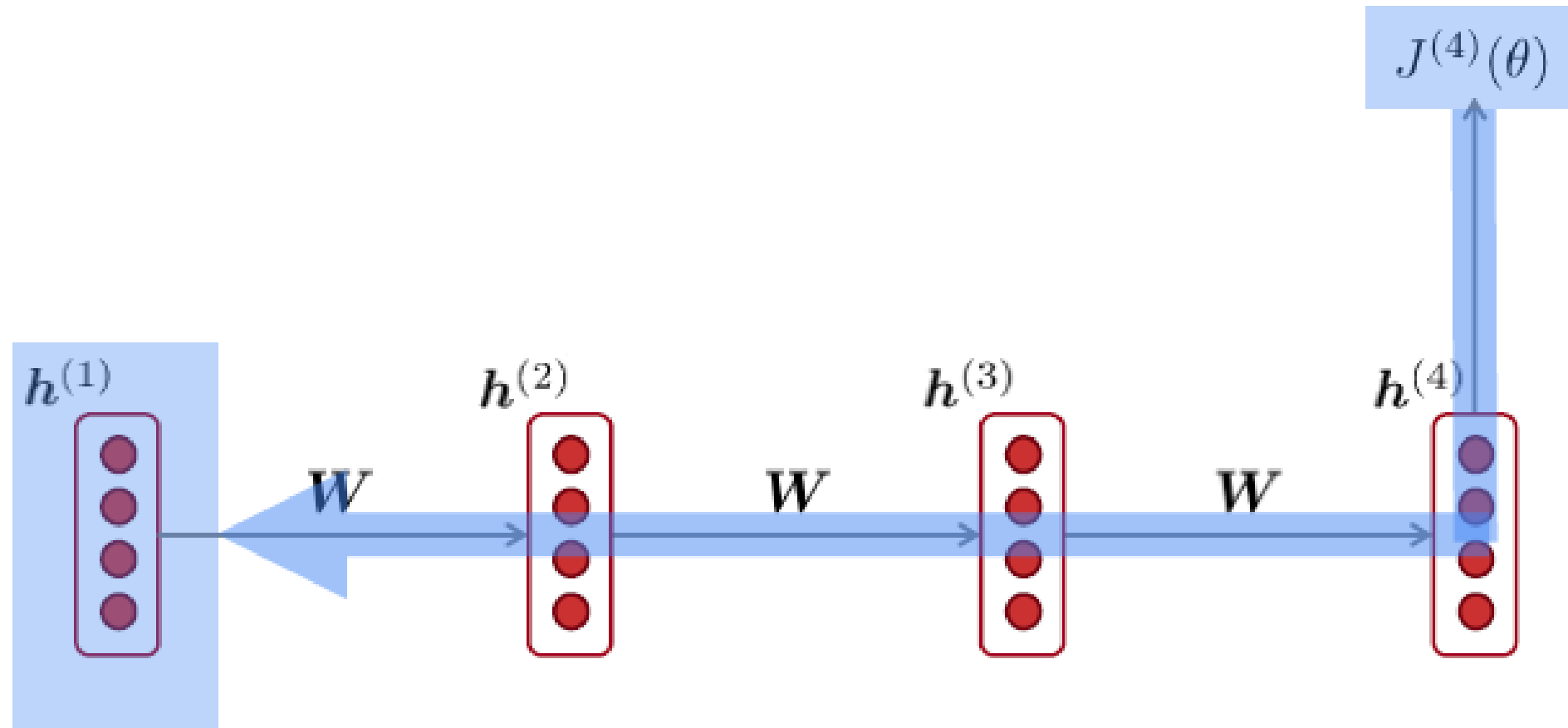
$$\frac{\partial h_{15}}{\partial h_2} = \frac{\partial h_{15}}{\partial h_{14}} \frac{\partial h_{14}}{\partial h_{13}} \cdots \frac{\partial h_2}{\partial h_1}$$



