CS7015 (Deep Learning) : Lecture 13

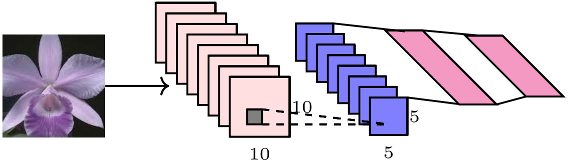
Sequence Learning Problems, Recurrent Neural Networks, Backpropagation Through Time (BPTT), Vanishing and Exploding Gradients, Truncated BPTT

Mitesh M. Khapra

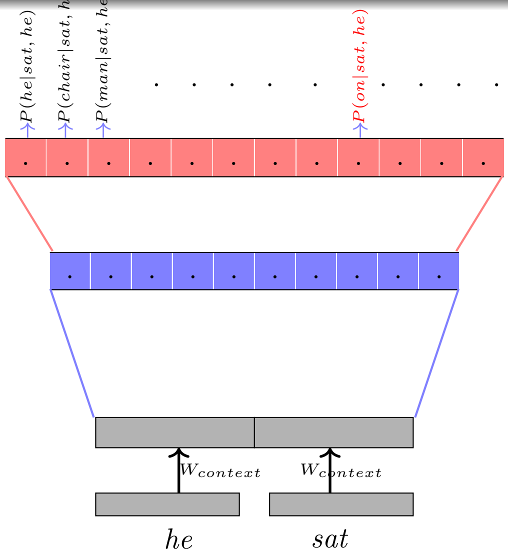
Department of Computer Science and Engineering
Indian Institute of Technology Madras

Module : Sequence Learning Problems

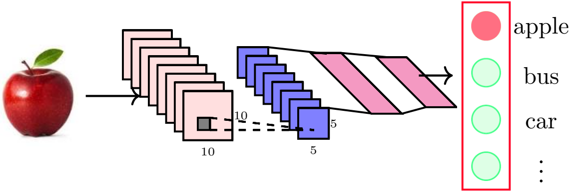
- In feedforward and convolutional neural networks the size of the input was always fixed



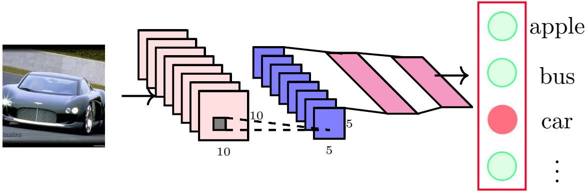
- In feedforward and convolutional neural networks the size of the input was always fixed
- For example, we fed fixed size (32×32) images to convolutional neural networks for image classification



- In feedforward and convolutional neural networks the size of the input was always fixed
- For example, we fed fixed size (32×32) images to convolutional neural networks for image classification
- Similarly in word2vec, we fed a fixed window (k) of words to the network



- In feedforward and convolutional neural networks the size of the input was always fixed
- For example, we fed fixed size (32×32) images to convolutional neural networks for image classification
- Similarly in word2vec, we fed a fixed window (k) of words to the network
- Further, each input to the network was independent of the previous or future inputs



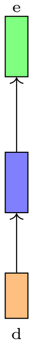
- In feedforward and convolutional neural networks the size of the input was always fixed
- For example, we fed fixed size (32×32) images to convolutional neural networks for image classification
- Similarly in word2vec, we fed a fixed window (k) of words to the network
- Further, each input to the network was independent of the previous or future inputs
- For example, the computatations, outputs and decisions for two successive images are completely independent of each other

- In many applications the input is not of a fixed size

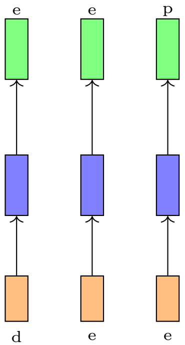
- In many applications the input is not of a fixed size
- Further successive inputs may not be independent of each other

- In many applications the input is not of a fixed size
- Further successive inputs may not be independent of each other
- For example, consider the task of auto completion

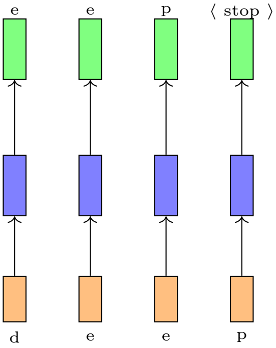
- In many applications the input is not of a fixed size
- Further successive inputs may not be independent of each other
- For example, consider the task of auto completion
- Given the first character 'd' you want to predict the next character 'e' and so on



- In many applications the input is not of a fixed size
- Further successive inputs may not be independent of each other
- For example, consider the task of auto completion
- Given the first character ‘d’ you want to predict the next character ‘e’ and so on

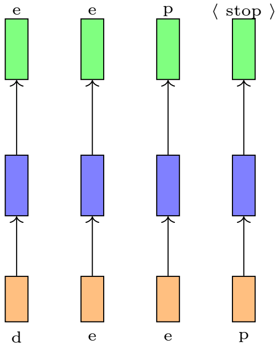


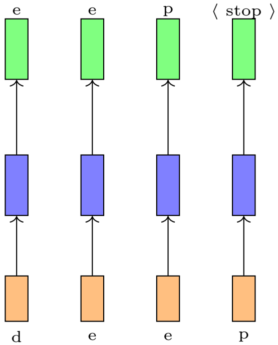
- In many applications the input is not of a fixed size
- Further successive inputs may not be independent of each other
- For example, consider the task of auto completion
- Given the first character 'd' you want to predict the next character 'e' and so on



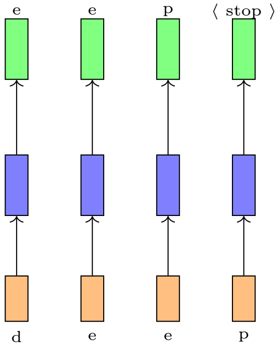
- In many applications the input is not of a fixed size
- Further successive inputs may not be independent of each other
- For example, consider the task of auto completion
- Given the first character 'd' you want to predict the next character 'e' and so on

- Notice a few things

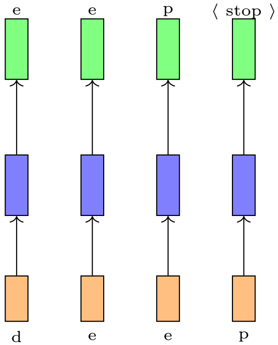




- Notice a few things
- First, successive inputs are no longer independent (while predicting 'e' you would want to know what the previous input was in addition to the current input)

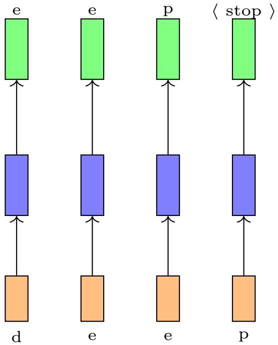


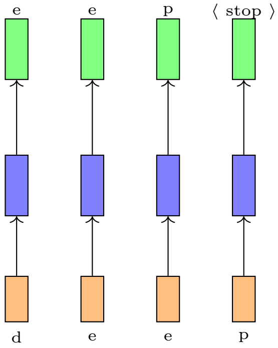
- Notice a few things
- First, successive inputs are no longer independent (while predicting ‘e’ you would want to know what the previous input was in addition to the current input)
- Second, the length of the inputs and the number of predictions you need to make is not fixed (for example, “learn”, “deep”, “machine” have different number of characters)



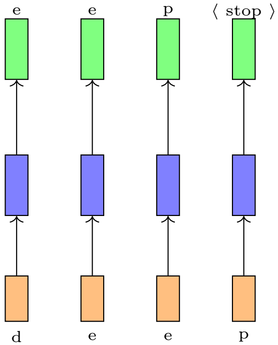
- Notice a few things
- First, successive inputs are no longer independent (while predicting ‘e’ you would want to know what the previous input was in addition to the current input)
- Second, the length of the inputs and the number of predictions you need to make is not fixed (for example, “learn”, “deep”, “machine” have different number of characters)
- Third, each network (orange-blue-green structure) is performing the same task (**input** : character **output** : character)

- These are known as sequence learning problems

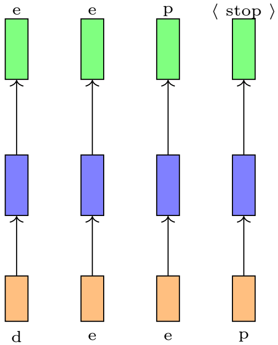




- These are known as sequence learning problems
- We need to look at a sequence of (dependent) inputs and produce an output (or outputs)



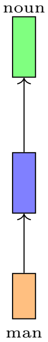
- These are known as sequence learning problems
- We need to look at a sequence of (dependent) inputs and produce an output (or outputs)
- Each input corresponds to one time step



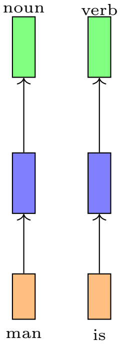
- These are known as sequence learning problems
- We need to look at a sequence of (dependent) inputs and produce an output (or outputs)
- Each input corresponds to one time step
- Let us look at some more examples of such problems

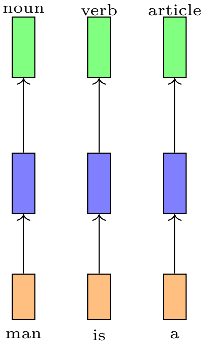
- Consider the task of predicting the part of speech tag (noun, adverb, adjective verb) of each word in a sentence

- Consider the task of predicting the part of speech tag (noun, adverb, adjective verb) of each word in a sentence

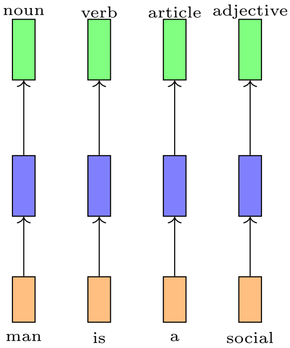


- Consider the task of predicting the part of speech tag (noun, adverb, adjective verb) of each word in a sentence



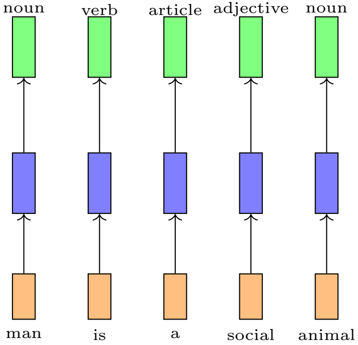


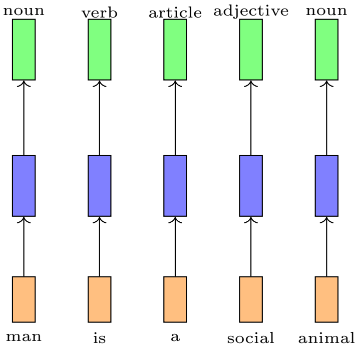
- Consider the task of predicting the part of speech tag (noun, adverb, adjective verb) of each word in a sentence



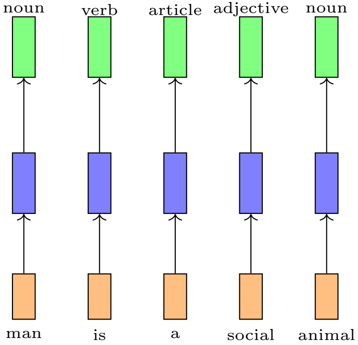
- Consider the task of predicting the part of speech tag (noun, adverb, adjective verb) of each word in a sentence

- Consider the task of predicting the part of speech tag (noun, adverb, adjective verb) of each word in a sentence

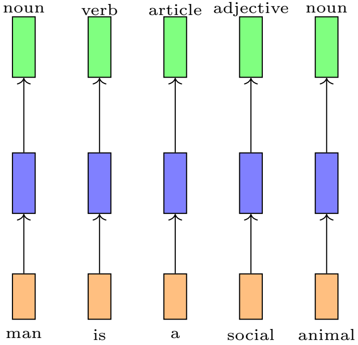




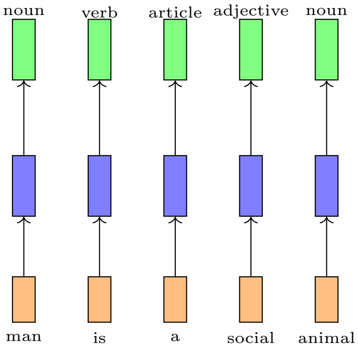
- Consider the task of predicting the part of speech tag (noun, adverb, adjective verb) of each word in a sentence
- Once we see an adjective (social) we are almost sure that the next word should be a noun (man)



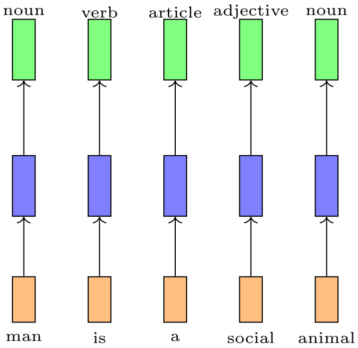
- Consider the task of predicting the part of speech tag (noun, adverb, adjective verb) of each word in a sentence
- Once we see an adjective (social) we are almost sure that the next word should be a noun (man)
- Thus the current output (noun) depends on the current input as well as the previous input



- Consider the task of predicting the part of speech tag (noun, adverb, adjective verb) of each word in a sentence
- Once we see an adjective (social) we are <u>almost</u> sure that the next word should be a noun (man)
- Thus the current output (noun) depends on the current input as well as the previous input
- Further the size of the input is not fixed (sentences could have arbitrary number of words)



- Consider the task of predicting the part of speech tag (noun, adverb, adjective verb) of each word in a sentence
- Once we see an adjective (social) we are almost sure that the next word should be a noun (man)
- Thus the current output (noun) depends on the current input as well as the previous input
- Further the size of the input is not fixed (sentences could have arbitrary number of words)
- Notice that here we are interested in producing an output at each time step

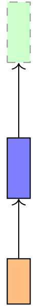


- Consider the task of predicting the part of speech tag (noun, adverb, adjective verb) of each word in a sentence
- Once we see an adjective (social) we are almost sure that the next word should be a noun (man)
- Thus the current output (noun) depends on the current input as well as the previous input
- Further the size of the input is not fixed (sentences could have arbitrary number of words)
- Notice that here we are interested in producing an output at each time step
- Each network is performing the same task (**input** : word, **output** : tag)

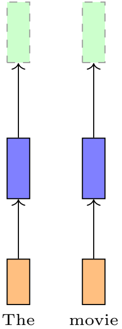
- Sometimes we may not be interested in producing an output at every stage

- Sometimes we may not be interested in producing an output at every stage
- Instead we would look at the full sequence and then produce an output

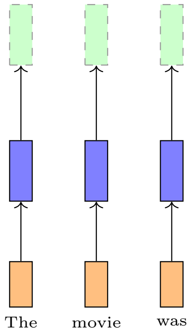
- Sometimes we may not be interested in producing an output at every stage
- Instead we would look at the full sequence and then produce an output
- For example, consider the task of predicting the polarity of a movie review



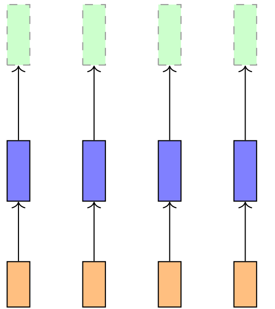
- Sometimes we may not be interested in producing an output at every stage
- Instead we would look at the full sequence and then produce an output
- For example, consider the task of predicting the polarity of a movie review



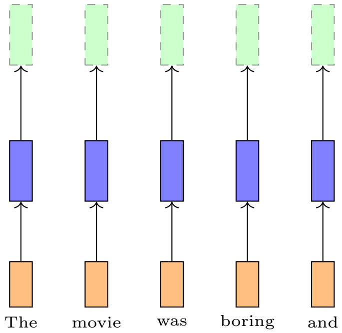
- Sometimes we may not be interested in producing an output at every stage
- Instead we would look at the full sequence and then produce an output
- For example, consider the task of predicting the polarity of a movie review



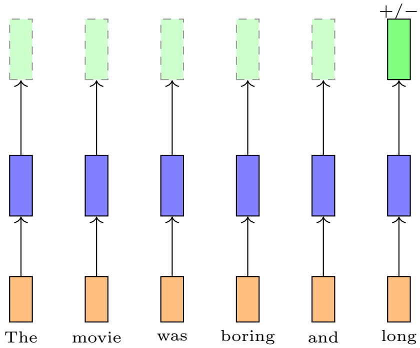
- Sometimes we may not be interested in producing an output at every stage
- Instead we would look at the full sequence and then produce an output
- For example, consider the task of predicting the polarity of a movie review



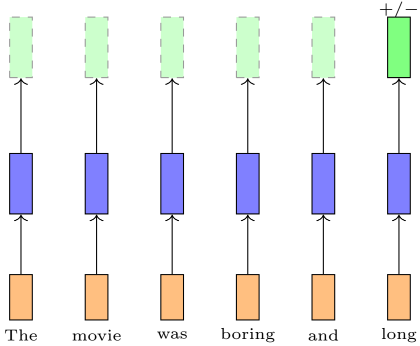
- Sometimes we may not be interested in producing an output at every stage
- Instead we would look at the full sequence and then produce an output
- For example, consider the task of predicting the polarity of a movie review



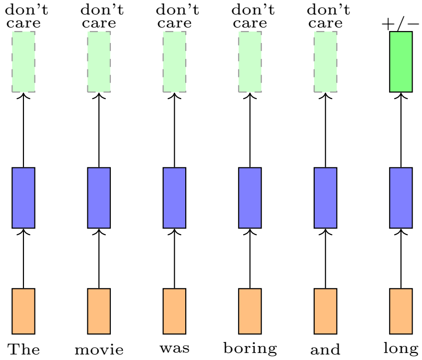
- Sometimes we may not be interested in producing an output at every stage
- Instead we would look at the full sequence and then produce an output
- For example, consider the task of predicting the polarity of a movie review



- Sometimes we may not be interested in producing an output at every stage
- Instead we would look at the full sequence and then produce an output
- For example, consider the task of predicting the polarity of a movie review



- Sometimes we may not be interested in producing an output at every stage
- Instead we would look at the full sequence and then produce an output
- For example, consider the task of predicting the polarity of a movie review
- The prediction clearly does not depend only on the last word but also on some words which appear before

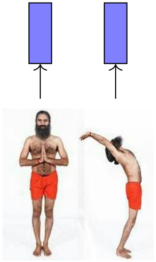


- Sometimes we may not be interested in producing an output at every stage
- Instead we would look at the full sequence and then produce an output
- For example, consider the task of predicting the polarity of a movie review
- The prediction clearly does not depend only on the last word but also on some words which appear before
- Here again we could think that the network is performing the same task at each step (input : word, output : $+/-$) but it's just that we don't care about intermediate outputs

- Sequences could be composed of anything (not just words)

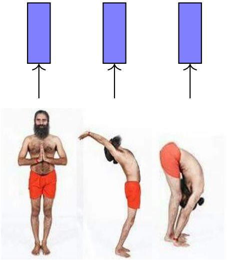
- Sequences could be composed of anything (not just words)
- For example, a video could be treated as a sequence of images

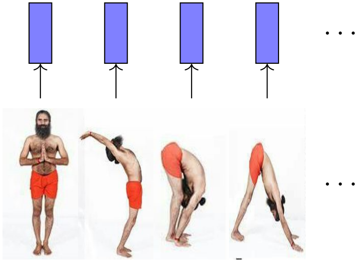
- Sequences could be composed of anything (not just words)
- For example, a video could be treated as a sequence of images



- Sequences could be composed of anything (not just words)
- For example, a video could be treated as a sequence of images

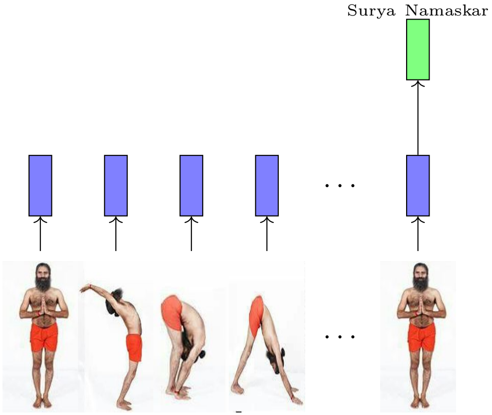
- Sequences could be composed of anything (not just words)
- For example, a video could be treated as a sequence of images

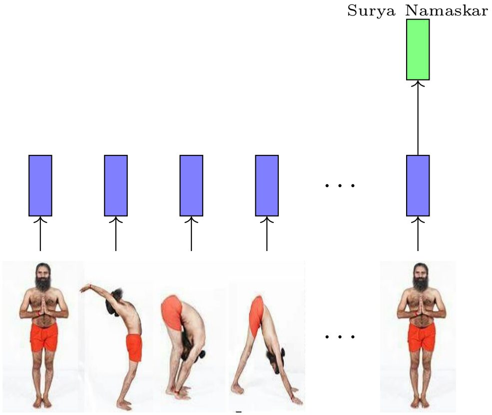




- Sequences could be composed of anything (not just words)
- For example, a video could be treated as a sequence of images

- Sequences could be composed of anything (not just words)
- For example, a video could be treated as a sequence of images





- Sequences could be composed of anything (not just words)
- For example, a video could be treated as a sequence of images
- We may want to look at the entire sequence and detect the activity being performed

Module : Recurrent Neural Networks

How do we model such tasks involving sequences ?

