Recurrent Neural Networks with Attention

CS7150, Spring 2025

Prof. Huaizu Jiang

Northeastern University

In-class midterm

- On Friday, February 28
- Cover all topics by Friday, February 21
- Work on paper with pens, no coding! Similar to the in-class quizzes.
- A practice exam will be released
- Everyone is expected to show up in the classroom
  - Unless you have valid reasons
  - Contact the instructor if you do by Tuesday, February 21
- If you need accommodation, send your request to DRC@northeastern.edu
  - Someone will work the instructor together to figure out a solution

Access to Discovery

- Your account is ready there
- You can log into Discovery following the instructions here https://rc-docs.northeastern.edu/en/latest/first_steps/connect_mac.html
  - ssh h.jiang@login.discovery.neu.edu
- Documentation of how to use GPUs on Discovery https://rc-docs.northeastern.edu/en/latest/using-discovery/workingwithgpu.html

Extra credit in PA2

Part 3: Extra credits (14 points).

Let's do something fun here. You can do whatever you can use. Earn the full credits by achieving at least 91% accuracy on the testing set with the following restrictions:

- Train the model for no more than 5 epochs.
- Use a model whose number of parameters is smaller than 2M.

Note: If you have to override any function you implemented earlier, write new code below. Do not change the function definition in previous sections so that we can grade your implementation appopriately.

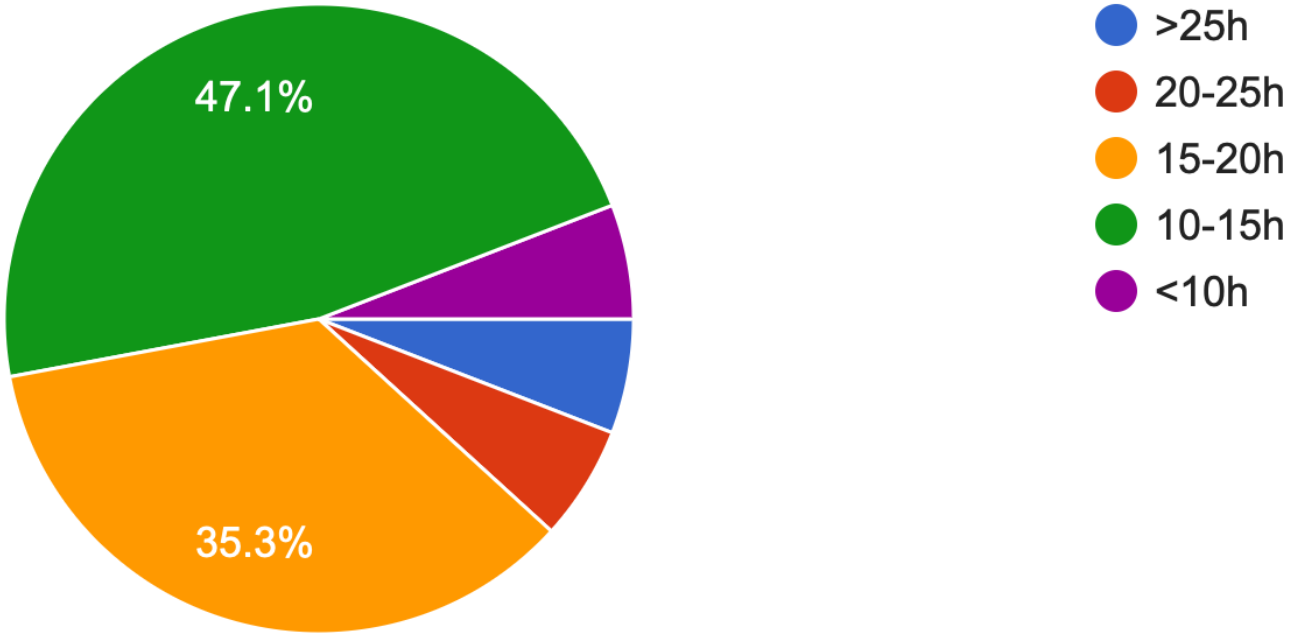
No partial credits will be given to this part. In other words, you won't get any credits if your final testing accuracy is lower than 91%.

- Do whatever you can think of.
- Be aware of the constraints.

Feedback of working on PA1

How many hours did you spend on it?

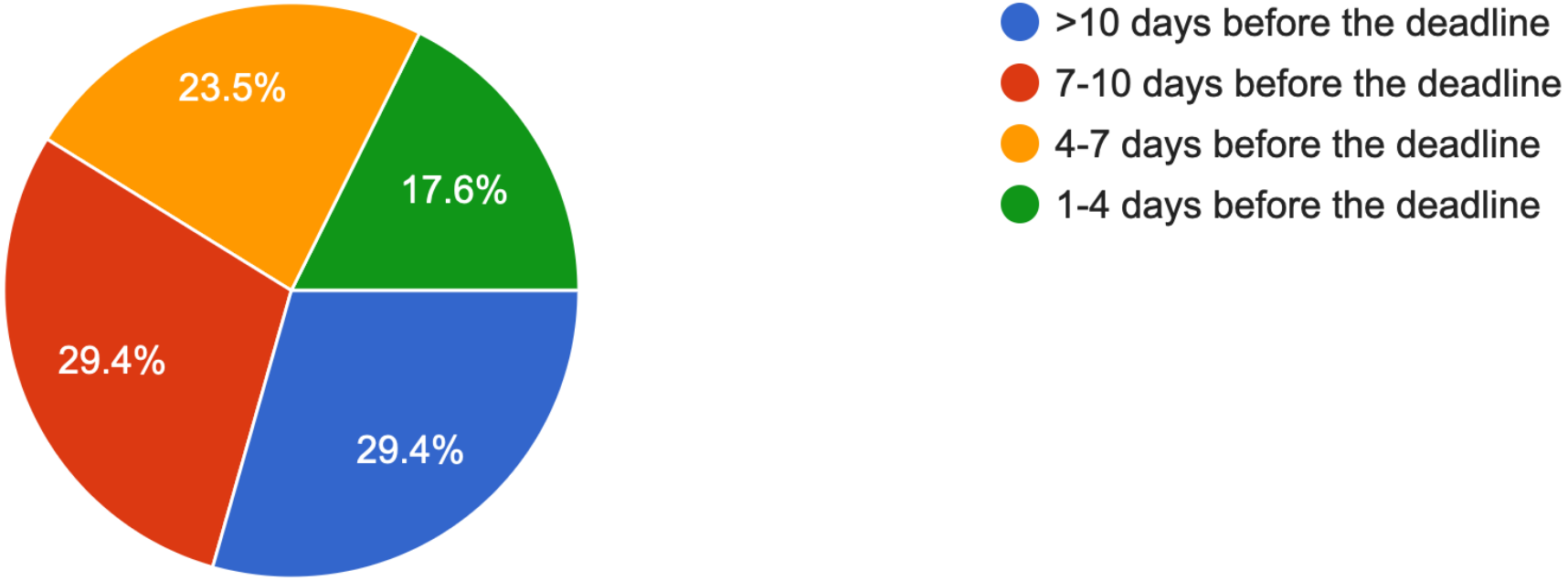
17 responses



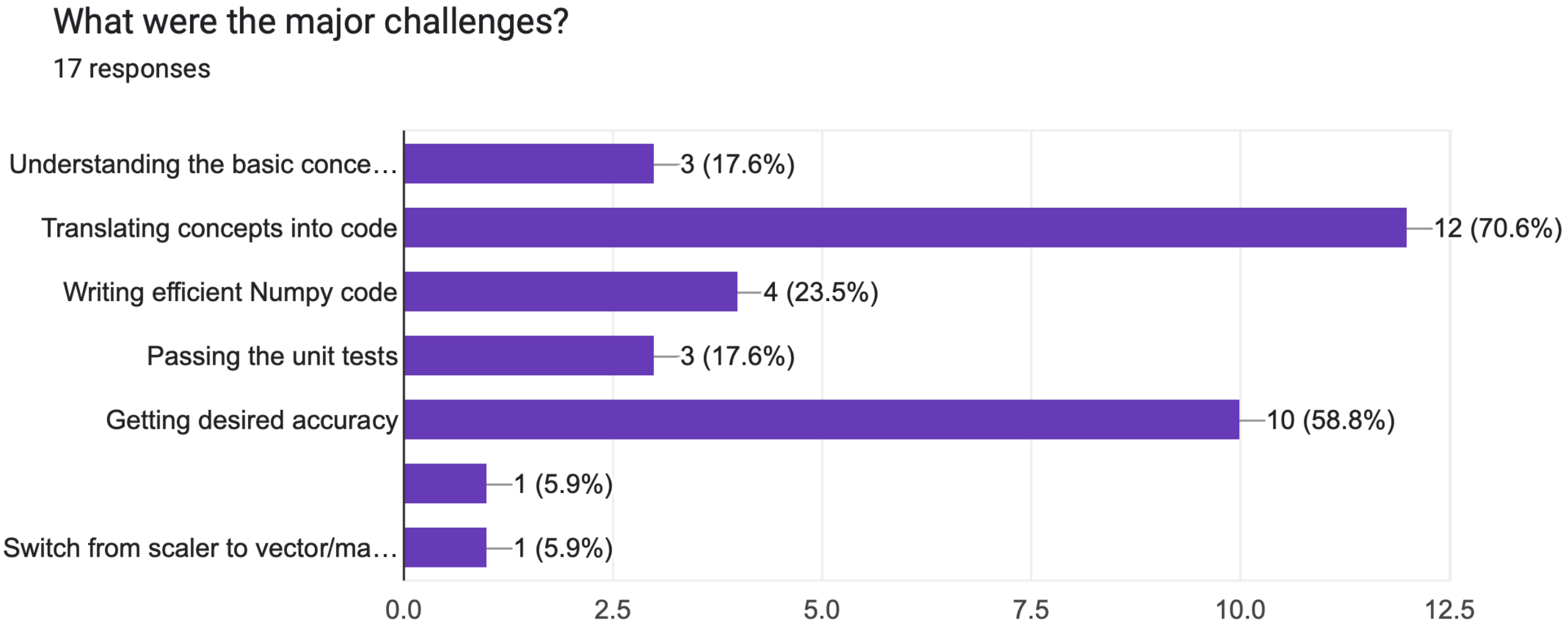
Feedback of working on PA1

When did you start working on it?