# Vanishing gradient intuition



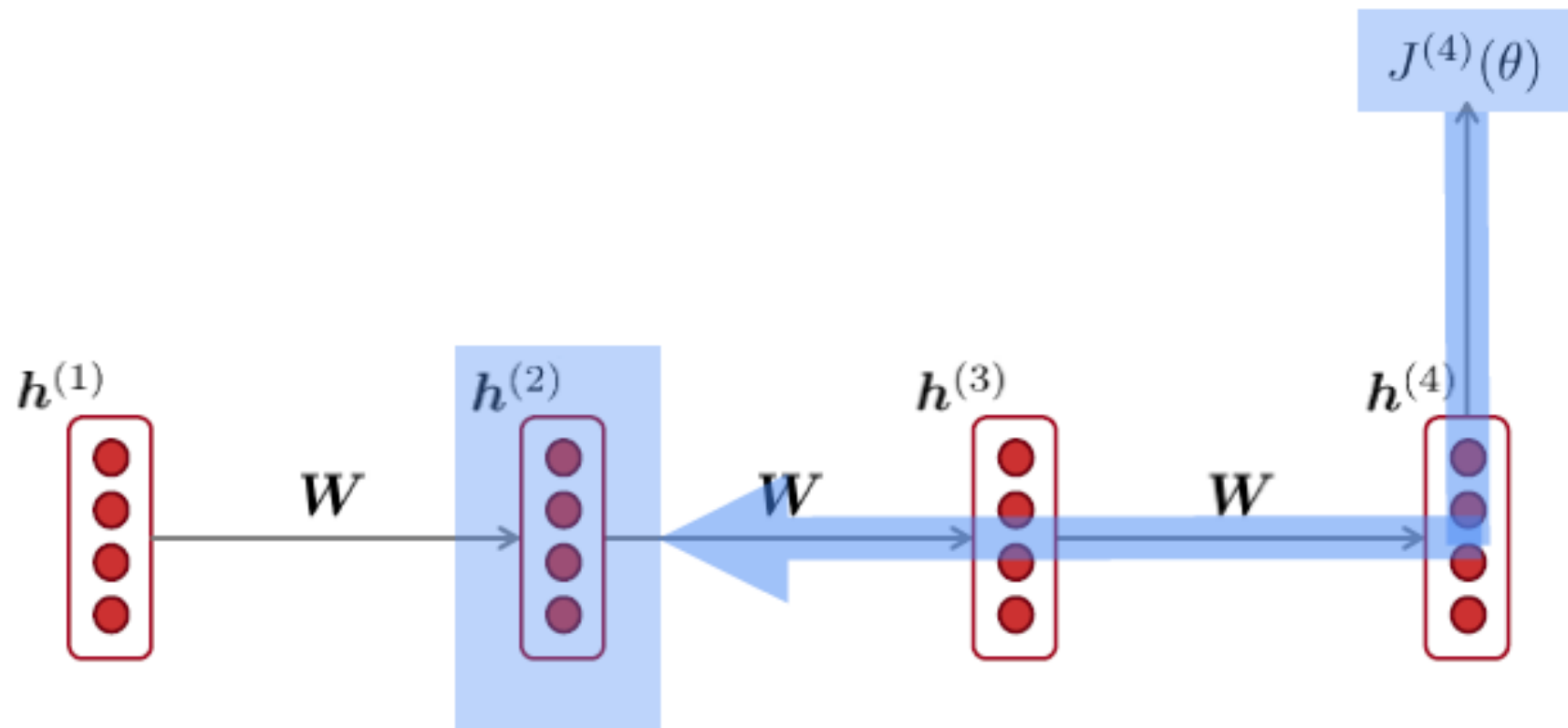
# Vanishing gradient intuition



$$\frac{\partial J^{(4)}}{\partial h^{(1)}} = ?$$



# Vanishing gradient intuition