Wishlist

- Account for dependence between inputs

Wishlist

- Account for dependence between inputs
- Account for variable number of inputs

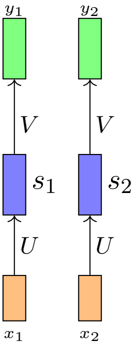
Wishlist

- Account for dependence between inputs
- Account for variable number of inputs
- Make sure that the function executed at each time step is the same

Wishlist

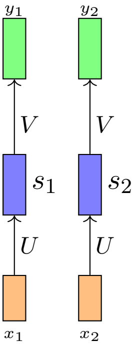
- Account for dependence between inputs
- Account for variable number of inputs
- Make sure that the function executed at each time step is the same
- We will focus on each of these to arrive at a model for dealing with sequences

- What is the function being executed at each time step ?



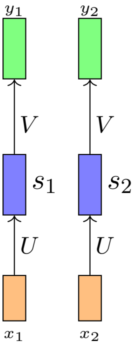
- What is the function being executed at each time step ?

$$s_i = \sigma(Ux_i + b)$$



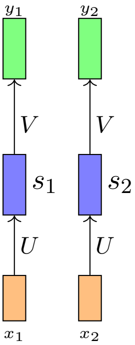
- What is the function being executed at each time step ?

$$s_i = \sigma(Ux_i + b)$$
$$y_i = \mathcal{O}(Vs_i + c)$$



- What is the function being executed at each time step ?

$$s_i = \sigma(Ux_i + b)$$
$$y_i = \mathcal{O}(Vs_i + c)$$
$$i = \text{timestep}$$

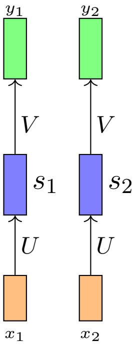


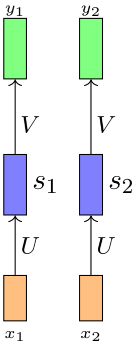
- What is the function being executed at each time step ?

$$s_i = \sigma(Ux_i + b)$$
$$y_i = \mathcal{O}(Vs_i + c)$$
$$i = \text{timestep}$$

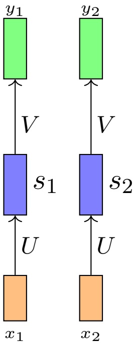
- Since we want the same function to be executed at each timestep we should share the same network (i.e., same parameters at each timestep)

- This parameter sharing also ensures that the network becomes agnostic to the length (size) of the input

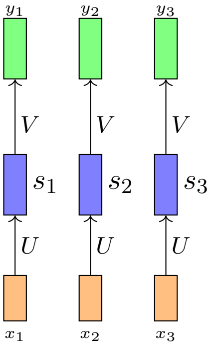




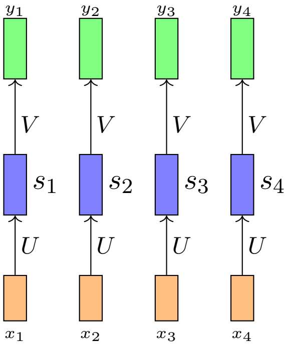
- This parameter sharing also ensures that the network becomes agnostic to the length (size) of the input
- Since we are simply going to compute the same function (with same parameters) at each timestep, the number of timesteps doesn't matter



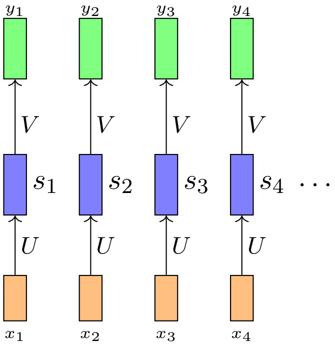
- This parameter sharing also ensures that the network becomes agnostic to the length (size) of the input
- Since we are simply going to compute the same function (with same parameters) at each timestep, the number of timesteps doesn't matter
- We just create multiple copies of the network and execute them at each timestep



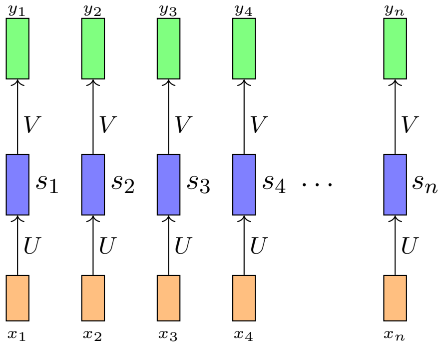
- This parameter sharing also ensures that the network becomes agnostic to the length (size) of the input
- Since we are simply going to compute the same function (with same parameters) at each timestep, the number of timesteps doesn't matter
- We just create multiple copies of the network and execute them at each timestep



- This parameter sharing also ensures that the network becomes agnostic to the length (size) of the input
- Since we are simply going to compute the same function (with same parameters) at each timestep, the number of timesteps doesn't matter
- We just create multiple copies of the network and execute them at each timestep

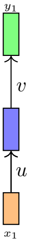


- This parameter sharing also ensures that the network becomes agnostic to the length (size) of the input
- Since we are simply going to compute the same function (with same parameters) at each timestep, the number of timesteps doesn't matter
- We just create multiple copies of the network and execute them at each timestep

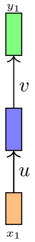


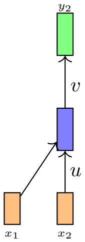
- This parameter sharing also ensures that the network becomes agnostic to the length (size) of the input
- Since we are simply going to compute the same function (with same parameters) at each timestep, the number of timesteps doesn't matter
- We just create multiple copies of the network and execute them at each timestep

- How do we account for dependence between inputs ?

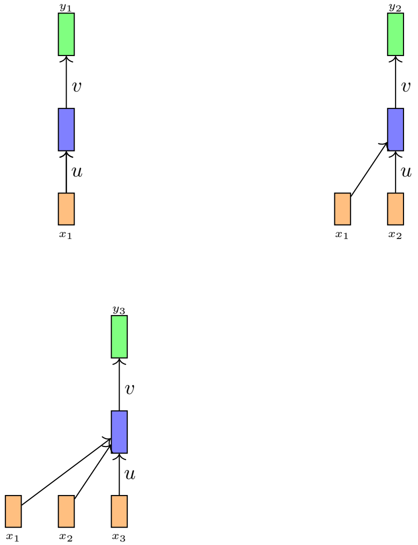


- How do we account for dependence between inputs ?
- Let us first see an infeasible way of doing this

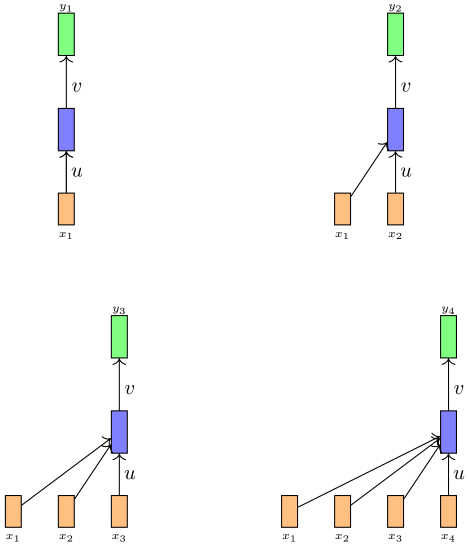




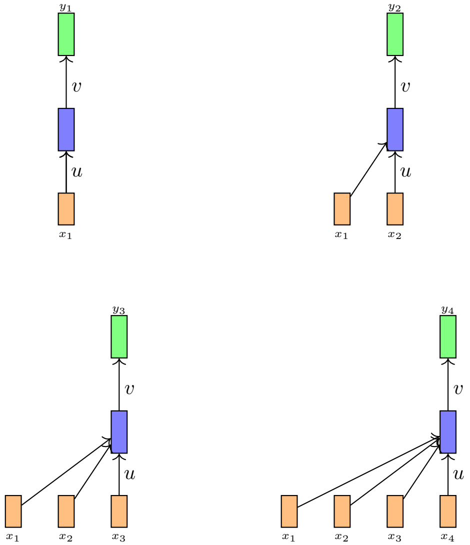
- How do we account for dependence between inputs ?
- Let us first see an infeasible way of doing this
- At each timestep we will feed all the previous inputs to the network



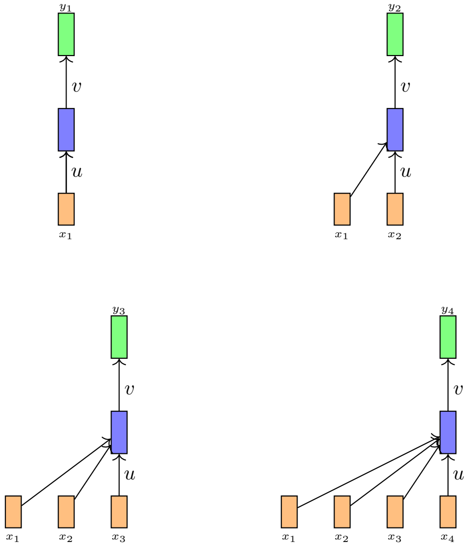
- How do we account for dependence between inputs ?
- Let us first see an infeasible way of doing this
- At each timestep we will feed all the previous inputs to the network



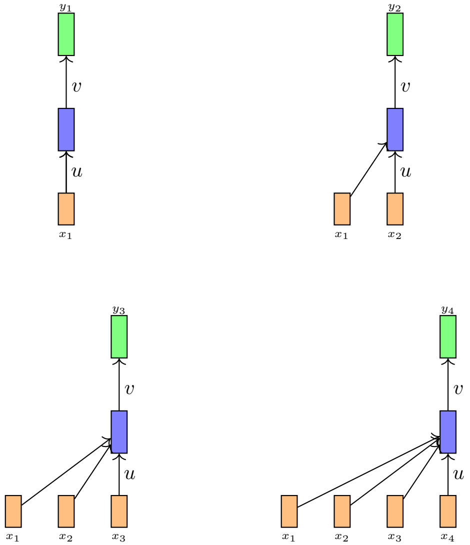
- How do we account for dependence between inputs ?
- Let us first see an infeasible way of doing this
- At each timestep we will feed all the previous inputs to the network



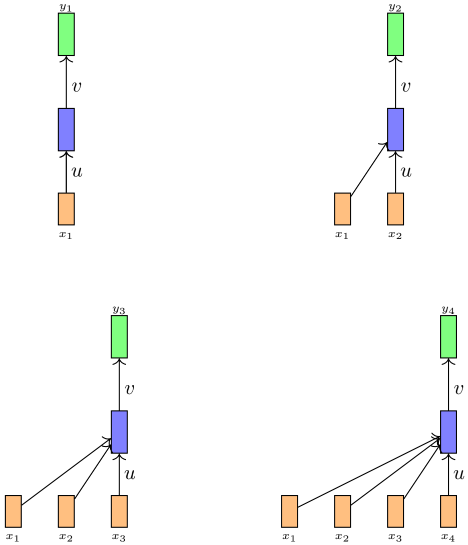
- How do we account for dependence between inputs ?
- Let us first see an infeasible way of doing this
- At each timestep we will feed all the previous inputs to the network
- Is this okay ?



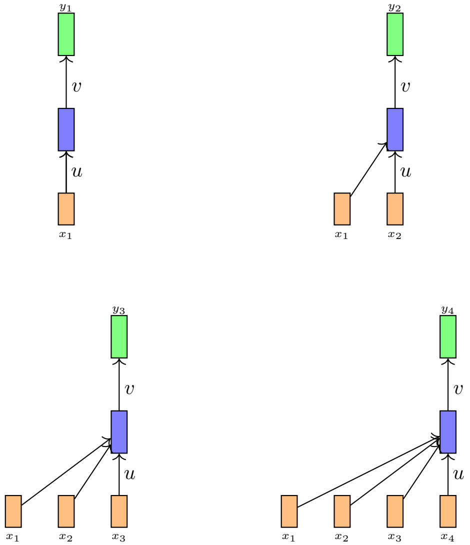
- How do we account for dependence between inputs ?
- Let us first see an infeasible way of doing this
- At each timestep we will feed all the previous inputs to the network
- Is this okay ?
- No, it violates the other two items on our wishlist



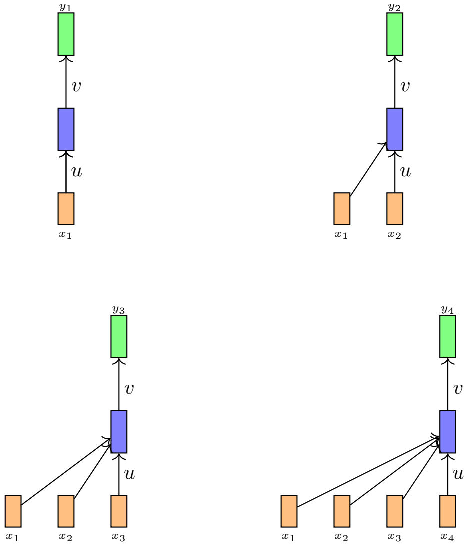
- How do we account for dependence between inputs ?
- Let us first see an infeasible way of doing this
- At each timestep we will feed all the previous inputs to the network
- Is this okay ?
- No, it violates the other two items on our wishlist
- How ?



- How do we account for dependence between inputs ?
- Let us first see an infeasible way of doing this
- At each timestep we will feed all the previous inputs to the network
- Is this okay ?
- No, it violates the other two items on our wishlist
- How ? Let us see

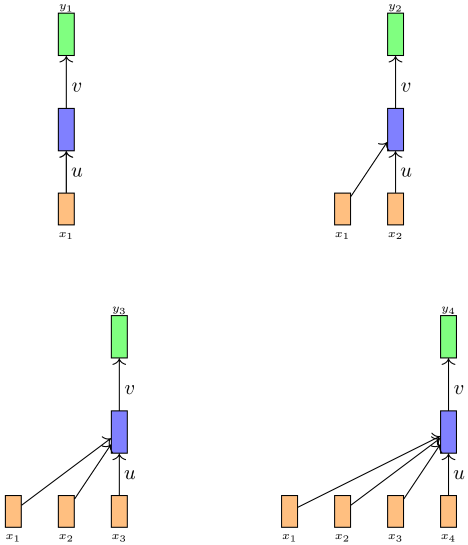


- First, the function being computed at each time-step now is different



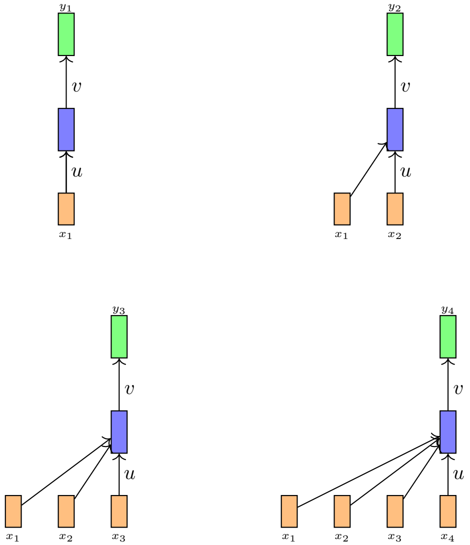
- First, the function being computed at each time-step now is different

$$y_1 = f_1(x_1)$$



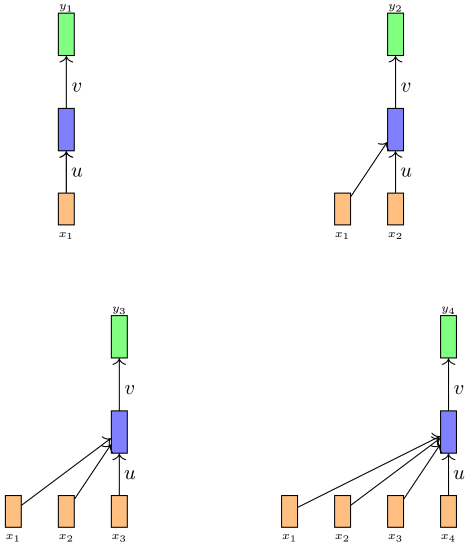
- First, the function being computed at each time-step now is different

$$y_1 = f_1(x_1)$$
$$y_2 = f_2(x_1, x_2)$$



- First, the function being computed at each time-step now is different

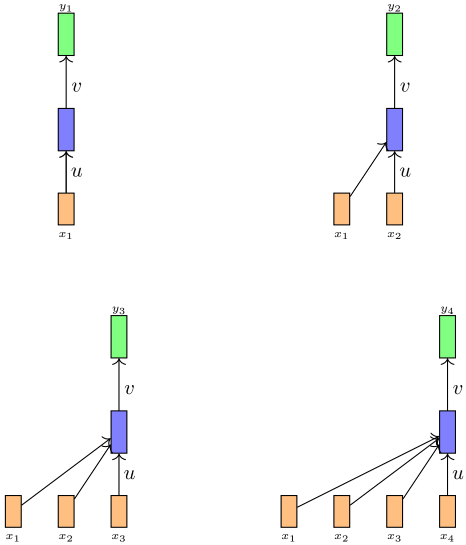
$$y_1 = f_1(x_1)$$
$$y_2 = f_2(x_1, x_2)$$
$$y_3 = f_3(x_1, x_2, x_3)$$