17 responses



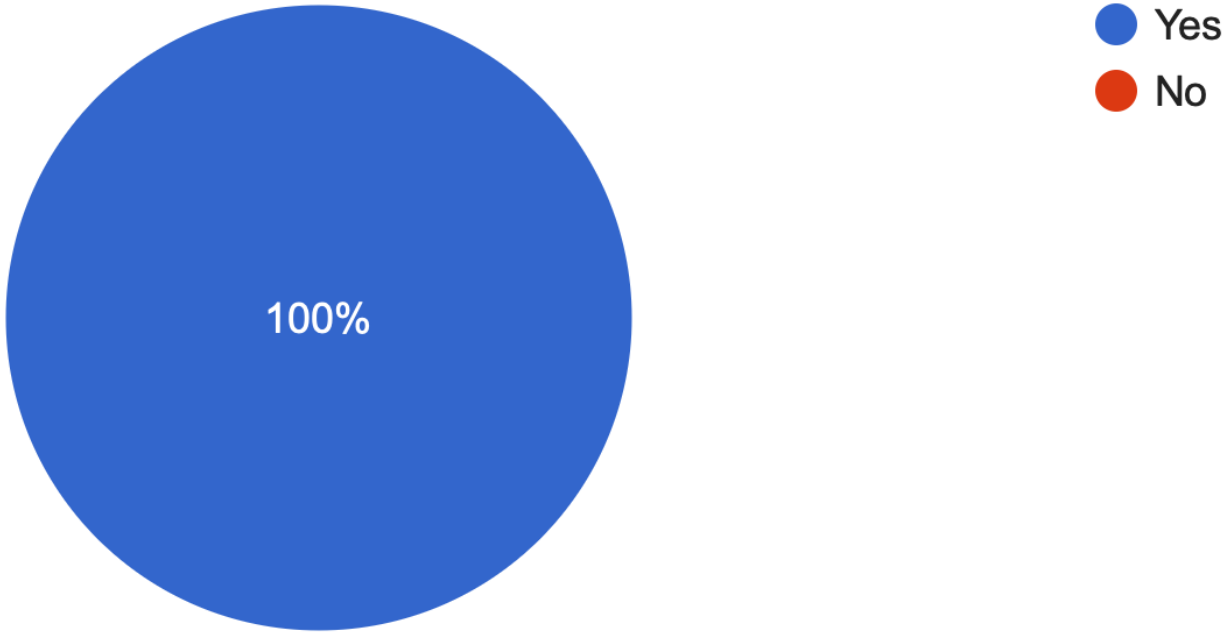
Feedback of working on PA1



Feedback of working on PA1

Did you find it useful for learning neural networks?

17 responses



Feedback of working on PA1

Could you please elaborate your previous choice especially if you selected "No"?

5 responses

I found the assignment very useful to understand the concepts from a practical point of view. It made me understand backpropagation much better, and how it actually works. This kinds of exercises/practical implementation would be perfect to consolidate the concepts before the quizzes.

Yes, I thought I knew how backprop works but realized it's somewhat more complicated than I initially assumed

Enhance the understanding.

It was useful to go through the process of implementing the networks manually.

N/A

Feedback of working on PA1

What could you have done better?

10 responses

do earlier to have more time doing hyperparameters tuning

didn't get the desired accuracy.. would be great if there were some hints or structures given

Nothing, I found it very clear.

checked deadline more carefully, thought it was midnight not 6pm

I felt this assignment was good.

More questions like this

Before writing the assignment, I will first choose the review the slides first, being more familar with the whole architectures to help me transfer all the concepts to the code.

Should start ealier.

I believe I did a pretty good job, but likely could have added more error handling. I just assumed we would be working with clean data.

N/A

Feedback of working on PA1

What could the instructor have done better?

11 responses

Bring candy has a student reinforcement learning training reward

like a markdown passage describing each section of the code in detail what each could should do, what is expected from the function and some structures and hints should have been provided

It would be nice to receive comments on the grade, if mistakes were made, to understand how to improve/avoid them. (I am filling this survey before getting the assignment grade, so this could already be put into practice)

could we get midnight deadlines instead?

NA

More time to solve quiz questions

The professor illustrated each components detailedly in class. It just necessary for me to review all the things before or during doing the assignment.

Feedback of working on PA1

n/a

I don't think there is anything you really need to improve for this assignment. It is relatively straightforward. If students are really struggling to find hyperparameters to meet accuracy requirements, maybe you can suggest a range for them to try.

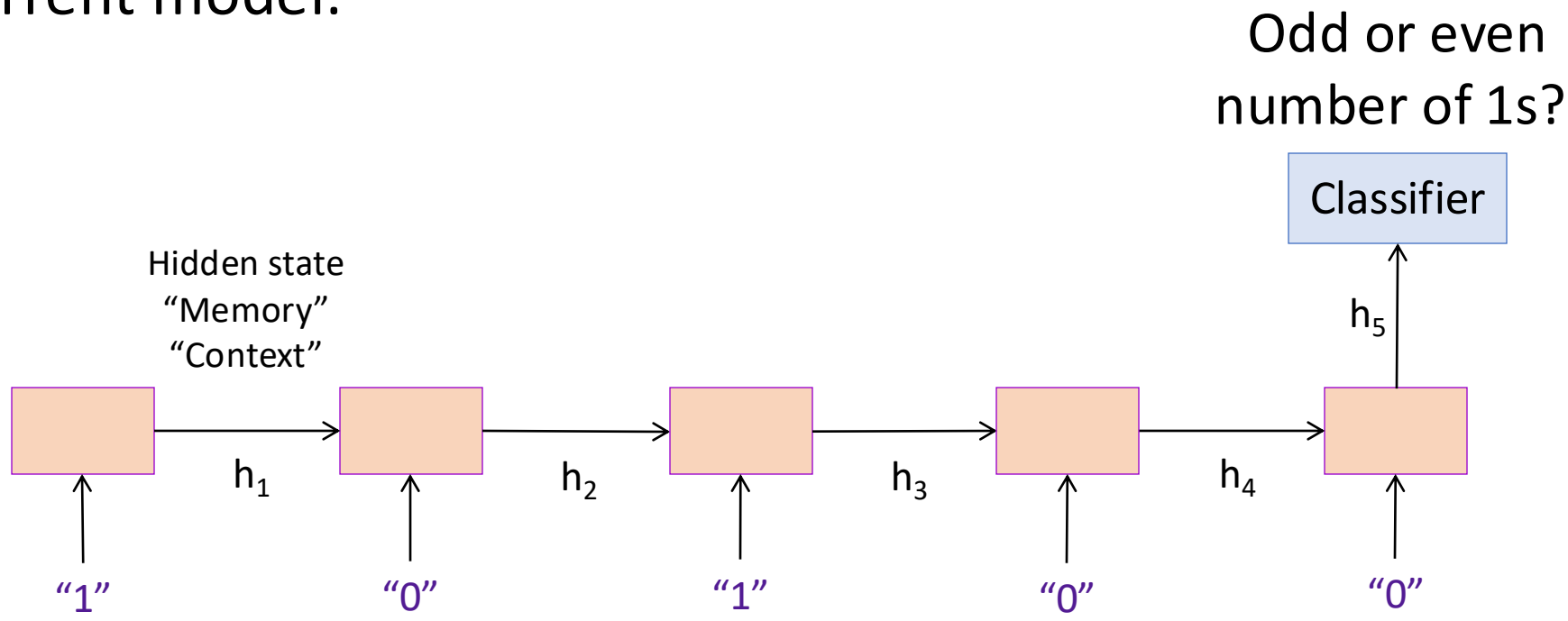
N/A

Maybe combine more code implementation with concepts covered in the class.

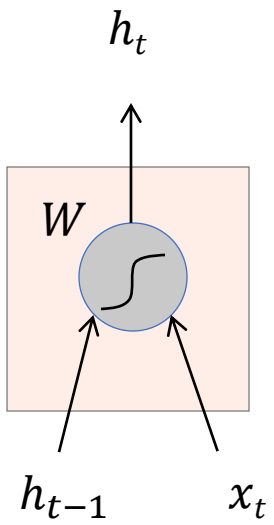
Recap

Sequence classification using a RNN

- Recurrent model:



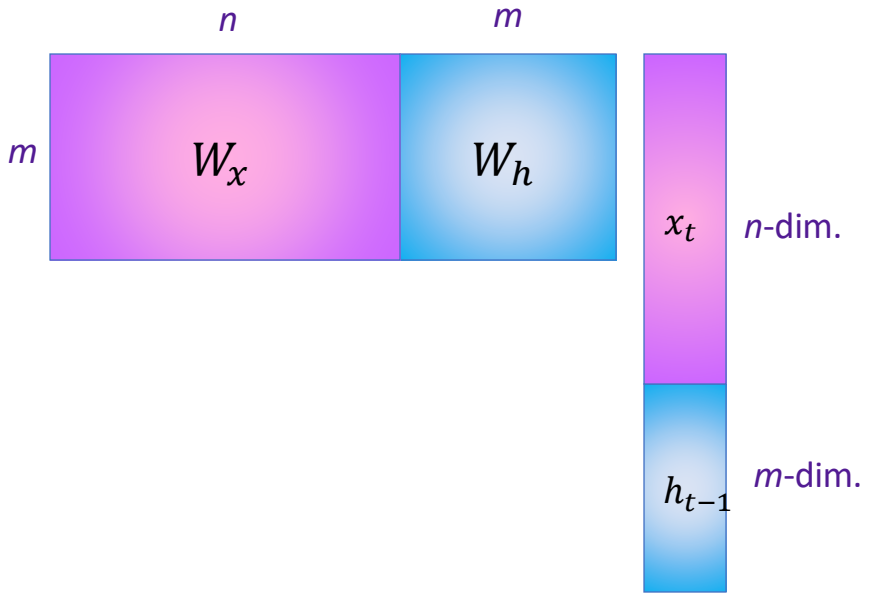
Vanilla RNN cell



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

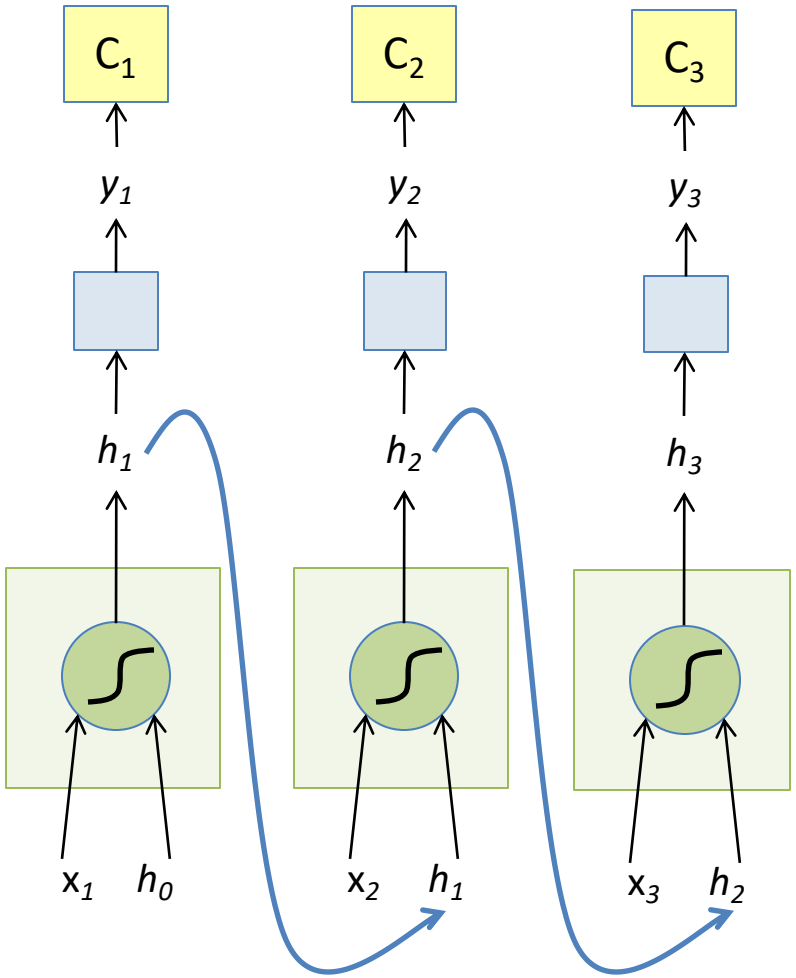
$$= \tanh W \begin{pmatrix} x_t \\ h_{t-1} \end{pmatrix}$$

