Advanced Computer Vision

Tanveer Hussain
htanveer3797@gmail.com

جامعة الملك عبدالله
للعلوم والتقنية
King Abdullah University of
Science and Technology

KAUST Academy
King Abdullah University of Science and Technology

Course designed by **Naeem Ullah Khan** (naeemullah.khan@kaust.edu.sa), updated by Tanveer Hussain

Computer Vision Applications

Image Classification

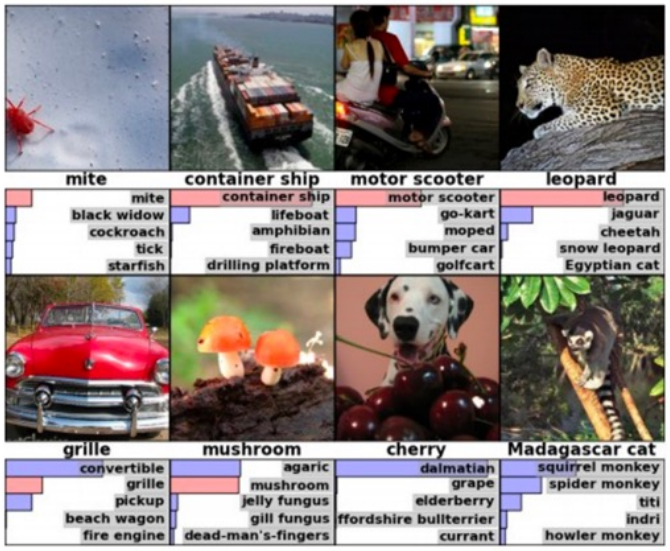
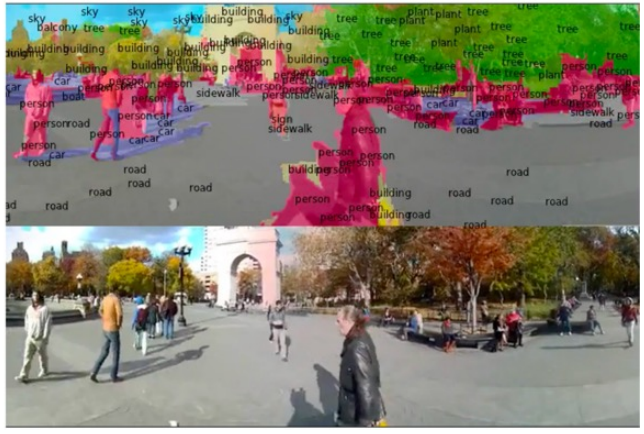


Image Retrieval

Image Segmentation



Fully Connected Neural Networks

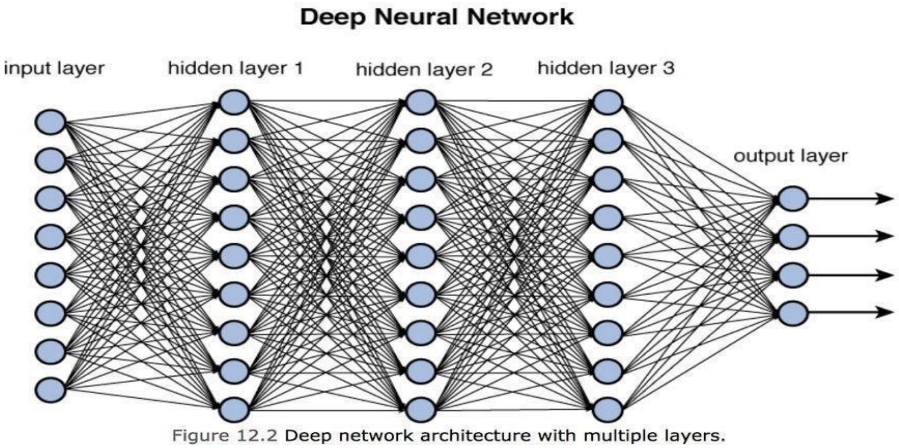
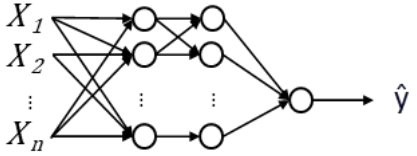


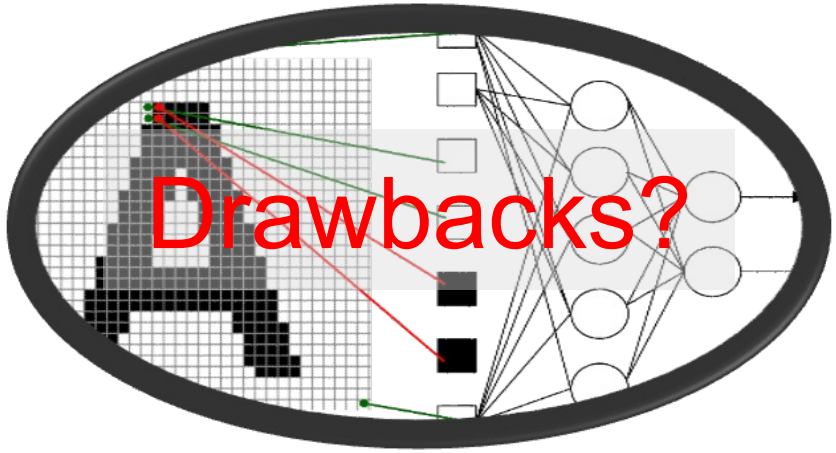
Figure 12.2 Deep network architecture with multiple layers.

64x64x3

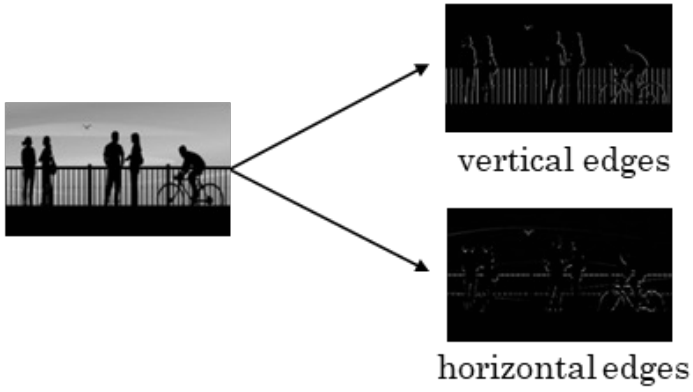
Cat? (0/1)

1000x1000x3





Features Extraction Techniques



3	0	1	2	7	4
1	5	8	9	3	1
2	7	2	5	1	3
0	1	3	1	7	8
4	2	1	6	2	8
2	4	5	2	3	9

* =

10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0

*

1	0	-1
1	0	-1
1	0	-1

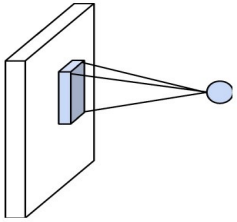
=

0	30	30	0
0	30	30	0
0	30	30	0
0	30	30	0

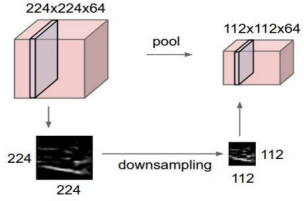
*

Components of CNNs

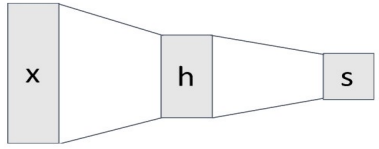
Convolution Layers



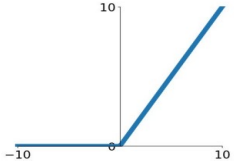
Pooling Layers



Fully-Connected Layers



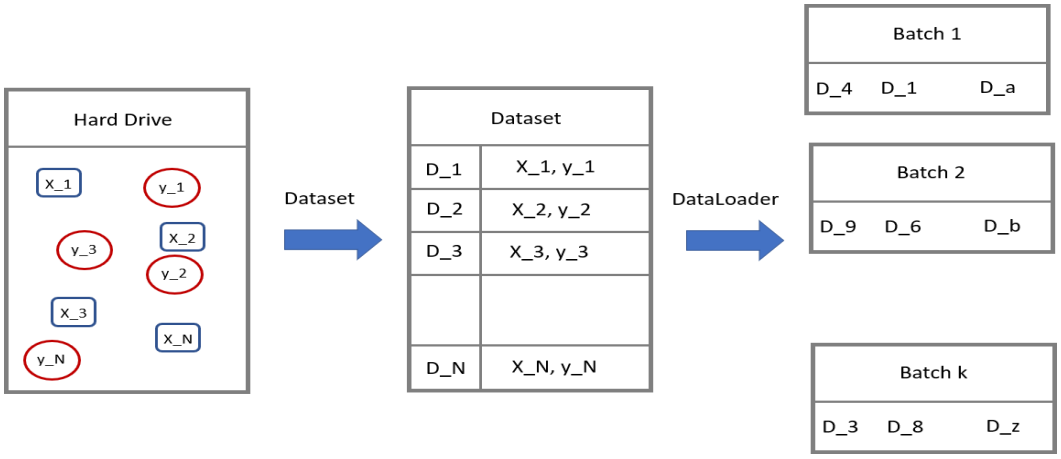
Activation Function

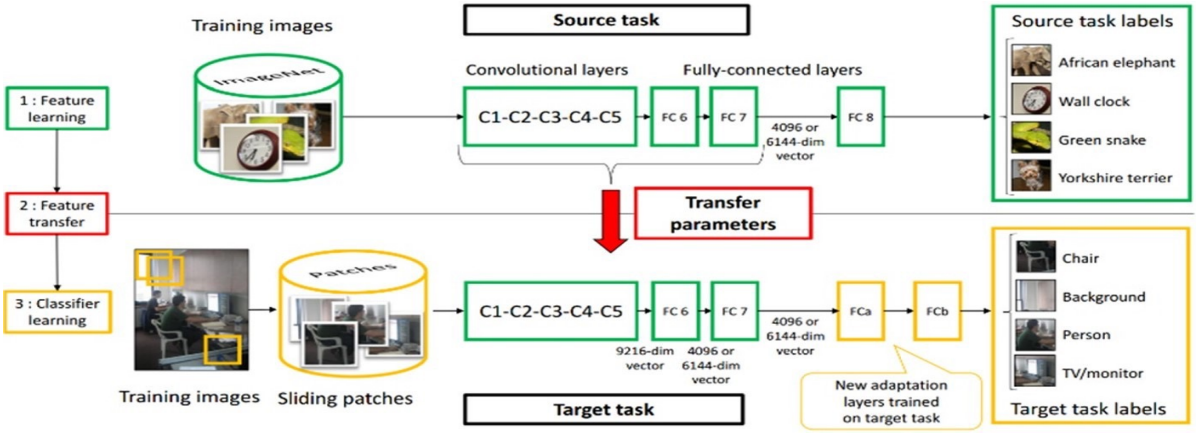


Normalization

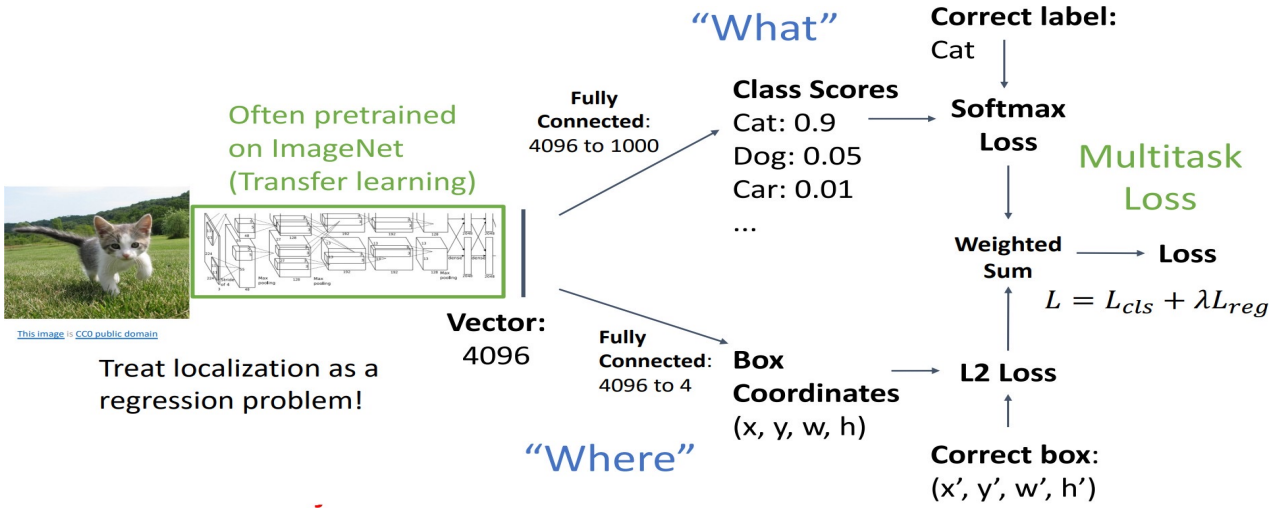
$$\hat{x}_{i,j} = \frac{x_{i,j} - \mu_j}{\sqrt{\sigma_j^2 + \varepsilon}}$$