- First, the function being computed at each time-step now is different

$$y_1 = f_1(x_1)$$
$$y_2 = f_2(x_1, x_2)$$
$$y_3 = f_3(x_1, x_2, x_3)$$

- The network is now sensitive to the length of the sequence

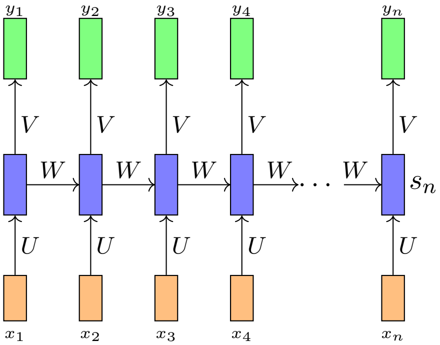


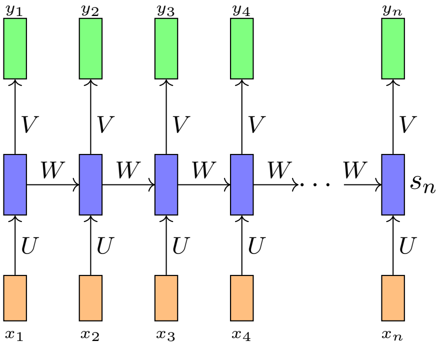
- First, the function being computed at each time-step now is different

$$
\begin{aligned}
y_1 &= f_1(x_1) \\
y_2 &= f_2(x_1, x_2) \\
y_3 &= f_3(x_1, x_2, x_3)
\end{aligned}
$$

- The network is now sensitive to the length of the sequence
- For example a sequence of length 10 will require $f_1, \ldots, f_{10}$ whereas a sequence of length 100 will require $f_1, \ldots, f_{100}$

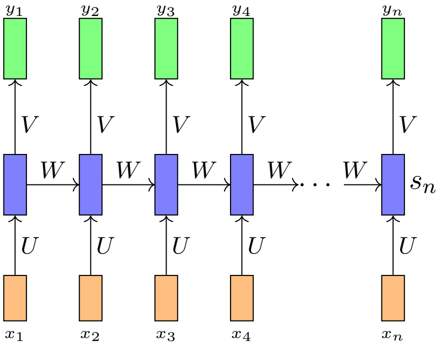
- The solution is to add a recurrent connection in the network,





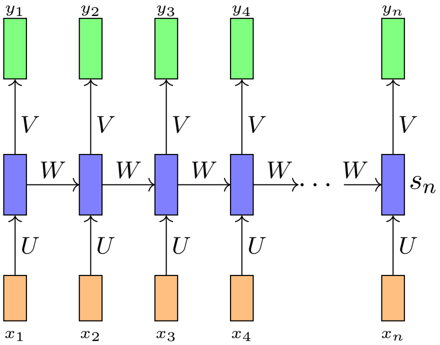
- The solution is to add a recurrent connection in the network,

$$s_i = \sigma(Ux_i + Ws_{i-1} + b)$$



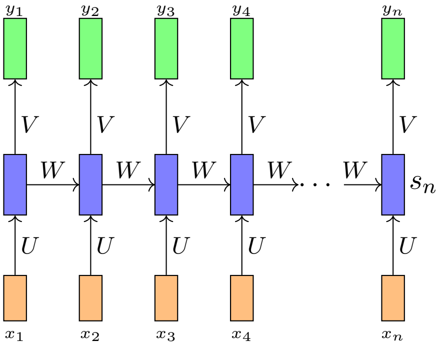
- The solution is to add a recurrent connection in the network,

$$s_i = \sigma(Ux_i + Ws_{i-1} + b)$$
$$y_i = \mathcal{O}(Vs_i + c)$$



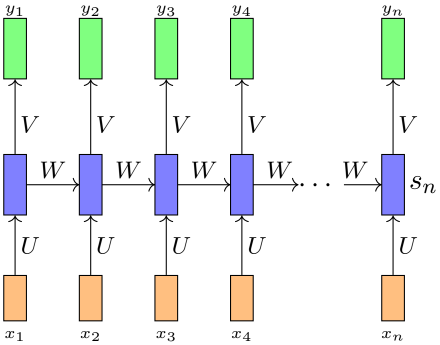
- The solution is to add a recurrent connection in the network,

$$s_i = \sigma(Ux_i + Ws_{i-1} + b)$$
$$y_i = \mathcal{O}(Vs_i + c)$$
$$or$$



- The solution is to add a recurrent connection in the network,

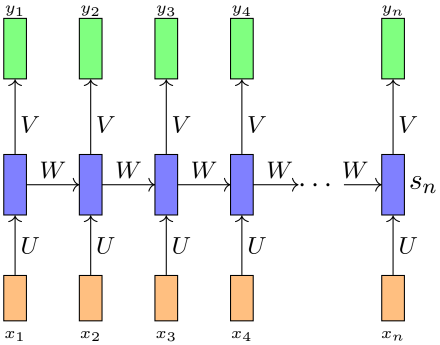
$$s_i = \sigma(Ux_i + Ws_{i-1} + b)$$
$$y_i = \mathcal{O}(Vs_i + c)$$
$$or$$
$$y_i = f(x_i, s_{i-1}, W, U, V, b, c)$$



- The solution is to add a recurrent connection in the network,

$$s_i = \sigma(Ux_i + Ws_{i-1} + b)$$
$$y_i = \mathcal{O}(Vs_i + c)$$
$$or$$
$$y_i = f(x_i, s_{i-1}, W, U, V, b, c)$$

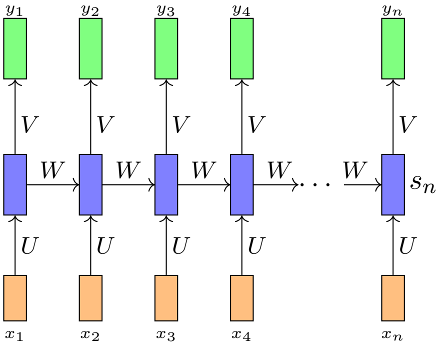
- s_i is the state of the network at timestep i



- The solution is to add a recurrent connection in the network,

$$s_i = \sigma(Ux_i + Ws_{i-1} + b)$$
$$y_i = \mathcal{O}(Vs_i + c)$$
$$or$$
$$y_i = f(x_i, s_{i-1}, W, U, V, b, c)$$

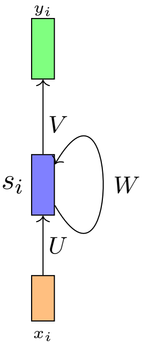
- s_i is the state of the network at timestep i
- The parameters are W, U, V, c, b which are shared across timesteps



- The solution is to add a recurrent connection in the network,

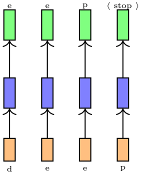
$$
\begin{aligned}
s_i &= \sigma(Ux_i + Ws_{i-1} + b) \\
y_i &= \mathcal{O}(Vs_i + c) \\
& \quad or \\
y_i &= f(x_i, s_{i-1}, W, U, V, b, c)
\end{aligned}
$$

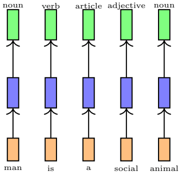
- s_i is the state of the network at timestep i
- The parameters are W, U, V, c, b which are shared across timesteps
- The same network (and parameters) can be used to compute $y_1, y_2, \ldots, y_{10}$ or y_{100}

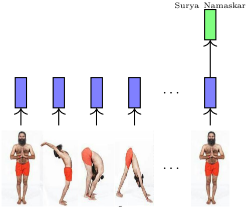


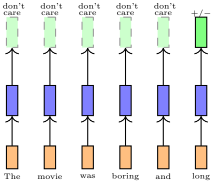
- This can be represented more compactly

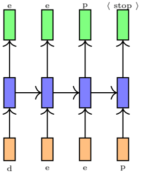
- Let us revisit the sequence learning problems that we saw earlier

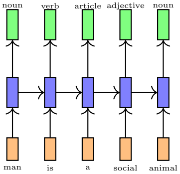


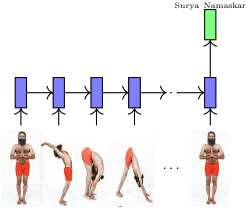


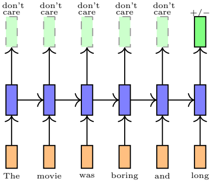






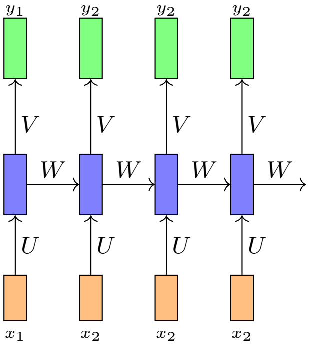




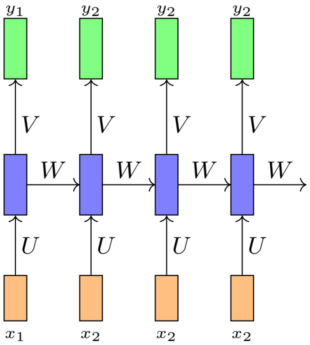


- Let us revisit the sequence learning problems that we saw earlier
- We now have recurrent connections between time steps which account for dependence between inputs

Module : Backpropagation through time

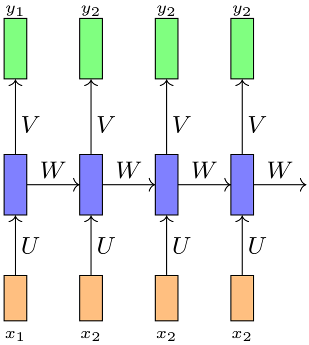


- Before proceeding let us look at the dimensions of the parameters carefully



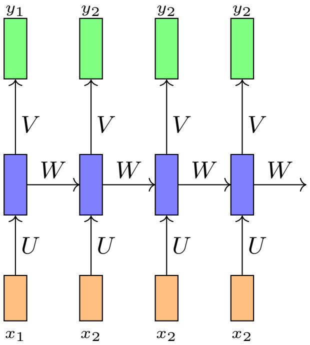
- Before proceeding let us look at the dimensions of the parameters carefully

$$x_i \in \mathbb{R}^n \quad \text{(n-dimensional input)}$$



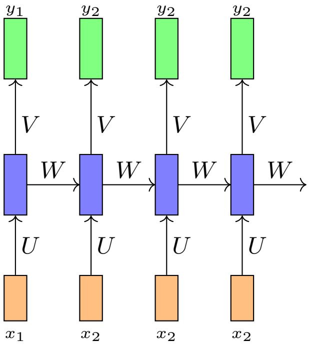
- Before proceeding let us look at the dimensions of the parameters carefully

$$x_i \in \mathbb{R}^n \quad \text{(n-dimensional input)}$$
$$s_i \in \mathbb{R}^d \quad \text{(d-dimensional state)}$$



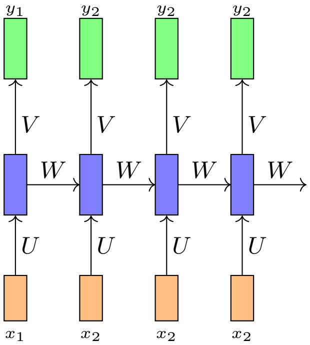
- Before proceeding let us look at the dimensions of the parameters carefully

$$x_i \in \mathbb{R}^n \quad \text{(n-dimensional input)}$$
$$s_i \in \mathbb{R}^d \quad \text{(d-dimensional state)}$$
$$y_i \in \mathbb{R}^k \quad \text{(say k classes)}$$



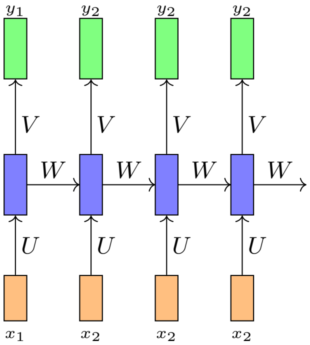
- Before proceeding let us look at the dimensions of the parameters carefully

$$x_i \in \mathbb{R}^n \quad \text{(n-dimensional input)}$$
$$s_i \in \mathbb{R}^d \quad \text{(d-dimensional state)}$$
$$y_i \in \mathbb{R}^k \quad \text{(say k classes)}$$
$$U \in$$



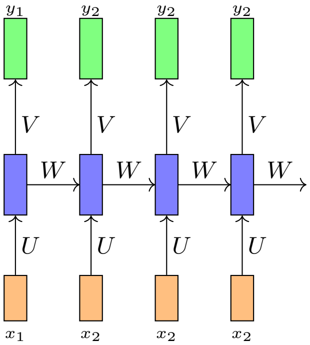
- Before proceeding let us look at the dimensions of the parameters carefully

$$x_i \in \mathbb{R}^n \quad \text{(n-dimensional input)}$$
$$s_i \in \mathbb{R}^d \quad \text{(d-dimensional state)}$$
$$y_i \in \mathbb{R}^k \quad \text{(say k classes)}$$
$$U \in \mathbb{R}^{n \times d}$$



- Before proceeding let us look at the dimensions of the parameters carefully

$$x_i \in \mathbb{R}^n \quad \text{(n-dimensional input)}$$
$$s_i \in \mathbb{R}^d \quad \text{(d-dimensional state)}$$
$$y_i \in \mathbb{R}^k \quad \text{(say k classes)}$$
$$U \in \mathbb{R}^{n \times d}$$
$$V \in$$



- Before proceeding let us look at the dimensions of the parameters carefully

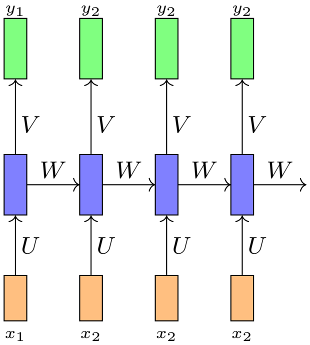
$$x_i \in \mathbb{R}^n \quad \text{(n-dimensional input)}$$

$$s_i \in \mathbb{R}^d \quad \text{(d-dimensional state)}$$

$$y_i \in \mathbb{R}^k \quad \text{(say k classes)}$$

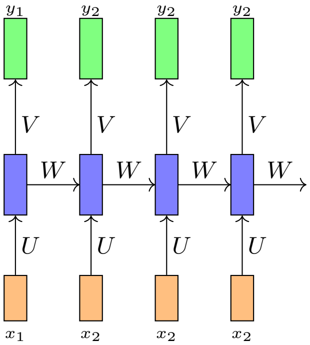
$$U \in \mathbb{R}^{n \times d}$$

$$V \in \mathbb{R}^{d \times k}$$



- Before proceeding let us look at the dimensions of the parameters carefully

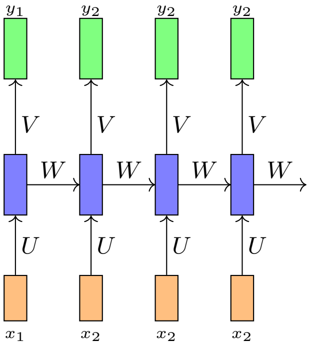
$$x_i \in \mathbb{R}^n \quad \text{(n-dimensional input)}$$
$$s_i \in \mathbb{R}^d \quad \text{(d-dimensional state)}$$
$$y_i \in \mathbb{R}^k \quad \text{(say k classes)}$$
$$U \in \mathbb{R}^{n \times d}$$
$$V \in \mathbb{R}^{d \times k}$$
$$W \in$$



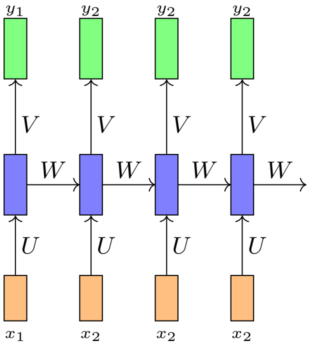
- Before proceeding let us look at the dimensions of the parameters carefully

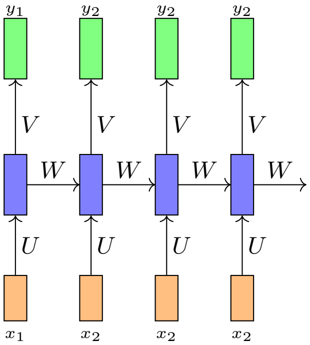
$$x_i \in \mathbb{R}^n \quad \text{(n-dimensional input)}$$
$$s_i \in \mathbb{R}^d \quad \text{(d-dimensional state)}$$
$$y_i \in \mathbb{R}^k \quad \text{(say k classes)}$$
$$U \in \mathbb{R}^{n \times d}$$
$$V \in \mathbb{R}^{d \times k}$$
$$W \in \mathbb{R}^{d \times d}$$

- How do we train this network ?