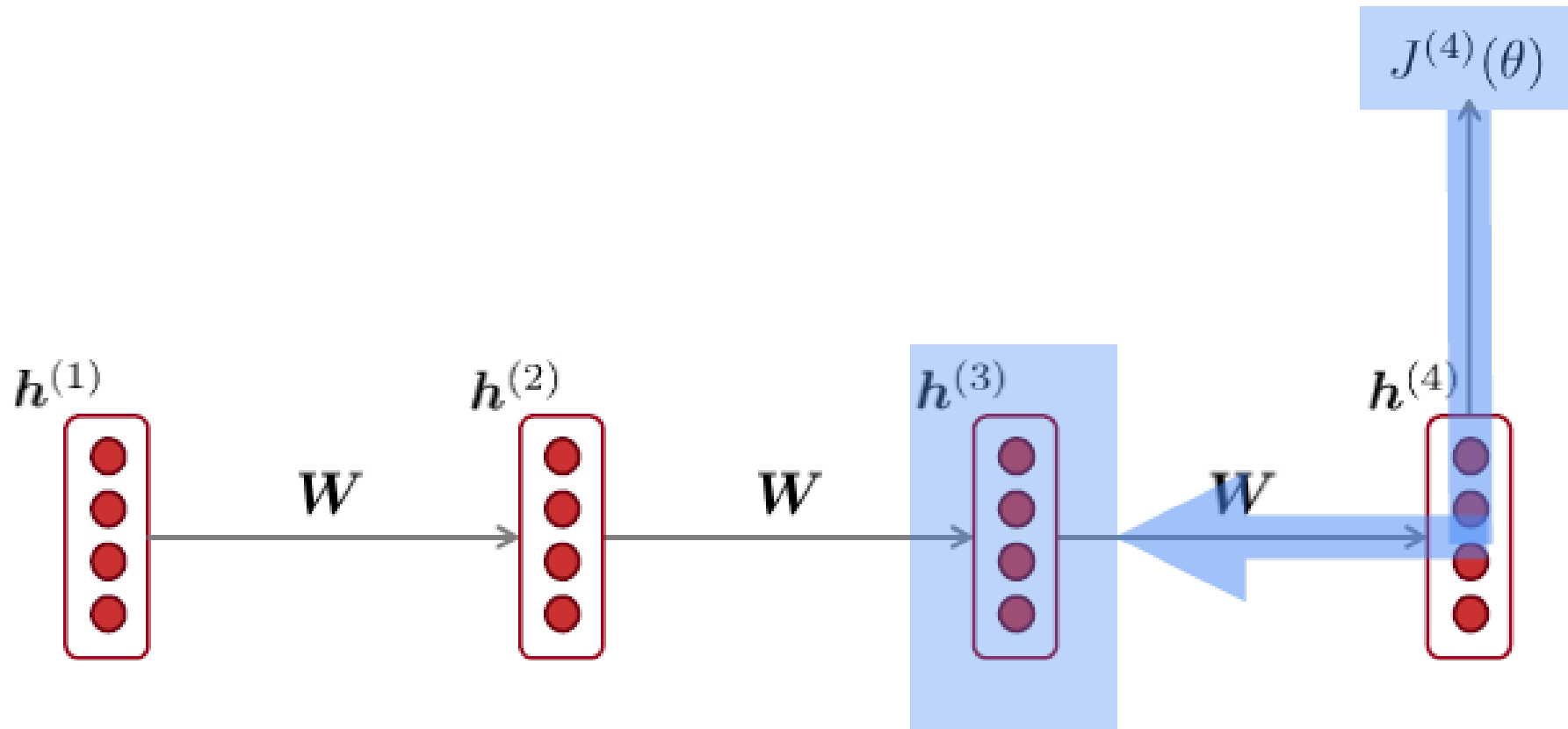


$$\frac{\partial J^{(4)}}{\partial h^{(1)}} = \frac{\partial h^{(2)}}{\partial h^{(1)}} \times \frac{\partial J^{(4)}}{\partial h^{(2)}}$$

chain rule!