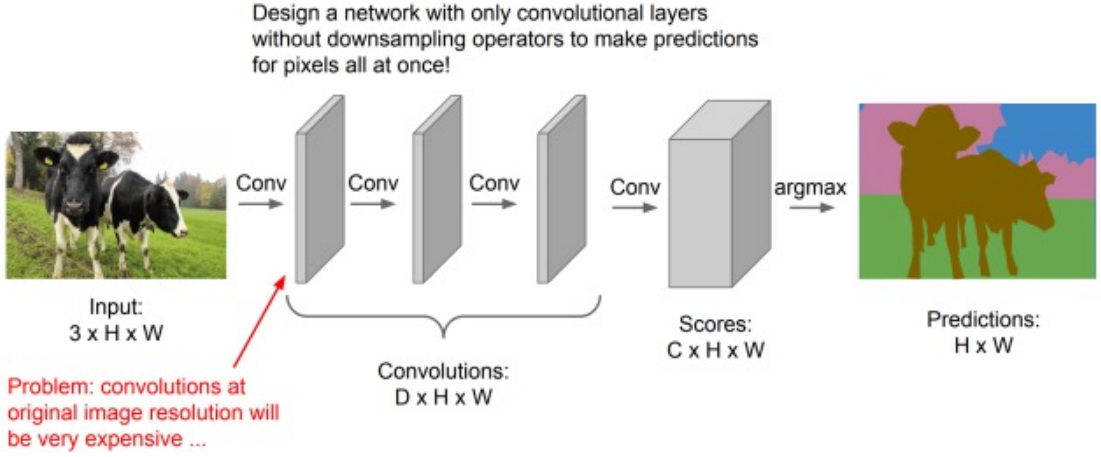
Data Loaders and Transfer Learning





Objects Detection and Segmentation

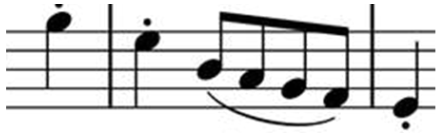
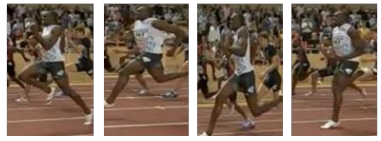




This Course

1. Recurrent Neural Networks and Self Attention
2. Attention and ViT
3. Video Classification
4. Generative AI (Part 1)
5. Generative AI (Part 2), CLIP model

Examples of Sequence Data

Music generation	∅	→	
Sentiment classification	"There is nothing to like in this movie."	→	
DNA sequence analysis	AGCCCCTGTGAGGAACTAG	→	AGCCCCTGTGAGGAACTAG
Machine translation	Voulez-vous chanter avec moi?	→	Do you want to sing with me?
Video activity recognition		→	Running
Name entity recognition	Yesterday, Harry Potter met Hermione Granger.	→	Yesterday, Harry Potter met Hermione Granger.

Andrew Ng

Representing Words

Named Entity Recognition

x: Tom ($x^{<1>}$) and ($x^{<2>}$) Jerry went to the store to buy food. ($x^{<Tx>}$)

y: 1 ($y^{<1>}$) 0 ($y^{<2>}$) 1 ..... 0 ($y^{<Ty>}$)

Vocabulary	Indexes	One-hot
A		
Aa		
…		
…		
…		
And		
…		
Jerry		
…		
Tom		
…		

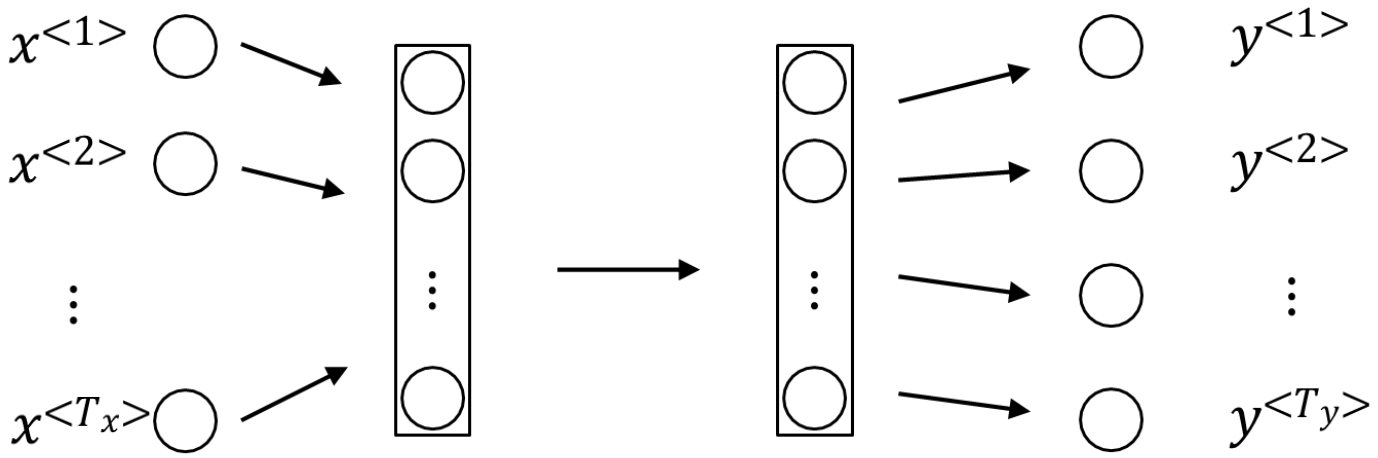
Unknown words?

End of sentence?

<unk>

<EOS>

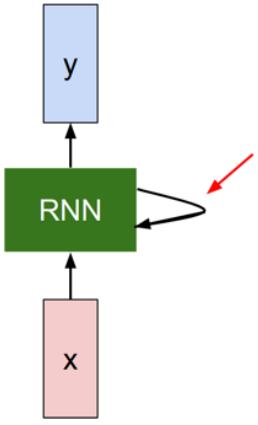
Why Sequence Models?



Problems?

- Inputs, outputs can be different lengths in different examples.
- Doesn't share features learned across different positions of text.

Rolled Diagram of RNNs



Key idea: RNNs have an "internal state" that is updated as a sequence is processed

$$h_t = f_W(h_{t-1}, x_t)$$

new state

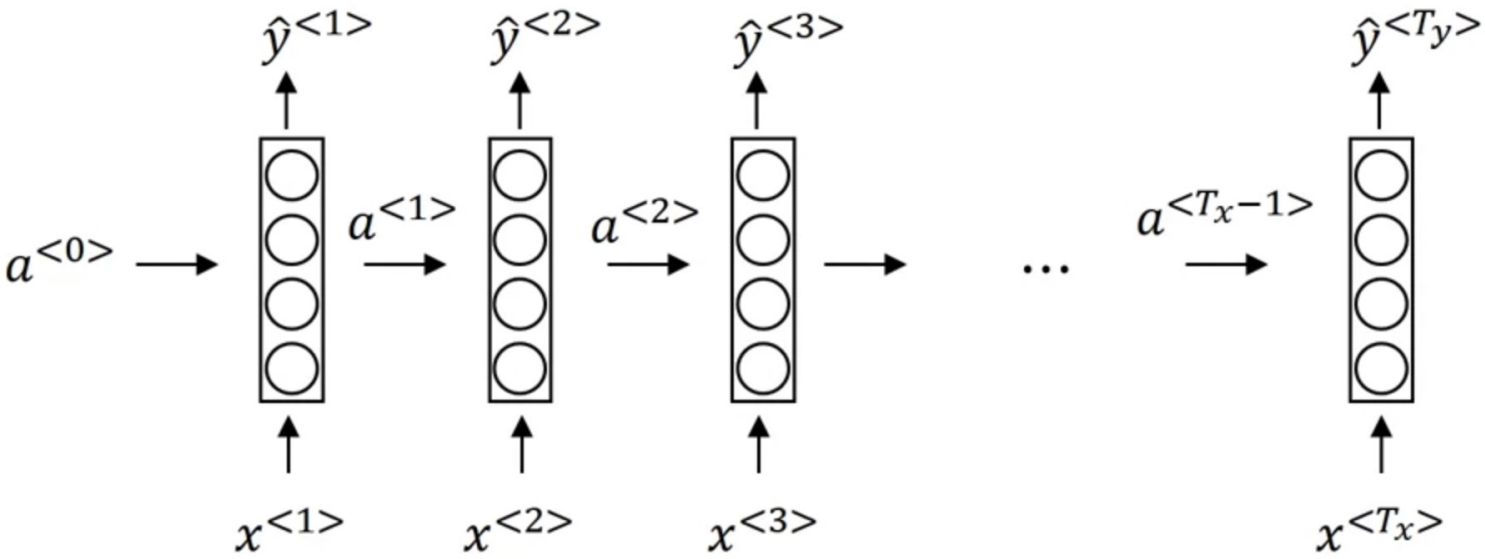
some function with parameters W

old state

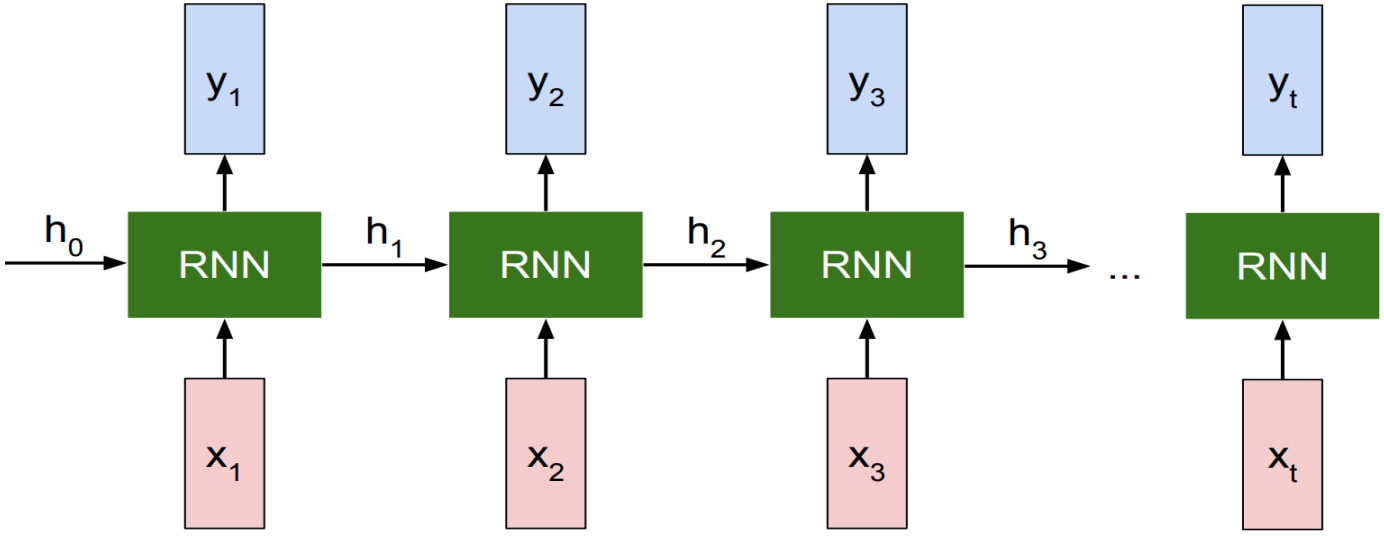
input vector at some time step

Lets elaborate/unroll it.