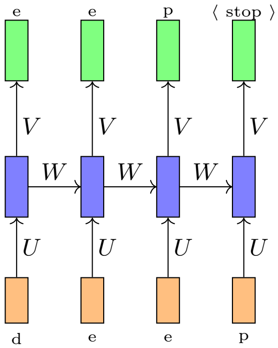
- How do we train this network ? (**Ans**: using backpropagation)

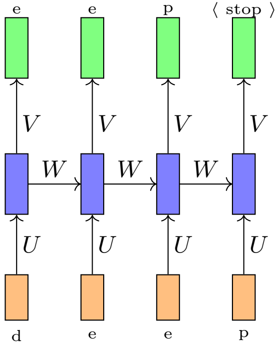




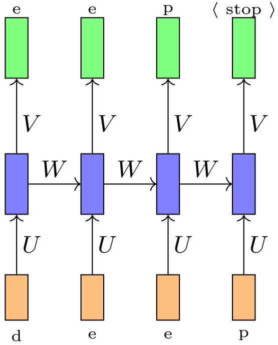
- How do we train this network ? (**Ans**: using backpropagation)
- Let us understand this with a concrete example

- Suppose we consider our task of auto-completion (predicting the next character)

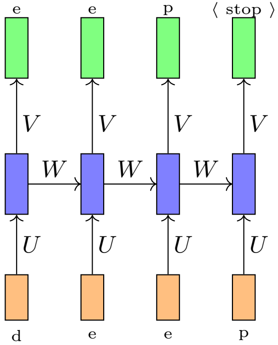




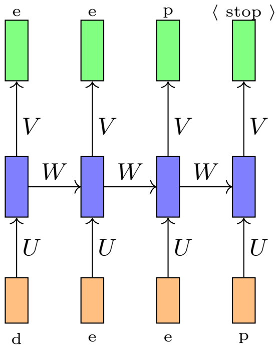
- Suppose we consider our task of auto-completion (predicting the next character)
- For simplicity we assume that there are only 4 characters in our vocabulary (d,e,p, <stop>)



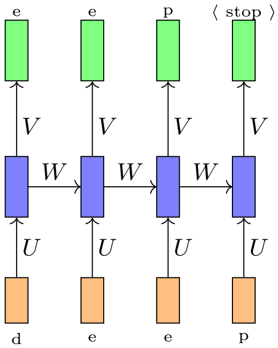
- Suppose we consider our task of auto-completion (predicting the next character)
- For simplicity we assume that there are only 4 characters in our vocabulary (d,e,p, <stop>)
- At each timestep we want to predict one of these 4 characters



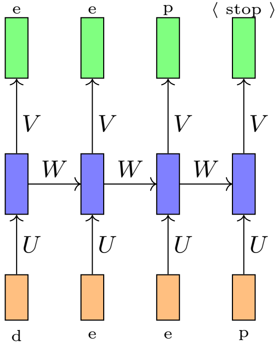
- Suppose we consider our task of auto-completion (predicting the next character)
- For simplicity we assume that there are only 4 characters in our vocabulary (d,e,p, <stop>)
- At each timestep we want to predict one of these 4 characters
- What is a suitable output function for this task ?



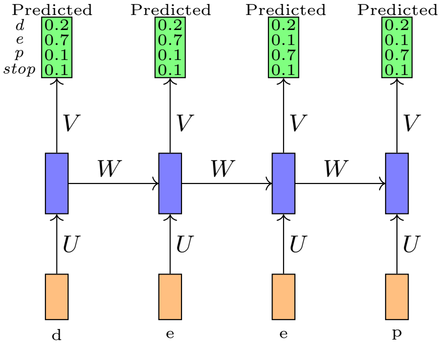
- Suppose we consider our task of auto-completion (predicting the next character)
- For simplicity we assume that there are only 4 characters in our vocabulary (d,e,p, <stop>)
- At each timestep we want to predict one of these 4 characters
- What is a suitable output function for this task ? **(softmax)**



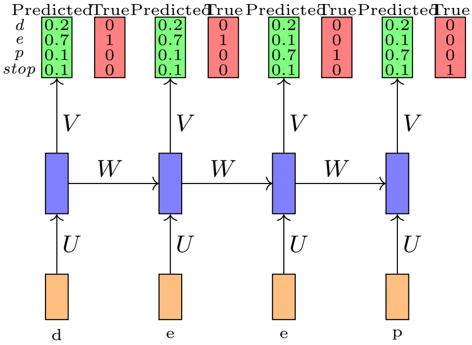
- Suppose we consider our task of auto-completion (predicting the next character)
- For simplicity we assume that there are only 4 characters in our vocabulary (d,e,p, <stop>)
- At each timestep we want to predict one of these 4 characters
- What is a suitable output function for this task ? **(softmax)**
- What is a suitable loss function for this task ?



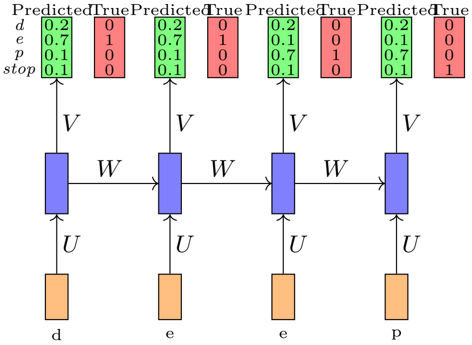
- Suppose we consider our task of auto-completion (predicting the next character)
- For simplicity we assume that there are only 4 characters in our vocabulary (d,e,p, <stop>)
- At each timestep we want to predict one of these 4 characters
- What is a suitable output function for this task ? **(softmax)**
- What is a suitable loss function for this task ? **(cross entropy)**



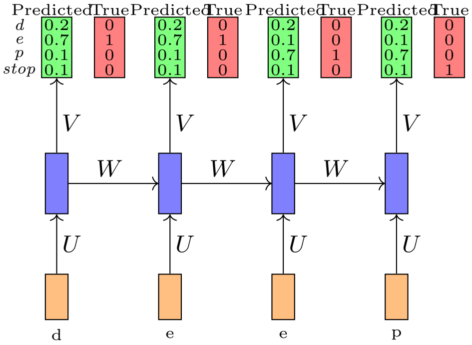
- Suppose we initialize U, V, W randomly and the network predicts the probabilities as shown



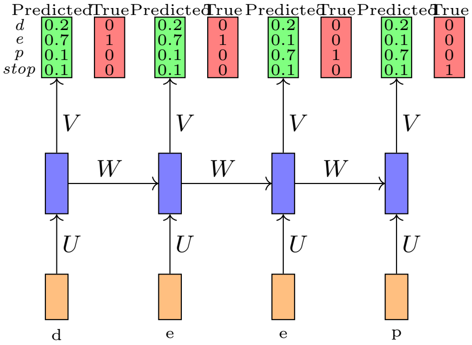
- Suppose we initialize U, V, W randomly and the network predicts the probabilities as shown
- And the true probabilities are as shown



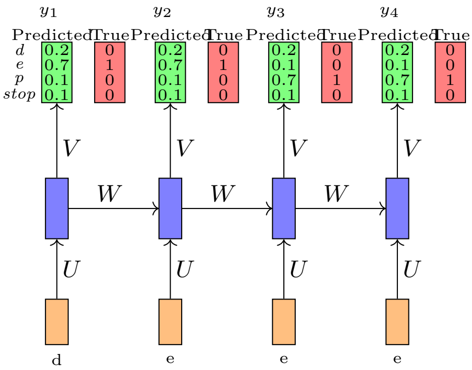
- Suppose we initialize U, V, W randomly and the network predicts the probabilities as shown
- And the true probabilities are as shown
- We need to answer two questions



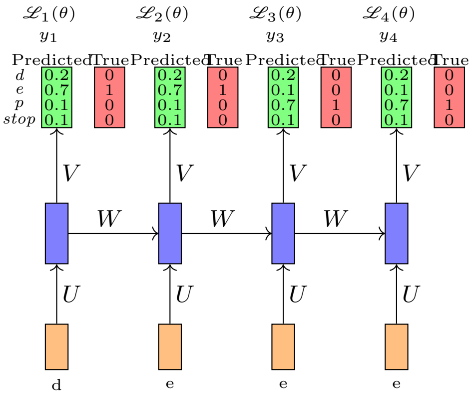
- Suppose we initialize U, V, W randomly and the network predicts the probabilities as shown
- And the true probabilities are as shown
- We need to answer two questions
- What is the total loss made by the model ?



- Suppose we initialize U, V, W randomly and the network predicts the probabilities as shown
- And the true probabilities are as shown
- We need to answer two questions
- What is the total loss made by the model ?
- How do we backpropagate this loss and update the parameters ($\theta = \{U, V, W, b, c\}$) of the network ?

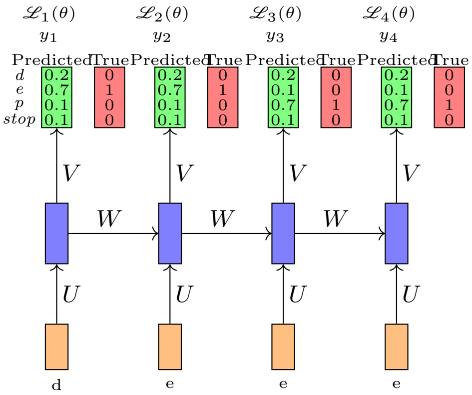


- The total loss is simply the sum of the loss over all time-steps



- The total loss is simply the sum of the loss over all time-steps

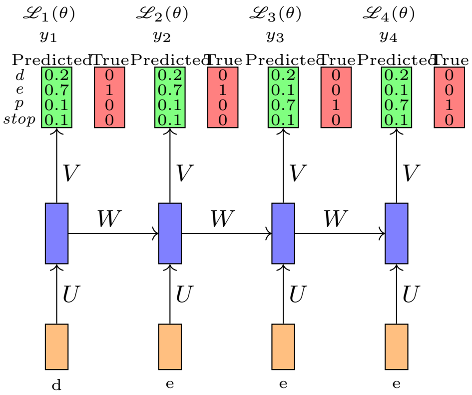
$$\mathscr{L}(\theta) = \sum_{t=1}^{T} \mathscr{L}_t(\theta)$$



- The total loss is simply the sum of the loss over all time-steps

$$\mathscr{L}(\theta) = \sum_{t=1}^{T} \mathscr{L}_t(\theta)$$

$$\mathscr{L}_t(\theta) = -log(y_{tc})$$

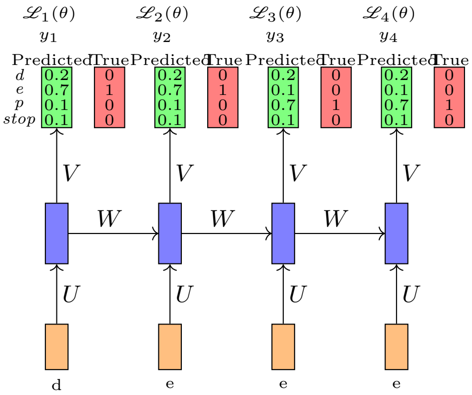


- The total loss is simply the sum of the loss over all time-steps

$$\mathscr{L}(\theta) = \sum_{t=1}^{T} \mathscr{L}_t(\theta)$$

$$\mathscr{L}_t(\theta) = -log(y_{tc})$$

$$y_{tc} = \text{predicted probability of true character at time-step } t$$



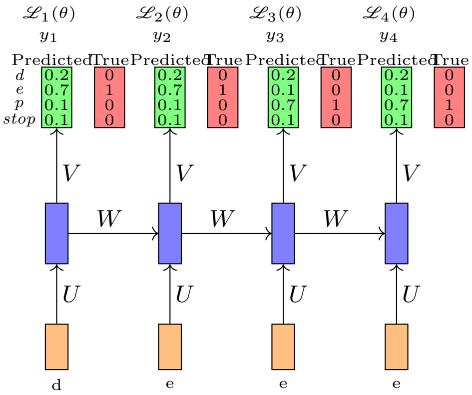
- The total loss is simply the sum of the loss over all time-steps

$$\mathscr{L}(\theta) = \sum_{t=1}^{T} \mathscr{L}_t(\theta)$$

$$\mathscr{L}_t(\theta) = -log(y_{tc})$$

$$y_{tc} = \text{predicted probability of true character at time-step } t$$

$$T = \text{number of timesteps}$$



- The total loss is simply the sum of the loss over all time-steps

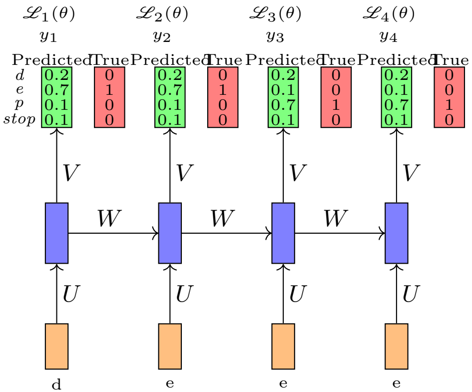
$$\mathscr{L}(\theta) = \sum_{t=1}^{T} \mathscr{L}_t(\theta)$$

$$\mathscr{L}_t(\theta) = -log(y_{tc})$$

$$y_{tc} = \text{predicted probability of true character at time-step } t$$

$$T = \text{number of timesteps}$$

- For backpropagation we need to compute the gradients w.r.t. W, U, V, b, c



- The total loss is simply the sum of the loss over all time-steps

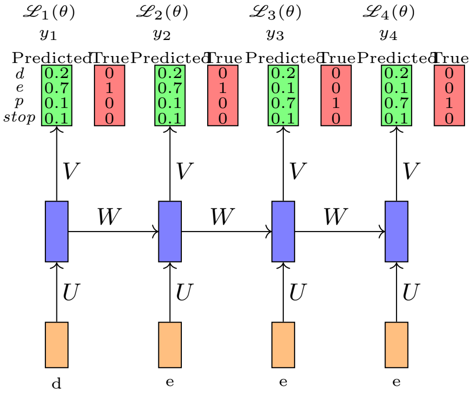
$$\mathscr{L}(\theta) = \sum_{t=1}^{T} \mathscr{L}_t(\theta)$$

$$\mathscr{L}_t(\theta) = -log(y_{tc})$$

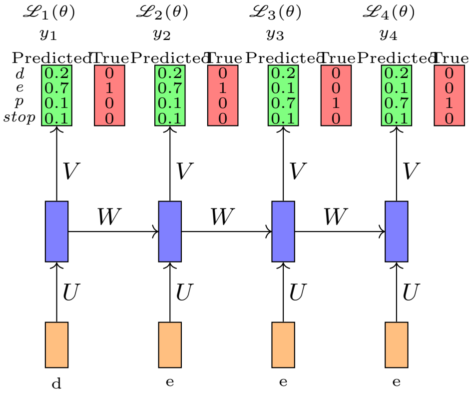
$$y_{tc} = \text{predicted probability of true character at time-step } t$$

$$T = \text{number of timesteps}$$

- For backpropagation we need to compute the gradients w.r.t. W, U, V, b, c
- Let us see how to do that

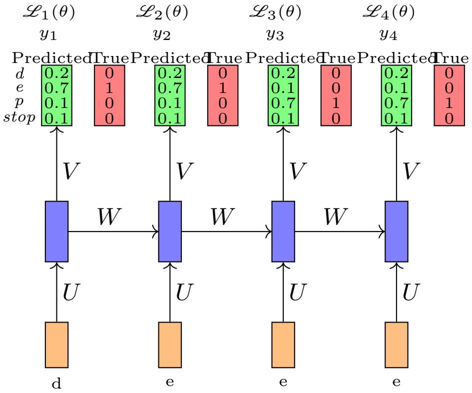


- Let us consider $\frac{\partial \mathscr{L}(\theta)}{\partial V}$ (V is a matrix so ideally we should write $\nabla_v \mathscr{L}(\theta)$)



- Let us consider $\frac{\partial \mathscr{L}(\theta)}{\partial V}$ (V is a matrix so ideally we should write $\nabla_v \mathscr{L}(\theta)$)

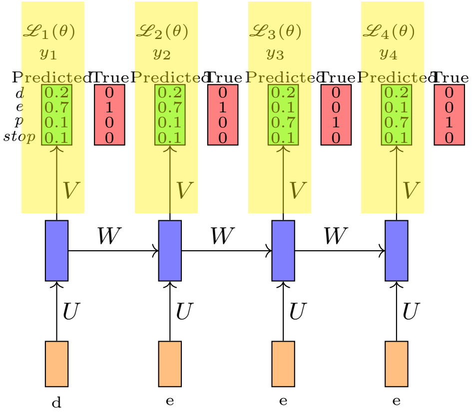
$$\frac{\partial \mathscr{L}(\theta)}{\partial V} = \sum_{t=1}^{T} \frac{\partial \mathscr{L}_t(\theta)}{\partial V}$$



- Let us consider $\frac{\partial \mathscr{L}(\theta)}{\partial V}$ (V is a matrix so ideally we should write $\nabla_v \mathscr{L}(\theta)$)

$$\frac{\partial \mathscr{L}(\theta)}{\partial V} = \sum_{t=1}^{T} \frac{\partial \mathscr{L}_t(\theta)}{\partial V}$$

- Each term is the summation is simply the derivative of the loss w.r.t. the weights in the output layer

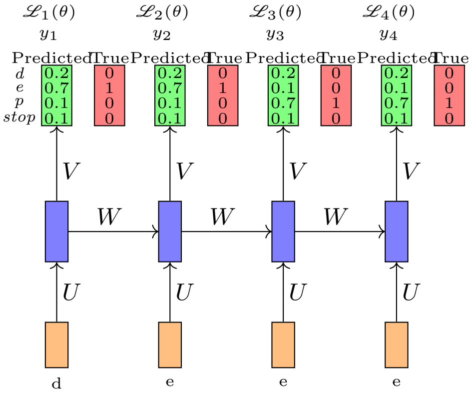


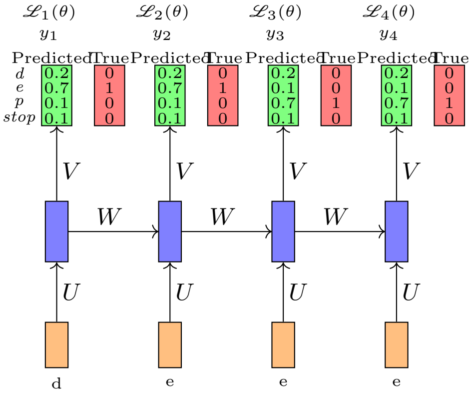
- Let us consider $\frac{\partial \mathscr{L}(\theta)}{\partial V}$ (V is a matrix so ideally we should write $\nabla_v \mathscr{L}(\theta)$)

$$\frac{\partial \mathscr{L}(\theta)}{\partial V} = \sum_{t=1}^{T} \frac{\partial \mathscr{L}_t(\theta)}{\partial V}$$

- Each term is the summation is simply the derivative of the loss w.r.t. the weights in the output layer
- We have already seen how to do this when we studied backpropagation

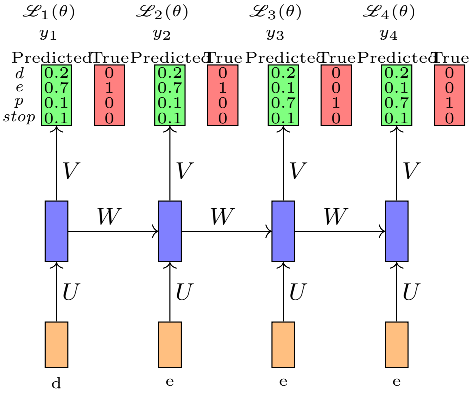
- Let us consider the derivative $\frac{\partial \mathscr{L}(\theta)}{\partial W}$





- Let us consider the derivative $\frac{\partial \mathscr{L}(\theta)}{\partial W}$

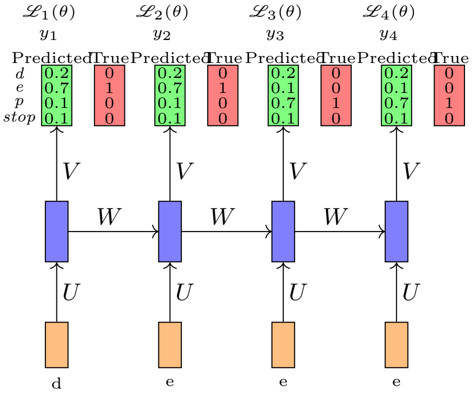
$$\frac{\partial \mathscr{L}(\theta)}{\partial W} = \sum_{t=1}^{T} \frac{\partial \mathscr{L}_t(\theta)}{\partial W}$$



- Let us consider the derivative $\frac{\partial \mathscr{L}(\theta)}{\partial W}$

$$\frac{\partial \mathscr{L}(\theta)}{\partial W} = \sum_{t=1}^{T} \frac{\partial \mathscr{L}_t(\theta)}{\partial W}$$

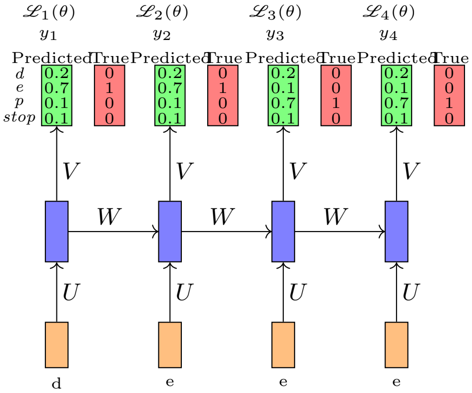
- By the chain rule of derivatives we know that $\frac{\partial \mathscr{L}_t(\theta)}{\partial W}$ is obtained by summing gradients along all the paths from $\mathscr{L}_t(\theta)$ to W



- Let us consider the derivative $\frac{\partial \mathscr{L}(\theta)}{\partial W}$

$$\frac{\partial \mathscr{L}(\theta)}{\partial W} = \sum_{t=1}^{T} \frac{\partial \mathscr{L}_t(\theta)}{\partial W}$$

- By the chain rule of derivatives we know that $\frac{\partial \mathscr{L}_t(\theta)}{\partial W}$ is obtained by summing gradients along all the paths from $\mathscr{L}_t(\theta)$ to W
- What are the paths connecting $\mathscr{L}_t(\theta)$ to W ?