# Vanishing gradient intuition



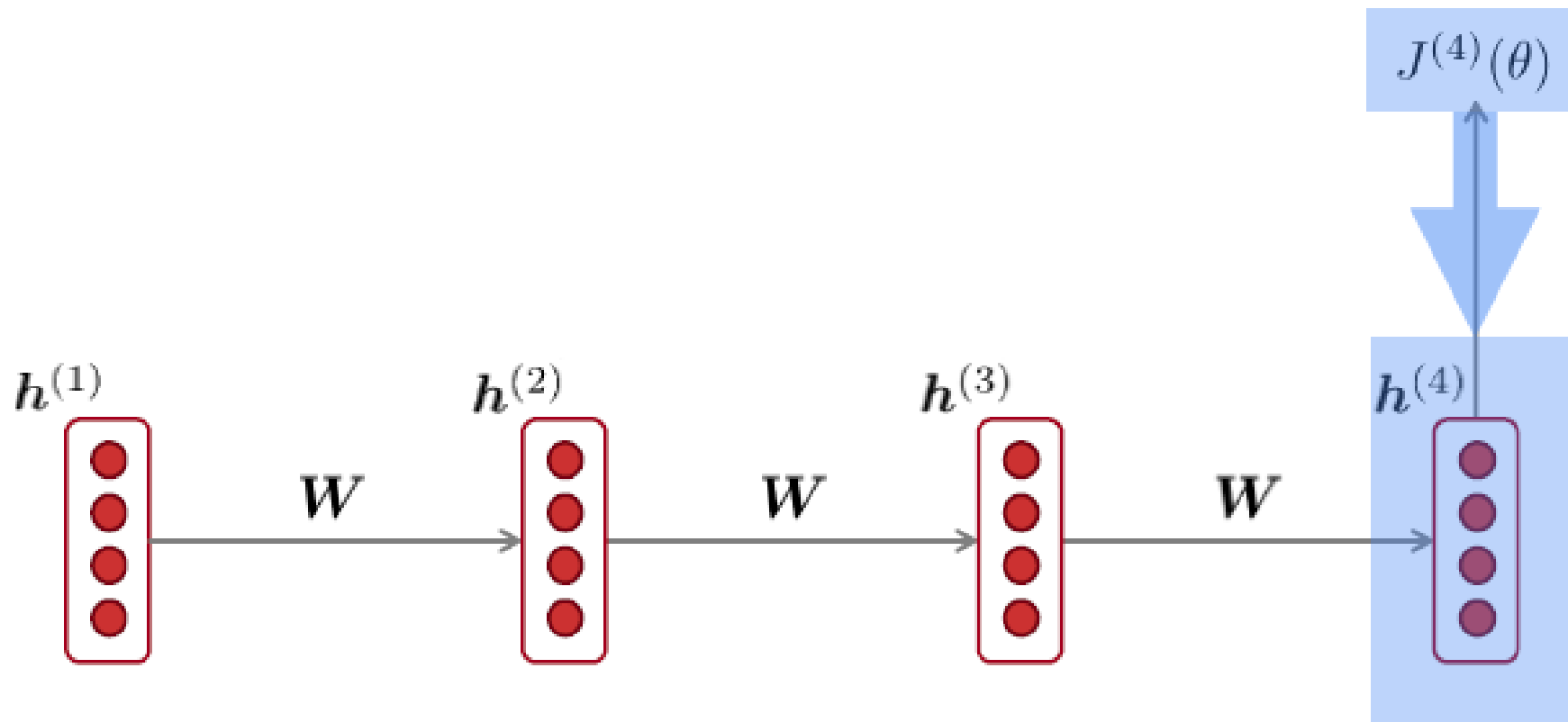
$$\frac{\partial J^{(4)}}{\partial h^{(1)}} = \frac{\partial h^{(2)}}{\partial h^{(1)}} \times \dots$$

$$\frac{\partial h^{(3)}}{\partial h^{(2)}} \times \frac{\partial J^{(4)}}{\partial h^{(3)}}$$

chain rule!



# Vanishing gradient intuition



$$\frac{\partial J^{(4)}}{\partial h^{(1)}} = \frac{\partial h^{(2)}}{\partial h^{(1)}} \times \dots$$