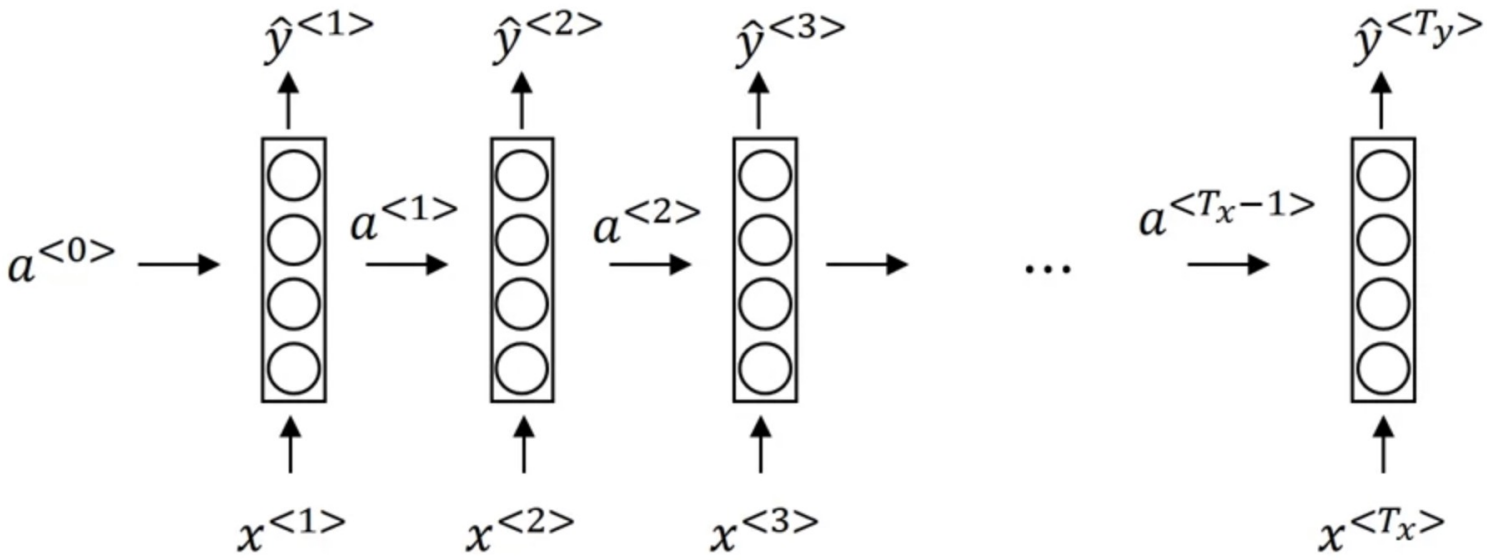
Recurrent Neural Networks (RNN)



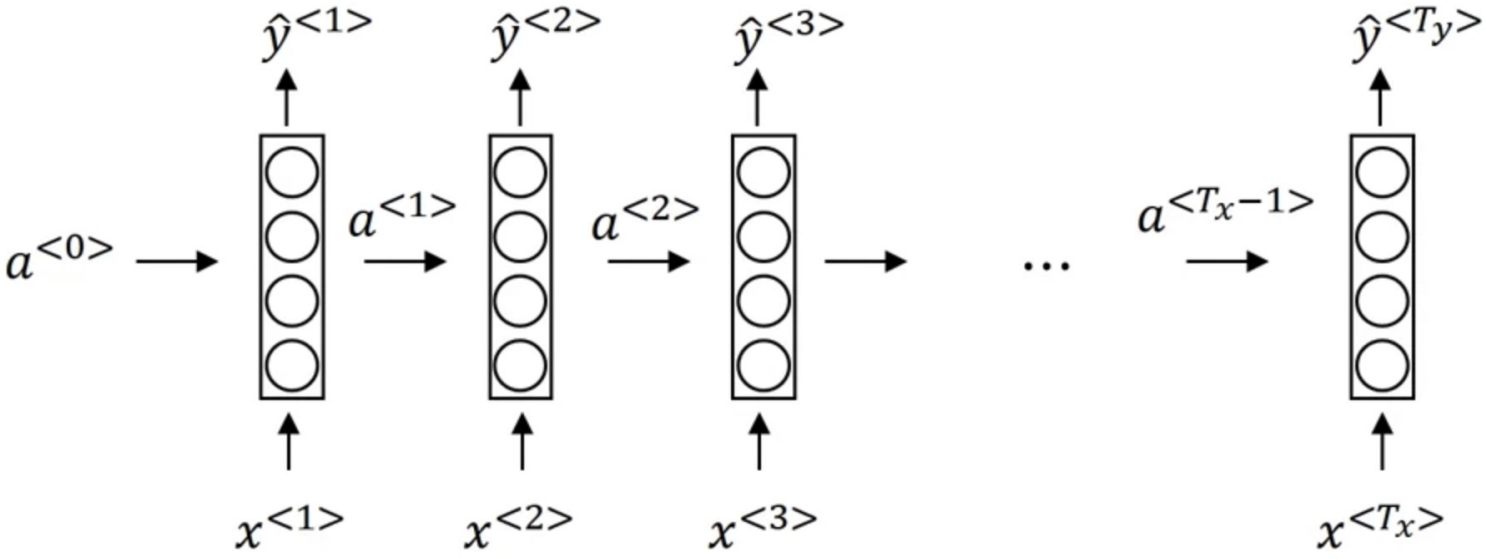
Another Representation



Forward propagation



Backward propagation



Recurrent Neural Networks (RNN)

- RNN Advantages:
  - Can process any length input
  - Computation for step t can (in theory) use information from many steps back
  - Model size doesn't increase for longer input
  - Same weights applied on every timestep, so there is symmetry in how inputs are processed.

Recurrent Neural Networks (RNN)

- RNN Advantages:
  - Can process any length input
  - Computation for step t can (in theory) use information from many steps back
  - Model size doesn't increase for longer input
  - Same weights applied on every timestep, so there is symmetry in how inputs are processed.
- RNN Disadvantages:
  - Recurrent computation is slow
  - In practice, difficult to access information from many steps back

Types of RNNs

Many-to-many ($T_x = T_y$)

Types of RNNs

Many-to-one

Types of RNNs

One-to-many

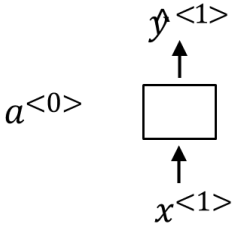
Types of RNNs

One-to-one

Types of RNNs

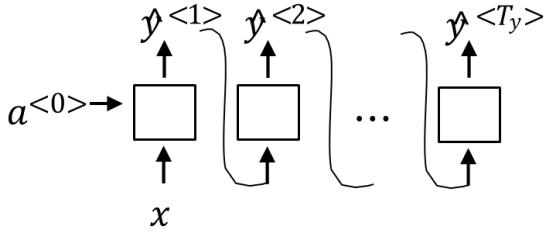
Many-to-many
See the whole input first.

Types of RNNs



One to one

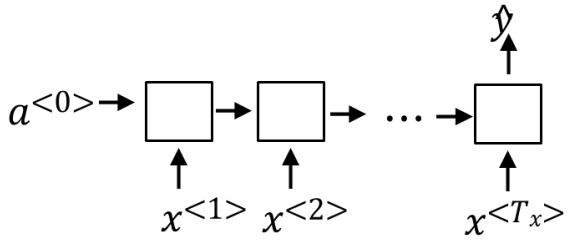
Standard generic NN.



One to many

Music notes generation

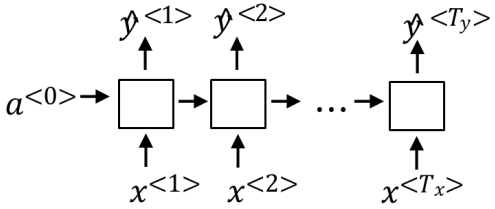
-1/**



Many to one

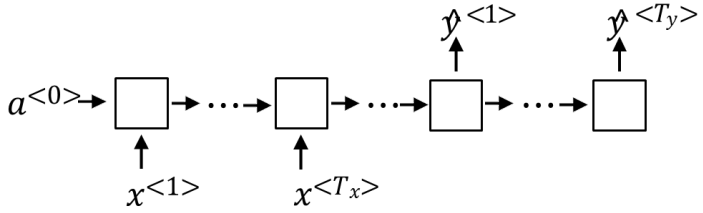
The food is worst in this restaurant.

Sentiment classification



Many to many

Named entity recognition



Many to many

First: Read the input, then generate output.

