Recurrent Neural Networks

CS 6956: Deep Learning for NLP

Overview

1. Modeling sequences
2. Recurrent neural networks: An abstraction
3. Usage patterns for RNNs
4. BiDirectional RNNs
5. A concrete example: The Elman RNN
6. The vanishing gradient problem
7. Long short-term memory units

Overview

1. Modeling sequences
2. Recurrent neural networks: An abstraction
3. Usage patterns for RNNs
4. BiDirectional RNNs
5. A concrete example: The Elman RNN
6. *The vanishing gradient problem*
7. Gating and Long short-term memory units

A simple RNN

1. How to generate the current state using the previous state and the current input?

   Next state $\mathbf{s}_t = g(\mathbf{s}_{t-1}\mathbf{W}_S + \mathbf{x}_t\mathbf{W}_I + \mathbf{b})$

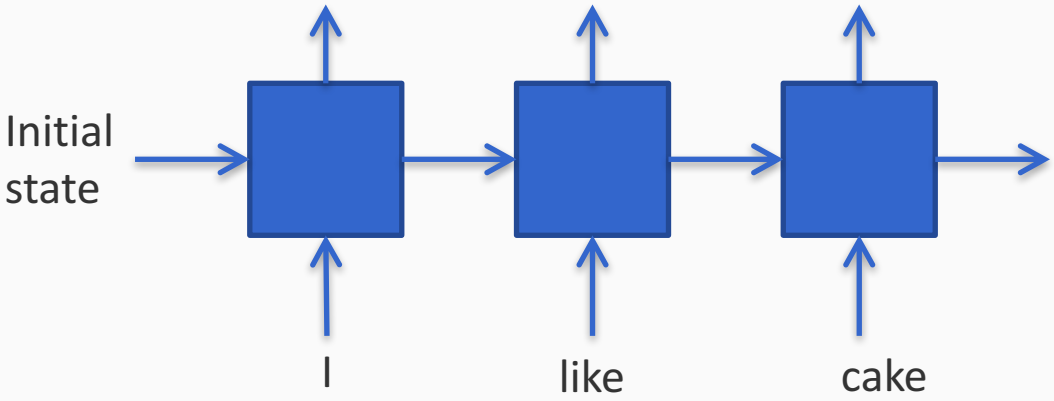
2. How to generate the current output using the current state?

   The output is the state. That is, $\boldsymbol{y}_t = \mathbf{s}_t$

How do we train a recurrent network?

We need to specify a problem first. Let's take an example.

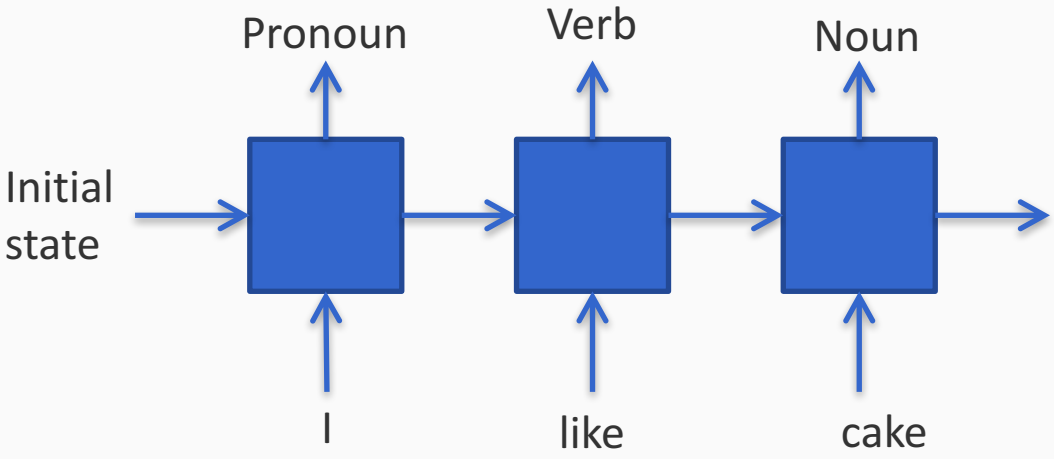
- Inputs are sequences (say, of words)



How do we train a recurrent network?

We need to specify a problem first. Let's take an example.