$$\frac{\partial h^{(3)}}{\partial h^{(2)}} \times$$

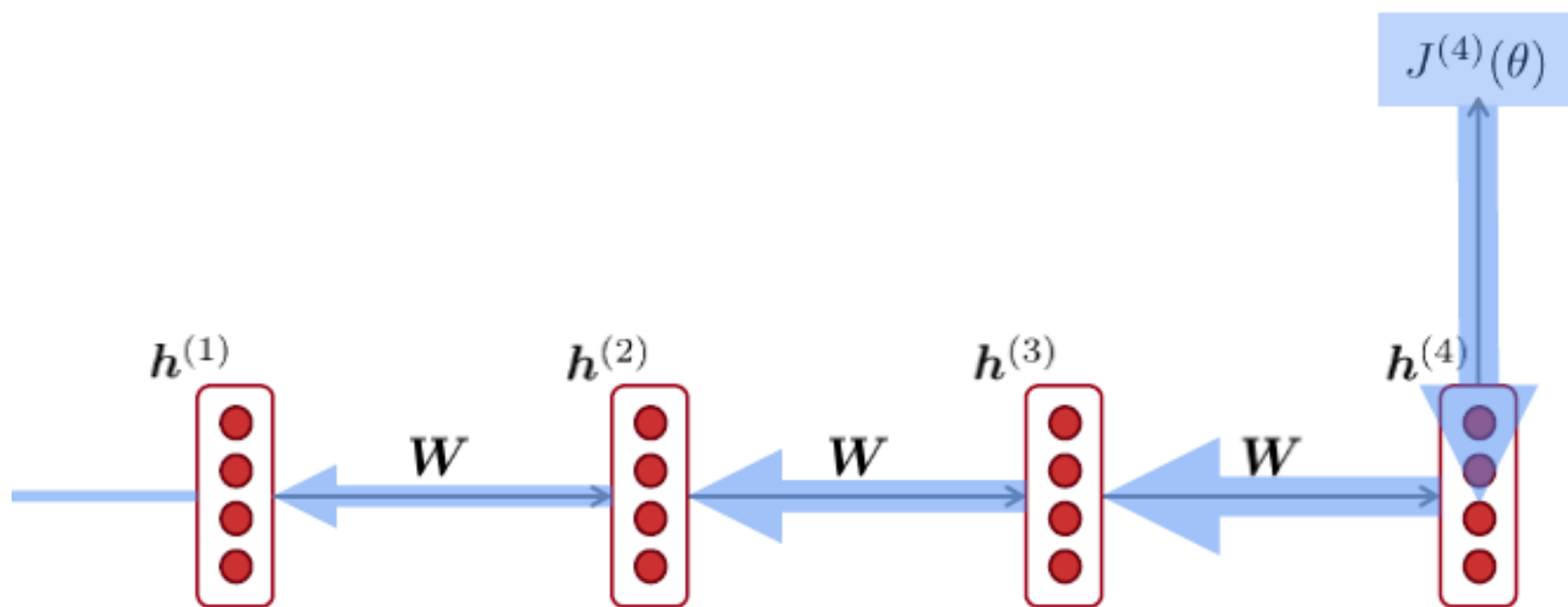
$$\frac{\partial h^{(4)}}{\partial h^{(3)}} \times \frac{\partial J^{(4)}}{\partial h^{(4)}}$$

chain rule!