Machine translation

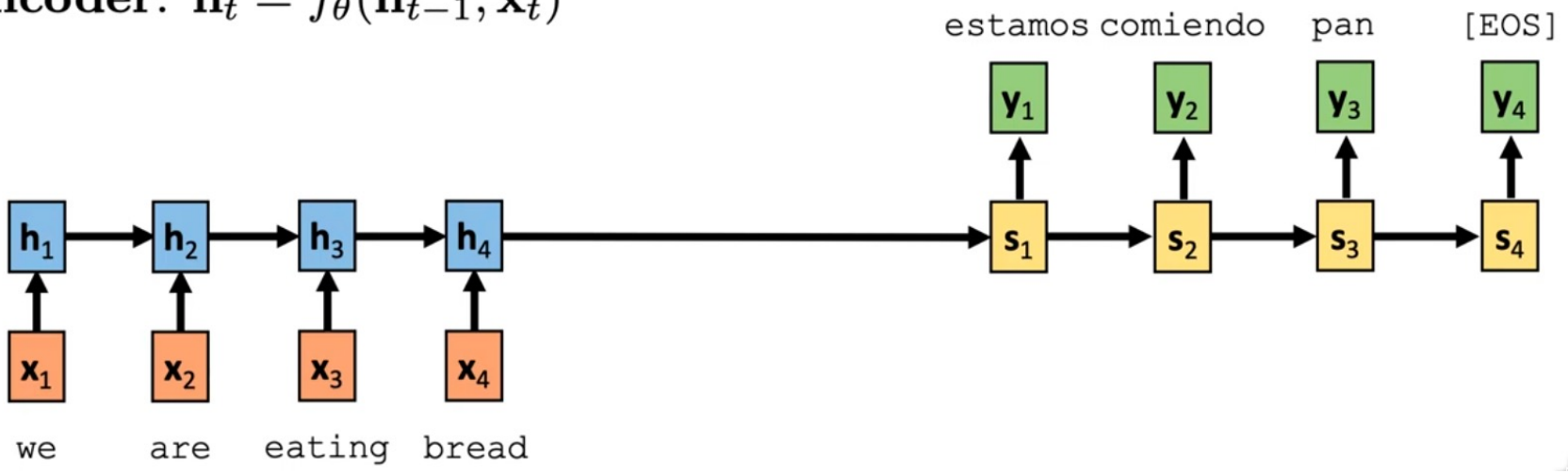
Seq2seq: Many-to-one + one-to-many

Input: Sequence $\mathbf{x}_1, \ldots, \mathbf{x}_T$

Output: Sequence $\mathbf{y}_1, \ldots, \mathbf{y}_{T'}$

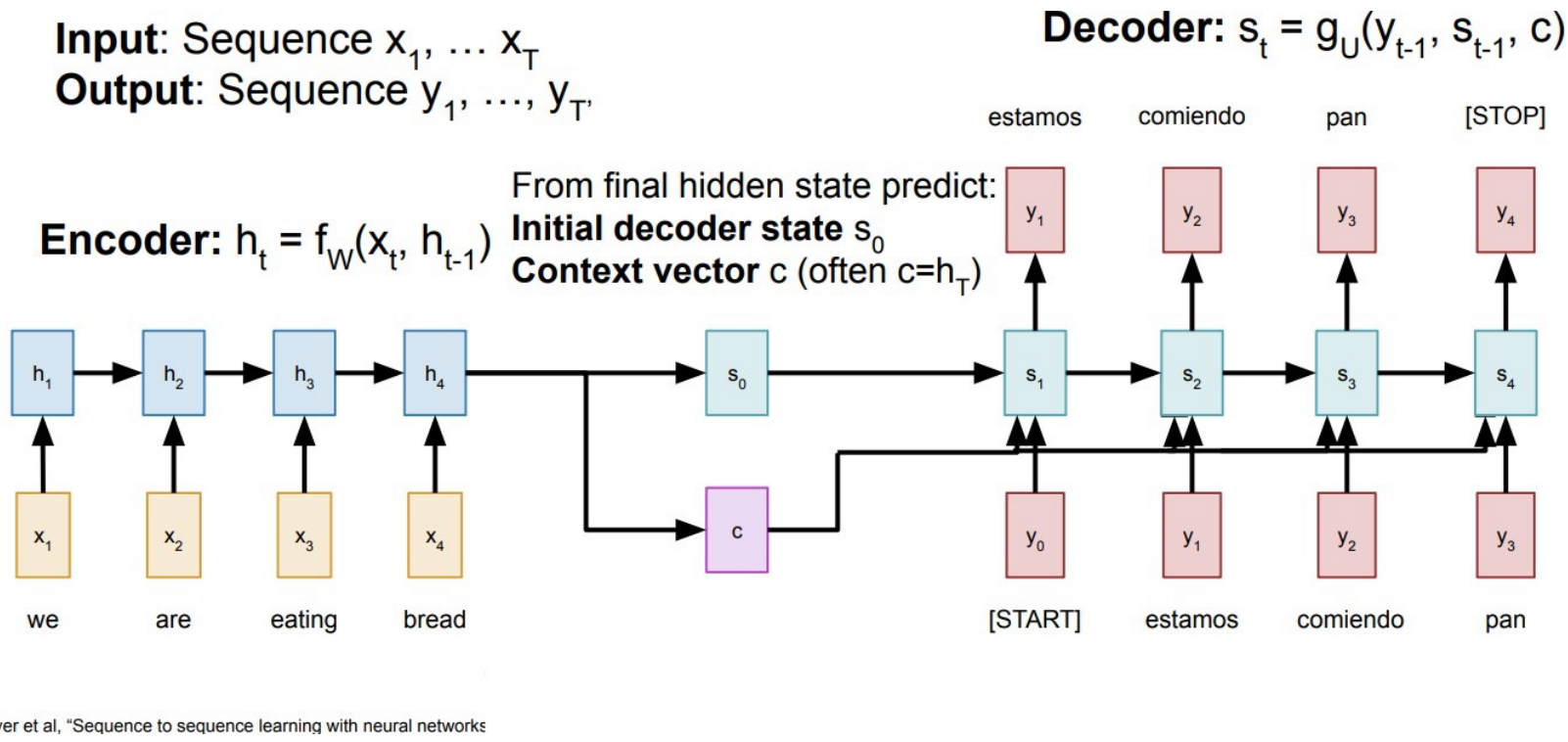
Encoder: $\mathbf{h}_t = f_\theta(\mathbf{h}_{t-1}, \mathbf{x}_t)$

Decoder: $\mathbf{s}_t, \mathbf{y}_t = g_{\theta'}(\mathbf{s}_{t-1})$



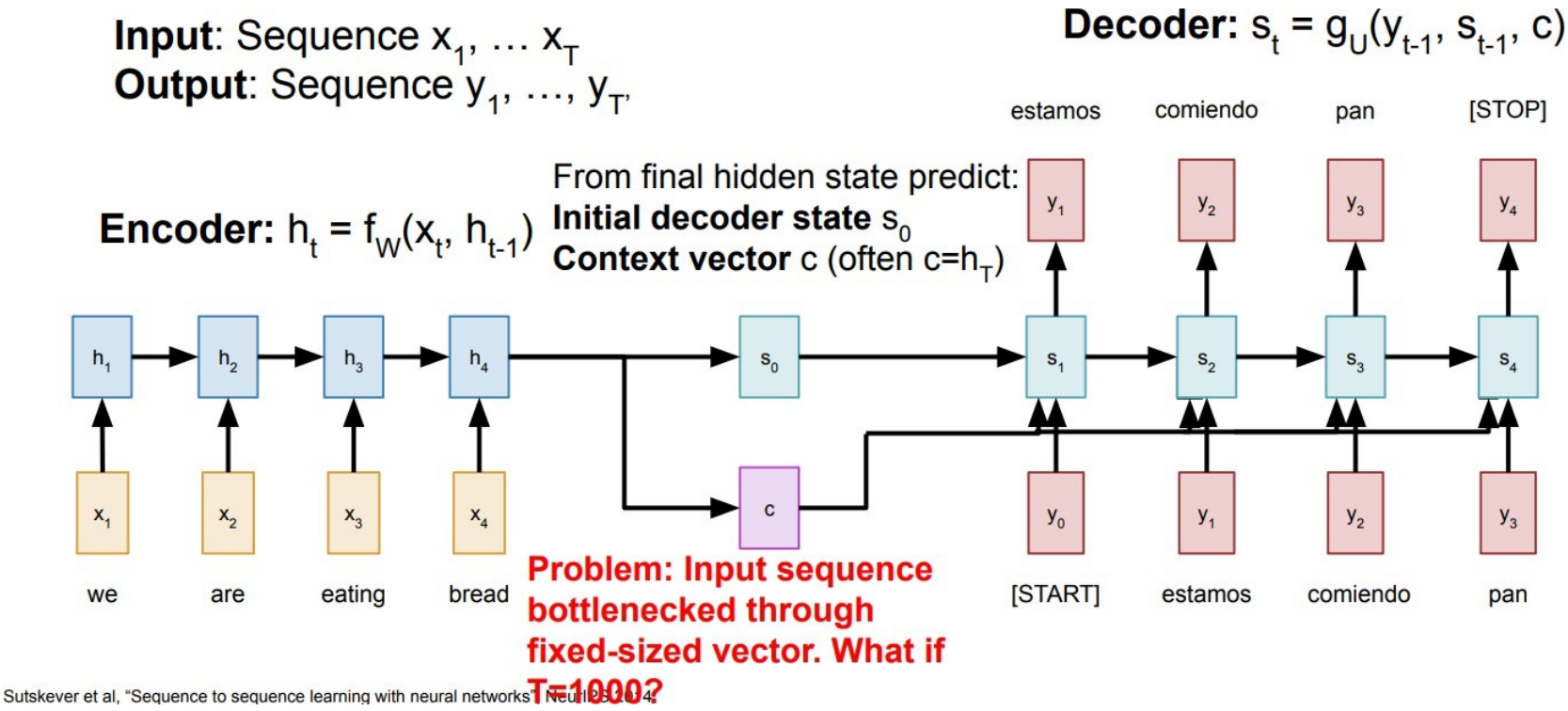
Machine translation: English to French

Seq2seq: Many-to-one + one-to-many



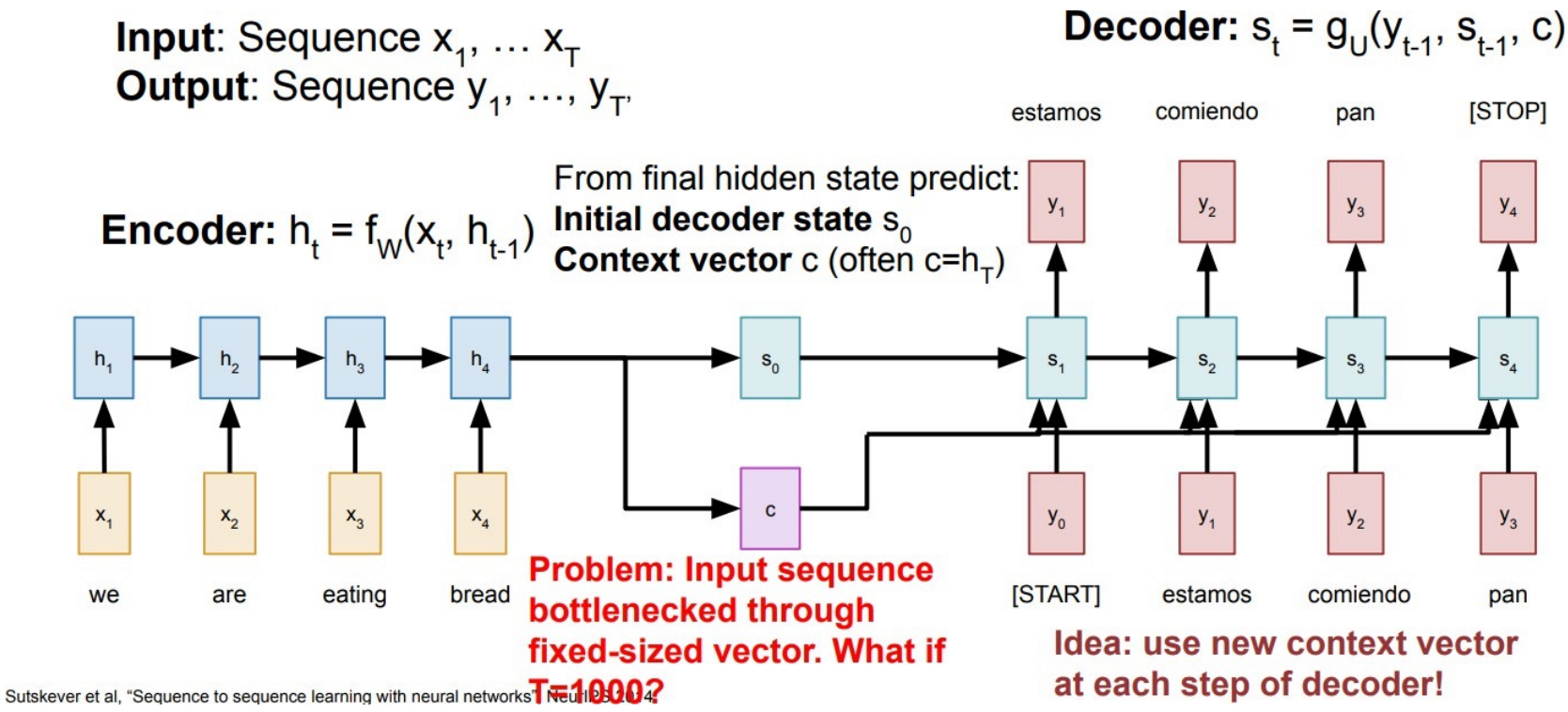
Sutskever et al, "Sequence to sequence learning with neural networks

Sequence to Sequence with RNNs



Sutskever et al, "Sequence to sequence learning with neural networks," NeurIPS 2014.