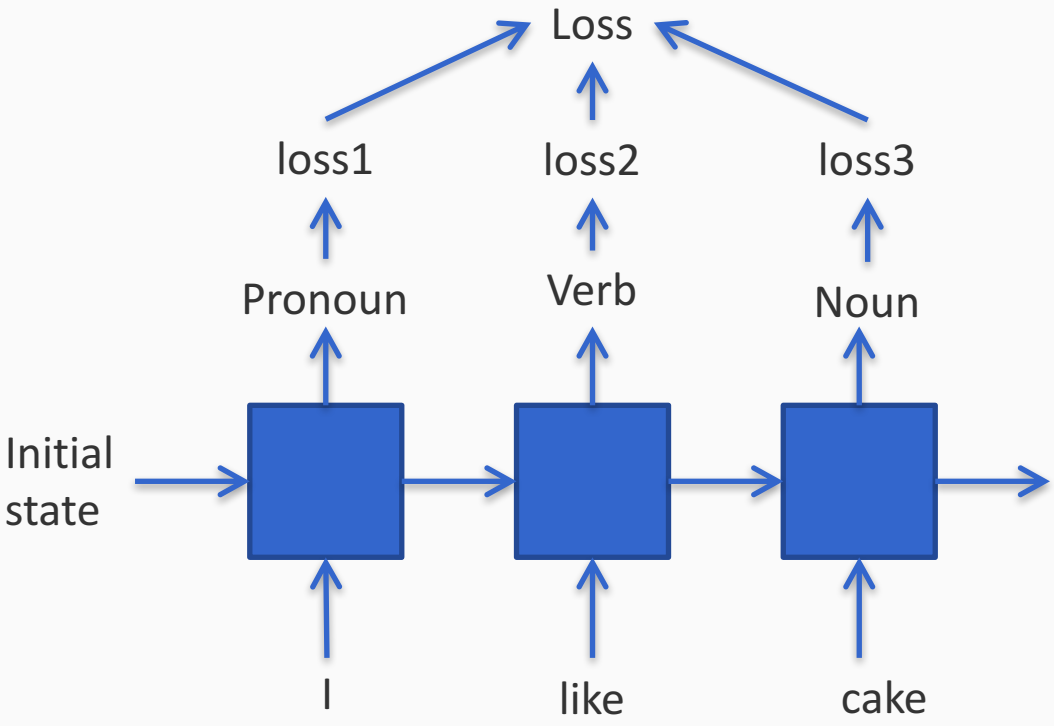
- Inputs are sequences (say, of words)
- The outputs are labels associated with each word



How do we train a recurrent network?

We need to specify a problem first. Let's take an example.

- Inputs are sequences (say, of words)
- The outputs are labels associated with each word
- Losses for each word are added up



Gradients to the rescue

- We have a computation graph
- Use back propagation to compute gradients of the loss with respect to the parameters ($\mathbf{W}_S, \mathbf{W}_I, \mathbf{b}$)
  - Sometimes called *Backpropagation Through Time (BPTT)*
- Update gradients using SGD or a variant
  - Adam, for example

A simple RNN

1. How to generate the current state using the previous state and the current input?

   Next state $\mathbf{s}_t = g(\mathbf{s}_{t-1}\mathbf{W}_S + \mathbf{x}_t\mathbf{W}_I + \mathbf{b})$

2. How to generate the current output using the current state?

   The output is the state. That is, $\boldsymbol{y}_t = \mathbf{s}_t$

Does this work? Let's see a simple example

To avoid complicating the notation more than necessary, suppose

1. The inputs, states and outputs are all scalars
2. The loss at each step is a function f of the state at that step

Does this work? Let's see a simple example

To avoid complicating the notation more than necessary, suppose
1. The inputs, states and outputs are all scalars
2. The loss at each step is a function f of the state at that step

First input: x_1
Transform: $t_1 = s_0 W_S + x_1 W_I + b$
State: $s_1 = g(t_1)$
Loss: $l_1 = f(s_1)$

Does this work? Let's see a simple example

To avoid complicating the notation more than necessary, suppose
1. The inputs, states and outputs are all scalars
2. The loss at each step is a function f of the state at that step

First input: x_1
Transform: $t_1 = s_0 W_S + x_1 W_I + b$
State: $s_1 = g(t_1)$
Loss: $l_1 = f(s_1)$

Let's compute the derivative of the loss with respect to the parameter W_I

Does this work? Let's see a simple example

To avoid complicating the notation more than necessary, suppose
1. The inputs, states and outputs are all scalars
2. The loss at each step is a function f of the state at that step

First input: x_1
Transform: $t_1 = s_0 W_S + x_1 W_I + b$
State: $s_1 = g(t_1)$
Loss: $l_1 = f(s_1)$

Let's compute the derivative of the loss with respect to the parameter W_I

$$\frac{\partial l_1}{\partial W_I} =$$

Follows the chain rule

Does this work? Let's see a simple example

To avoid complicating the notation more than necessary, suppose
1. The inputs, states and outputs are all scalars
2. The loss at each step is a function f of the state at that step

First input: x_1
Transform: $t_1 = s_0 W_S + x_1 W_I + b$
State: $s_1 = g(t_1)$
Loss: $l_1 = f(s_1)$

Let's compute the derivative of the loss with respect to the parameter W_I

$$\frac{\partial l_1}{\partial W_I} = \frac{\partial l_1}{\partial s_1} \cdot$$

Follows the chain rule

Does this work? Let's see a simple example

To avoid complicating the notation more than necessary, suppose
1. The inputs, states and outputs are all scalars
2. The loss at each step is a function f of the state at that step

First input: x_1
Transform: $t_1 = s_0 W_S + x_1 W_I + b$
State: $s_1 = g(t_1)$
Loss: $l_1 = f(s_1)$

Let's compute the derivative of the loss with respect to the parameter W_I

$$\frac{\partial l_1}{\partial W_I} = \frac{\partial l_1}{\partial s_1} \cdot \frac{\partial s_1}{\partial t_1} \cdot$$