# Vanishing gradient intuition



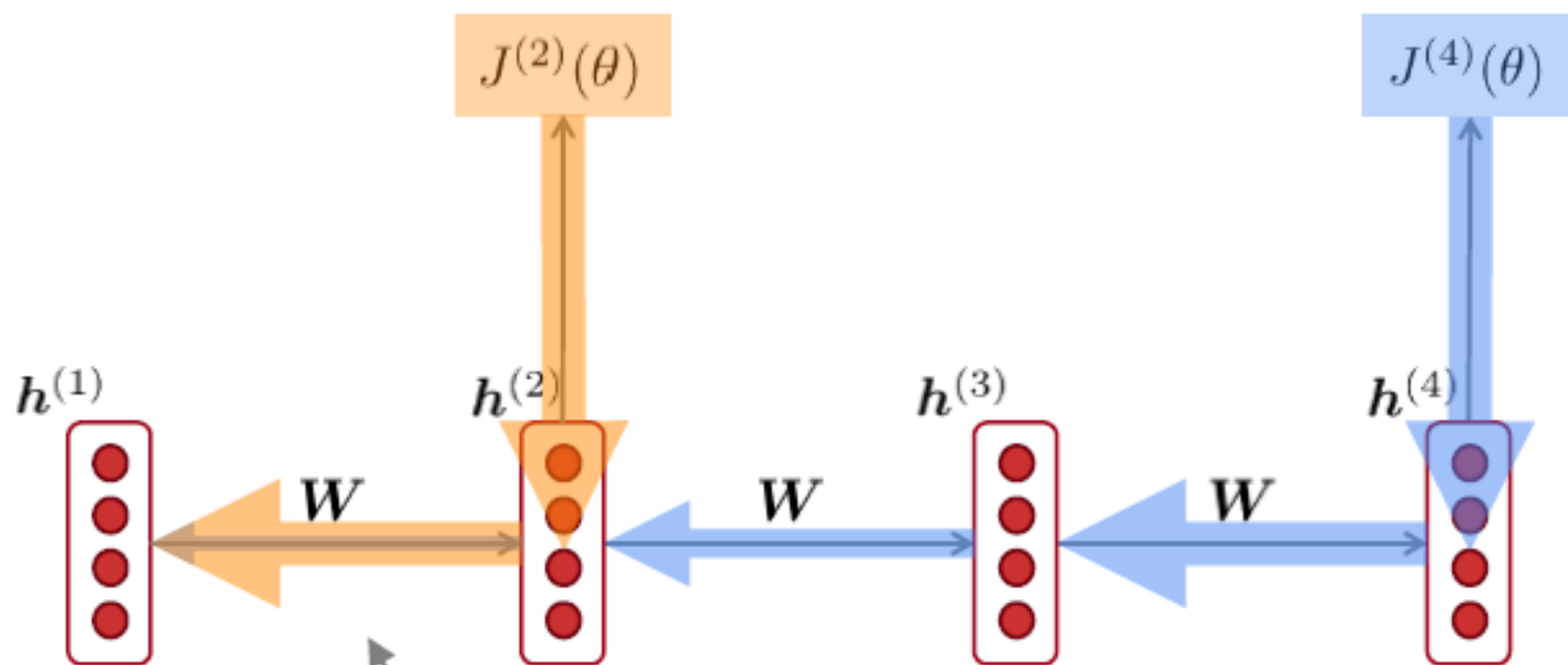
$$\frac{\partial J^{(4)}}{\partial h^{(1)}} = \left[ \frac{\partial h^{(2)}}{\partial h^{(1)}} \right] \times \left[ \frac{\partial h^{(3)}}{\partial h^{(2)}} \right] \times \left[ \frac{\partial h^{(4)}}{\partial h^{(3)}} \right] \times \frac{\partial J^{(4)}}{\partial h^{(4)}}$$

What happens if these are small?

Vanishing gradient problem:  
When these are small, the gradient signal gets smaller and smaller as it backpropagates further



# Why is vanishing gradient a problem?



Gradient signal from faraway is lost because it's much smaller than gradient signal from close-by.

So model weights are only updated only with respect to near effects, not long-term effects.



# Why is vanishing gradient a problem?

- Another explanation: Gradient can be viewed as a measure of *the effect of the past on the future*
- If the gradient becomes vanishingly small over longer distances (step  $t$  to step  $t+n$ ), then we can't tell whether:
  1. There's *no dependency* between step  $t$  and  $t+n$  in the data
  2. We have *wrong parameters* to capture the true dependency between  $t$  and  $t+n$

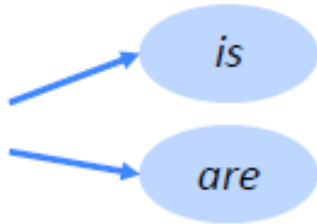




## Effect of vanishing gradient on RNN-LM

- **LM task:** *When she tried to print her tickets, she found that the printer was out of toner. She went to the stationery store to buy more toner. It was very overpriced. After installing the toner into the printer, she finally printed her \_\_\_\_\_*
- To learn from this training example, the RNN-LM needs to **model the dependency** between “tickets” on the 7<sup>th</sup> step and the target word “tickets” at the end.
- But if gradient is small, the model **can't learn this dependency**
  - So the model is **unable to predict similar long-distance dependencies** at test time



## Effect of vanishing gradient on RNN-LM

- **LM task:** *The writer of the books \_\_\_\_* 
  - is*
  - are*
- **Correct answer:** *The writer of the books is planning a sequel*
- **Syntactic recency:** *The writer of the books is* (correct) 
- **Sequential recency:** *The writer of the books are* (incorrect) 
- Due to vanishing gradient, RNN-LMs are better at learning from **sequential recency** than **syntactic recency**, so they make this type of error more often than we'd like [Linzen et al 2016]



## Why is exploding gradient a problem?

- If the gradient becomes too big, then the SGD update step becomes too big:

$$\theta^{new} = \theta^{old} - \overset{\text{learning rate}}{\alpha} \underbrace{\nabla_{\theta} J(\theta)}_{\text{gradient}}$$