Sequence to Sequence with RNNs



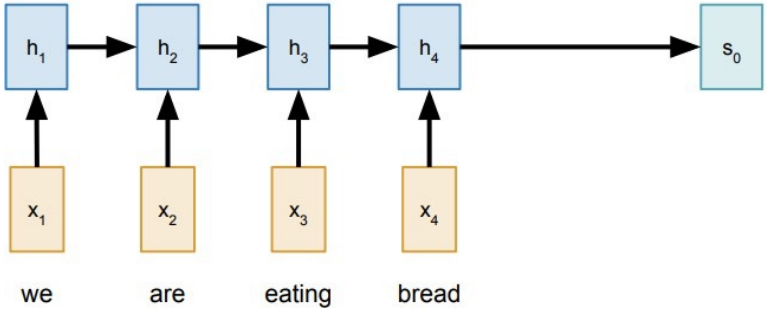
Sutskever et al, "Sequence to sequence learning with neural networks" NeurIPS 2014

Sequence to Sequence with RNNs and Attention

Input: Sequence $x_1, \ldots x_T$
Output: Sequence $y_1, \ldots, y_{T'}$

Encoder: $h_t = f_W(x_t, h_{t-1})$

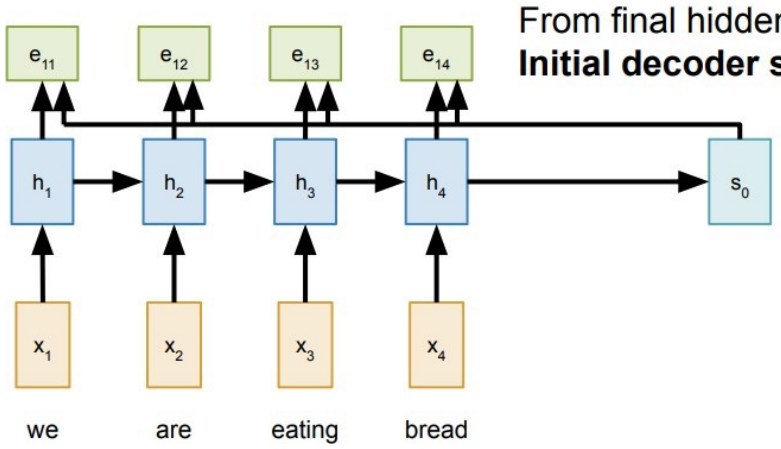
From final hidden state:
Initial decoder state s_0



Sequence to Sequence with RNNs and Attention

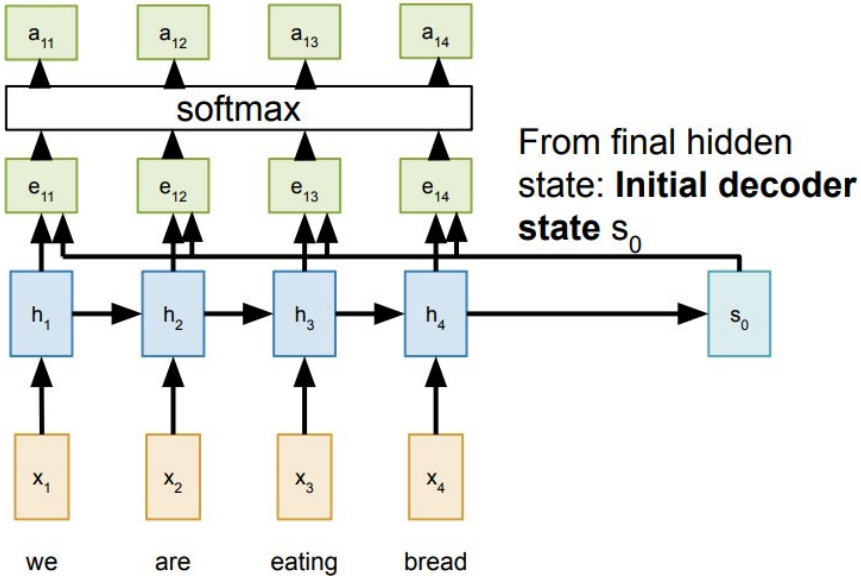
How important is h_i [from encoder] for predicting output at S_t?

Compute (scalar) **alignment scores**
$e_{t,i} = f_{att}(s_{t-1}, h_i)$ (f_{att} is an MLP)



From final hidden state:
Initial decoder state s_0

Sequence to Sequence with RNNs and Attention



From final hidden state: **Initial decoder state** s_0

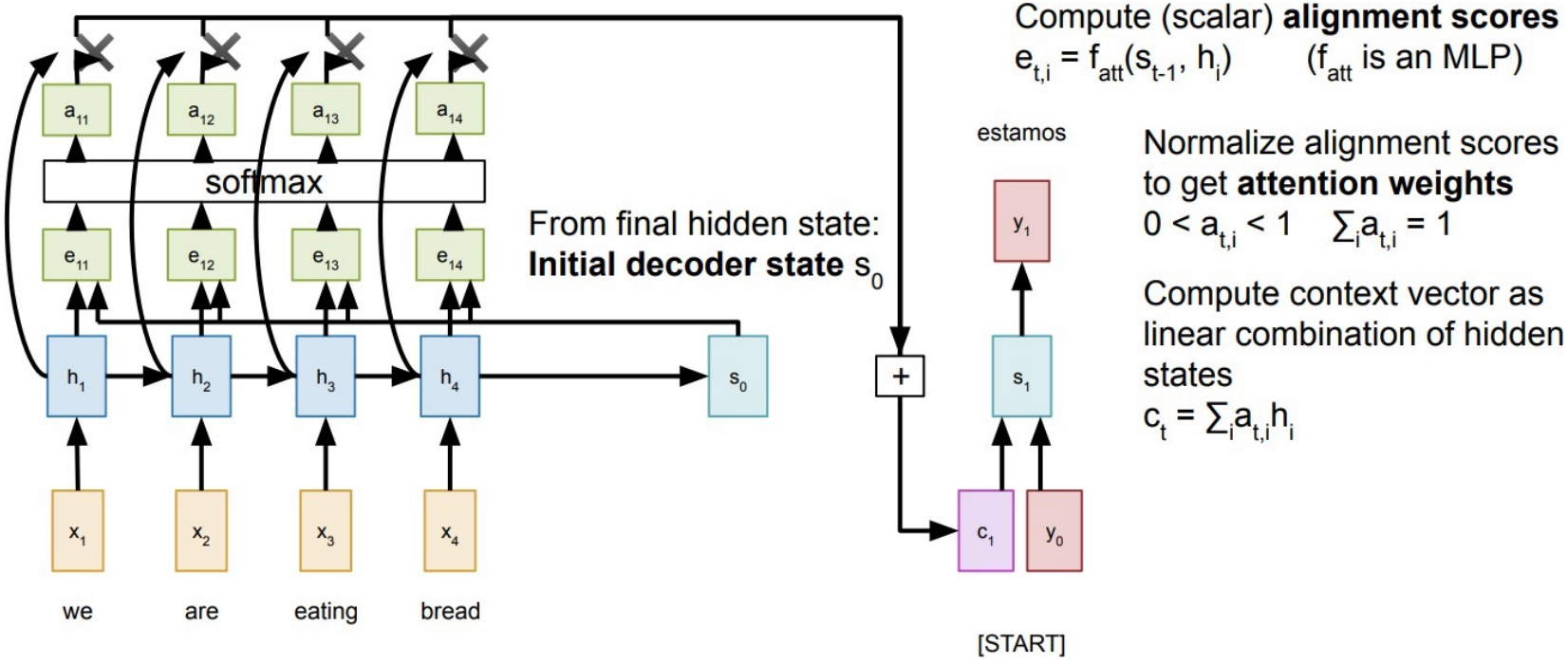
Compute (scalar) **alignment scores**

$e_{t,i} = f_{att}(s_{t-1}, h_i)$ (f_{att} is an MLP)

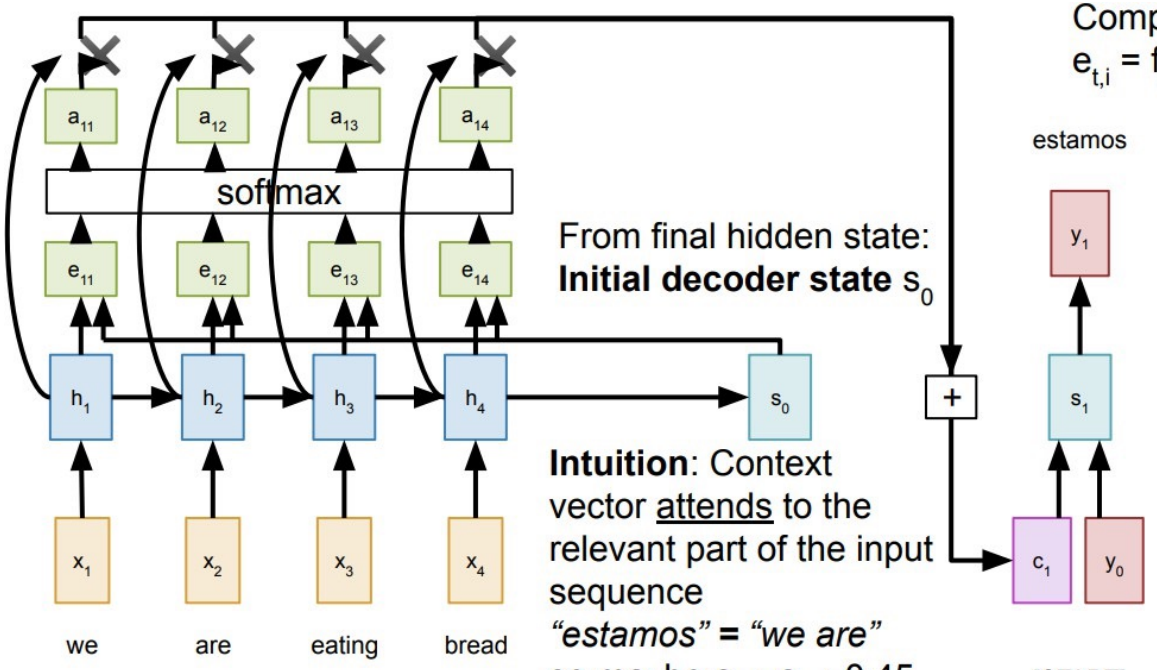
Normalize alignment scores to get **attention weights**

$0 < a_{t,i} < 1 \quad \sum_i a_{t,i} = 1$

Sequence to Sequence with RNNs and Attention



Sequence to Sequence with RNNs and Attention



From final hidden state:
Initial decoder state s_0

Intuition: Context vector attends to the relevant part of the input sequence
"estamos" = "we are"
so maybe $a_{11}=a_{12}=0.45$, $a_{13}=a_{14}=0.05$

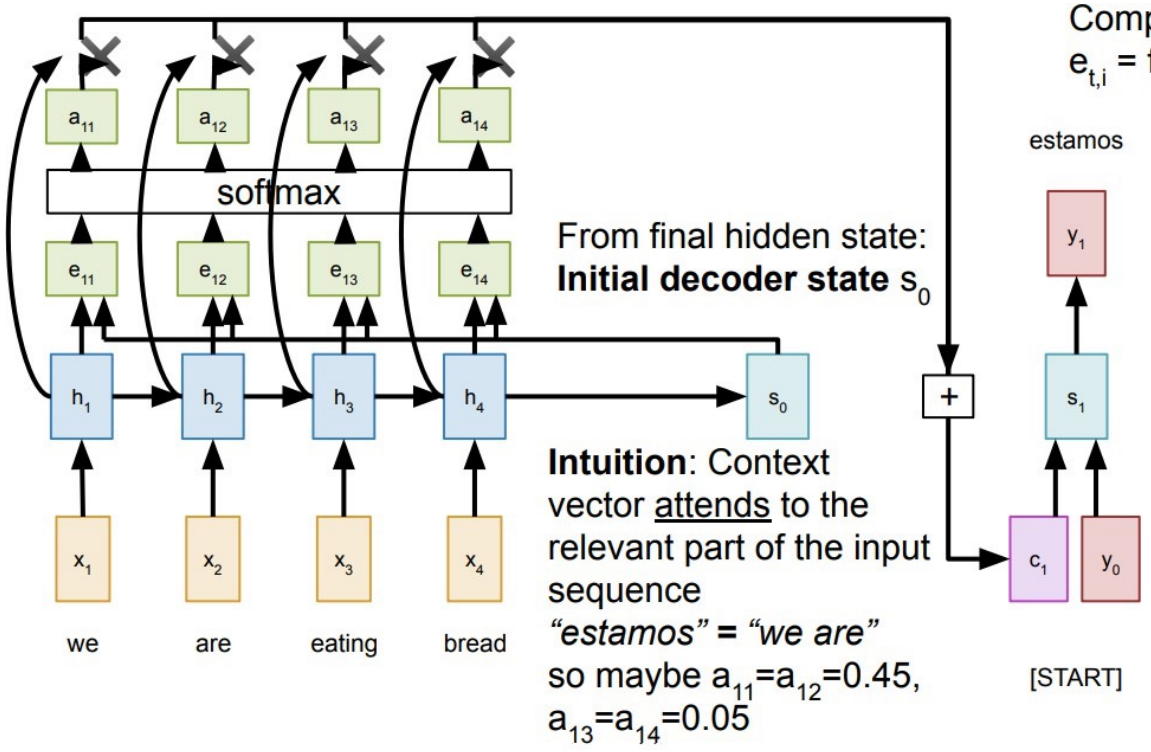
Compute (scalar) **alignment scores**
$e_{t,i} = f_{att}(s_{t-1}, h_i)$ (f_{att} is an MLP)