Follows the chain rule

Does this work? Let's see a simple example

To avoid complicating the notation more than necessary, suppose
1. The inputs, states and outputs are all scalars
2. The loss at each step is a function f of the state at that step

First input: x_1
Transform: $t_1 = s_0 W_S + x_1 W_I + b$
State: $s_1 = g(t_1)$
Loss: $l_1 = f(s_1)$

Let's compute the derivative of the loss with respect to the parameter W_I

$$\frac{\partial l_1}{\partial W_I} = \frac{\partial l_1}{\partial s_1} \cdot \frac{\partial s_1}{\partial t_1} \cdot \frac{\partial t_1}{\partial W_I}$$

Follows the chain rule

Does this work? Let's see a simple example

To avoid complicating the notation more than necessary, suppose
1. The inputs, states and outputs are all scalars
2. The loss at each step is a function f of the state at that step

First input: x_1
Transform: $t_1 = s_0 W_S + x_1 W_I + b$
State: $s_1 = g(t_1)$
Loss: $l_1 = f(s_1)$

Let's compute the derivative of the loss with respect to the parameter W_I

$$\frac{\partial l_1}{\partial W_I} = \frac{\partial l_1}{\partial s_1} \cdot \frac{\partial s_1}{\partial t_1} \cdot \frac{\partial t_1}{\partial W_I}$$

Let us examine the non-linearity in this system due to the activation function

Does this work? Let's see a simple example

To avoid complicating the notation more than necessary, suppose
1. The inputs, states and outputs are all scalars
2. The loss at each step is a function f of the state at that step

First input: x_1
Transform: $t_1 = s_0 W_S + x_1 W_I + b$
State: $s_1 = g(t_1)$
Loss: $l_1 = f(s_1)$

Let's compute the derivative of the loss with respect to the parameter W_I

$$\frac{\partial l_1}{\partial W_I} = \frac{\partial l_1}{\partial s_1} \cdot \frac{\partial s_1}{\partial t_1} \cdot \frac{\partial t_1}{\partial W_I}$$

Suppose $g(z) = \tanh(z)$

Does this work? Let's see a simple example

To avoid complicating the notation more than necessary, suppose
1. The inputs, states and outputs are all scalars
2. The loss at each step is a function f of the state at that step

First input: x_1
Transform: $t_1 = s_0 W_S + x_1 W_I + b$
State: $s_1 = g(t_1)$
Loss: $l_1 = f(s_1)$

Let's compute the derivative of the loss with respect to the parameter W_I

$$\frac{\partial l_1}{\partial W_I} = \frac{\partial l_1}{\partial s_1} \cdot \frac{\partial s_1}{\partial t_1} \cdot \frac{\partial t_1}{\partial W_I}$$

Suppose $g(z) = \tanh(z)$

Then $\frac{dg}{dz} = 1 - \tanh^2(z)$

Does this work? Let's see a simple example

To avoid complicating the notation more than necessary, suppose
1. The inputs, states and outputs are all scalars
2. The loss at each step is a function f of the state at that step

First input: x_1
Transform: $t_1 = s_0 W_S + x_1 W_I + b$
State: $s_1 = g(t_1)$
Loss: $l_1 = f(s_1)$

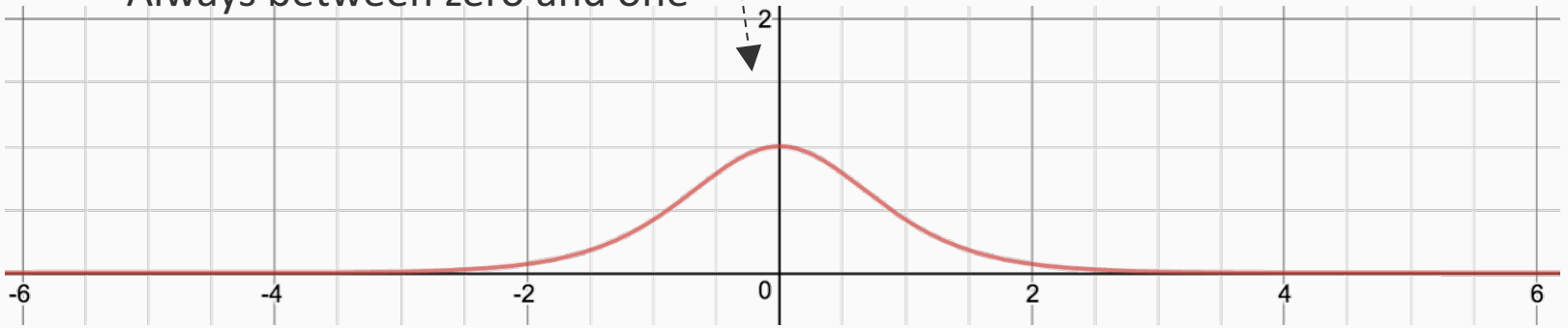
Let's compute the derivative of the loss with respect to the parameter W_I

$$\frac{\partial l_1}{\partial W_I} = \frac{\partial l_1}{\partial s_1} \cdot \frac{\partial s_1}{\partial t_1} \cdot \frac{\partial t_1}{\partial W_I}$$