$$= \tanh(W_x x_t + W_h h_{t-1})$$



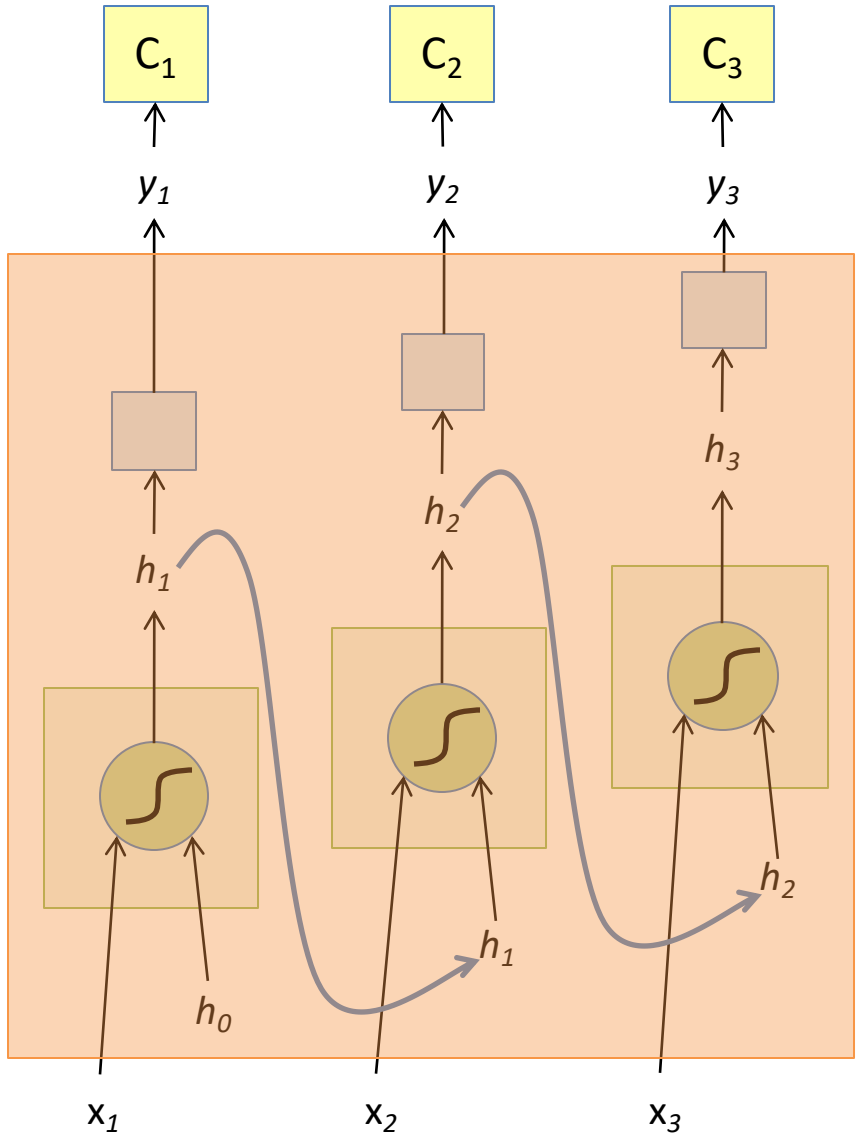
Slide credit: S. Lazebnik

The Unfolded Vanilla RNN Forward



Slide credit: A. Mallya

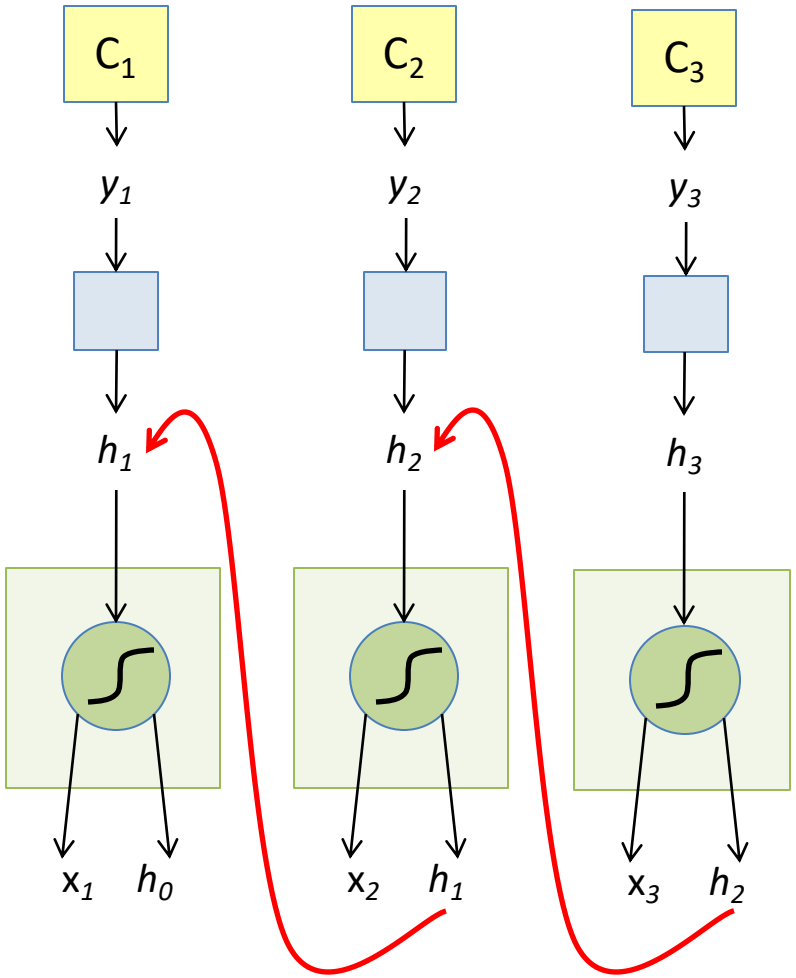
The Unfolded Vanilla RNN



- Treat the unfolded network as one big feed-forward network!
- This big network takes in entire sequence as an input
- Compute gradients through the usual backpropagation
- Update shared weights

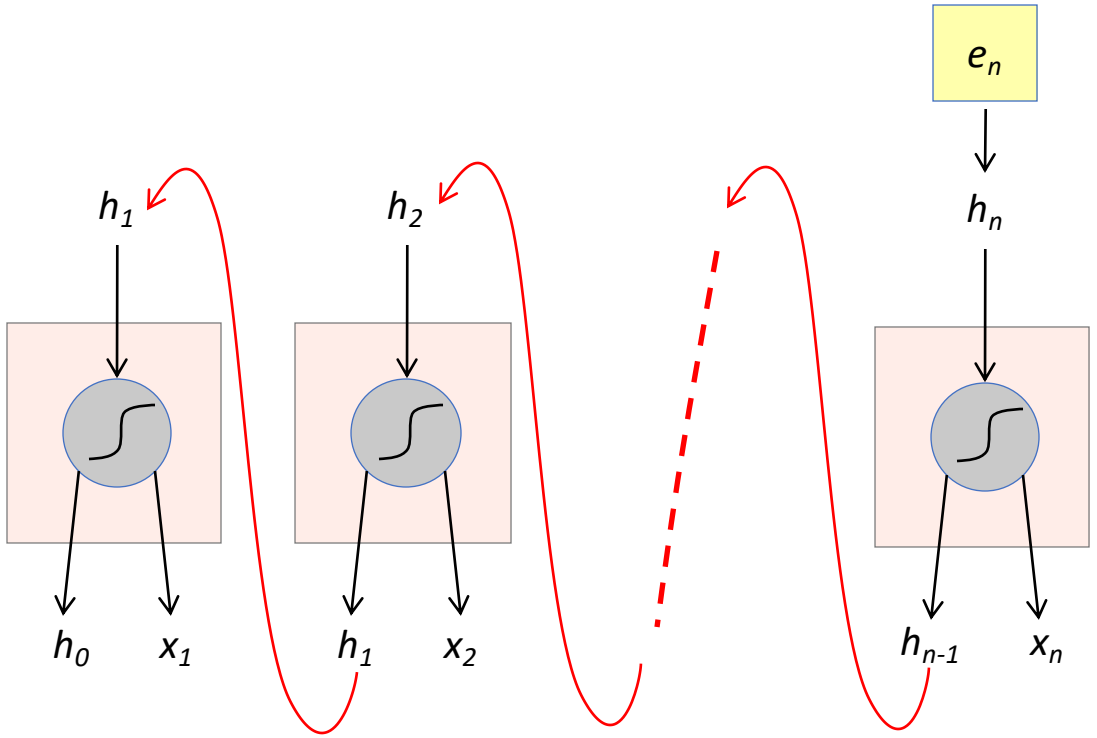
Slide credit: A. Mallya

The Unfolded Vanilla RNN Backward



Slide credit: A. Mallya

Vanishing and exploding gradients



$$\frac{\partial e}{\partial h_{t-1}} = W_h^T\left(1 - \tanh^2(W_x x_t + W_h h_{t-1})\right) \odot \frac{\partial e}{\partial h_t}$$

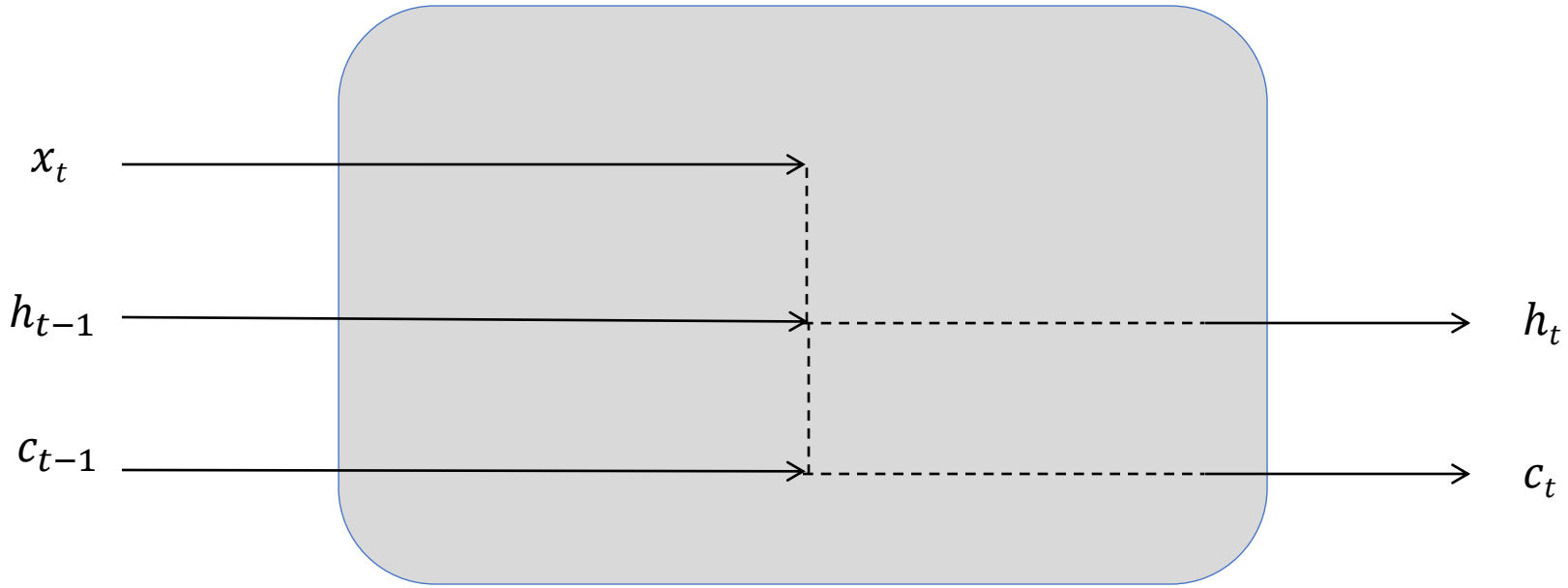
Computing gradient for h_0
involves many multiplications by W_h^T
(and rescalings between 0 and 1)