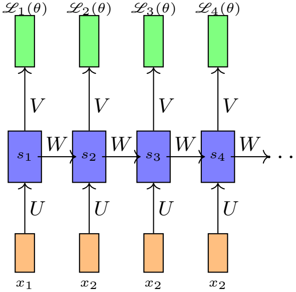


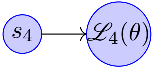
- Let us consider the derivative $\frac{\partial \mathscr{L}(\theta)}{\partial W}$

$$\frac{\partial \mathscr{L}(\theta)}{\partial W} = \sum_{t=1}^{T} \frac{\partial \mathscr{L}_t(\theta)}{\partial W}$$

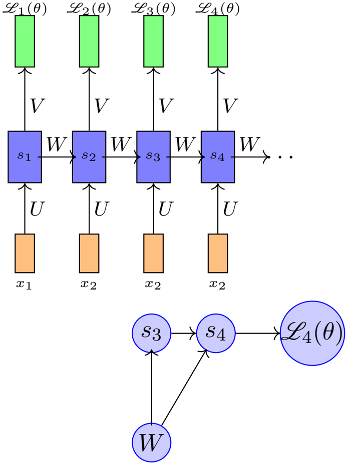
- By the chain rule of derivatives we know that $\frac{\partial \mathscr{L}_t(\theta)}{\partial W}$ is obtained by summing gradients along all the paths from $\mathscr{L}_t(\theta)$ to W
- What are the paths connecting $\mathscr{L}_t(\theta)$ to W ?
- Let us see this by considering $\mathscr{L}_4(\theta)$

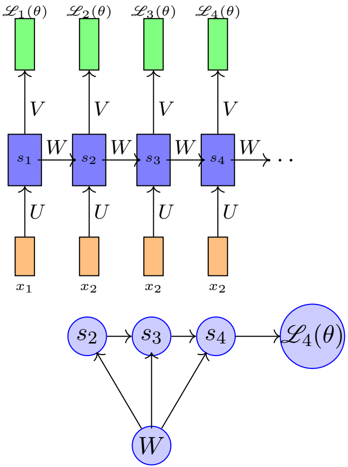
- $\mathscr{L}_4(\theta)$ depends on s_4



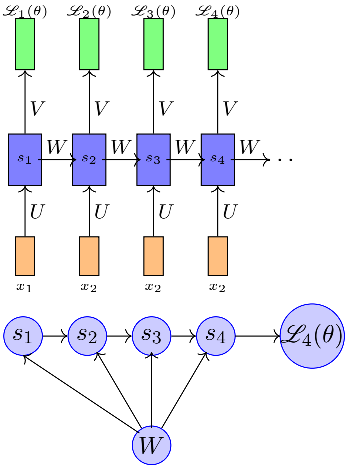


- $\mathscr{L}_4(\theta)$ depends on s_4
- s_4 in turn depends on s_3 and W

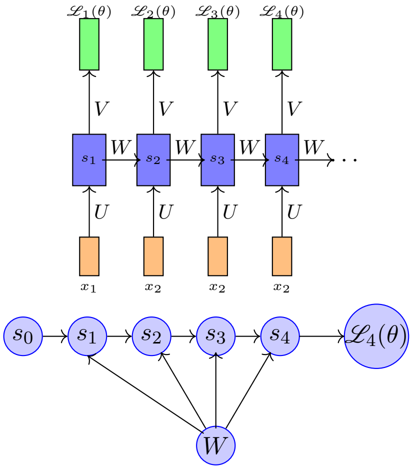




- $\mathscr{L}_4(\theta)$ depends on s_4
- s_4 in turn depends on s_3 and W
- s_3 in turn depends on s_2 and W

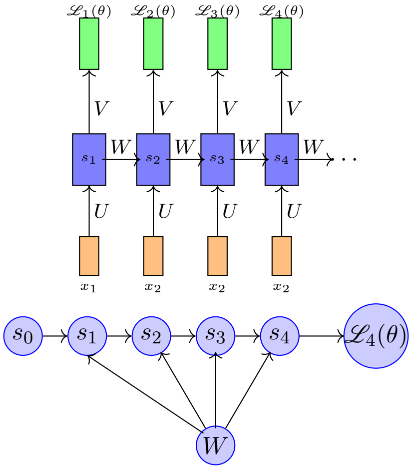


- $\mathscr{L}_4(\theta)$ depends on s_4
- s_4 in turn depends on s_3 and W
- s_3 in turn depends on s_2 and W
- s_2 in turn depends on s_1 and W

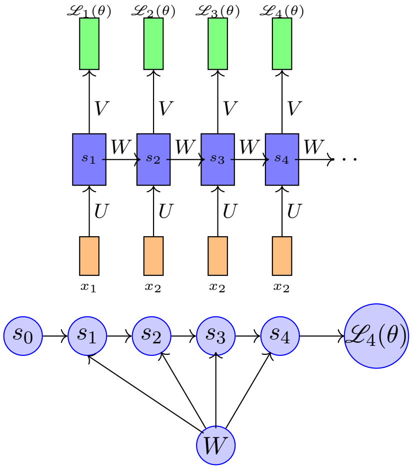


- $\mathscr{L}_4(\theta)$ depends on s_4
- s_4 in turn depends on s_3 and W
- s_3 in turn depends on s_2 and W
- s_2 in turn depends on s_1 and W
- s_1 in turn depends on s_0 and W

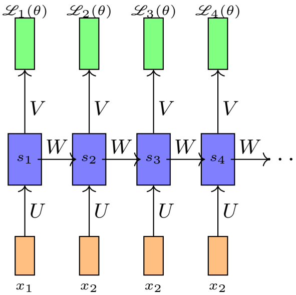
  where s_0 is a constant starting state.

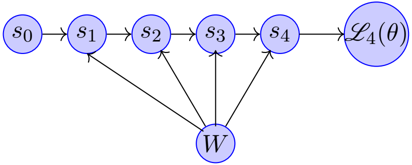


- What we have here is an ordered network



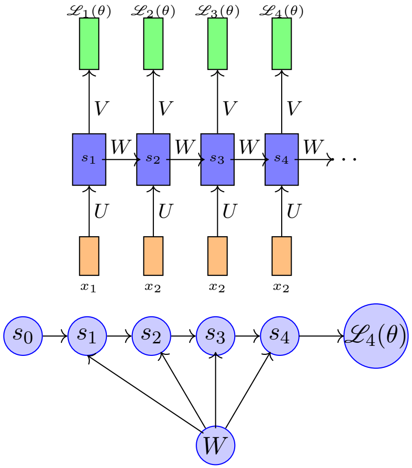
- What we have here is an ordered network
- In an ordered network each state variable is computed one at a time in a specified order (first s_1, then s_2 and so on)





- What we have here is an ordered network
- In an ordered network each state variable is computed one at a time in a specified order (first s_1, then s_2 and so on)
- Now we have

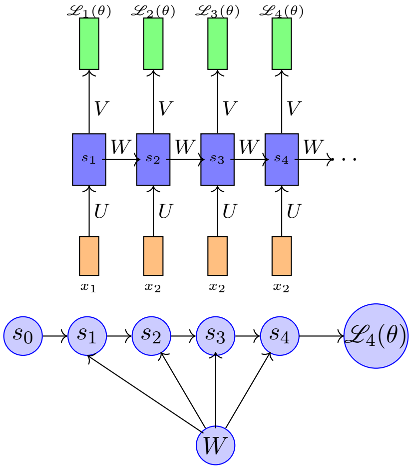
$$\frac{\partial \mathscr{L}_4(\theta)}{\partial W} = \frac{\partial \mathscr{L}_4(\theta)}{\partial s_4} \frac{\partial s_4}{\partial W}$$



- What we have here is an ordered network
- In an ordered network each state variable is computed one at a time in a specified order (first s_1, then s_2 and so on)
- Now we have

$$\frac{\partial \mathscr{L}_4(\theta)}{\partial W} = \frac{\partial \mathscr{L}_4(\theta)}{\partial s_4}\frac{\partial s_4}{\partial W}$$

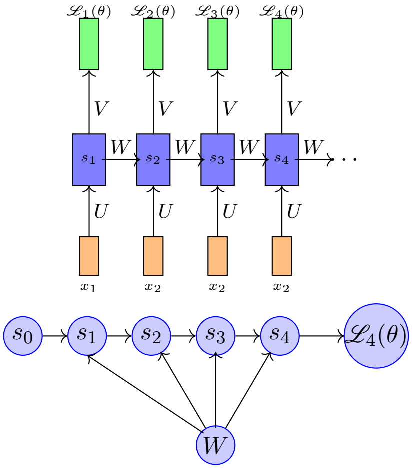
- We have already seen how to compute $\frac{\partial \mathscr{L}_4(\theta)}{\partial s_4}$ when we studied backprop



- What we have here is an ordered network
- In an ordered network each state variable is computed one at a time in a specified order (first s_1, then s_2 and so on)
- Now we have

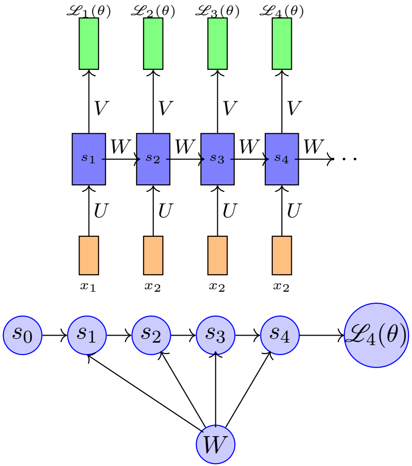
$$\frac{\partial \mathscr{L}_4(\theta)}{\partial W} = \frac{\partial \mathscr{L}_4(\theta)}{\partial s_4} \frac{\partial s_4}{\partial W}$$

- We have already seen how to compute $\frac{\partial \mathscr{L}_4(\theta)}{\partial s_4}$ when we studied backprop
- But how do we compute $\frac{\partial s_4}{\partial W}$



- Recall that

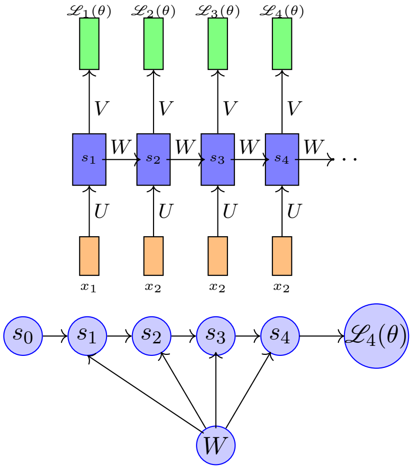
$$s_4 = \sigma(Ws_3 + b)$$



- Recall that

$$s_4 = \sigma(Ws_3 + b)$$

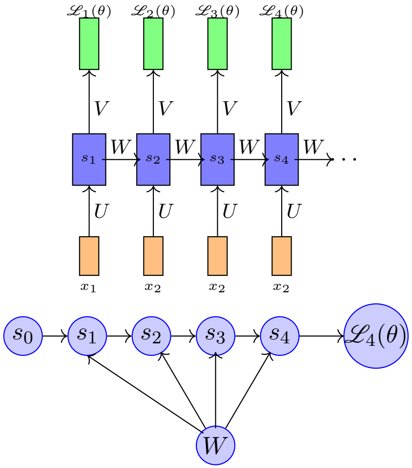
- In such an ordered network, we can't compute $\frac{\partial s_4}{\partial W}$ by simply treating s_3 as a constant (because it also depends on W)



- Recall that

$$s_4 = \sigma(Ws_3 + b)$$

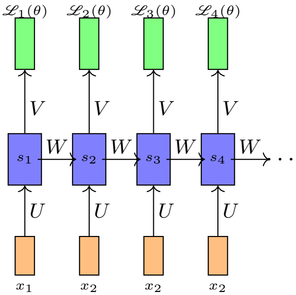
- In such an ordered network, we can't compute $\frac{\partial s_4}{\partial W}$ by simply treating s_3 as a constant (because it also depends on W)
- In such networks the total derivative $\frac{\partial s_4}{\partial W}$ has two parts

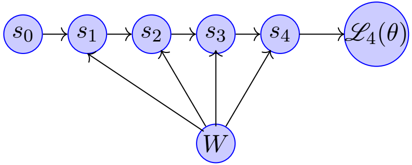


- Recall that

$$s_4 = \sigma(Ws_3 + b)$$

- In such an ordered network, we can't compute $\frac{\partial s_4}{\partial W}$ by simply treating s_3 as a constant (because it also depends on W)
- In such networks the total derivative $\frac{\partial s_4}{\partial W}$ has two parts
- **Explicit** : $\frac{\partial^+ s_4}{\partial W}$, treating all other inputs as constant

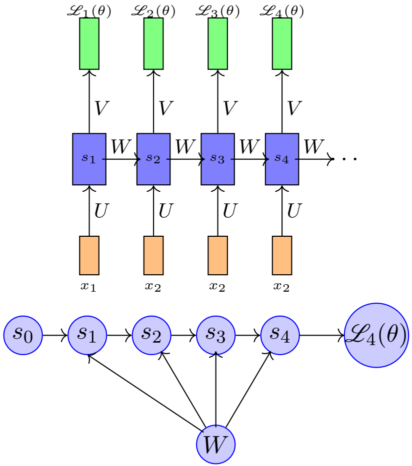




- Recall that

$$s_4 = \sigma(Ws_3 + b)$$

- In such an ordered network, we can't compute $\frac{\partial s_4}{\partial W}$ by simply treating s_3 as a constant (because it also depends on W)
- In such networks the total derivative $\frac{\partial s_4}{\partial W}$ has two parts
- **Explicit** : $\frac{\partial^+ s_4}{\partial W}$, treating all other inputs as constant
- **Implicit** : Summing over all indirect paths from s_4 to W