Suppose $g(z) = \tanh(z)$

Then $\frac{dg}{dz} = 1 - \tanh^2(z)$

Always between zero and one



Does this work? Let's see a simple example

To avoid complicating the notation more than necessary, suppose
1. The inputs, states and outputs are all scalars
2. The loss at each step is a function f of the state at that step

First input: x_1
Transform: $t_1 = s_0 W_S + x_1 W_I + b$
State: $s_1 = g(t_1)$
Loss: $l_1 = f(s_1)$

Let's compute the derivative of the loss with respect to the parameter W_I

$$\frac{\partial l_1}{\partial W_I} = \frac{\partial l_1}{\partial s_1} \cdot \frac{\partial s_1}{\partial t_1} \cdot \frac{\partial t_1}{\partial W_I}$$

Suppose $g(z) = \tanh(z)$

Then $\frac{dg}{dz} = 1 - \tanh^2(z)$

That is $\frac{\partial s_1}{\partial t_1} = 1 - \tanh^2 t_1$

Does this work? Let's see a simple example

To avoid complicating the notation more than necessary, suppose
1. The inputs, states and outputs are all scalars
2. The loss at each step is a function f of the state at that step

First input: x_1
Transform: $t_1 = s_0 W_S + x_1 W_I + b$
State: $s_1 = g(t_1)$
Loss: $l_1 = f(s_1)$

Let's compute the derivative of the loss with respect to the parameter W_I

$$\frac{\partial l_1}{\partial W_I} = \frac{\partial l_1}{\partial s_1} \cdot \frac{\partial s_1}{\partial t_1} \cdot \frac{\partial t_1}{\partial W_I}$$

Suppose $g(z) = \tanh(z)$

Then $\frac{dg}{dz} = 1 - \tanh^2(z)$

That is $\frac{\partial s_1}{\partial t_1} = 1 - \tanh^2 t_1$

A number between zero and one.

Does this work? Let's see a simple example

To avoid complicating the notation more than necessary, suppose
1. The inputs, states and outputs are all scalars
2. The loss at each step is a function f of the state at that step

First input: x_1
Transform: $t_1 = s_0 W_S + x_1 W_I + b$
State: $s_1 = g(t_1)$
Loss: $l_1 = f(s_1)$

Let's see what happens with another input

Does this work? Let's see a simple example

To avoid complicating the notation more than necessary, suppose
1. The inputs, states and outputs are all scalars
2. The loss at each step is a function f of the state at that step

First input: x_1
Transform: $t_1 = s_0 W_S + x_1 W_I + b$
State: $s_1 = g(t_1)$
Loss: $l_1 = f(s_1)$

Second input: x_2
Transform: $t_2 = s_1 W_S + x_2 W_I + b$
State: $s_2 = g(t_2)$
Loss: $l_2 = f(s_2)$

Does this work? Let's see a simple example

To avoid complicating the notation more than necessary, suppose
1. The inputs, states and outputs are all scalars
2. The loss at each step is a function f of the state at that step

First input: x_1
Transform: $t_1 = s_0 W_S + x_1 W_I + b$
State: $s_1 = g(t_1)$
Loss: $l_1 = f(s_1)$

Second input: x_2
Transform: $t_2 = s_1 W_S + x_2 W_I + b$
State: $s_2 = g(t_2)$
Loss: $l_2 = f(s_2)$

Let's compute the derivative of the loss with respect to the parameter W_I

Once again, the chain rule

Does this work? Let's see a simple example

To avoid complicating the notation more than necessary, suppose
1. The inputs, states and outputs are all scalars
2. The loss at each step is a function f of the state at that step

First input: x_1
Transform: $t_1 = s_0 W_S + x_1 W_I + b$
State: $s_1 = g(t_1)$
Loss: $l_1 = f(s_1)$

Second input: x_2
Transform: $t_2 = s_1 W_S + x_2 W_I + b$
State: $s_2 = g(t_2)$
Loss: $l_2 = f(s_2)$

Let's compute the derivative of the loss with respect to the parameter W_I

Once again, the chain rule

$$\frac{\partial l_2}{\partial W_I} =$$

Does this work? Let's see a simple example

To avoid complicating the notation more than necessary, suppose
1. The inputs, states and outputs are all scalars
2. The loss at each step is a function f of the state at that step

First input: x_1
Transform: $t_1 = s_0 W_S + x_1 W_I + b$
State: $s_1 = g(t_1)$
Loss: $l_1 = f(s_1)$

Second input: x_2
Transform: $t_2 = s_1 W_S + x_2 W_I + b$
State: $s_2 = g(t_2)$
Loss: $l_2 = f(s_2)$

Let's compute the derivative of the loss with respect to the parameter W_I

Once again, the chain rule

$$\frac{\partial l_2}{\partial W_I} = \frac{\partial l_2}{\partial s_2} \cdot$$

Does this work? Let's see a simple example

To avoid complicating the notation more than necessary, suppose
1. The inputs, states and outputs are all scalars
2. The loss at each step is a function f of the state at that step