Gradients will *vanish* if largest singular value of W_h is less than 1
and *explode* if it's greater than 1

Slide credit: S. Lazebnik

Long short-term memory (LSTM)

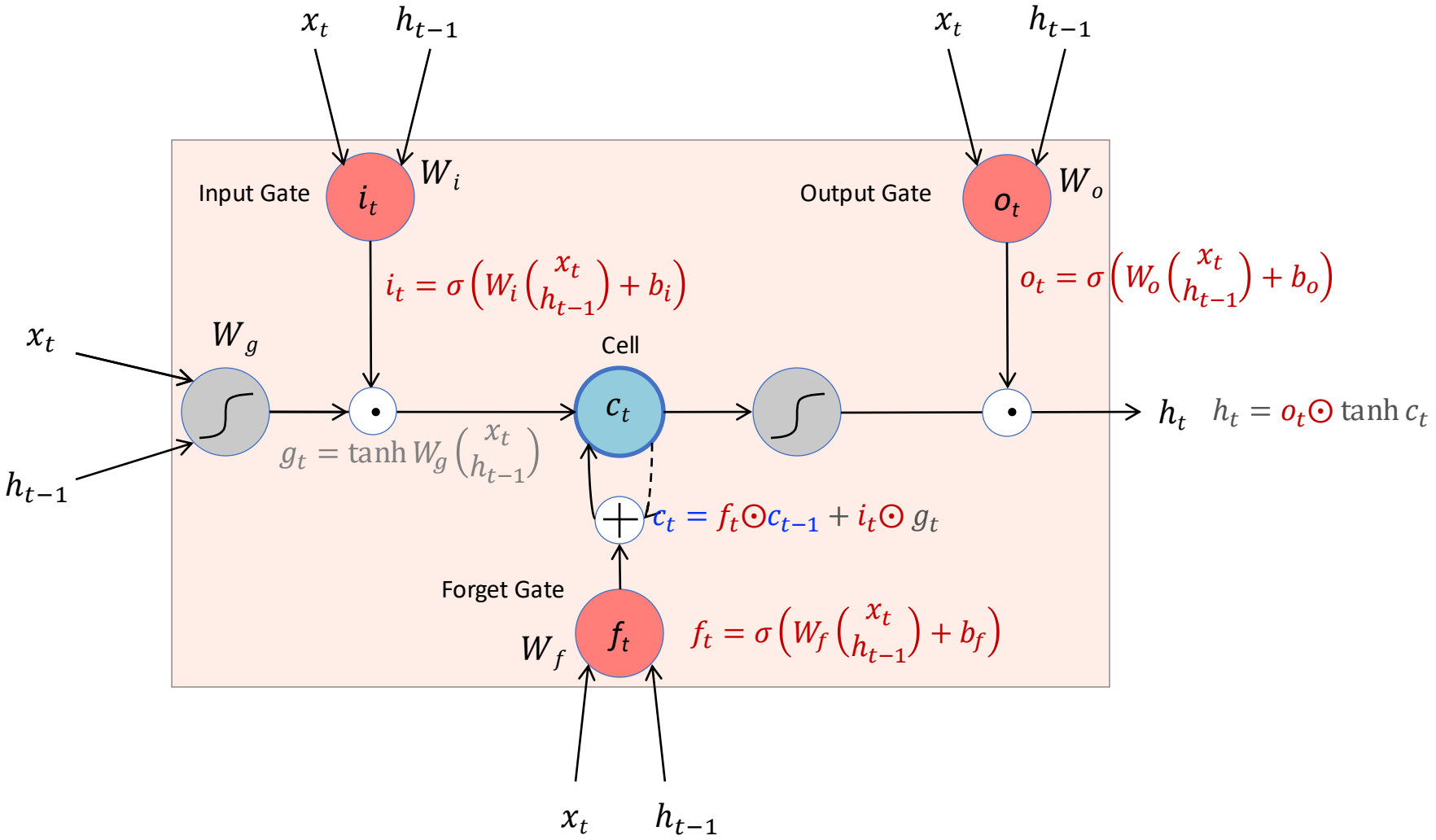
- Add a *memory cell* that is not subject to matrix multiplication or squishing, thereby avoiding gradient decay



S. Hochreiter and J. Schmidhuber, Long short-term memory, Neural Computation 9 (8), pp. 1735–1780, 1997

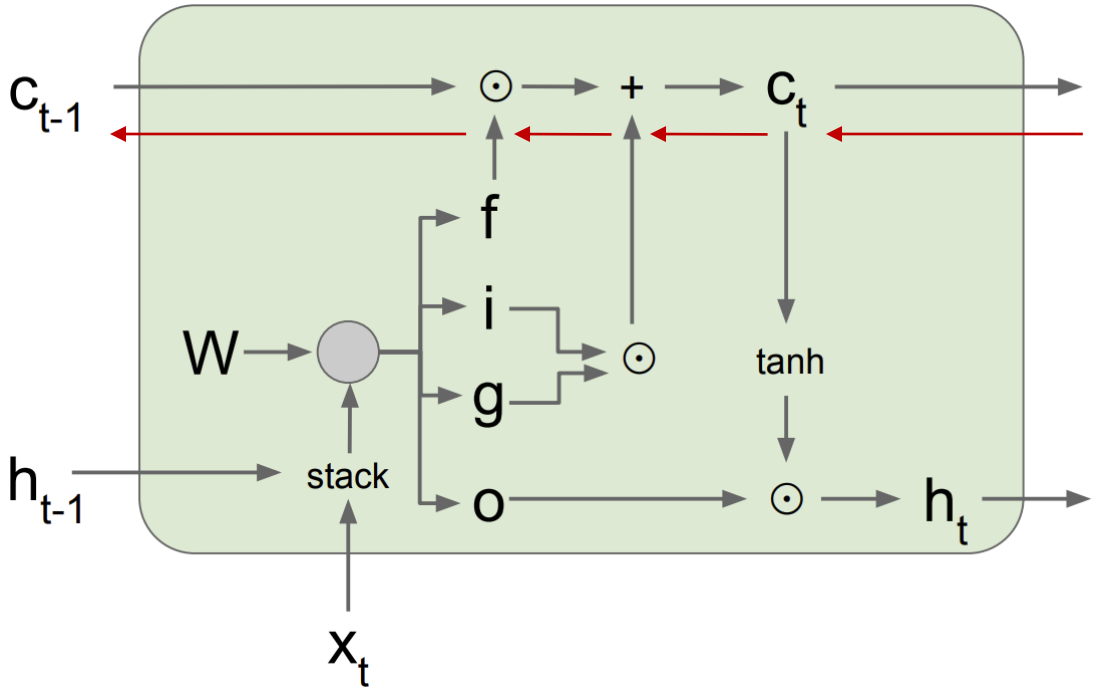
Slide credit: S. Lazebnik

The LSTM cell



Slide credit: S. Lazebnik

LSTM backward pass



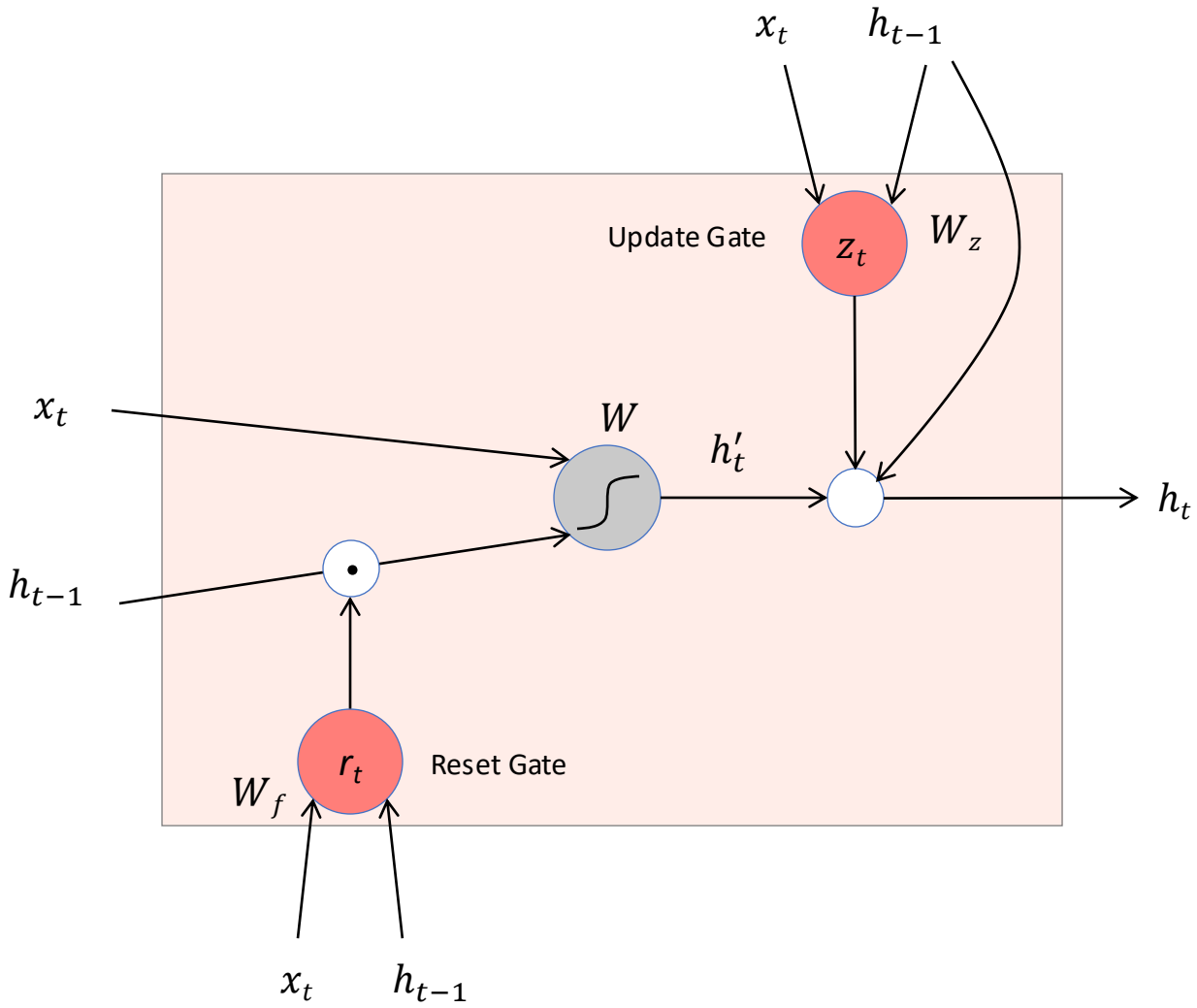
Gradient flow from c_t to c_{t-1} only involves back-propagating through addition and elementwise multiplication, not matrix multiplication or tanh

For complete details: Illustrated LSTM Forward and Backward Pass

Figure source

Slide credit: S. Lazebnik

Gated recurrent unit (GRU)



- Get rid of separate cell state
- Merge "forget" and "output" gates into "update" gate