- Recall that

$$s_4 = \sigma(Ws_3 + b)$$

- In such an ordered network, we can't compute $\frac{\partial s_4}{\partial W}$ by simply treating s_3 as a constant (because it also depends on W)
- In such networks the total derivative $\frac{\partial s_4}{\partial W}$ has two parts
- **Explicit** : $\frac{\partial^+ s_4}{\partial W}$, treating all other inputs as constant
- **Implicit** : Summing over all indirect paths from s_4 to W
- Let us see how to do this

$$\frac{\partial s_4}{\partial W} = \underbrace{\frac{\partial^+ s_4}{\partial W}}_{\text{explicit}} + \underbrace{\frac{\partial s_4}{\partial s_3}\frac{\partial s_3}{\partial W}}_{\text{implicit}}$$

$$\frac{\partial s_4}{\partial W} = \underbrace{\frac{\partial^+ s_4}{\partial W}}_{\text{explicit}} + \underbrace{\frac{\partial s_4}{\partial s_3}\frac{\partial s_3}{\partial W}}_{\text{implicit}}$$

$$= \frac{\partial^+ s_4}{\partial W} + \frac{\partial s_4}{\partial s_3}\left[\underbrace{\frac{\partial^+ s_3}{\partial W}}_{\text{explicit}} + \underbrace{\frac{\partial s_3}{\partial s_2}\frac{\partial s_2}{\partial W}}_{\text{implicit}}\right]$$

$$
\begin{aligned}
\frac{\partial s_4}{\partial W} &= \underbrace{\frac{\partial^+ s_4}{\partial W}}_{\text{explicit}} + \underbrace{\frac{\partial s_4}{\partial s_3}\frac{\partial s_3}{\partial W}}_{\text{implicit}} \\
&= \frac{\partial^+ s_4}{\partial W} + \frac{\partial s_4}{\partial s_3}\left[\underbrace{\frac{\partial^+ s_3}{\partial W}}_{\text{explicit}} + \underbrace{\frac{\partial s_3}{\partial s_2}\frac{\partial s_2}{\partial W}}_{\text{implicit}}\right] \\
&= \frac{\partial^+ s_4}{\partial W} + \frac{\partial s_4}{\partial s_3}\frac{\partial^+ s_3}{\partial W} + \frac{\partial s_4}{\partial s_3}\frac{\partial s_3}{\partial s_2}\left[\frac{\partial^+ s_2}{\partial W} + \frac{\partial s_2}{\partial s_1}\frac{\partial s_1}{\partial W}\right]
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial s_4}{\partial W} &= \underbrace{\frac{\partial^+ s_4}{\partial W}}_{\text{explicit}} + \underbrace{\frac{\partial s_4}{\partial s_3}\frac{\partial s_3}{\partial W}}_{\text{implicit}} \\
&= \frac{\partial^+ s_4}{\partial W} + \frac{\partial s_4}{\partial s_3}\Big[\underbrace{\frac{\partial^+ s_3}{\partial W}}_{\text{explicit}} + \underbrace{\frac{\partial s_3}{\partial s_2}\frac{\partial s_2}{\partial W}}_{\text{implicit}}\Big] \\
&= \frac{\partial^+ s_4}{\partial W} + \frac{\partial s_4}{\partial s_3}\frac{\partial^+ s_3}{\partial W} + \frac{\partial s_4}{\partial s_3}\frac{\partial s_3}{\partial s_2}\Big[\frac{\partial^+ s_2}{\partial W} + \frac{\partial s_2}{\partial s_1}\frac{\partial s_1}{\partial W}\Big] \\
&= \frac{\partial^+ s_4}{\partial W} + \frac{\partial s_4}{\partial s_3}\frac{\partial^+ s_3}{\partial W} + \frac{\partial s_4}{\partial s_3}\frac{\partial s_3}{\partial s_2}\frac{\partial^+ s_2}{\partial W} + \frac{\partial s_4}{\partial s_3}\frac{\partial s_3}{\partial s_2}\frac{\partial s_2}{\partial s_1}\Big[\frac{\partial^+ s_1}{\partial W}\Big]
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial s_4}{\partial W} &= \underbrace{\frac{\partial^+ s_4}{\partial W}}_{\text{explicit}} + \underbrace{\frac{\partial s_4}{\partial s_3}\frac{\partial s_3}{\partial W}}_{\text{implicit}} \\
&= \frac{\partial^+ s_4}{\partial W} + \frac{\partial s_4}{\partial s_3}\left[\underbrace{\frac{\partial^+ s_3}{\partial W}}_{\text{explicit}} + \underbrace{\frac{\partial s_3}{\partial s_2}\frac{\partial s_2}{\partial W}}_{\text{implicit}}\right] \\
&= \frac{\partial^+ s_4}{\partial W} + \frac{\partial s_4}{\partial s_3}\frac{\partial^+ s_3}{\partial W} + \frac{\partial s_4}{\partial s_3}\frac{\partial s_3}{\partial s_2}\left[\frac{\partial^+ s_2}{\partial W} + \frac{\partial s_2}{\partial s_1}\frac{\partial s_1}{\partial W}\right] \\
&= \frac{\partial^+ s_4}{\partial W} + \frac{\partial s_4}{\partial s_3}\frac{\partial^+ s_3}{\partial W} + \frac{\partial s_4}{\partial s_3}\frac{\partial s_3}{\partial s_2}\frac{\partial^+ s_2}{\partial W} + \frac{\partial s_4}{\partial s_3}\frac{\partial s_3}{\partial s_2}\frac{\partial s_2}{\partial s_1}\left[\frac{\partial^+ s_1}{\partial W}\right]
\end{aligned}
$$

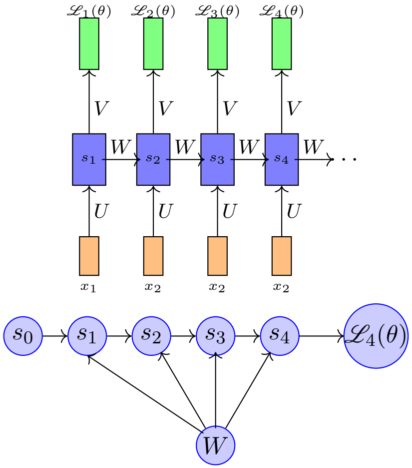
For simplicity we will short-circuit some of the paths

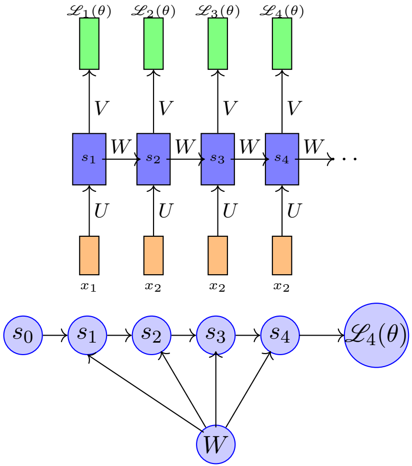
$$
\begin{aligned}
\frac{\partial s_4}{\partial W} &= \underbrace{\frac{\partial^+ s_4}{\partial W}}_{\text{explicit}} + \underbrace{\frac{\partial s_4}{\partial s_3}\frac{\partial s_3}{\partial W}}_{\text{implicit}} \\
&= \frac{\partial^+ s_4}{\partial W} + \frac{\partial s_4}{\partial s_3}\Big[\underbrace{\frac{\partial^+ s_3}{\partial W}}_{\text{explicit}} + \underbrace{\frac{\partial s_3}{\partial s_2}\frac{\partial s_2}{\partial W}}_{\text{implicit}}\Big] \\
&= \frac{\partial^+ s_4}{\partial W} + \frac{\partial s_4}{\partial s_3}\frac{\partial^+ s_3}{\partial W} + \frac{\partial s_4}{\partial s_3}\frac{\partial s_3}{\partial s_2}\Big[\frac{\partial^+ s_2}{\partial W} + \frac{\partial s_2}{\partial s_1}\frac{\partial s_1}{\partial W}\Big] \\
&= \frac{\partial^+ s_4}{\partial W} + \frac{\partial s_4}{\partial s_3}\frac{\partial^+ s_3}{\partial W} + \frac{\partial s_4}{\partial s_3}\frac{\partial s_3}{\partial s_2}\frac{\partial^+ s_2}{\partial W} + \frac{\partial s_4}{\partial s_3}\frac{\partial s_3}{\partial s_2}\frac{\partial s_2}{\partial s_1}\Big[\frac{\partial^+ s_1}{\partial W}\Big]
\end{aligned}
$$

For simplicity we will short-circuit some of the paths

$$
\frac{\partial s_4}{\partial W} = \frac{\partial s_4}{\partial s_4}\frac{\partial^+ s_4}{\partial W} + \frac{\partial s_4}{\partial s_3}\frac{\partial^+ s_3}{\partial W} + \frac{\partial s_4}{\partial s_2}\frac{\partial^+ s_2}{\partial W} + \frac{\partial s_4}{\partial s_1}\frac{\partial^+ s_1}{\partial W} = \sum_{k=1}^{4}\frac{\partial s_4}{\partial s_k}\frac{\partial^+ s_k}{\partial W}
$$

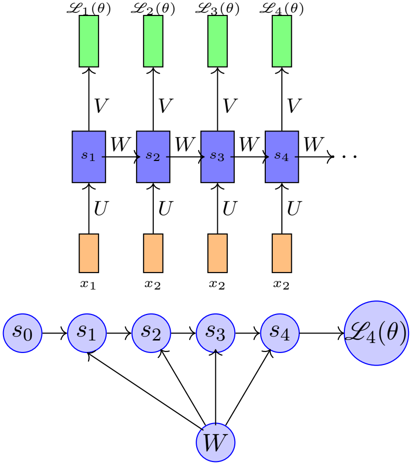
- Finally we have





- Finally we have

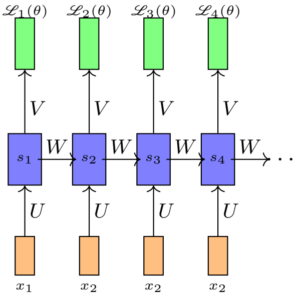
$$\frac{\partial \mathscr{L}_4(\theta)}{\partial W} = \frac{\partial \mathscr{L}_4(\theta)}{\partial s_4} \frac{\partial s_4}{\partial W}$$

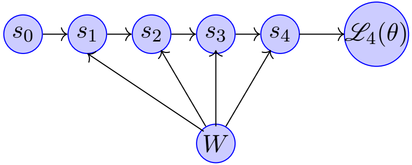


- Finally we have

$$\frac{\partial \mathscr{L}_4(\theta)}{\partial W} = \frac{\partial \mathscr{L}_4(\theta)}{\partial s_4}\frac{\partial s_4}{\partial W}$$

$$\frac{\partial s_4}{\partial W} = \sum_{k=1}^{4} \frac{\partial s_4}{\partial s_k}\frac{\partial^+ s_k}{\partial W}$$



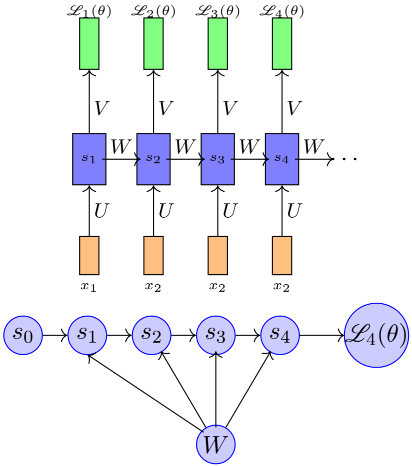


- Finally we have

$$\frac{\partial \mathscr{L}_4(\theta)}{\partial W} = \frac{\partial \mathscr{L}_4(\theta)}{\partial s_4}\frac{\partial s_4}{\partial W}$$

$$\frac{\partial s_4}{\partial W} = \sum_{k=1}^{4}\frac{\partial s_4}{\partial s_k}\frac{\partial^+ s_k}{\partial W}$$

$$\therefore \frac{\partial \mathscr{L}_t(\theta)}{\partial W} = \frac{\partial \mathscr{L}_t(\theta)}{\partial s_t}\sum_{k=1}^{t}\frac{\partial s_t}{\partial s_k}\frac{\partial^+ s_k}{\partial W}$$



- Finally we have

$$\frac{\partial \mathscr{L}_4(\theta)}{\partial W} = \frac{\partial \mathscr{L}_4(\theta)}{\partial s_4}\frac{\partial s_4}{\partial W}$$

$$\frac{\partial s_4}{\partial W} = \sum_{k=1}^{4} \frac{\partial s_4}{\partial s_k}\frac{\partial^+ s_k}{\partial W}$$

$$\therefore \frac{\partial \mathscr{L}_t(\theta)}{\partial W} = \frac{\partial \mathscr{L}_t(\theta)}{\partial s_t}\sum_{k=1}^{t} \frac{\partial s_t}{\partial s_k}\frac{\partial^+ s_k}{\partial W}$$

- This algorithm is called backpropagation through time (BPTT) as we backpropagate over all previous time steps

Module : The problem of Exploding and Vanishing Gradients

- We will now focus on $\frac{\partial s_t}{\partial s_k}$ and highlight an important problem in training RNN's using BPTT

- We will now focus on $\frac{\partial s_t}{\partial s_k}$ and highlight an important problem in training RNN's using BPTT

$$\frac{\partial s_t}{\partial s_k} = \frac{\partial s_t}{\partial s_{t-1}} \frac{\partial s_{t-1}}{\partial s_{t-2}} \cdots \frac{\partial s_{k+1}}{\partial s_k}$$

- We will now focus on $\frac{\partial s_t}{\partial s_k}$ and highlight an important problem in training RNN's using BPTT

$$\frac{\partial s_t}{\partial s_k} = \frac{\partial s_t}{\partial s_{t-1}} \frac{\partial s_{t-1}}{\partial s_{t-2}} \cdots \frac{\partial s_{k+1}}{\partial s_k}$$
$$= \prod_{j=k}^{t-1} \frac{\partial s_{j+1}}{\partial s_j}$$

- We will now focus on $\frac{\partial s_t}{\partial s_k}$ and highlight an important problem in training RNN's using BPTT

$$\frac{\partial s_t}{\partial s_k} = \frac{\partial s_t}{\partial s_{t-1}} \frac{\partial s_{t-1}}{\partial s_{t-2}} \dots \frac{\partial s_{k+1}}{\partial s_k}$$
$$= \prod_{j=k}^{t-1} \frac{\partial s_{j+1}}{\partial s_j}$$

- Let us look at one such term in the product (i.e., $\frac{\partial s_{j+1}}{\partial s_j}$)

- We are interested in $\frac{\partial s_j}{\partial s_{j-1}}$

- We are interested in $\frac{\partial s_j}{\partial s_{j-1}}$

$$a_j = W s_j + b$$

$$s_j = \sigma(a_j)$$

- We are interested in $\frac{\partial s_j}{\partial s_{j-1}}$

$$a_j = W s_j + b$$
$$s_j = \sigma(a_j)$$

$$\frac{\partial s_j}{\partial s_{j-1}} = \frac{\partial s_j}{\partial a_j}\frac{\partial a_j}{\partial s_{j-1}}$$

- We are interested in $\frac{\partial s_j}{\partial s_{j-1}}$

$$a_j = [a_{j1}, a_{j2}, a_{j3}, \ldots a_{jd},]$$
$$s_j = [\sigma(a_{j1}), \sigma(a_{j2}), \ldots \sigma(a_{jd})]$$

$$a_j = W s_j + b$$
$$s_j = \sigma(a_j)$$

$$\frac{\partial s_j}{\partial s_{j-1}} = \frac{\partial s_j}{\partial a_j} \frac{\partial a_j}{\partial s_{j-1}}$$

- We are interested in $\frac{\partial s_j}{\partial s_{j-1}}$

$$a_j = [a_{j1}, a_{j2}, a_{j3}, \ldots a_{jd},]$$
$$s_j = [\sigma(a_{j1}), \sigma(a_{j2}), \ldots \sigma(a_{jd})]$$

$$\frac{\partial s_j}{\partial a_j} = \begin{bmatrix} \\ \\ \\ \end{bmatrix}$$

$$a_j = W s_j + b$$
$$s_j = \sigma(a_j)$$

$$\frac{\partial s_j}{\partial s_{j-1}} = \frac{\partial s_j}{\partial a_j} \frac{\partial a_j}{\partial s_{j-1}}$$

- We are interested in $\frac{\partial s_j}{\partial s_{j-1}}$

$$a_j = [a_{j1}, a_{j2}, a_{j3}, \ldots a_{jd},]$$
$$s_j = [\sigma(a_{j1}), \sigma(a_{j2}), \ldots \sigma(a_{jd})]$$

$$\frac{\partial s_j}{\partial a_j} = \begin{bmatrix} \frac{\partial s_{j1}}{\partial a_{j1}} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix}$$

$$a_j = W s_j + b$$
$$s_j = \sigma(a_j)$$

$$\frac{\partial s_j}{\partial s_{j-1}} = \frac{\partial s_j}{\partial a_j} \frac{\partial a_j}{\partial s_{j-1}}$$

- We are interested in $\frac{\partial s_j}{\partial s_{j-1}}$

$$a_j = Ws_j + b$$
$$s_j = \sigma(a_j)$$

$$\frac{\partial s_j}{\partial s_{j-1}} = \frac{\partial s_j}{\partial a_j}\frac{\partial a_j}{\partial s_{j-1}}$$

$$a_j = [a_{j1}, a_{j2}, a_{j3}, \ldots a_{jd},]$$
$$s_j = [\sigma(a_{j1}), \sigma(a_{j2}), \ldots \sigma(a_{jd})]$$

$$\frac{\partial s_j}{\partial a_j} = \begin{bmatrix} \frac{\partial s_{j1}}{\partial a_{j1}} & \frac{\partial s_{j2}}{\partial a_{j1}} & & \\ & & & \\ & & & \end{bmatrix}$$

- We are interested in $\frac{\partial s_j}{\partial s_{j-1}}$

$$a_j = [a_{j1}, a_{j2}, a_{j3}, \dots a_{jd},]$$
$$s_j = [\sigma(a_{j1}), \sigma(a_{j2}), \dots \sigma(a_{jd})]$$

$$a_j = W s_j + b$$
$$s_j = \sigma(a_j)$$

$$\frac{\partial s_j}{\partial a_j} = \begin{bmatrix} \frac{\partial s_{j1}}{\partial a_{j1}} & \frac{\partial s_{j2}}{\partial a_{j1}} & \frac{\partial s_{j3}}{\partial a_{j1}} & \\ & & & \\ & & & \end{bmatrix}$$

$$\frac{\partial s_j}{\partial s_{j-1}} = \frac{\partial s_j}{\partial a_j} \frac{\partial a_j}{\partial s_{j-1}}$$

- We are interested in $\frac{\partial s_j}{\partial s_{j-1}}$

$$a_j = [a_{j1}, a_{j2}, a_{j3}, \dots a_{jd},]$$
$$s_j = [\sigma(a_{j1}), \sigma(a_{j2}), \dots \sigma(a_{jd})]$$

$$a_j = W s_j + b$$
$$s_j = \sigma(a_j)$$

$$\frac{\partial s_j}{\partial s_{j-1}} = \frac{\partial s_j}{\partial a_j} \frac{\partial a_j}{\partial s_{j-1}}$$

$$\frac{\partial s_j}{\partial a_j} = \begin{bmatrix} \frac{\partial s_{j1}}{\partial a_{j1}} & \frac{\partial s_{j2}}{\partial a_{j1}} & \frac{\partial s_{j3}}{\partial a_{j1}} & \dots \\ & & & \\ & & & \end{bmatrix}$$

- We are interested in $\frac{\partial s_j}{\partial s_{j-1}}$

$$a_j = [a_{j1}, a_{j2}, a_{j3}, \ldots a_{jd},]$$
$$s_j = [\sigma(a_{j1}), \sigma(a_{j2}), \ldots \sigma(a_{jd})]$$

$$a_j = W s_j + b$$
$$s_j = \sigma(a_j)$$

$$\frac{\partial s_j}{\partial s_{j-1}} = \frac{\partial s_j}{\partial a_j} \frac{\partial a_j}{\partial s_{j-1}}$$

$$\frac{\partial s_j}{\partial a_j} = \begin{bmatrix} \frac{\partial s_{j1}}{\partial a_{j1}} & \frac{\partial s_{j2}}{\partial a_{j1}} & \frac{\partial s_{j3}}{\partial a_{j1}} & \cdots \\ \frac{\partial s_{j1}}{\partial a_{j2}} & \frac{\partial s_{j2}}{\partial a_{j2}} & \ddots & \\ & & & \end{bmatrix}$$