First input: x_1
Transform: $t_1 = s_0 W_S + x_1 W_I + b$
State: $s_1 = g(t_1)$
Loss: $l_1 = f(s_1)$

Second input: x_2
Transform: $t_2 = s_1 W_S + x_2 W_I + b$
State: $s_2 = g(t_2)$
Loss: $l_2 = f(s_2)$

Let's compute the derivative of the loss with respect to the parameter W_I

Once again, the chain rule

$$\frac{\partial l_2}{\partial W_I} = \frac{\partial l_2}{\partial s_2} \cdot \frac{\partial s_2}{\partial t_2} \cdot$$

Does this work? Let's see a simple example

To avoid complicating the notation more than necessary, suppose
1. The inputs, states and outputs are all scalars
2. The loss at each step is a function f of the state at that step

First input: x_1
Transform: $t_1 = s_0 W_S + x_1 W_I + b$
State: $s_1 = g(t_1)$
Loss: $l_1 = f(s_1)$

Second input: x_2
Transform: $t_2 = s_1 W_S + x_2 W_I + b$
State: $s_2 = g(t_2)$
Loss: $l_2 = f(s_2)$

Two dependencies on W_I

Let's compute the derivative of the loss with respect to the parameter W_I

Once again, the chain rule

$$\frac{\partial l_2}{\partial W_I} = \frac{\partial l_2}{\partial s_2} \cdot \frac{\partial s_2}{\partial t_2} \cdot \left(\qquad \right)$$

Does this work? Let's see a simple example

To avoid complicating the notation more than necessary, suppose
1. The inputs, states and outputs are all scalars
2. The loss at each step is a function f of the state at that step

First input: x_1
Transform: $t_1 = s_0 W_S + x_1 W_I + b$
State: $s_1 = g(t_1)$
Loss: $l_1 = f(s_1)$

Second input: x_2
Transform: $t_2 = s_1 W_S + x_2 W_I + b$
State: $s_2 = g(t_2)$
Loss: $l_2 = f(s_2)$

Let's compute the derivative of the loss with respect to the parameter W_I

Once again, the chain rule

$$\frac{\partial l_2}{\partial W_I} = \frac{\partial l_2}{\partial s_2} \cdot \frac{\partial s_2}{\partial t_2} \cdot \left(\frac{\partial t_2}{\partial W_I} + \right)$$

Does this work? Let's see a simple example

To avoid complicating the notation more than necessary, suppose
1. The inputs, states and outputs are all scalars
2. The loss at each step is a function f of the state at that step

First input: x_1
Transform: $t_1 = s_0 W_S + x_1 W_I + b$
State: $s_1 = g(t_1)$
Loss: $l_1 = f(s_1)$

Second input: x_2
Transform: $t_2 = s_1 W_S + x_2 W_I + b$
State: $s_2 = g(t_2)$
Loss: $l_2 = f(s_2)$

Let's compute the derivative of the loss with respect to the parameter W_I

Once again, the chain rule

$$\frac{\partial l_2}{\partial W_I} = \frac{\partial l_2}{\partial s_2} \cdot \frac{\partial s_2}{\partial t_2} \cdot \left(\frac{\partial t_2}{\partial W_I} + \frac{\partial t_2}{\partial s_1} \cdot \right)$$

Does this work? Let's see a simple example

To avoid complicating the notation more than necessary, suppose
1. The inputs, states and outputs are all scalars
2. The loss at each step is a function f of the state at that step

First input: x_1
Transform: $t_1 = s_0 W_S + x_1 W_I + b$
State: $s_1 = g(t_1)$
Loss: $l_1 = f(s_1)$

Second input: x_2
Transform: $t_2 = s_1 W_S + x_2 W_I + b$
State: $s_2 = g(t_2)$
Loss: $l_2 = f(s_2)$

Let's compute the derivative of the loss with respect to the parameter W_I

Once again, the chain rule

$$\frac{\partial l_2}{\partial W_I} = \frac{\partial l_2}{\partial s_2} \cdot \frac{\partial s_2}{\partial t_2} \cdot \left(\frac{\partial t_2}{\partial W_I} + \frac{\partial t_2}{\partial s_1} \cdot \frac{\partial s_1}{\partial t_1} \cdot \right)$$

Does this work? Let's see a simple example

To avoid complicating the notation more than necessary, suppose
1. The inputs, states and outputs are all scalars
2. The loss at each step is a function f of the state at that step

First input: x_1
Transform: $t_1 = s_0 W_S + x_1 W_I + b$
State: $s_1 = g(t_1)$
Loss: $l_1 = f(s_1)$

Second input: x_2
Transform: $t_2 = s_1 W_S + x_2 W_I + b$
State: $s_2 = g(t_2)$
Loss: $l_2 = f(s_2)$