K. Cho et al., Learning phrase representations using RNN encoder-decoder for statistical machine translation, ACL 2014

Slide credit: S. Lazebnik

RNN, LSTM, and GRU in Practice

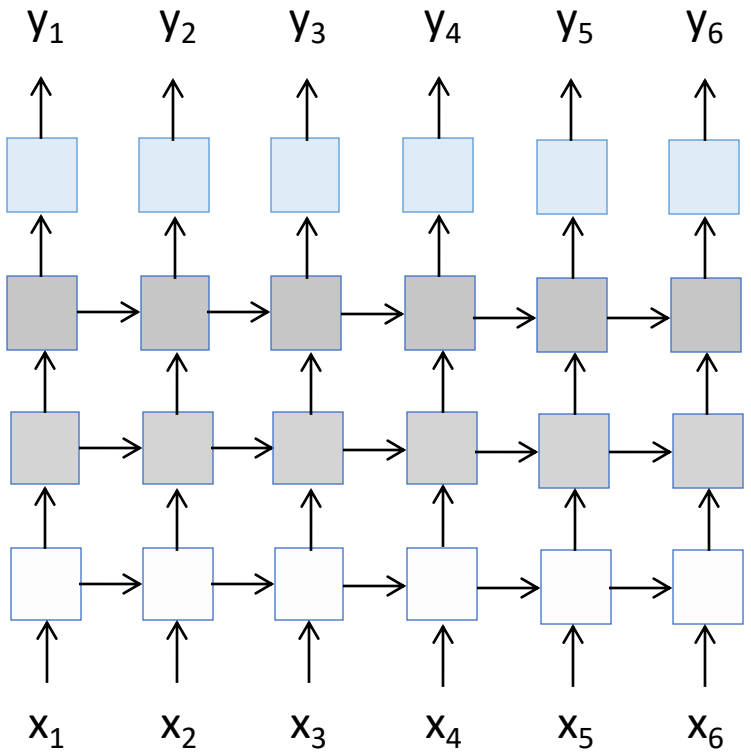
- Always use LSTM or GRU
  - No vanilla/raw RNN
  - Unless for educational purposes ☺
- LSTM vs GRU
  - Sometimes LSTM works well, sometimes GRU works well
  - GRU is simpler, less parameters, slightly faster

Today’s Class

- Attention in seq2seq model
- Transformer

Multi-layer RNNs

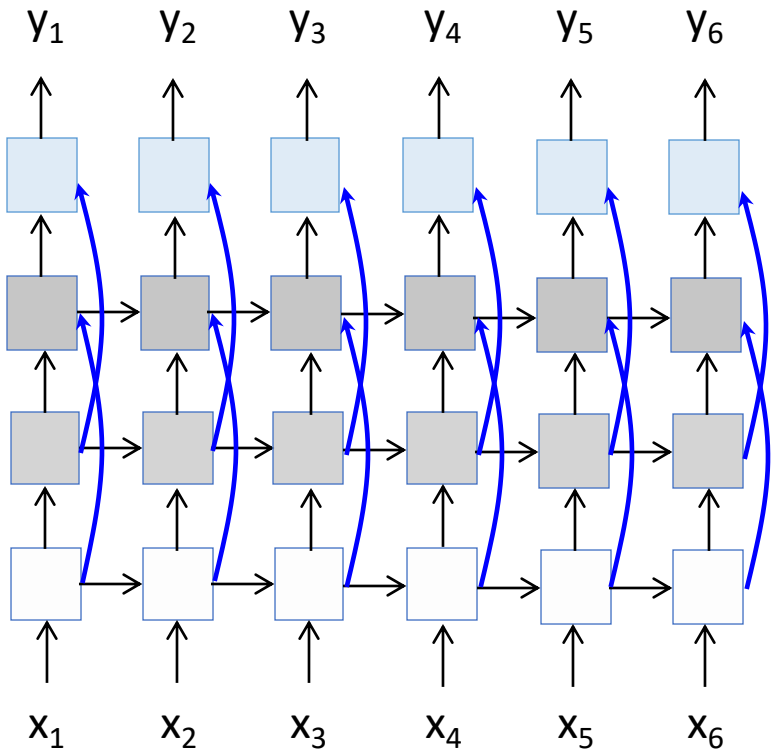
- We can of course design RNNs with multiple hidden layers



- Anything goes: skip connections across layers, across time, …

Multi-layer RNNs

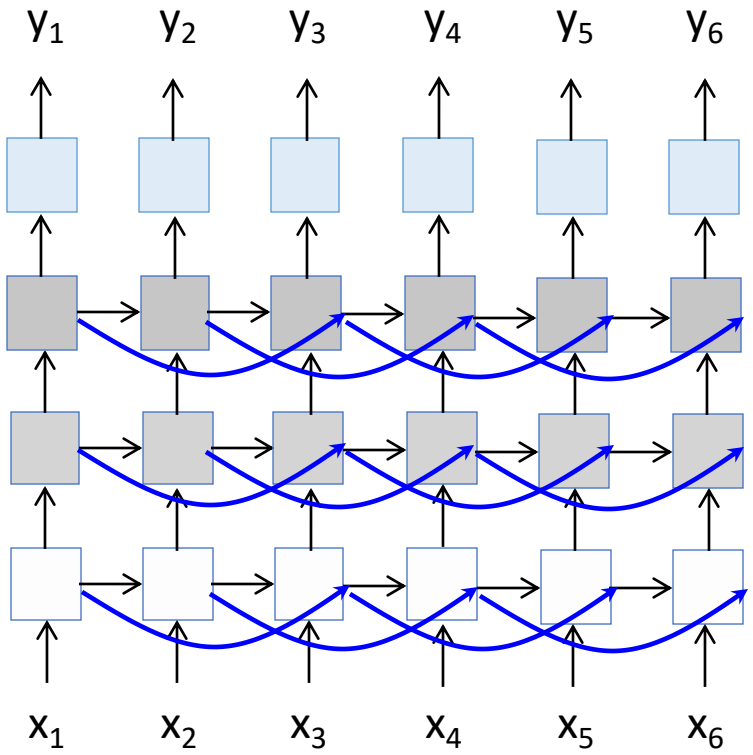
- We can of course design RNNs with multiple hidden layers



- Anything goes: skip connections across layers, across time, …

Multi-layer RNNs

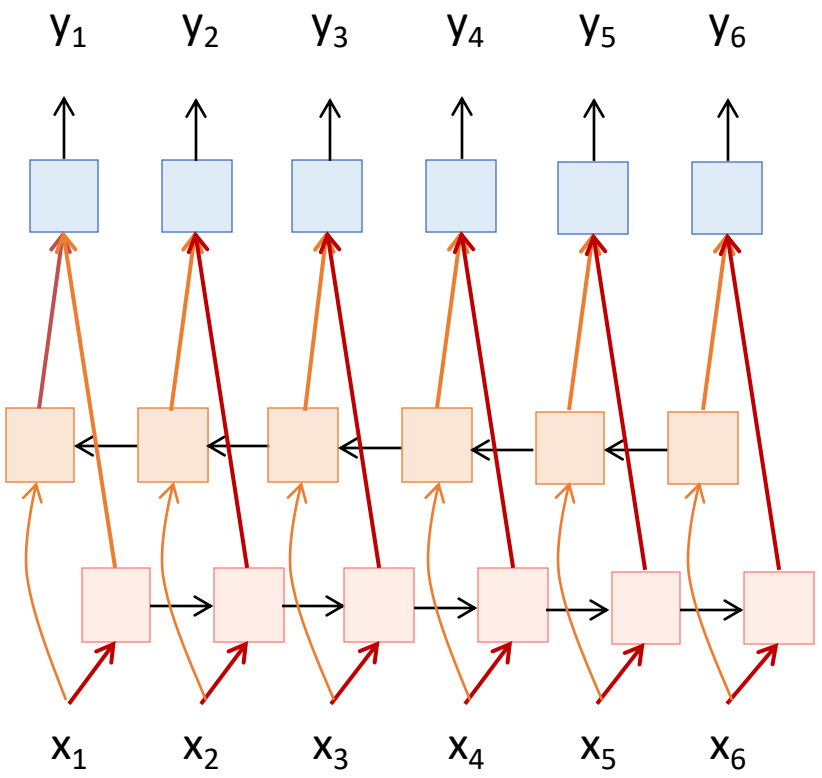
- We can of course design RNNs with multiple hidden layers



- Anything goes: skip connections across layers, across time, …

Bi-directional RNNs

- RNNs can process the input sequence in forward and in the reverse direction (common in speech recognition)



Recall: Batch Normalization for ConvNets

Batch Normalization for
fully-connected networks

$$x : N \times D$$

Normalize ↓

$$\mu, \sigma : 1 \times D$$

$$\gamma, \beta : 1 \times D$$

$$y = \frac{(x - \mu)}{\sigma}\gamma + \beta$$

Batch Normalization for
convolutional networks
(Spatial Batchnorm, BatchNorm2D)

$$x : N \times C \times H \times W$$

Normalize ↓ ↓ ↓

$$\mu, \sigma : 1 \times C \times 1 \times 1$$

$$\gamma, \beta : 1 \times C \times 1 \times 1$$

$$y = \frac{(x - \mu)}{\sigma}\gamma + \beta$$

How to use BatchNorm in an RNN?

Slide credit: D Fouhey & J Johnson

Naïve way of applying BN in RNN

Why it is not a good idea?

1. Variable lengths in the input.
2. The recurrent nature of RNN.

Batch Normalization for **recurrent** networks

$$x : N \times L \times C$$

$$\mu, \sigma : 1 \times 1 \times C$$

$$\gamma, \beta : 1 \times 1 \times C$$

$$y = \frac{(x - \mu)}{\sigma}\gamma + \beta$$

Layer Normalization

Computing of μ, σ is independent of the batch and length dimension.

But the learnable parameters γ, β are still for each channel.

Largely adopted in Transformer.