Normalize alignment scores to get **attention weights**
$0 < a_{t,i} < 1 \quad \sum_i a_{t,i} = 1$

Compute context vector as linear combination of hidden states
$c_t = \sum_i a_{t,i} h_i$

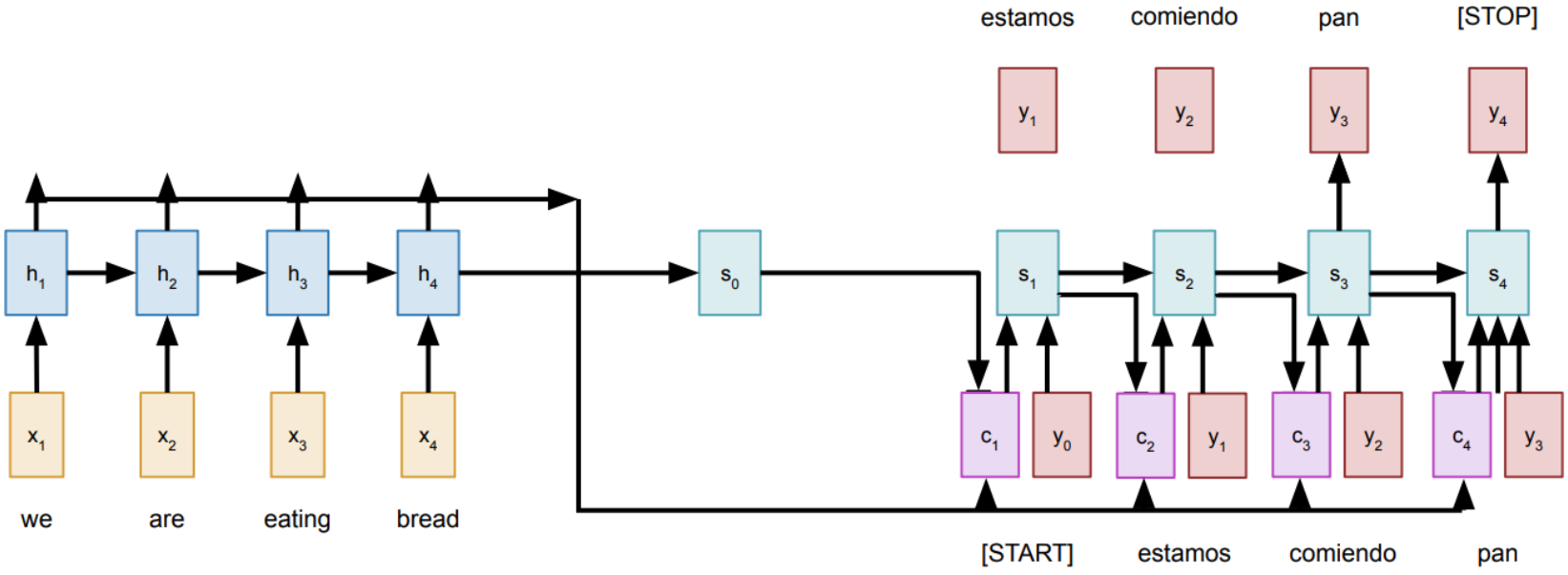
Use context vector in decoder: $s_t = g_U(y_{t-1}, s_{t-1}, c_t)$

Sequence to Sequence with RNNs and Attention

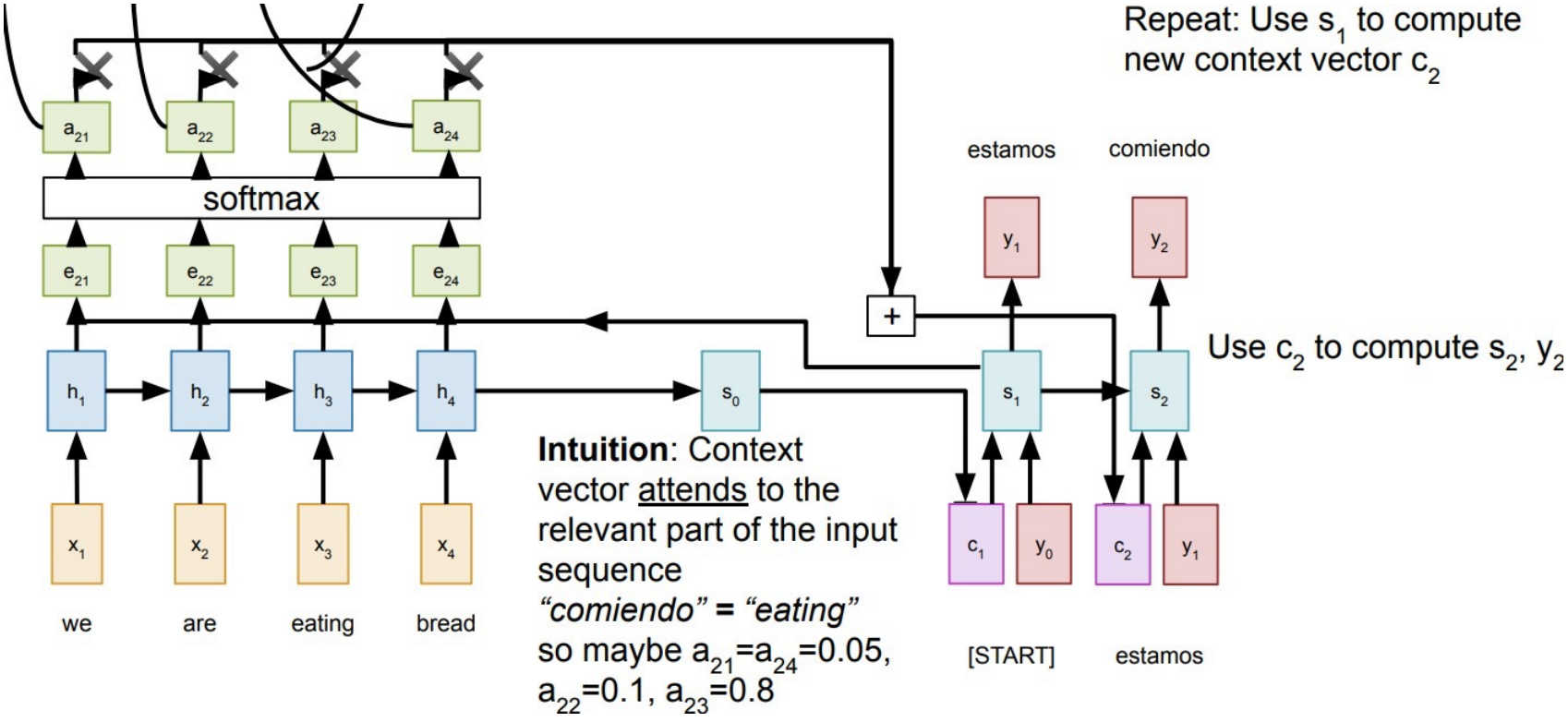


Sequence to Sequence with RNNs and Attention

- Use a different context vector in each timestep of decoder
- Input sequence not bottlenecked through single vector
- At each timestep of decoder, context vector ”looks at” different parts of the input sequence



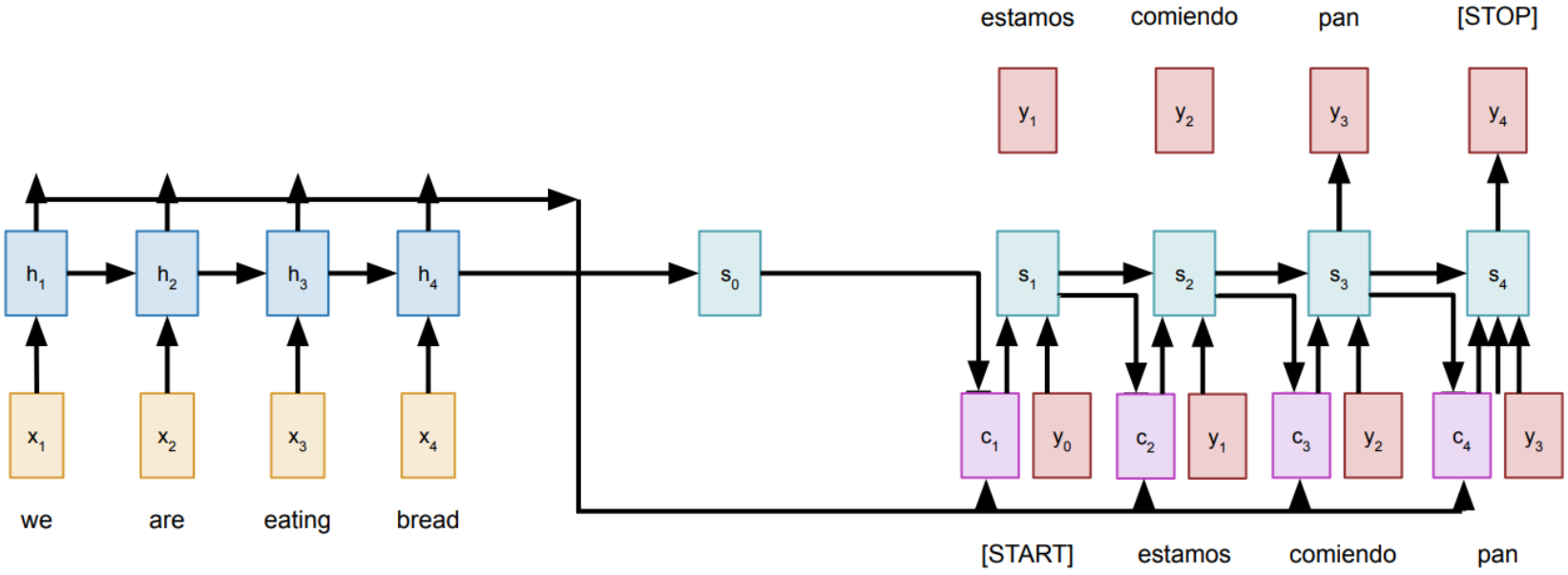
Sequence to Sequence with RNNs and Attention



Sequence to Sequence with RNNs and Attention

- What is the cost of generating attention?
  - Quadratic cost, at acceptable level.

  Research is going to reduce the cost.



- Can we use this architecture for any type of hidden vectors?