Let's compute the derivative of the loss with respect to the parameter W_I

Once again, the chain rule

$$\frac{\partial l_2}{\partial W_I} = \frac{\partial l_2}{\partial s_2} \cdot \frac{\partial s_2}{\partial t_2} \cdot \left(\frac{\partial t_2}{\partial W_I} + \frac{\partial t_2}{\partial s_1} \cdot \frac{\partial s_1}{\partial t_1} \cdot \frac{\partial t_1}{\partial W_I} \right)$$

Does this work? Let's see a simple example

To avoid complicating the notation more than necessary, suppose
1. The inputs, states and outputs are all scalars
2. The loss at each step is a function f of the state at that step

First input: x_1
Transform: $t_1 = s_0 W_S + x_1 W_I + b$
State: $s_1 = g(t_1)$
Loss: $l_1 = f(s_1)$

Second input: x_2
Transform: $t_2 = s_1 W_S + x_2 W_I + b$
State: $s_2 = g(t_2)$
Loss: $l_2 = f(s_2)$

Let's compute the derivative of the loss with respect to the parameter W_I

Once again, the chain rule

$$\frac{\partial l_2}{\partial W_I} = \frac{\partial l_2}{\partial s_2} \cdot \frac{\partial s_2}{\partial t_2} \cdot \left(\frac{\partial t_2}{\partial W_I} + \frac{\partial t_2}{\partial s_1} \cdot \frac{\partial s_1}{\partial t_1} \cdot \frac{\partial t_1}{\partial W_I} \right)$$

How does the first input affect the loss for the second term?

Does this work? Let's see a simple example

To avoid complicating the notation more than necessary, suppose
1. The inputs, states and outputs are all scalars
2. The loss at each step is a function f of the state at that step

First input: x_1
Transform: $t_1 = s_0 W_S + x_1 W_I + b$
State: $s_1 = g(t_1)$
Loss: $l_1 = f(s_1)$

Second input: x_2
Transform: $t_2 = s_1 W_S + x_2 W_I + b$
State: $s_2 = g(t_2)$
Loss: $l_2 = f(s_2)$

Let's compute the derivative of the loss with respect to the parameter W_I

Once again, the chain rule

$$\frac{\partial l_2}{\partial W_I} = \frac{\partial l_2}{\partial s_2} \cdot \frac{\partial s_2}{\partial t_2} \cdot \left(\frac{\partial t_2}{\partial W_I} + \frac{\partial t_2}{\partial s_1} \cdot \frac{\partial s_1}{\partial t_1} \cdot \frac{\partial t_1}{\partial W_I} \right)$$

How does the first input affect the loss for the second term?
Through this term here

Does this work? Let's see a simple example

To avoid complicating the notation more than necessary, suppose
1. The inputs, states and outputs are all scalars
2. The loss at each step is a function f of the state at that step

First input: x_1
Transform: $t_1 = s_0 W_S + x_1 W_I + b$
State: $s_1 = g(t_1)$
Loss: $l_1 = f(s_1)$

Second input: x_2
Transform: $t_2 = s_1 W_S + x_2 W_I + b$
State: $s_2 = g(t_2)$
Loss: $l_2 = f(s_2)$

Let's compute the derivative of the loss with respect to the parameter W_I

Once again, the chain rule

$$\frac{\partial l_2}{\partial W_I} = \frac{\partial l_2}{\partial s_2} \cdot \frac{\partial s_2}{\partial t_2} \cdot \left(\frac{\partial t_2}{\partial W_I} + \frac{\partial t_2}{\partial s_1} \cdot \frac{\partial s_1}{\partial t_1} \cdot \frac{\partial t_1}{\partial W_I} \right)$$

But this gradient is multiplied by all these other terms

Does this work? Let's see a simple example

To avoid complicating the notation more than necessary, suppose
1. The inputs, states and outputs are all scalars
2. The loss at each step is a function f of the state at that step

First input: x_1
Transform: $t_1 = s_0 W_S + x_1 W_I + b$
State: $s_1 = g(t_1)$
Loss: $l_1 = f(s_1)$

Second input: x_2
Transform: $t_2 = s_1 W_S + x_2 W_I + b$
State: $s_2 = g(t_2)$
Loss: $l_2 = f(s_2)$

Let's compute the derivative of the loss with respect to the parameter W_I

Once again, the chain rule

$$\frac{\partial l_2}{\partial W_I} = \frac{\partial l_2}{\partial s_2} \cdot \frac{\partial s_2}{\partial t_2} \cdot \left(\frac{\partial t_2}{\partial W_I} + \frac{\partial t_2}{\partial s_1} \cdot \frac{\partial s_1}{\partial t_1} \cdot \frac{\partial t_1}{\partial W_I} \right)$$

Let's focus on the impact of the activation terms

Does this work? Let's see a simple example

To avoid complicating the notation more than necessary, suppose
1. The inputs, states and outputs are all scalars
2. The loss at each step is a function f of the state at that step

First input: x_1
Transform: $t_1 = s_0 W_S + x_1 W_I + b$
State: $s_1 = g(t_1)$
Loss: $l_1 = f(s_1)$

Second input: x_2
Transform: $t_2 = s_1 W_S + x_2 W_I + b$
State: $s_2 = g(t_2)$
Loss: $l_2 = f(s_2)$