Layer Normalization for **recurrent** networks

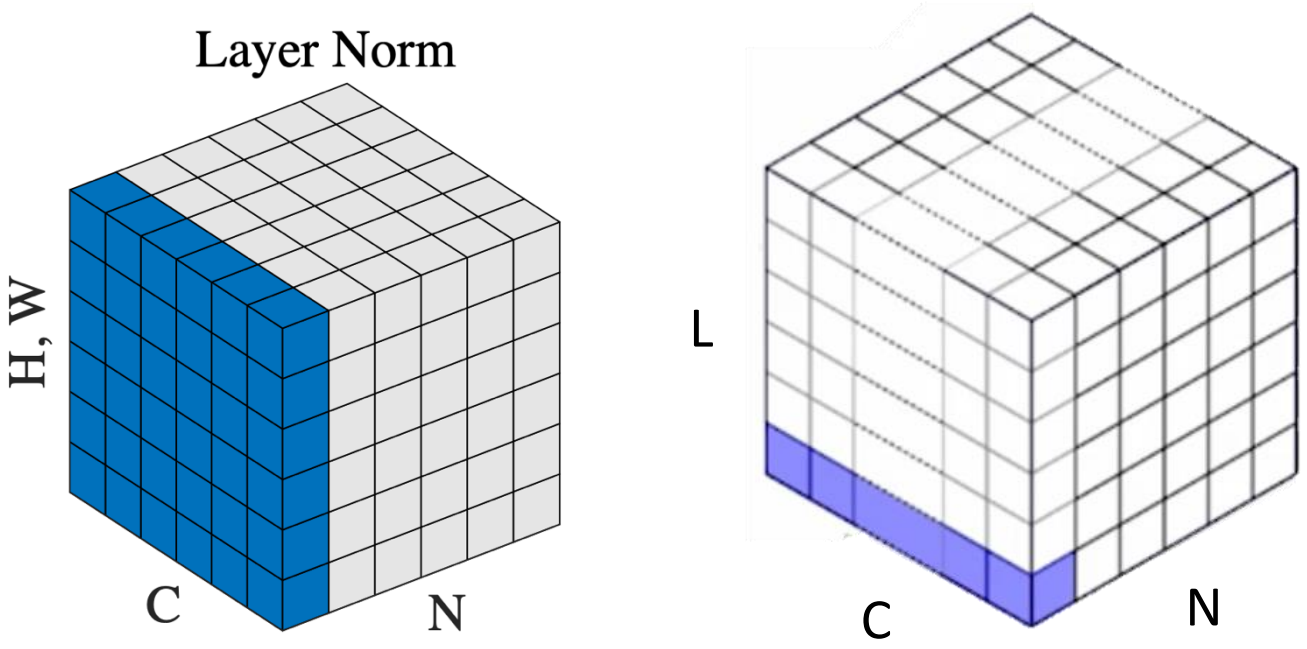
$$x : N \times L \times C$$

$$\mu, \sigma : N \times L \times 1$$

$$\gamma, \beta : 1 \times 1 \times C$$

$$y = \frac{(x - \mu)}{\sigma}\gamma + \beta$$

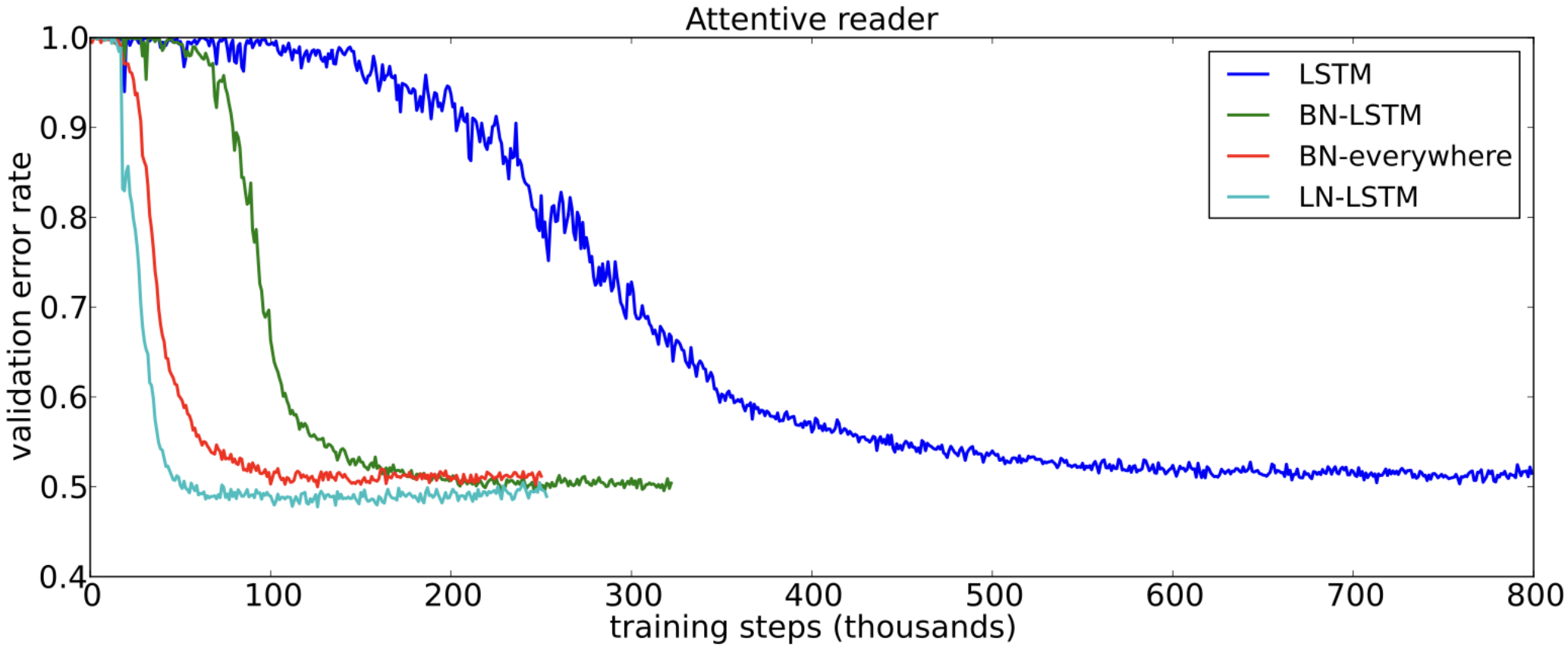
A subtle thing: LayerNorm in CNN vs RNN



In feedforward networks (i.e., CNNs), H, W are processed in parallel.
But in recurrent networks (i.e., RNNs), L are processed sequentially.

[Wu and He. Group Normalization. ECCV 2018. Best paper honorable mention.]

Normalization in RNN

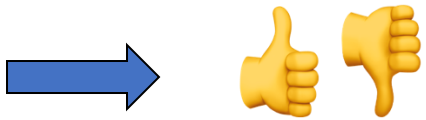


BatchNorm is possible (but not the raw BN): https://arxiv.org/pdf/1603.09025.pdf

Use LayerNorm instead: https://arxiv.org/pdf/1607.06450.pdf

Sequence classification

"The food is usually not so good"



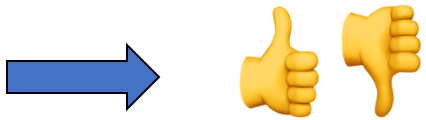
One-hot encoding

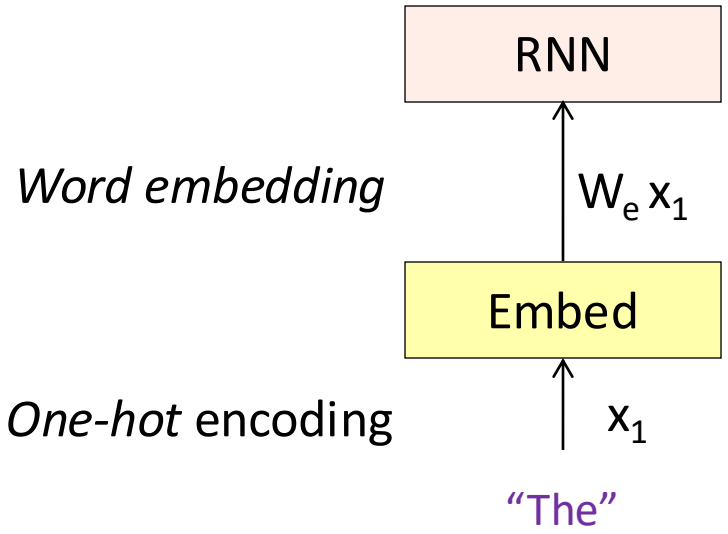
x_1

"The"

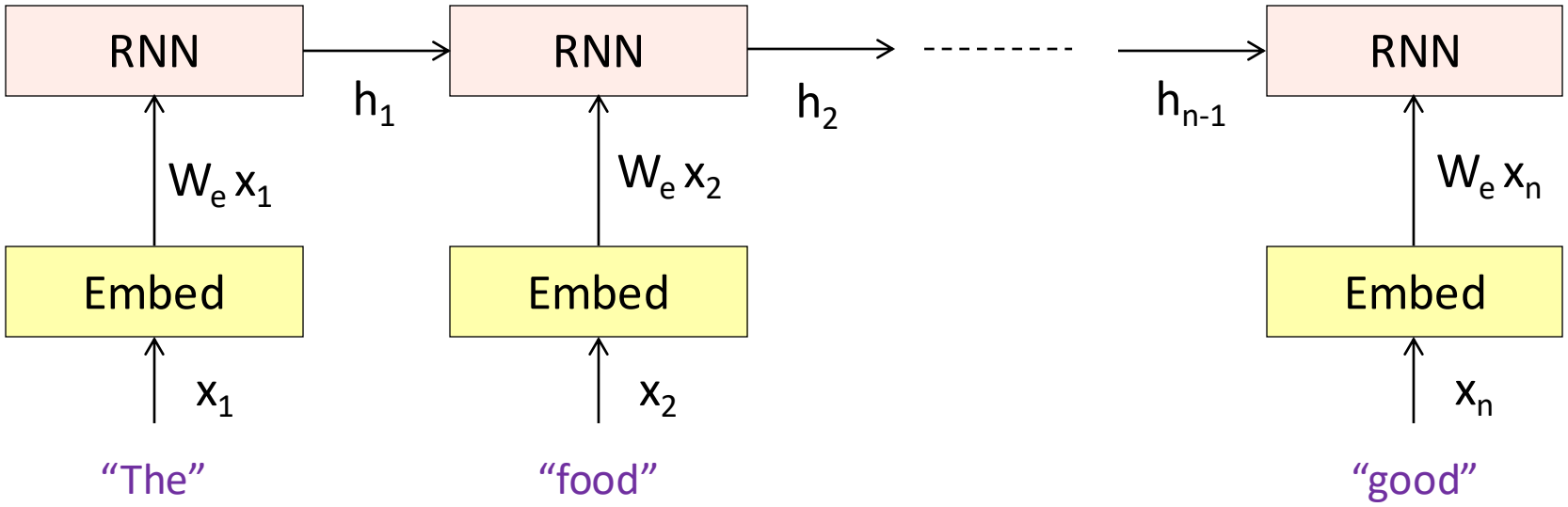
Sequence classification

"The food is usually not so good"