- We are interested in $\frac{\partial s_j}{\partial s_{j-1}}$

$$a_j = [a_{j1}, a_{j2}, a_{j3}, \ldots a_{jd},]$$
$$s_j = [\sigma(a_{j1}), \sigma(a_{j2}), \ldots \sigma(a_{jd})]$$

$$a_j = Ws_j + b$$
$$s_j = \sigma(a_j)$$

$$\frac{\partial s_j}{\partial a_j} = \begin{bmatrix} \frac{\partial s_{j1}}{\partial a_{j1}} & \frac{\partial s_{j2}}{\partial a_{j1}} & \frac{\partial s_{j3}}{\partial a_{j1}} & \cdots \\ \frac{\partial s_{j1}}{\partial a_{j2}} & \frac{\partial s_{j2}}{\partial a_{j2}} & \ddots & \\ \vdots & \vdots & \vdots & \frac{\partial s_{jd}}{\partial a_{jd}} \end{bmatrix}$$

$$\frac{\partial s_j}{\partial s_{j-1}} = \frac{\partial s_j}{\partial a_j}\frac{\partial a_j}{\partial s_{j-1}}$$

- We are interested in $\frac{\partial s_j}{\partial s_{j-1}}$

$$a_j = W s_j + b$$
$$s_j = \sigma(a_j)$$

$$\frac{\partial s_j}{\partial s_{j-1}} = \frac{\partial s_j}{\partial a_j} \frac{\partial a_j}{\partial s_{j-1}}$$

$$a_j = [a_{j1}, a_{j2}, a_{j3}, \ldots a_{jd},]$$
$$s_j = [\sigma(a_{j1}), \sigma(a_{j2}), \ldots \sigma(a_{jd})]$$

$$\frac{\partial s_j}{\partial a_j} = \begin{bmatrix} \frac{\partial s_{j1}}{\partial a_{j1}} & \frac{\partial s_{j2}}{\partial a_{j1}} & \frac{\partial s_{j3}}{\partial a_{j1}} & \cdots \\ \frac{\partial s_{j1}}{\partial a_{j2}} & \frac{\partial s_{j2}}{\partial a_{j2}} & \ddots & \\ \vdots & \vdots & \vdots & \frac{\partial s_{jd}}{\partial a_{jd}} \end{bmatrix}$$

$$= \begin{bmatrix} \sigma'(a_{j1}) & 0 & 0 & 0 \\ & & & \\ & & & \end{bmatrix}$$

- We are interested in $\frac{\partial s_j}{\partial s_{j-1}}$

$$a_j = [a_{j1}, a_{j2}, a_{j3}, \dots a_{jd},]$$
$$s_j = [\sigma(a_{j1}), \sigma(a_{j2}), \dots \sigma(a_{jd})]$$

$$a_j = W s_j + b$$
$$s_j = \sigma(a_j)$$

$$\frac{\partial s_j}{\partial s_{j-1}} = \frac{\partial s_j}{\partial a_j} \frac{\partial a_j}{\partial s_{j-1}}$$

$$\frac{\partial s_j}{\partial a_j} = \begin{bmatrix} \frac{\partial s_{j1}}{\partial a_{j1}} & \frac{\partial s_{j2}}{\partial a_{j1}} & \frac{\partial s_{j3}}{\partial a_{j1}} & \dots \\ \frac{\partial s_{j1}}{\partial a_{j2}} & \frac{\partial s_{j2}}{\partial a_{j2}} & \ddots & \\ \vdots & \vdots & \vdots & \frac{\partial s_{jd}}{\partial a_{jd}} \end{bmatrix}$$

$$= \begin{bmatrix} \sigma'(a_{j1}) & 0 & 0 & 0 \\ 0 & \sigma'(a_{j2}) & 0 & 0 \\ 0 & 0 & \ddots & \\ 0 & 0 & \dots & \sigma'(a_{jd}) \end{bmatrix}$$

- We are interested in $\frac{\partial s_j}{\partial s_{j-1}}$

$$a_j = Ws_j + b$$
$$s_j = \sigma(a_j)$$

$$\frac{\partial s_j}{\partial s_{j-1}} = \frac{\partial s_j}{\partial a_j} \frac{\partial a_j}{\partial s_{j-1}}$$

$$a_j = [a_{j1}, a_{j2}, a_{j3}, \dots a_{jd},]$$
$$s_j = [\sigma(a_{j1}), \sigma(a_{j2}), \dots \sigma(a_{jd})]$$

$$\frac{\partial s_j}{\partial a_j} = \begin{bmatrix} \frac{\partial s_{j1}}{\partial a_{j1}} & \frac{\partial s_{j2}}{\partial a_{j1}} & \frac{\partial s_{j3}}{\partial a_{j1}} & \dots \\ \frac{\partial s_{j1}}{\partial a_{j2}} & \frac{\partial s_{j2}}{\partial a_{j2}} & \ddots & \\ \vdots & \vdots & \vdots & \frac{\partial s_{jd}}{\partial a_{jd}} \end{bmatrix}$$
$$= \begin{bmatrix} \sigma'(a_{j1}) & 0 & 0 & 0 \\ 0 & \sigma'(a_{j2}) & 0 & 0 \\ 0 & 0 & \ddots & \\ 0 & 0 & \dots & \sigma'(a_{jd}) \end{bmatrix}$$
$$= diag(\sigma'(a_j))$$

- We are interested in $\frac{\partial s_j}{\partial s_{j-1}}$

$$a_j = [a_{j1}, a_{j2}, a_{j3}, \ldots a_{jd},]$$
$$s_j = [\sigma(a_{j1}), \sigma(a_{j2}), \ldots \sigma(a_{jd})]$$

$$a_j = W s_j + b$$
$$s_j = \sigma(a_j)$$

$$\frac{\partial s_j}{\partial a_j} = \begin{bmatrix} \frac{\partial s_{j1}}{\partial a_{j1}} & \frac{\partial s_{j2}}{\partial a_{j1}} & \frac{\partial s_{j3}}{\partial a_{j1}} & \cdots \\ \frac{\partial s_{j1}}{\partial a_{j2}} & \frac{\partial s_{j2}}{\partial a_{j2}} & \ddots & \\ \vdots & \vdots & \vdots & \frac{\partial s_{jd}}{\partial a_{jd}} \end{bmatrix}$$
$$= \begin{bmatrix} \sigma'(a_{j1}) & 0 & 0 & 0 \\ 0 & \sigma'(a_{j2}) & 0 & 0 \\ 0 & 0 & \ddots & \\ 0 & 0 & \ldots & \sigma'(a_{jd}) \end{bmatrix}$$
$$= diag(\sigma'(a_j))$$

$$\frac{\partial s_j}{\partial s_{j-1}} = \frac{\partial s_j}{\partial a_j} \frac{\partial a_j}{\partial s_{j-1}}$$
$$= diag(\sigma'(a_j))W$$

$$a_j = [a_{j1}, a_{j2}, a_{j3}, \dots a_{jd},]$$
$$s_j = [\sigma(a_{j1}), \sigma(a_{j2}), \dots \sigma(a_{jd})]$$

$$\frac{\partial s_j}{\partial a_j} = \begin{bmatrix} \frac{\partial s_{j1}}{\partial a_{j1}} & \frac{\partial s_{j2}}{\partial a_{j1}} & \frac{\partial s_{j3}}{\partial a_{j1}} & \cdots \\ \frac{\partial s_{j1}}{\partial a_{j2}} & \frac{\partial s_{j2}}{\partial a_{j2}} & \ddots & \\ \vdots & \vdots & \vdots & \frac{\partial s_{jd}}{\partial a_{jd}} \end{bmatrix}$$
$$= \begin{bmatrix} \sigma'(a_{j1}) & 0 & 0 & 0 \\ 0 & \sigma'(a_{j2}) & 0 & 0 \\ 0 & 0 & \ddots & \\ 0 & 0 & \dots & \sigma'(a_{jd}) \end{bmatrix}$$
$$= diag(\sigma'(a_j))$$

- We are interested in $\frac{\partial s_j}{\partial s_{j-1}}$

$$a_j = W s_j + b$$
$$s_j = \sigma(a_j)$$

$$\frac{\partial s_j}{\partial s_{j-1}} = \frac{\partial s_j}{\partial a_j} \frac{\partial a_j}{\partial s_{j-1}}$$
$$= diag(\sigma'(a_j))W$$

- We are interested in the magnitude of $\frac{\partial s_j}{\partial s_{j-1}}$ $\leftarrow$ if it is small (large) $\frac{\partial s_t}{\partial s_k}$ and hence $\frac{\partial \mathscr{L}_t}{\partial W}$ will vanish (explode)

$$\left\|\frac{\partial s_j}{\partial s_{j-1}}\right\| = \left\|diag(\sigma'(a_j))W\right\|$$

$$\left\| \frac{\partial s_j}{\partial s_{j-1}} \right\| = \left\| diag(\sigma^{'}(a_j))W \right\|$$

$$\leq \left\| diag(\sigma^{'}(a_j) \right\| \|W\|$$

$$\left\|\frac{\partial s_j}{\partial s_{j-1}}\right\| = \left\|diag(\sigma^{'}(a_j))W\right\|$$
$$\leq \left\|diag(\sigma^{'}(a_j)\right\| \|W\|$$

$\because \sigma(a_j)$ is a bounded function (sigmoid, tanh) $\sigma^{'}(a_j)$ is bounded

$$\left\|\frac{\partial s_j}{\partial s_{j-1}}\right\| = \left\|diag(\sigma^{'}(a_j))W\right\|$$
$$\leq \left\|diag(\sigma^{'}(a_j)\right\| \|W\|$$

$\because \sigma(a_j)$ is a bounded function (sigmoid, tanh) $\sigma^{'}(a_j)$ is bounded

$$\sigma^{'}(a_j) \leq \frac{1}{4} = \gamma\,[\text{if } \sigma \text{ is logistic }]$$

$$\left\| \frac{\partial s_j}{\partial s_{j-1}} \right\| = \left\| diag(\sigma^{'}(a_j))W \right\|$$
$$\leq \left\| diag(\sigma^{'}(a_j) \right\| \|W\|$$

$\because \sigma(a_j)$ is a bounded function (sigmoid, tanh) $\sigma^{'}(a_j)$ is bounded

$$\sigma^{'}(a_j) \leq \frac{1}{4} = \gamma \,[\text{if } \sigma \text{ is logistic }]$$
$$\leq 1 = \gamma \,[\text{if } \sigma \text{ is tanh }]$$

$$\left\|\frac{\partial s_j}{\partial s_{j-1}}\right\| = \left\|diag(\sigma^{'}(a_j))W\right\|$$
$$\leq \left\|diag(\sigma^{'}(a_j)\right\| \|W\|$$

$\because \sigma(a_j)$ is a bounded function (sigmoid, tanh) $\sigma^{'}(a_j)$ is bounded

$$\sigma^{'}(a_j) \leq \frac{1}{4} = \gamma \,[\text{if } \sigma \text{ is logistic }]$$
$$\leq 1 = \gamma \,[\text{if } \sigma \text{ is tanh }]$$
$$\left\|\frac{\partial s_j}{\partial s_{j-1}}\right\| \leq \gamma \|W\|$$
$$\leq \gamma\lambda$$

$$\left\| \frac{\partial s_j}{\partial s_{j-1}} \right\| = \left\| diag(\sigma^{'}(a_j))W \right\|$$
$$\leq \left\| diag(\sigma^{'}(a_j) \right\| \|W\|$$

$$\left\| \frac{\partial s_t}{\partial s_k} \right\| = \left\| \prod_{j=k+1}^{t} \frac{\partial s_j}{\partial s_{j-1}} \right\|$$

$\because \sigma(a_j)$ is a bounded function (sigmoid, tanh) $\sigma^{'}(a_j)$ is bounded

$$\sigma^{'}(a_j) \leq \frac{1}{4} = \gamma \,[\text{if } \sigma \text{ is logistic }]$$
$$\leq 1 = \gamma \,[\text{if } \sigma \text{ is tanh }]$$
$$\left\| \frac{\partial s_j}{\partial s_{j-1}} \right\| \leq \gamma \|W\|$$
$$\leq \gamma\lambda$$

$$\left\| \frac{\partial s_j}{\partial s_{j-1}} \right\| = \left\| diag(\sigma^{'}(a_j))W \right\|$$
$$\leq \left\| diag(\sigma^{'}(a_j) \right\| \|W\|$$

$\because \sigma(a_j)$ is a bounded function (sigmoid, tanh) $\sigma^{'}(a_j)$ is bounded

$$\sigma^{'}(a_j) \leq \frac{1}{4} = \gamma \text{ [if } \sigma \text{ is logistic]}$$
$$\leq 1 = \gamma \text{ [if } \sigma \text{ is tanh]}$$
$$\left\| \frac{\partial s_j}{\partial s_{j-1}} \right\| \leq \gamma \|W\|$$
$$\leq \gamma\lambda$$

$$\left\| \frac{\partial s_t}{\partial s_k} \right\| = \left\| \prod_{j=k+1}^{t} \frac{\partial s_j}{\partial s_{j-1}} \right\|$$
$$\leq \prod_{j=k+1}^{t} \gamma\lambda$$

$$\left\|\frac{\partial s_j}{\partial s_{j-1}}\right\| = \left\|diag(\sigma^{'}(a_j))W\right\|$$
$$\leq \left\|diag(\sigma^{'}(a_j)\right\| \|W\|$$

$\because \sigma(a_j)$ is a bounded function (sigmoid, tanh) $\sigma^{'}(a_j)$ is bounded

$$\sigma^{'}(a_j) \leq \frac{1}{4} = \gamma \, [\text{if } \sigma \text{ is logistic }]$$
$$\leq 1 = \gamma \, [\text{if } \sigma \text{ is tanh }]$$
$$\left\|\frac{\partial s_j}{\partial s_{j-1}}\right\| \leq \gamma \|W\|$$
$$\leq \gamma\lambda$$

$$\left\|\frac{\partial s_t}{\partial s_k}\right\| = \left\|\prod_{j=k+1}^{t} \frac{\partial s_j}{\partial s_{j-1}}\right\|$$
$$\leq \prod_{j=k+1}^{t} \gamma\lambda$$
$$\leq (\gamma\lambda)^{t-k}$$

$$\left\| \frac{\partial s_j}{\partial s_{j-1}} \right\| = \left\| diag(\sigma^{'}(a_j))W \right\|$$
$$\leq \left\| diag(\sigma^{'}(a_j) \right\| \|W\|$$

$\because \sigma(a_j)$ is a bounded function (sigmoid, tanh) $\sigma^{'}(a_j)$ is bounded

$$\sigma^{'}(a_j) \leq \frac{1}{4} = \gamma \, [\text{if } \sigma \text{ is logistic }]$$
$$\leq 1 = \gamma \, [\text{if } \sigma \text{ is tanh }]$$
$$\left\| \frac{\partial s_j}{\partial s_{j-1}} \right\| \leq \gamma \|W\|$$
$$\leq \gamma\lambda$$

$$\left\| \frac{\partial s_t}{\partial s_k} \right\| = \left\| \prod_{j=k+1}^{t} \frac{\partial s_j}{\partial s_{j-1}} \right\|$$
$$\leq \prod_{j=k+1}^{t} \gamma\lambda$$
$$\leq (\gamma\lambda)^{t-k}$$

- If $\gamma\lambda < 1$ the gradient will vanish

$$\left\| \frac{\partial s_j}{\partial s_{j-1}} \right\| = \left\| diag(\sigma^{'}(a_j))W \right\|$$
$$\leq \left\| diag(\sigma^{'}(a_j) \right\| \|W\|$$

$\because$ $\sigma(a_j)$ is a bounded function (sigmoid, tanh) $\sigma^{'}(a_j)$ is bounded

$$\sigma^{'}(a_j) \leq \frac{1}{4} = \gamma \,[\text{if } \sigma \text{ is logistic }]$$
$$\leq 1 = \gamma \,[\text{if } \sigma \text{ is tanh }]$$

$$\left\| \frac{\partial s_j}{\partial s_{j-1}} \right\| \leq \gamma \|W\|$$
$$\leq \gamma\lambda$$

$$\left\| \frac{\partial s_t}{\partial s_k} \right\| = \left\| \prod_{j=k+1}^{t} \frac{\partial s_j}{\partial s_{j-1}} \right\|$$
$$\leq \prod_{j=k+1}^{t} \gamma\lambda$$
$$\leq (\gamma\lambda)^{t-k}$$

- If $\gamma\lambda < 1$ the gradient will vanish
- If $\gamma\lambda > 1$ the gradient could explode

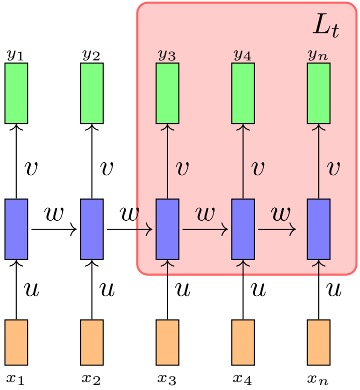
$$\left\| \frac{\partial s_j}{\partial s_{j-1}} \right\| = \left\| diag(\sigma^{'}(a_j))W \right\|$$
$$\leq \left\| diag(\sigma^{'}(a_j) \right\| \|W\|$$

$\because \sigma(a_j)$ is a bounded function (sigmoid, tanh) $\sigma^{'}(a_j)$ is bounded

$$\sigma^{'}(a_j) \leq \frac{1}{4} = \gamma \text{ [if } \sigma \text{ is logistic]}$$
$$\leq 1 = \gamma \text{ [if } \sigma \text{ is tanh]}$$
$$\left\| \frac{\partial s_j}{\partial s_{j-1}} \right\| \leq \gamma \|W\|$$
$$\leq \gamma\lambda$$

$$\left\| \frac{\partial s_t}{\partial s_k} \right\| = \left\| \prod_{j=k+1}^{t} \frac{\partial s_j}{\partial s_{j-1}} \right\|$$
$$\leq \prod_{j=k+1}^{t} \gamma\lambda$$
$$\leq (\gamma\lambda)^{t-k}$$

- If $\gamma\lambda < 1$ the gradient will vanish
- If $\gamma\lambda > 1$ the gradient could explode
- This is known as the problem of vanishing/ exploding gradients



- One simple way of avoiding this is to use truncated backpropogation where we restrict the product to $\tau(< t - k)$ terms