Suppose $g(z) = \tanh(z)$

Then $\frac{dg}{dz} = 1 - \tanh^2(z)$

Let's compute the derivative of the loss with respect to the parameter W_I

Once again, the chain rule

$$\frac{\partial l_2}{\partial W_I} = \frac{\partial l_2}{\partial s_2} \cdot \frac{\partial s_2}{\partial t_2} \cdot \left(\frac{\partial t_2}{\partial W_I} + \frac{\partial t_2}{\partial s_1} \cdot \frac{\partial s_1}{\partial t_1} \cdot \frac{\partial t_1}{\partial W_I} \right)$$

Let's focus on the impact of the activation terms

Does this work? Let's see a simple example

To avoid complicating the notation more than necessary, suppose
1. The inputs, states and outputs are all scalars
2. The loss at each step is a function f of the state at that step

First input: x_1
Transform: $t_1 = s_0 W_S + x_1 W_I + b$
State: $s_1 = g(t_1)$
Loss: $l_1 = f(s_1)$

Second input: x_2
Transform: $t_2 = s_1 W_S + x_2 W_I + b$
State: $s_2 = g(t_2)$
Loss: $l_2 = f(s_2)$

Suppose $g(z) = \tanh(z)$

Then $\frac{dg}{dz} = 1 - \tanh^2(z)$

Let's compute the derivative of the loss with respect to the parameter W_I

Once again, the chain rule

$$\frac{\partial l_2}{\partial W_I} = \frac{\partial l_2}{\partial s_2} \cdot \frac{\partial s_2}{\partial t_2} \cdot \left(\frac{\partial t_2}{\partial W_I} + \frac{\partial t_2}{\partial s_1} \cdot \frac{\partial s_1}{\partial t_1} \cdot \frac{\partial t_1}{\partial W_I} \right)$$

Let's focus on the impact of the activation terms

Both these gradients are numbers between zero and one. **Multiplying them scales the gradient down**

Does this work? Let's see a simple example

To avoid complicating the notation more than necessary, suppose
1. The inputs, states and outputs are all scalars
2. The loss at each step is a function f of the state at that step

First input: x_1
Transform: $t_1 = s_0 W_S + x_1 W_I + b$
State: $s_1 = g(t_1)$
Loss: $l_1 = f(s_1)$

Second input: x_2
Transform: $t_2 = s_1 W_S + x_2 W_I + b$
State: $s_2 = g(t_2)$
Loss: $l_2 = f(s_2)$

With ***one*** input, the contribution of the ***first*** input towards the gradient of the loss of the ***first*** output is scaled by ***one*** term between zero and one.

Does this work? Let's see a simple example

To avoid complicating the notation more than necessary, suppose
1. The inputs, states and outputs are all scalars
2. The loss at each step is a function f of the state at that step

First input: x_1
Transform: $t_1 = s_0 W_S + x_1 W_I + b$
State: $s_1 = g(t_1)$
Loss: $l_1 = f(s_1)$

Second input: x_2
Transform: $t_2 = s_1 W_S + x_2 W_I + b$
State: $s_2 = g(t_2)$
Loss: $l_2 = f(s_2)$

With ***two*** inputs, the contribution of the *first* input towards the gradient of the loss of the ***second*** output is scaled by ***two*** terms between zero and one.

Does this work? Let's see a simple example

To avoid complicating the notation more than necessary, suppose
1. The inputs, states and outputs are all scalars
2. The loss at each step is a function f of the state at that step

First input: x_1
Transform: $t_1 = s_0 W_S + x_1 W_I + b$
State: $s_1 = g(t_1)$
Loss: $l_1 = f(s_1)$

Second input: x_2
Transform: $t_2 = s_1 W_S + x_2 W_I + b$
State: $s_2 = g(t_2)$
Loss: $l_2 = f(s_2)$

n[th] input: x_n
Transform: $t_n = s_{n-1} W_S + x_n W_I + b$
State: $s_n = g(t_n)$
Loss: $l_n = f(s_n)$

With ***n*** inputs, the contribution of the ***first*** input towards the gradient of the loss of the ***n[th]*** output is scaled by ***n*** terms between zero and one.

The vanishing gradient problem

[Bengio et al 1994]

- As the length of the sequence grows, the impact of the far away inputs diminishes because the gradient vanishes
- We saw an example where states and inputs are scalars.
  - Applies when the states and inputs are vectors/matrices as in usual networks

The vanishing gradient problem

- As the length of the sequence grows, the impact of the far away inputs diminishes because the gradient vanishes
- We saw an example where states and inputs are scalars.
  - Applies when the states and inputs are vectors/matrices as in usual networks

Why is this a problem?

The vanishing gradient problem

- As the length of the sequence grows, the impact of the far away inputs diminishes because the gradient vanishes
- We saw an example where states and inputs are scalars.
  - Applies when the states and inputs are vectors/matrices as in usual networks

Why is this a problem?

I have a banana and an apple. My friend ate the banana and I ate the _____?