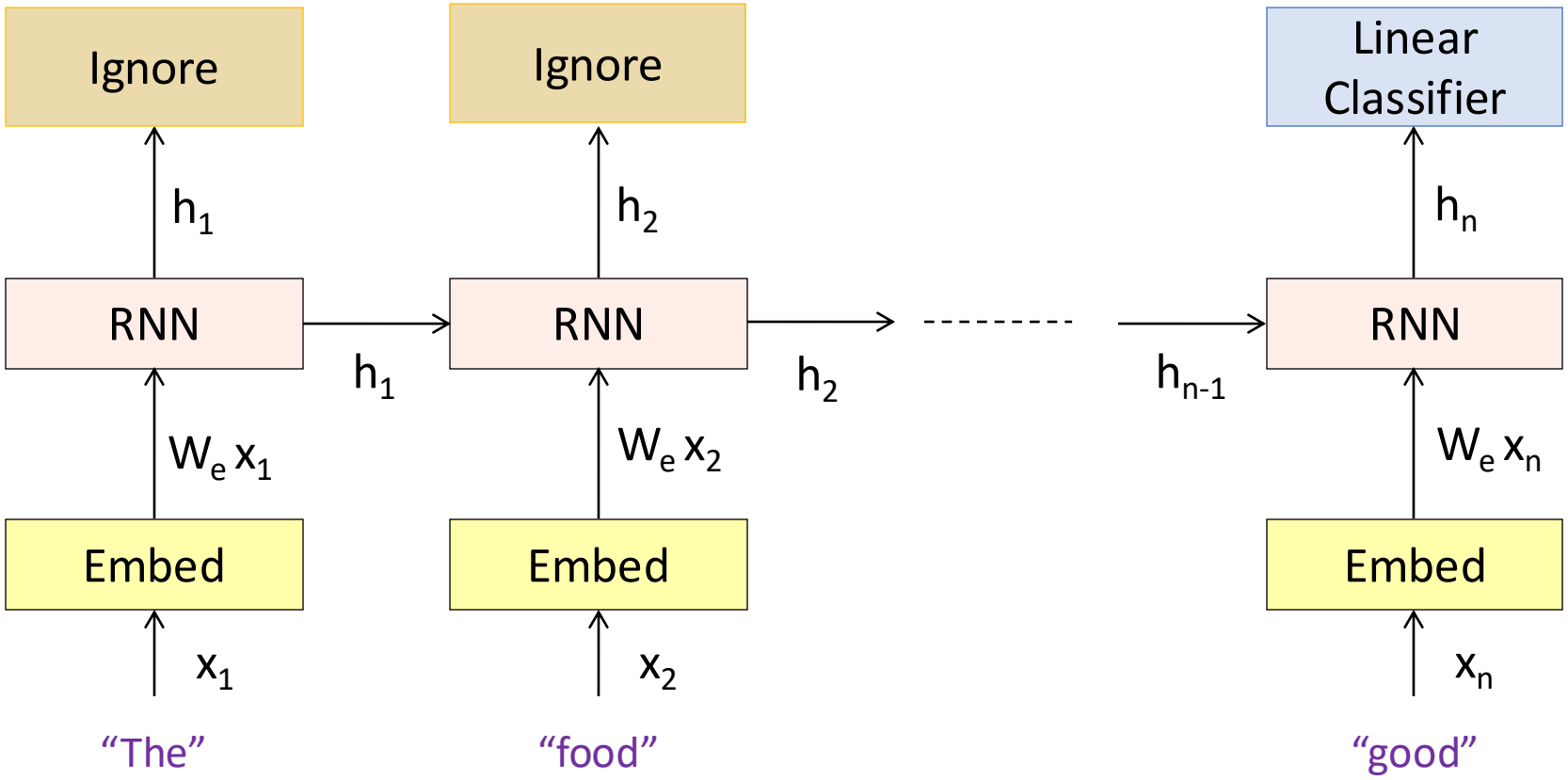




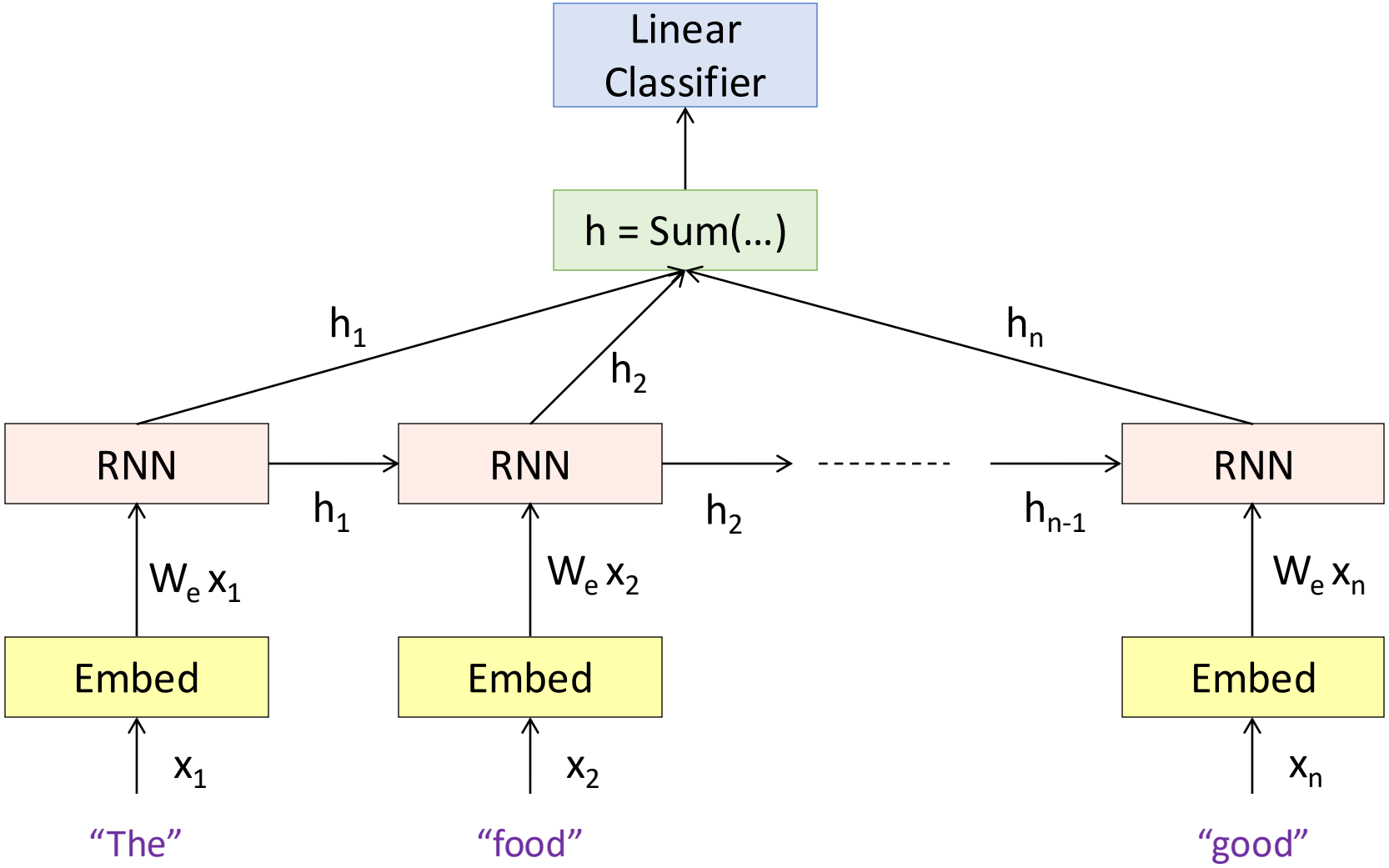
Sequence classification



Sequence classification

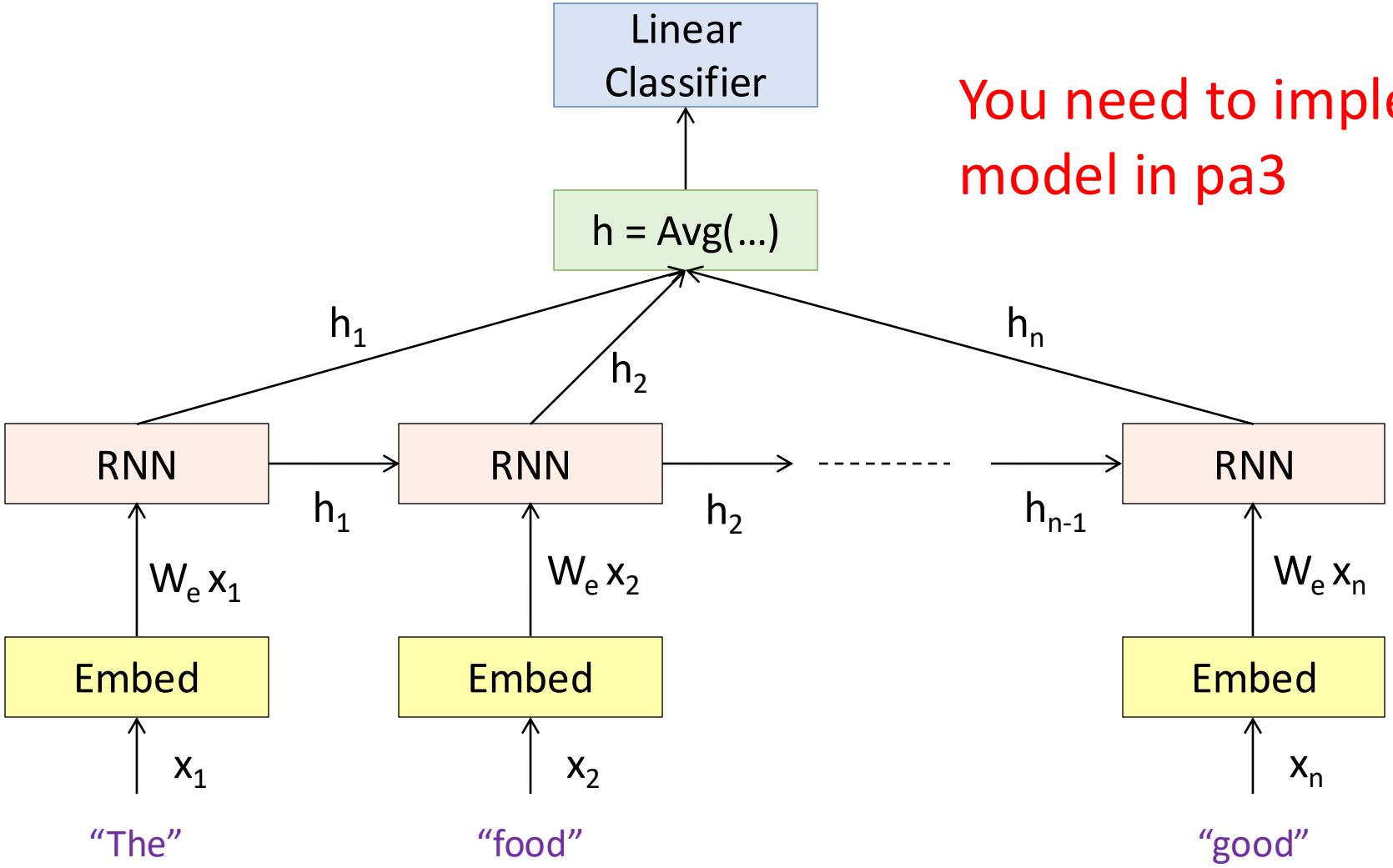


Sequence classification



http://deeplearning.net/tutorial/lstm.html

Sequence classification



You need to implement this model in pa3

http://deeplearning.net/tutorial/lstm.html

Sequence classification