Module : Some Gory Details

$$\underbrace{\frac{\partial \mathscr{L}_t(\theta)}{\partial W}} = \underbrace{\frac{\partial \mathscr{L}_t(\theta)}{\partial s_t}} \sum_{k=1}^{t} \underbrace{\frac{\partial s_t}{\partial s_k}} \quad \underbrace{\frac{\partial^+ s_k}{\partial W}}$$

$$\underbrace{\frac{\partial \mathscr{L}_t(\theta)}{\partial W}}_{\in \mathbb{R}^{d \times d}} = \underbrace{\frac{\partial \mathscr{L}_t(\theta)}{\partial s_t}} \sum_{k=1}^{t} \underbrace{\frac{\partial s_t}{\partial s_k}} \quad \underbrace{\frac{\partial^+ s_k}{\partial W}}$$

$$\underbrace{\frac{\partial \mathscr{L}_t(\theta)}{\partial W}}_{\in \mathbb{R}^{d\times d}} = \underbrace{\frac{\partial \mathscr{L}_t(\theta)}{\partial s_t}}_{\in \mathbb{R}^{1\times d}} \sum_{k=1}^{t} \underbrace{\frac{\partial s_t}{\partial s_k}} \quad \underbrace{\frac{\partial^+ s_k}{\partial W}}$$

$$\underbrace{\frac{\partial \mathscr{L}_t(\theta)}{\partial W}}_{\in \mathbb{R}^{d\times d}} = \underbrace{\frac{\partial \mathscr{L}_t(\theta)}{\partial s_t}}_{\in \mathbb{R}^{1\times d}} \sum_{k=1}^{t} \underbrace{\frac{\partial s_t}{\partial s_k}}_{\in \mathbb{R}^{d\times d}} \underbrace{\frac{\partial^+ s_k}{\partial W}}$$

$$\underbrace{\frac{\partial \mathscr{L}_t(\theta)}{\partial W}}_{\in \mathbb{R}^{d \times d}} = \underbrace{\frac{\partial \mathscr{L}_t(\theta)}{\partial s_t}}_{\in \mathbb{R}^{1 \times d}} \sum_{k=1}^{t} \underbrace{\frac{\partial s_t}{\partial s_k}}_{\in \mathbb{R}^{d \times d}} \underbrace{\frac{\partial^+ s_k}{\partial W}}_{\in \mathbb{R}^{d \times d \times d}}$$

$$\underbrace{\frac{\partial \mathscr{L}_t(\theta)}{\partial W}}_{\in \mathbb{R}^{d \times d}} = \underbrace{\frac{\partial \mathscr{L}_t(\theta)}{\partial s_t}}_{\in \mathbb{R}^{1 \times d}} \sum_{k=1}^{t} \underbrace{\frac{\partial s_t}{\partial s_k}}_{\in \mathbb{R}^{d \times d}} \underbrace{\frac{\partial^+ s_k}{\partial W}}_{\in \mathbb{R}^{d \times d \times d}}$$

- We know how to compute $\frac{\partial \mathscr{L}_t(\theta)}{\partial s_t}$ (derivative of $\mathscr{L}_t(\theta)$ (scalar) w.r.t. last hidden layer (vector)) using backpropagation

$$\underbrace{\frac{\partial \mathscr{L}_t(\theta)}{\partial W}}_{\in \mathbb{R}^{d\times d}} = \underbrace{\frac{\partial \mathscr{L}_t(\theta)}{\partial s_t}}_{\in \mathbb{R}^{1\times d}} \sum_{k=1}^{t} \underbrace{\frac{\partial s_t}{\partial s_k}}_{\in \mathbb{R}^{d\times d}} \underbrace{\frac{\partial^+ s_k}{\partial W}}_{\in \mathbb{R}^{d\times d\times d}}$$

- We know how to compute $\frac{\partial \mathscr{L}_t(\theta)}{\partial s_t}$ (derivative of $\mathscr{L}_t(\theta)$ (scalar) w.r.t. last hidden layer (vector)) using backpropagation
- We just saw a formula for $\frac{\partial s_t}{\partial s_k}$ which is the derivative of a vector w.r.t. a vector)

$$\underbrace{\frac{\partial \mathscr{L}_t(\theta)}{\partial W}}_{\in \mathbb{R}^{d\times d}} = \underbrace{\frac{\partial \mathscr{L}_t(\theta)}{\partial s_t}}_{\in \mathbb{R}^{1\times d}} \sum_{k=1}^{t} \underbrace{\frac{\partial s_t}{\partial s_k}}_{\in \mathbb{R}^{d\times d}} \underbrace{\frac{\partial^+ s_k}{\partial W}}_{\in \mathbb{R}^{d\times d\times d}}$$

- We know how to compute $\frac{\partial \mathscr{L}_t(\theta)}{\partial s_t}$ (derivative of $\mathscr{L}_t(\theta)$ (scalar) w.r.t. last hidden layer (vector)) using backpropagation
- We just saw a formula for $\frac{\partial s_t}{\partial s_k}$ which is the derivative of a vector w.r.t. a vector)
- $\frac{\partial^+ s_k}{\partial W}$ is a tensor $\in \mathbb{R}^{d\times d\times d}$, the derivative of a vector $\in \mathbb{R}^{d}$ w.r.t. a matrix $\in \mathbb{R}^{d\times d}$

$$\underbrace{\frac{\partial \mathscr{L}_t(\theta)}{\partial W}}_{\in \mathbb{R}^{d\times d}} = \underbrace{\frac{\partial \mathscr{L}_t(\theta)}{\partial s_t}}_{\in \mathbb{R}^{1\times d}} \sum_{k=1}^{t} \underbrace{\frac{\partial s_t}{\partial s_k}}_{\in \mathbb{R}^{d\times d}} \underbrace{\frac{\partial^+ s_k}{\partial W}}_{\in \mathbb{R}^{d\times d\times d}}$$

- We know how to compute $\frac{\partial \mathscr{L}_t(\theta)}{\partial s_t}$ (derivative of $\mathscr{L}_t(\theta)$ (scalar) w.r.t. last hidden layer (vector)) using backpropagation
- We just saw a formula for $\frac{\partial s_t}{\partial s_k}$ which is the derivative of a vector w.r.t. a vector)
- $\frac{\partial^+ s_k}{\partial W}$ is a tensor $\in \mathbb{R}^{d\times d\times d}$, the derivative of a vector $\in \mathbb{R}^d$ w.r.t. a matrix $\in \mathbb{R}^{d\times d}$
- How do we compute $\frac{\partial^+ s_k}{\partial W}$? Let us see

- We just look at one element of this $\frac{\partial^+ s_k}{\partial W}$ tensor

- We just look at one element of this $\frac{\partial^+ s_k}{\partial W}$ tensor
- $\frac{\partial^+ s_{kp}}{\partial W_{qr}}$ is the (p, q, r)-th element of the 3d tensor

- We just look at one element of this $\frac{\partial^+ s_k}{\partial W}$ tensor
- $\frac{\partial^+ s_{kp}}{\partial W_{qr}}$ is the (p, q, r)-th element of the 3d tensor

  $a_k = W s_{k-1} + b$

- We just look at one element of this $\frac{\partial^+ s_k}{\partial W}$ tensor
- $\frac{\partial^+ s_{kp}}{\partial W_{qr}}$ is the (p, q, r)-th element of the 3d tensor

$$a_k = W s_{k-1} + b$$

$$s_k = \sigma(a_k)$$

$$a_k = W s_{k-1}$$

$$a_k = W s_{k-1}$$

$$\begin{bmatrix} a_{k1} \\ a_{k2} \\ \vdots \\ a_{kp} \\ \vdots \\ a_{kd} \end{bmatrix} = \begin{bmatrix} W_{11} & W_{12} & \dots & W_{1d} \\ \\ \vdots & \vdots & \vdots & \vdots \\ W_{p1} & W_{p2} & \dots & W_{pd} \\ \vdots & \vdots & \vdots & \vdots \\ \\ \end{bmatrix} \begin{bmatrix} s_{k-1,1} \\ s_{k-1,2} \\ \vdots \\ s_{k-1,p} \\ \vdots \\ s_{k-1,d} \end{bmatrix}$$

$$a_k = W s_{k-1}$$

$$\begin{bmatrix} a_{k1} \\ a_{k2} \\ \vdots \\ a_{kp} \\ \vdots \\ a_{kd} \end{bmatrix} = \begin{bmatrix} W_{11} & W_{12} & \dots & W_{1d} \\ \\ \vdots & \vdots & \vdots & \vdots \\ W_{p1} & W_{p2} & \dots & W_{pd} \\ \vdots & \vdots & \vdots & \vdots \\ \\ \end{bmatrix} \begin{bmatrix} s_{k-1,1} \\ s_{k-1,2} \\ \vdots \\ s_{k-1,p} \\ \vdots \\ s_{k-1,d} \end{bmatrix}$$

$$a_{kp} = \sum_{i=1}^{d} W_{pi} s_{k-1,i}$$

$$a_k = W s_{k-1}$$

$$\begin{bmatrix} a_{k1} \\ a_{k2} \\ \vdots \\ a_{kp} \\ \vdots \\ a_{kd} \end{bmatrix} = \begin{bmatrix} W_{11} & W_{12} & \dots & W_{1d} \\ \vdots & \vdots & \vdots & \vdots \\ W_{p1} & W_{p2} & \dots & W_{pd} \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix} \begin{bmatrix} s_{k-1,1} \\ s_{k-1,2} \\ \vdots \\ s_{k-1,p} \\ \vdots \\ s_{k-1,d} \end{bmatrix}$$

$$a_{kp} = \sum_{i=1}^{d} W_{pi} s_{k-1,i}$$

$$s_{kp} = \sigma(a_{kp})$$

$$a_k = W s_{k-1}$$

$$\begin{bmatrix} a_{k1} \\ a_{k2} \\ \vdots \\ a_{kp} \\ \vdots \\ a_{kd} \end{bmatrix} = \begin{bmatrix} W_{11} & W_{12} & \dots & W_{1d} \\ \vdots & \vdots & \vdots & \vdots \\ W_{p1} & W_{p2} & \dots & W_{pd} \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix} \begin{bmatrix} s_{k-1,1} \\ s_{k-1,2} \\ \vdots \\ s_{k-1,p} \\ \vdots \\ s_{k-1,d} \end{bmatrix}$$

$$a_{kp} = \sum_{i=1}^{d} W_{pi} s_{k-1,i}$$

$$s_{kp} = \sigma(a_{kp})$$

$$\frac{\partial s_{kp}}{\partial W_{qr}} = \frac{\partial s_{kp}}{\partial a_{kp}} \frac{\partial a_{kp}}{\partial W_{qr}}$$

$$a_k = W s_{k-1}$$

$$\begin{bmatrix} a_{k1} \\ a_{k2} \\ \vdots \\ a_{kp} \\ \vdots \\ a_{kd} \end{bmatrix} = \begin{bmatrix} W_{11} & W_{12} & \dots & W_{1d} \\ & & & \\ \vdots & \vdots & \vdots & \vdots \\ W_{p1} & W_{p2} & \dots & W_{pd} \\ \vdots & \vdots & \vdots & \vdots \\ & & & \end{bmatrix} \begin{bmatrix} s_{k-1,1} \\ s_{k-1,2} \\ \vdots \\ s_{k-1,p} \\ \vdots \\ s_{k-1,d} \end{bmatrix}$$

$$a_{kp} = \sum_{i=1}^{d} W_{pi} s_{k-1,i}$$

$$s_{kp} = \sigma(a_{kp})$$

$$\frac{\partial s_{kp}}{\partial W_{qr}} = \frac{\partial s_{kp}}{\partial a_{kp}} \frac{\partial a_{kp}}{\partial W_{qr}}$$

$$= \sigma'(a_{kp}) \frac{\partial a_{kp}}{\partial W_{qr}}$$

$$a_k = W s_{k-1}$$

$$\begin{bmatrix} a_{k1} \\ a_{k2} \\ \vdots \\ a_{kp} \\ \vdots \\ a_{kd} \end{bmatrix} = \begin{bmatrix} W_{11} & W_{12} & \dots & W_{1d} \\ \vdots & \vdots & \vdots & \vdots \\ W_{p1} & W_{p2} & \dots & W_{pd} \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix} \begin{bmatrix} s_{k-1,1} \\ s_{k-1,2} \\ \vdots \\ s_{k-1,p} \\ \vdots \\ s_{k-1,d} \end{bmatrix}$$

$$a_{kp} = \sum_{i=1}^{d} W_{pi} s_{k-1,i}$$

$$s_{kp} = \sigma(a_{kp})$$

$$\frac{\partial s_{kp}}{\partial W_{qr}} = \frac{\partial s_{kp}}{\partial a_{kp}} \frac{\partial a_{kp}}{\partial W_{qr}}$$

$$= \sigma'(a_{kp}) \frac{\partial a_{kp}}{\partial W_{qr}}$$

$$\frac{\partial a_{kp}}{\partial W_{qr}} = \frac{\partial \sum_{i=1}^{d} W_{pi} s_{k-1,i}}{\partial W_{qr}}$$

$$a_k = W s_{k-1}$$

$$\begin{bmatrix} a_{k1} \\ a_{k2} \\ \vdots \\ a_{kp} \\ \vdots \\ a_{kd} \end{bmatrix} = \begin{bmatrix} W_{11} & W_{12} & \dots & W_{1d} \\ \vdots & \vdots & \vdots & \vdots \\ W_{p1} & W_{p2} & \dots & W_{pd} \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix} \begin{bmatrix} s_{k-1,1} \\ s_{k-1,2} \\ \vdots \\ s_{k-1,p} \\ \vdots \\ s_{k-1,d} \end{bmatrix}$$

$$a_{kp} = \sum_{i=1}^{d} W_{pi} s_{k-1,i}$$

$$s_{kp} = \sigma(a_{kp})$$

$$\frac{\partial s_{kp}}{\partial W_{qr}} = \frac{\partial s_{kp}}{\partial a_{kp}} \frac{\partial a_{kp}}{\partial W_{qr}}$$

$$= \sigma'(a_{kp}) \frac{\partial a_{kp}}{\partial W_{qr}}$$

$$\frac{\partial a_{kp}}{\partial W_{qr}} = \frac{\partial \sum_{i=1}^{d} W_{pi} s_{k-1,i}}{\partial W_{qr}}$$

$$= s_{k-1,i} \quad \text{if} \quad p = q \quad \text{and} \quad i = r$$

$$a_k = W s_{k-1}$$

$$\begin{bmatrix} a_{k1} \\ a_{k2} \\ \vdots \\ a_{kp} \\ \vdots \\ a_{kd} \end{bmatrix} = \begin{bmatrix} W_{11} & W_{12} & \dots & W_{1d} \\ & & & \\ \vdots & \vdots & \vdots & \vdots \\ W_{p1} & W_{p2} & \dots & W_{pd} \\ \vdots & \vdots & \vdots & \vdots \\ & & & \end{bmatrix} \begin{bmatrix} s_{k-1,1} \\ s_{k-1,2} \\ \vdots \\ s_{k-1,p} \\ \vdots \\ s_{k-1,d} \end{bmatrix}$$

$$a_{kp} = \sum_{i=1}^{d} W_{pi} s_{k-1,i}$$

$$s_{kp} = \sigma(a_{kp})$$

$$\frac{\partial s_{kp}}{\partial W_{qr}} = \frac{\partial s_{kp}}{\partial a_{kp}} \frac{\partial a_{kp}}{\partial W_{qr}}$$

$$= \sigma'(a_{kp}) \frac{\partial a_{kp}}{\partial W_{qr}}$$

$$\frac{\partial a_{kp}}{\partial W_{qr}} = \frac{\partial \sum_{i=1}^{d} W_{pi} s_{k-1,i}}{\partial W_{qr}}$$

$$= s_{k-1,i} \quad \text{if} \quad p = q \quad \text{and} \quad i = r$$

$$= 0 \quad \text{otherwise}$$

$$a_k = W s_{k-1}$$

$$\begin{bmatrix} a_{k1} \\ a_{k2} \\ \vdots \\ a_{kp} \\ \vdots \\ a_{kd} \end{bmatrix} = \begin{bmatrix} W_{11} & W_{12} & \dots & W_{1d} \\ \\ \vdots & \vdots & \vdots & \vdots \\ W_{p1} & W_{p2} & \dots & W_{pd} \\ \vdots & \vdots & \vdots & \vdots \\ \\ \end{bmatrix} \begin{bmatrix} s_{k-1,1} \\ s_{k-1,2} \\ \vdots \\ s_{k-1,p} \\ \vdots \\ s_{k-1,d} \end{bmatrix}$$

$$a_{kp} = \sum_{i=1}^{d} W_{pi} s_{k-1,i}$$

$$s_{kp} = \sigma(a_{kp})$$

$$\frac{\partial s_{kp}}{\partial W_{qr}} = \frac{\partial s_{kp}}{\partial a_{kp}} \frac{\partial a_{kp}}{\partial W_{qr}}$$

$$= \sigma'(a_{kp}) \frac{\partial a_{kp}}{\partial W_{qr}}$$

$$\frac{\partial a_{kp}}{\partial W_{qr}} = \frac{\partial \sum_{i=1}^{d} W_{pi} s_{k-1,i}}{\partial W_{qr}}$$

$$= s_{k-1,i} \quad \text{if} \quad p = q \quad \text{and} \quad i = r$$

$$= 0 \quad \text{otherwise}$$

$$\frac{\partial s_{kp}}{\partial W_{qr}} = \sigma'(a_{kp}) s_{k-1,r} \quad \text{if} \quad p = q \quad \text{and} \quad i = r$$

$$a_k = W s_{k-1}$$

$$\begin{bmatrix} a_{k1} \\ a_{k2} \\ \vdots \\ a_{kp} \\ \vdots \\ a_{kd} \end{bmatrix} = \begin{bmatrix} W_{11} & W_{12} & \dots & W_{1d} \\ \vdots & \vdots & \vdots & \vdots \\ W_{p1} & W_{p2} & \dots & W_{pd} \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix} \begin{bmatrix} s_{k-1,1} \\ s_{k-1,2} \\ \vdots \\ s_{k-1,p} \\ \vdots \\ s_{k-1,d} \end{bmatrix}$$

$$a_{kp} = \sum_{i=1}^{d} W_{pi} s_{k-1,i}$$

$$s_{kp} = \sigma(a_{kp})$$

$$\frac{\partial s_{kp}}{\partial W_{qr}} = \frac{\partial s_{kp}}{\partial a_{kp}} \frac{\partial a_{kp}}{\partial W_{qr}}$$

$$= \sigma'(a_{kp}) \frac{\partial a_{kp}}{\partial W_{qr}}$$

$$\frac{\partial a_{kp}}{\partial W_{qr}} = \frac{\partial \sum_{i=1}^{d} W_{pi} s_{k-1,i}}{\partial W_{qr}}$$

$$= s_{k-1,i} \quad \text{if} \quad p = q \quad \text{and} \quad i = r$$

$$= 0 \quad \text{otherwise}$$

$$\frac{\partial s_{kp}}{\partial W_{qr}} = \sigma'(a_{kp}) s_{k-1,r} \quad \text{if} \quad p = q \quad \text{and} \quad i = r$$

$$= 0 \quad \text{otherwise}$$