The vanishing gradient problem

- As the length of the sequence grows, the impact of the far away inputs diminishes because the gradient vanishes
- We saw an example where states and inputs are scalars.
  - Applies when the states and inputs are vectors/matrices as in usual networks

Why is this a problem?

I have a banana and an apple. My friend ate the banana. I was hungry and wanted a fruit. So I ate the _____?

The vanishing gradient problem

- As the length of the sequence grows, the impact of the far away inputs diminishes because the gradient vanishes
- We saw an example where states and inputs are scalars.
  - Applies when the states and inputs are vectors/matrices as in usual networks

Why is this a problem?

I have a banana and an apple. My friend ate the banana. I was hungry and wanted a fruit. I really wished I had a banana as well, but we were all out. So I ate the _____?

The vanishing gradient problem

- As the length of the sequence grows, the impact of the far away inputs diminishes because the gradient vanishes
- We saw an example where states and inputs are scalars.
  - Applies when the states and inputs are vectors/matrices as in usual networks

Why is this a problem?

I have a banana and an apple. My friend ate the banana. I was hungry and wanted a fruit. I really wished I had a banana as well, but we were all out. So I ate the _____?

Consider a RNN language model for this task. If it makes a mistake in the final word, the signal for correcting it is far away.

The vanishing gradient problem

- As the length of the sequence grows, the impact of the far away inputs diminishes because the gradient vanishes
- We saw an example where states and inputs are scalars.
  - Applies when the states and inputs are vectors/matrices as in usual networks

Why is this a problem?

I have a banana and an apple. My friend ate the banana. I was hungry and wanted a fruit. I really wished I had a banana as well, but we were all out. So I ate the _____?

[Hochreiter and Schmidhuber 1997]:
"Backpropagation through time is too sensitive to recent distractions."

The vanishing gradient problem

- As the length of the sequence grows, the impact of the far away inputs diminishes because the gradient vanishes
- We saw an example where states and inputs are scalars.
  - Applies when the states and inputs are vectors/matrices as in usual networks
- Happens because the gradient of the non-linear activation is a number between zero and one
  - … and many such numbers are multiplied together

The vanishing gradient problem

- As the length of the sequence grows, the impact of the far away inputs diminishes because the gradient vanishes
- We saw an example where states and inputs are scalars.
  - Applies when the states and inputs are vectors/matrices as in usual networks
- Happens because the gradient of the non-linear activation is a number between zero and one
  - … and many such numbers are multiplied together
- Applicable not only to recurrent networks, but to any case where we have a long chain of such activations (i.e. in a deep network): Layers closer to the loss will get larger updates

Addressing the vanishing gradient problem

Approach 1: Change the activation

- The problem occurs because the derivatives of the activation function are small, so change it

Addressing the vanishing gradient problem

Approach 1: Change the activation

- The problem occurs because the derivatives of the activation function are small, so change it
- Commonly used: the rectified linear unit

$$\mathrm{ReLU}(z) = \max(0, z)$$

Addressing the vanishing gradient problem

Approach 1: Change the activation

- The problem occurs because the derivatives of the activation function are small, so change it
- Commonly used: the rectified linear unit

$$\mathrm{ReLU}(z) = \max(0, z)$$

What is its derivative?

Addressing the vanishing gradient problem

Approach 1: Change the activation

- The problem occurs because the derivatives of the activation function are small, so change it
- Commonly used: the rectified linear unit

$$\mathrm{ReLU}(z) = \max(0, z)$$

What is its derivative?

$$\frac{d\ RELU}{dz} = \begin{cases} 1 & \text{if } z \geq 0 \\ 0 & \text{else} \end{cases}$$

Addressing the vanishing gradient problem

Approach 1: Change the activation

- The problem occurs because the derivatives of the activation function are small, so change it
- Commonly used: the rectified linear unit

$$\mathrm{ReLU}(z) = \max(0, z)$$

What is its derivative?

$$\frac{d\ RELU}{dz} = \begin{cases} 1 & \text{if } z \geq 0 \\ 0 & \text{else} \end{cases}$$

Multiplying many of these won't vanish the gradient if the pre-activation value is positive.

But can completely erase the gradient if it is negative.

Exploding gradients

If our gradients are not fractional (e.g. with ReLUs), we might end up multiplying many large numbers during gradient computation

This could quickly give numeric overflow errors

The Exploding Gradient Problem

Addressing vanishing/exploding gradients

Approach 2: Don't take derivatives all the way to the beginning

- The problem occurs because we need to compute derivatives with respect to the early inputs
- Truncate the backpropagation process instead
- Called Truncated Backpropagation Through Time (TBPTT)

Essentially, this makes a Markov-like assumption.

Addressing vanishing/exploding gradients

Approach 3: Use a ReLU activation, but explicitly avoid exploding gradients

- If a gradient is larger than a certain threshold, truncate it

- ReLUs reduce vanishing gradients, and truncation takes care of exploding gradients

Addressing vanishing/exploding gradients

Approach 4: Changing the internals of the RNN more thoroughly…

> … by using a gated architecture such as an LSTM or a GRU unit