Image Captioning

- A computer Vision task in which we generate captions for images

A cat sitting on a suitcase on the floor

A cat is sitting on a tree branch

A dog is running in the grass with a frisbee

A white teddy bear sitting in the grass

Two people walking on the beach with surfboards

A tennis player in action on the court

Two giraffes standing in a grassy field

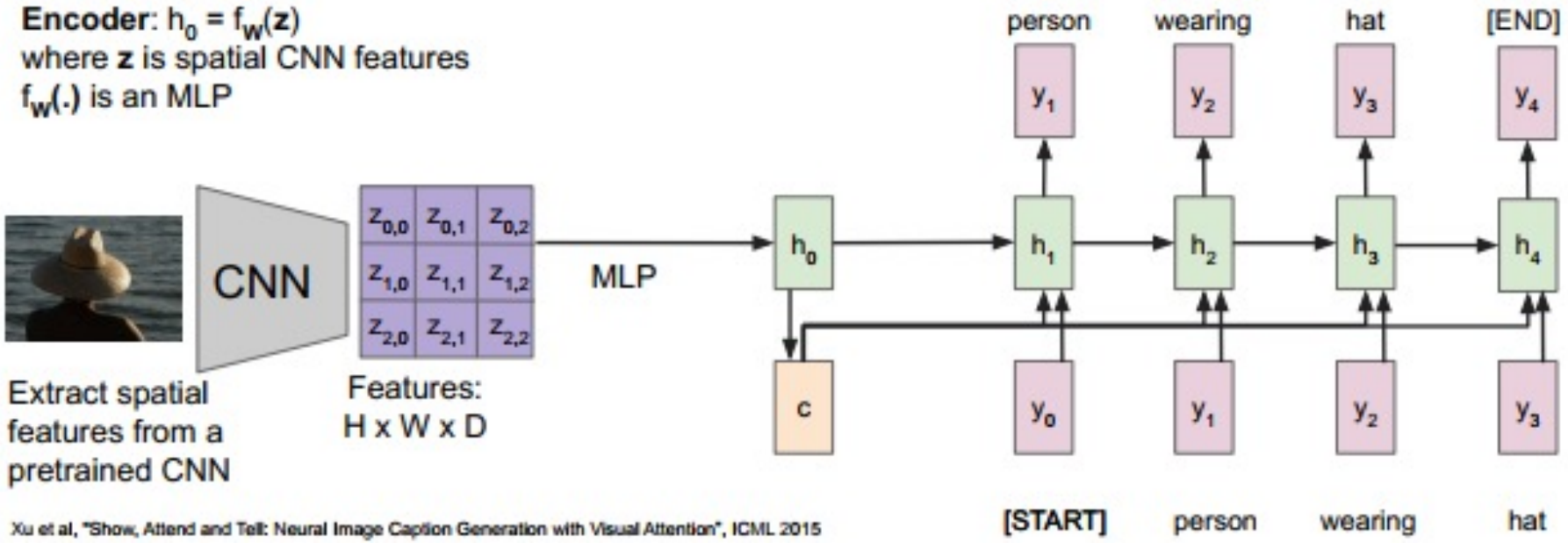
A man riding a dirt bike on a dirt track

Similar task in Computer Vision: Image Captioning Example

Input: Image **I**
Output: Sequence $\mathbf{y} = y_1, y_2, \ldots, y_T$

Decoder: $y_t = g_V(y_{t-1}, h_{t-1}, c)$
where context vector c is often $c = h_0$

Encoder: $h_0 = f_W(z)$
where **z** is spatial CNN features
$f_W(.)$ is an MLP



Xu et al, "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention", ICML 2015

Problem: Input is "bottlenecked" through c

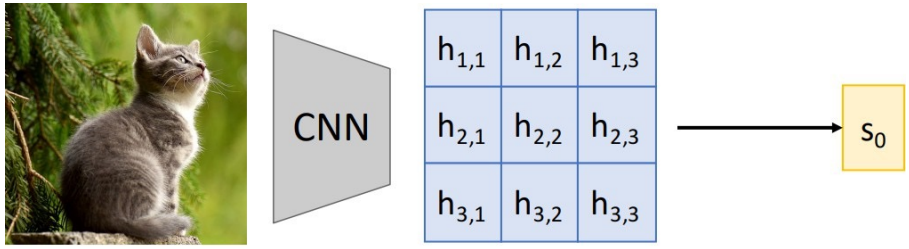
- Model needs to encode everything it wants to say within c

This is a problem if we want to generate really long descriptions? 100s of words long

Context vector at each time step

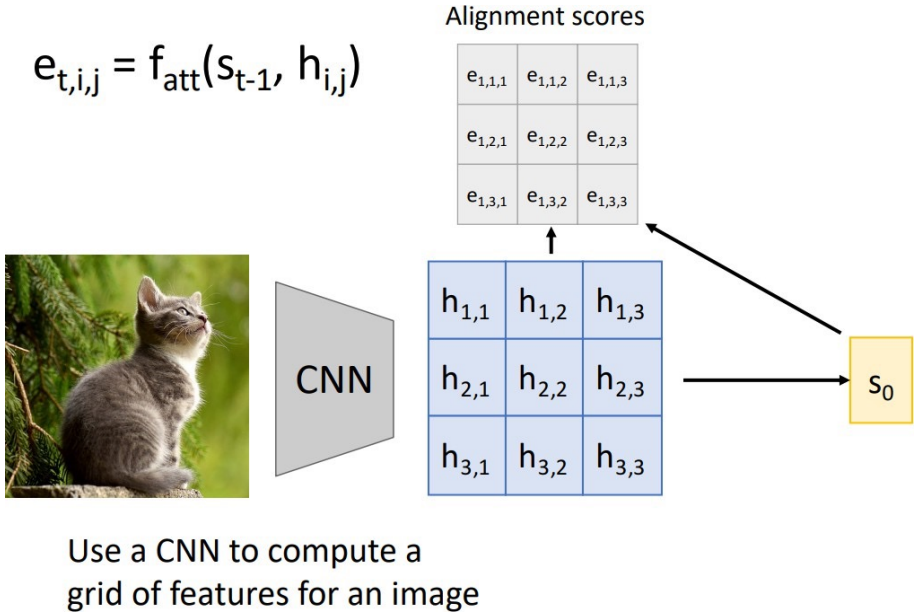
Each context vector attending different image regions

Image Captioning with RNNs and Attention



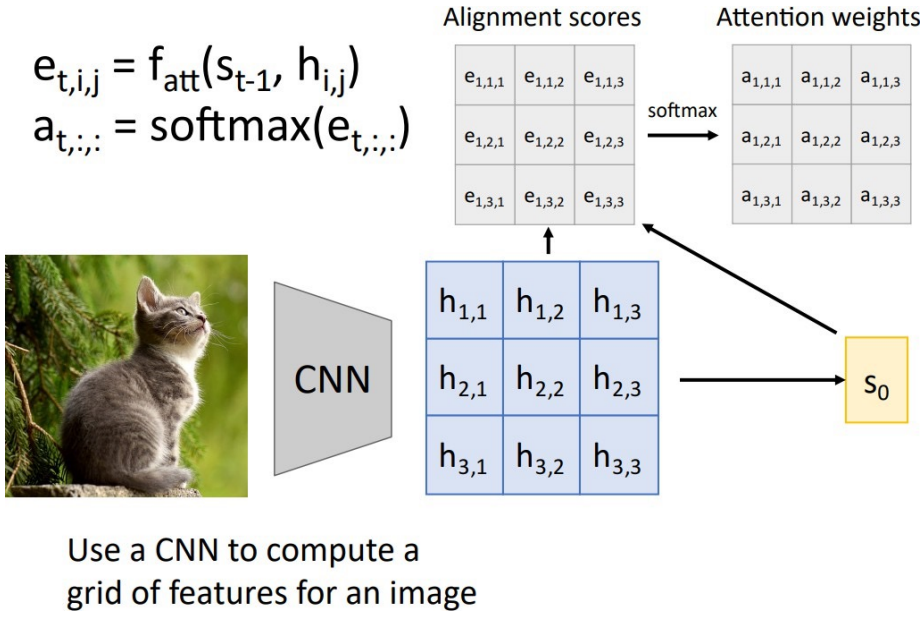
Use a CNN to compute a grid of features for an image