- This can cause **bad updates**: we take too large a step and reach a bad parameter configuration (with large loss)
- In the worst case, this will result in **Inf** or **NaN** in your network (then you have to restart training from an earlier checkpoint)



# Gradient clipping: solution for exploding gradient

- Gradient clipping: if the norm of the gradient is greater than some threshold, scale it down before applying SGD update

---

**Algorithm 1** Pseudo-code for norm clipping

---

```
 $\hat{\mathbf{g}} \leftarrow \frac{\partial \mathcal{E}}{\partial \theta}$   
if  $\|\hat{\mathbf{g}}\| \geq threshold$  then  
     $\hat{\mathbf{g}} \leftarrow \frac{threshold}{\|\hat{\mathbf{g}}\|} \hat{\mathbf{g}}$   
end if
```

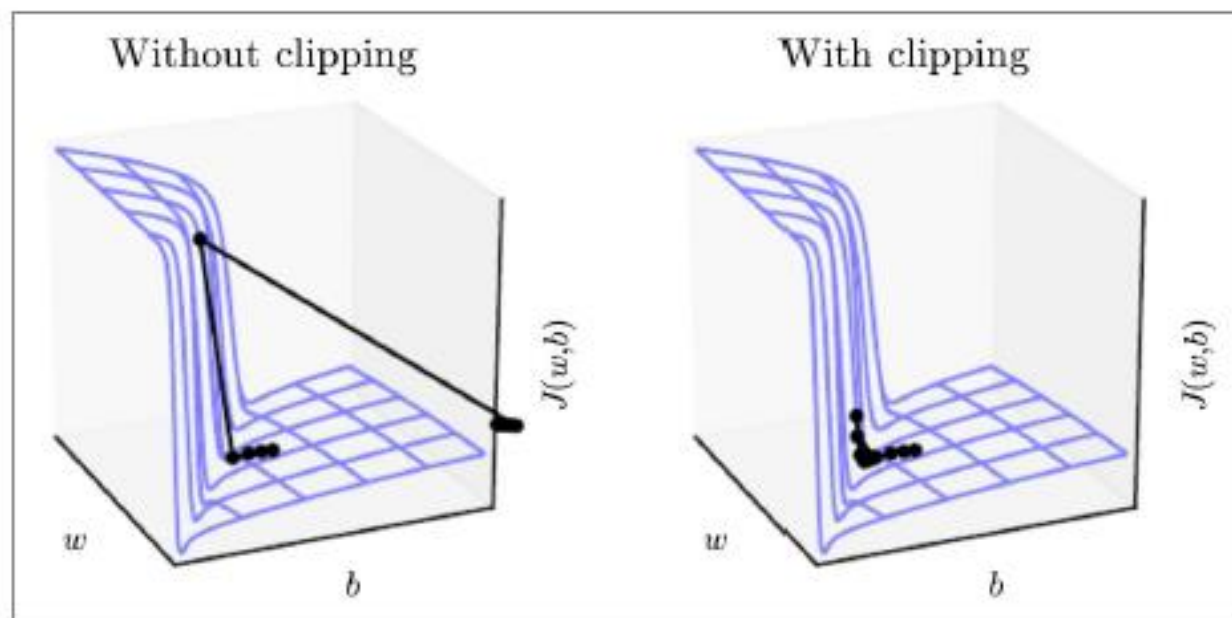
---

- Intuition: take a step in the same direction, but a smaller step





# Gradient clipping: solution for exploding gradient



- This shows the loss surface of a simple RNN (hidden state is a scalar not a vector)
- The “cliff” is dangerous because it has steep gradient
- On the left, gradient descent takes two very big steps due to steep gradient, resulting in climbing the cliff then shooting off to the right (both bad updates)
- On the right, gradient clipping reduces the size of those steps, so effect is less drastic





# How to fix vanishing gradient problem?

- The main problem is that *it's too difficult for the RNN to learn to preserve information over many timesteps.*
- In a vanilla RNN, the hidden state is constantly being rewritten

$$\mathbf{h}^{(t)} = \sigma \left( \mathbf{W}_h \mathbf{h}^{(t-1)} + \mathbf{W}_x \mathbf{x}^{(t)} + \mathbf{b} \right)$$

- How about a RNN with separate memory?



# Vanishing Gradient Problem solution

- GRU
- LSTM