Image Captioning with RNNs and Attention



Use a CNN to compute a grid of features for an image

Image Captioning with RNNs and Attention



Use a CNN to compute a grid of features for an image

Image Captioning with RNNs and Attention

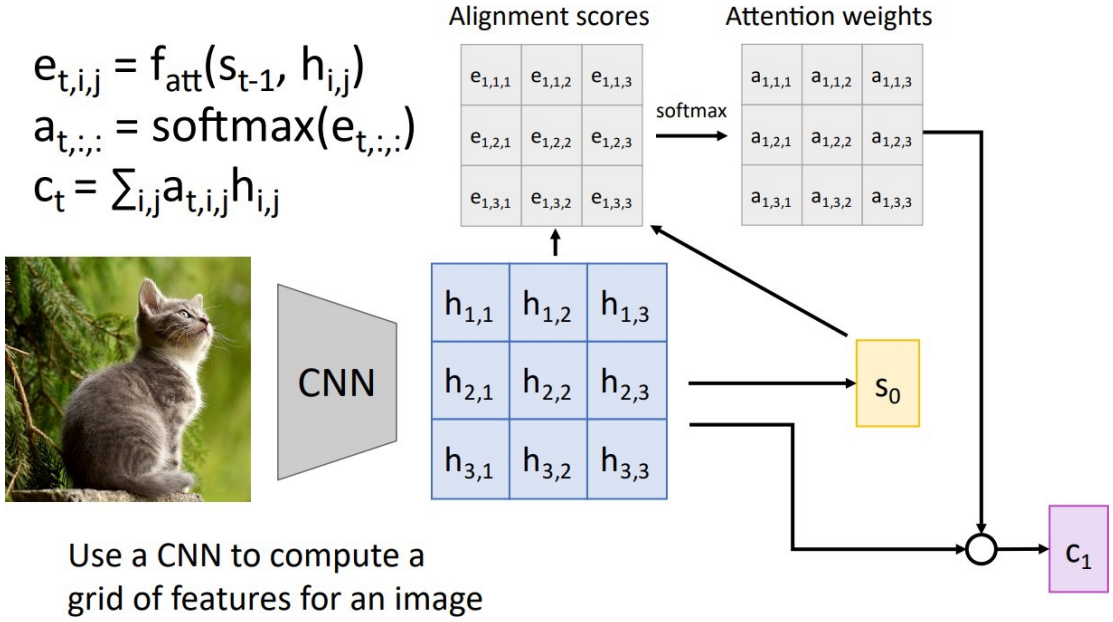


Image Captioning with RNNs and Attention

$e_{t,i,j} = f_{att}(s_{t-1}, h_{i,j})$

$a_{t,:,:} = \text{softmax}(e_{t,:,:})$

$c_t = \sum_{i,j} a_{t,i,j} h_{i,j}$

CNN

Use a CNN to compute a grid of features for an image

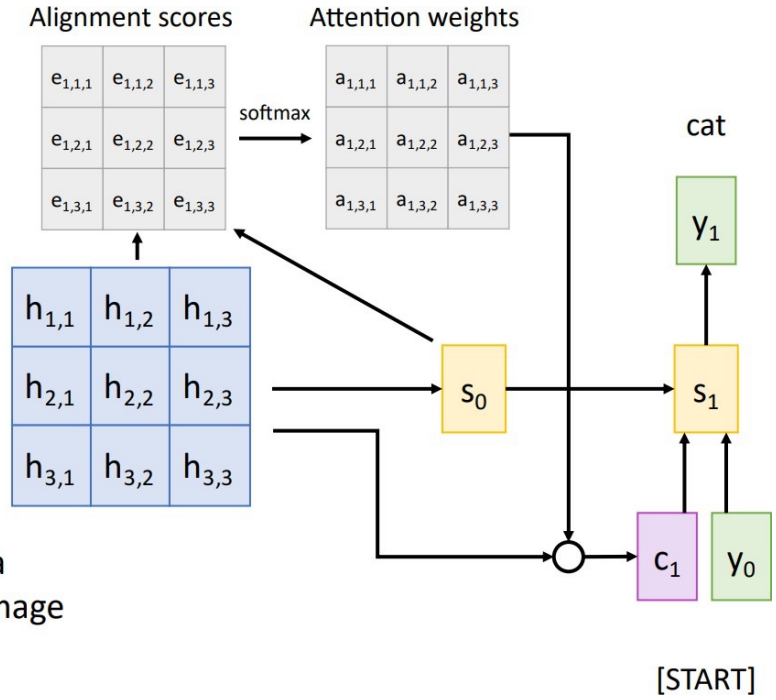


Image Captioning with RNNs and Attention

$e_{t,i,j} = f_{att}(s_{t-1}, h_{i,j})$

$a_{t,:,:} = \text{softmax}(e_{t,:,:})$

$c_t = \sum_{i,j} a_{t,i,j} h_{i,j}$

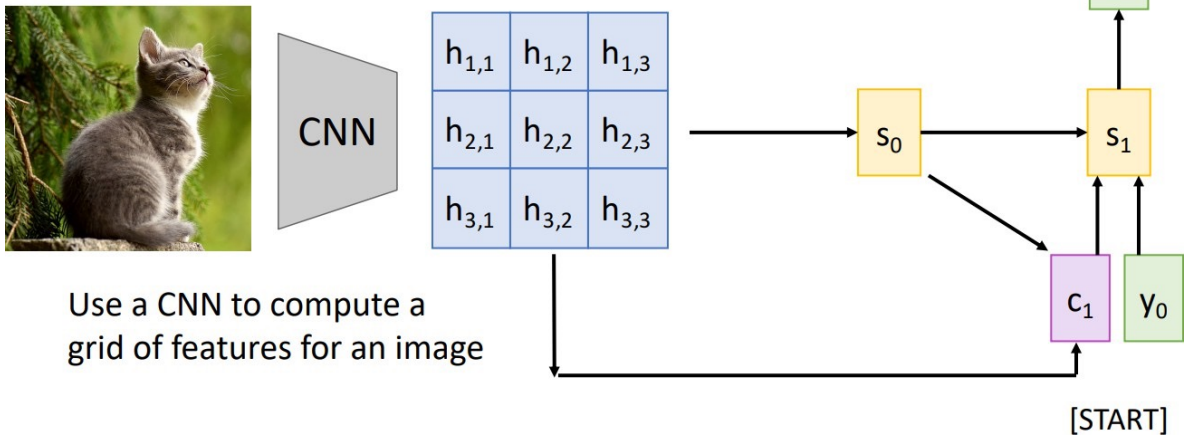


Image Captioning with RNNs and Attention

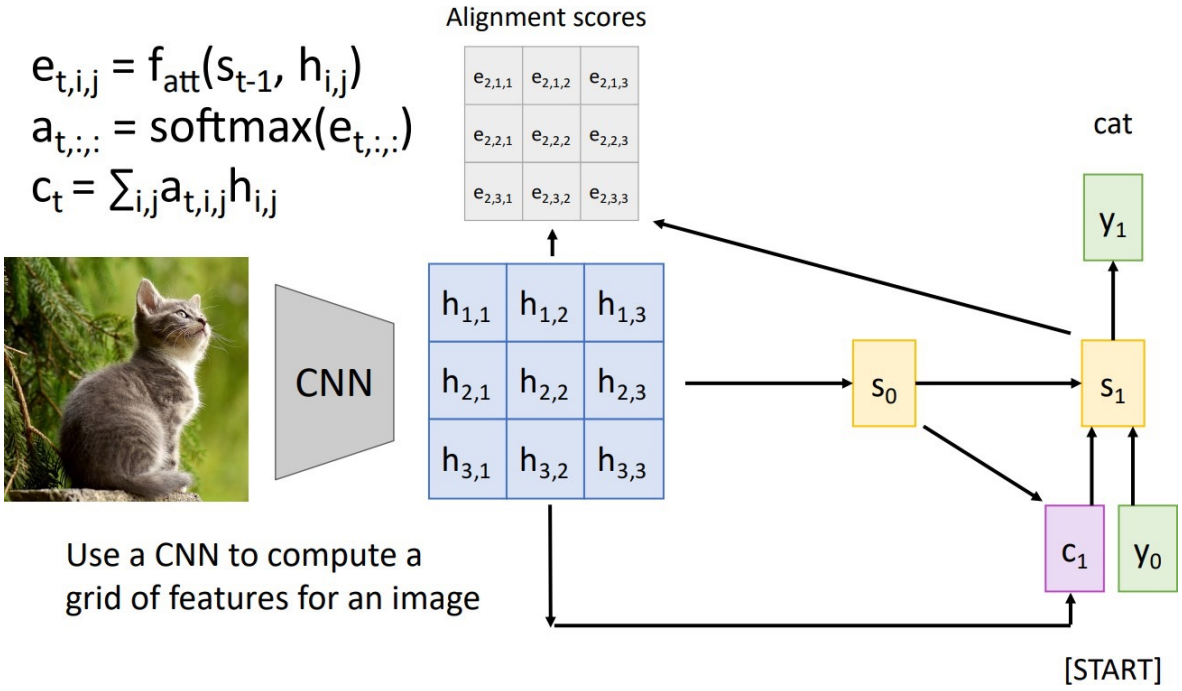


Image Captioning with RNNs and Attention

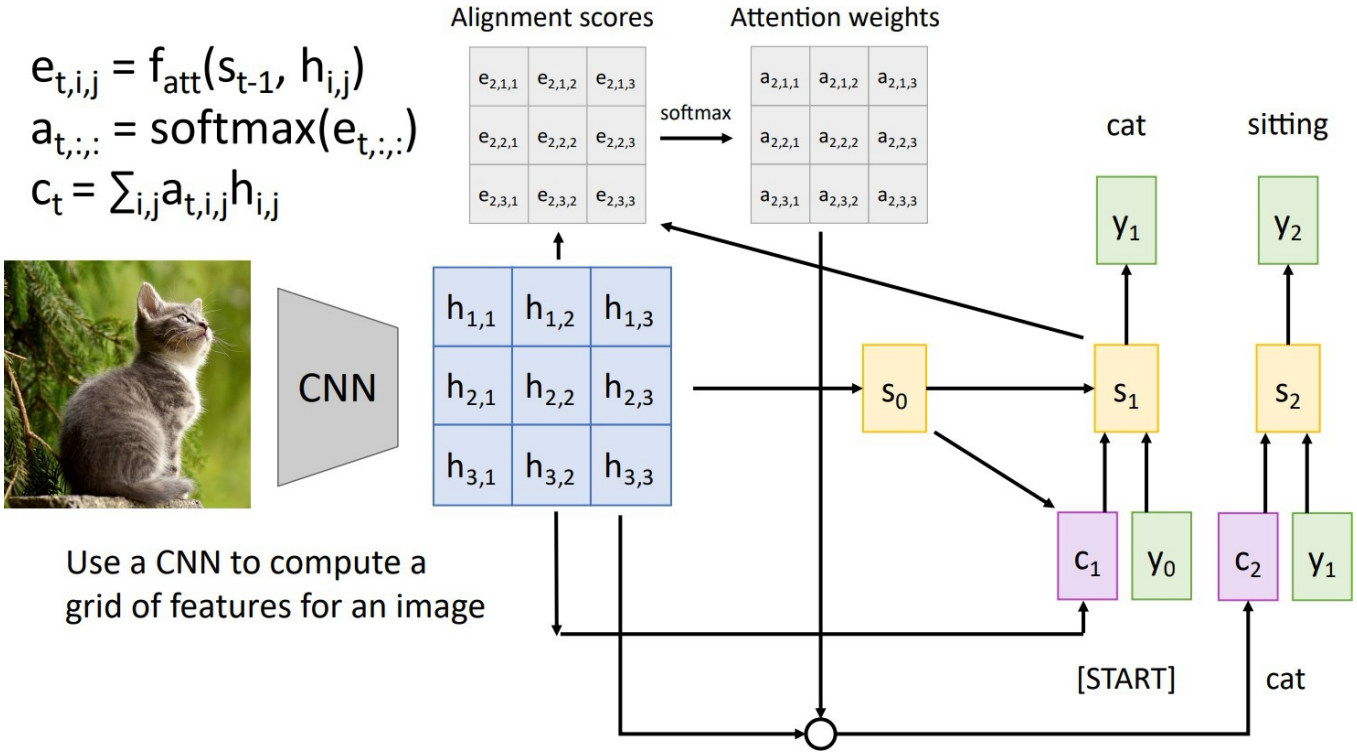


Image Captioning with RNNs and Attention

$e_{t,i,j} = f_{att}(s_{t-1}, h_{i,j})$
$a_{t,:,:} = \text{softmax}(e_{t,:,:})$
$c_t = \sum_{i,j} a_{t,i,j} h_{i,j}$

Each timestep of decoder uses a different context vector that looks at different parts of the input image

CNN

$h_{1,1}$ $h_{1,2}$ $h_{1,3}$
$h_{2,1}$ $h_{2,2}$ $h_{2,3}$
$h_{3,1}$ $h_{3,2}$ $h_{3,3}$

Use a CNN to compute a grid of features for an image

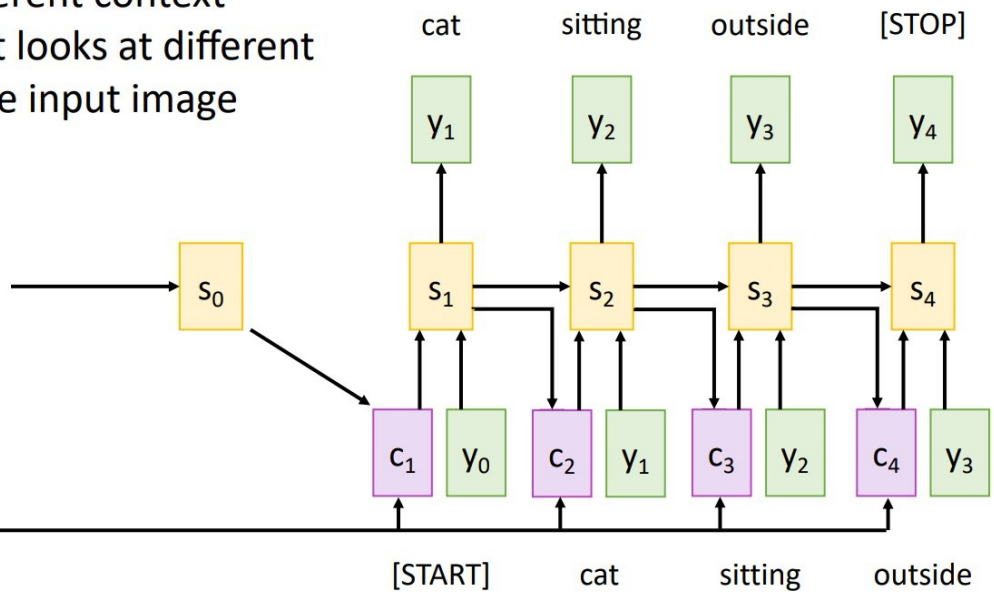


Image Captioning with RNNs and Attention

A dog is standing on a hardwood floor.

A stop sign is on a road with a mountain in the background.

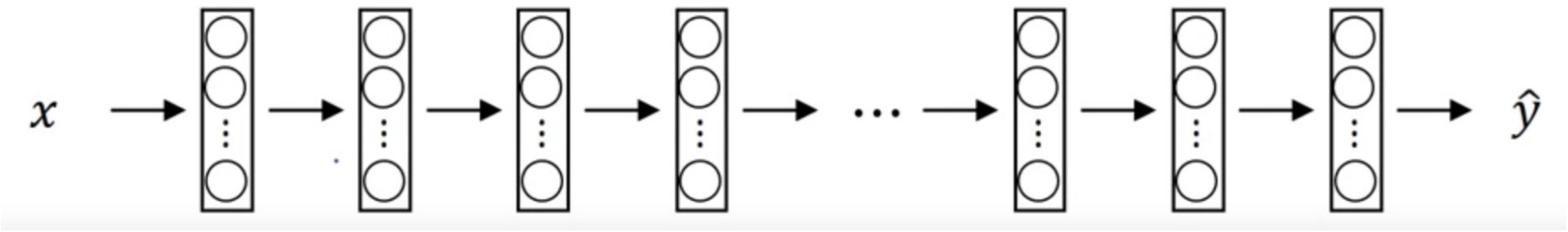
A group of people sitting on a boat in the water.

A giraffe standing in a forest with trees in the background.

[0]Xu et al, "Show, AKend, and Tell: Neural Image CapAon GeneraAon with Visual AKenAon", ICML 2015

Vanishing Gradients with RNNs

Long Sequences, Hard to Remember



Forward Prop: Left to right

Backward Prop: Right to left

Gradients from output > Hard times propagating back to effect the weights in the earlier layers.

Gated Recurrent Neural Networks

Quick Recap

1. Recurrent Neural Networks and Self Attention
2. Attention and ViT
3. Video Classification
4. Generative AI (Part 1)
5. Generative AI (Part 2), CLIP model

References

These slides have been adapted from

- Fei-Fei Li, Yunzhu Li & Ruohan Gao, Stanford CS231n: Deep Learning for Computer Vision
- Assaf Shocher, Shai Bagon, Meirav Galun & Tali Dekel, WAIC DL4CV Deep Learning for Computer Vision: Fundamentals and Applications
- Justin Johnson, UMich EECS 498.008/598.008: Deep Learning for Computer Vision
- Andrew Ng, Sequence Models