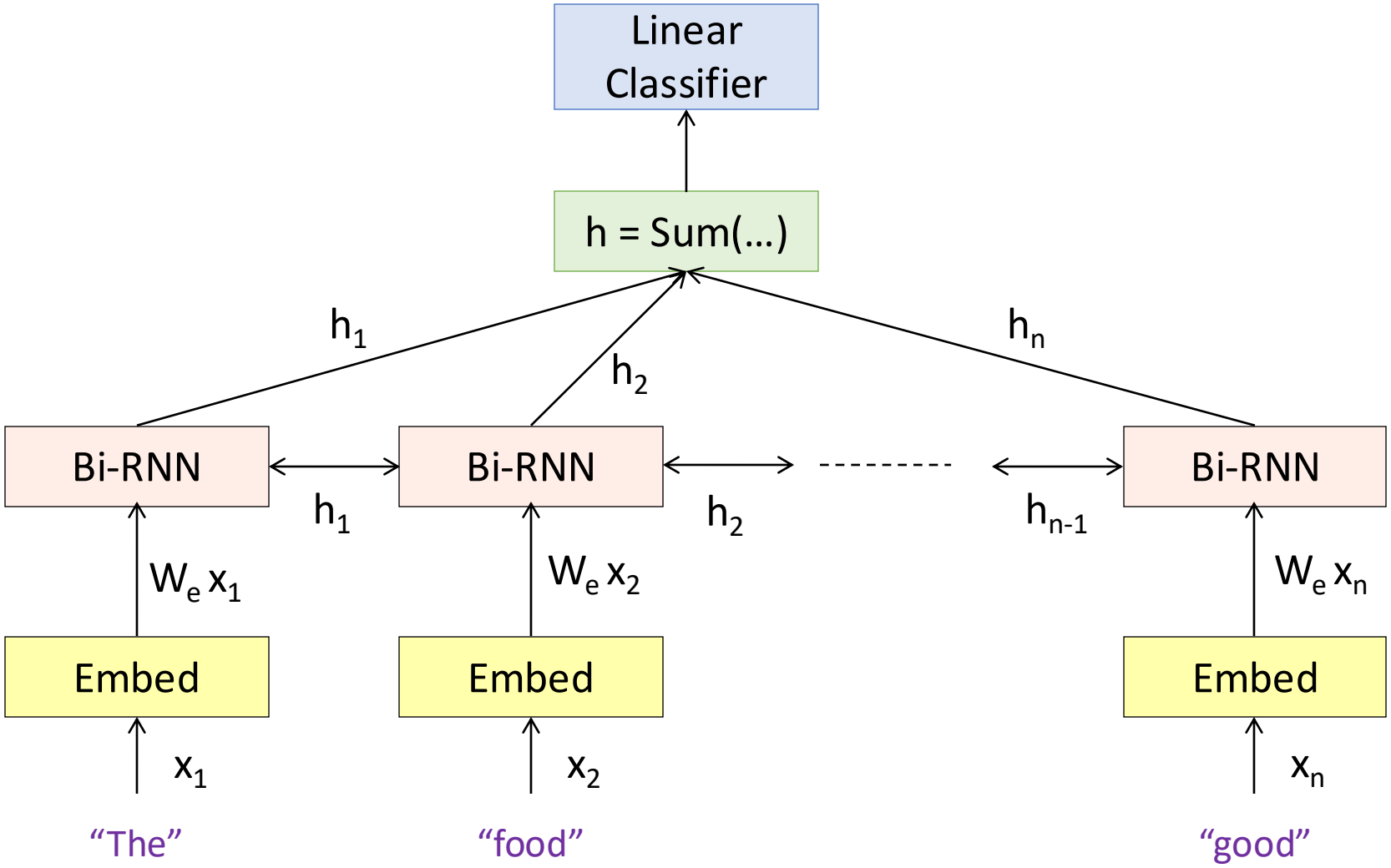
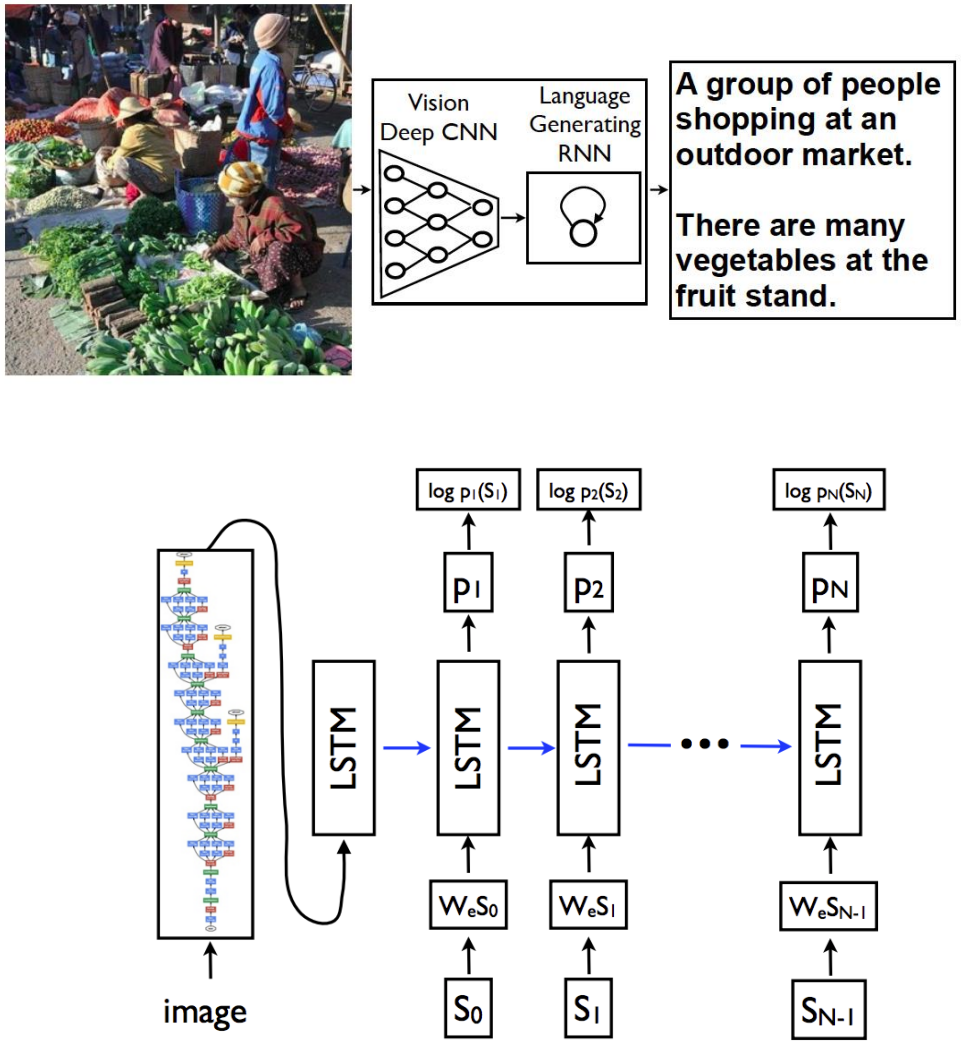


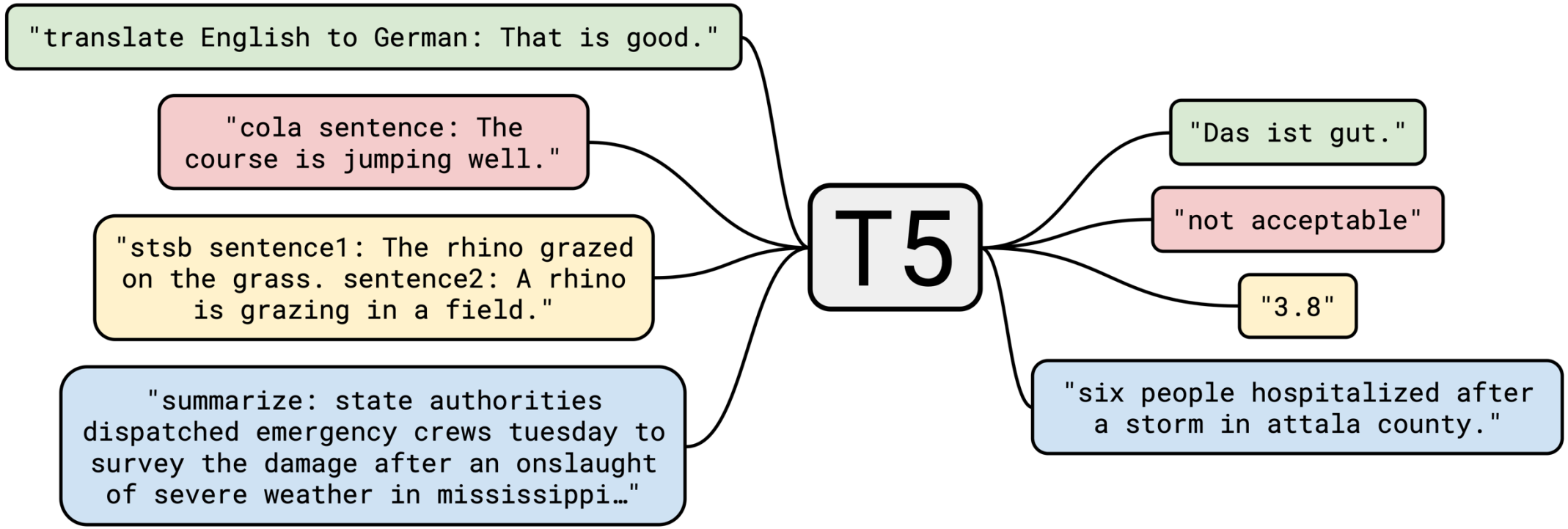
Image caption generation



O. Vinyals, A. Toshev, S. Bengio, D. Erhan, Show and Tell: A Neural Image Caption Generator, CVPR 2015

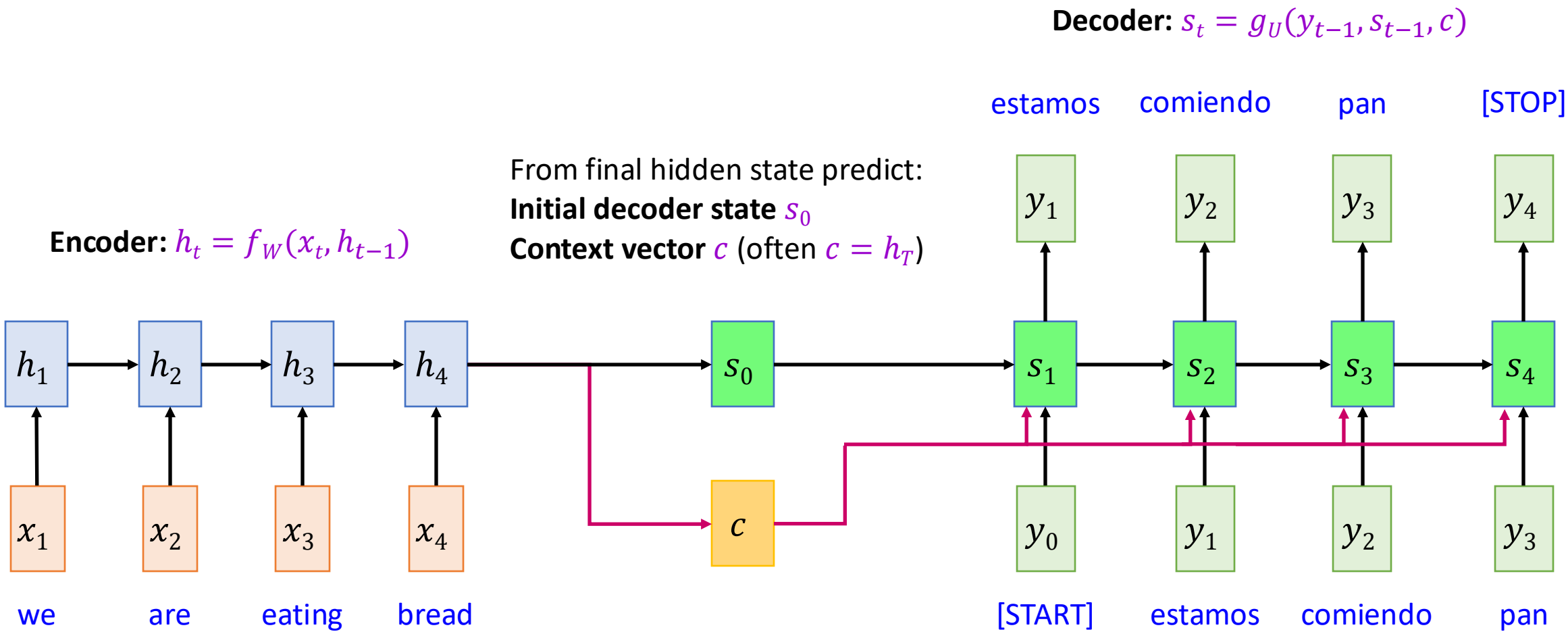
Sequence-to-sequence task

Text-to-text generation to unify different NLP tasks

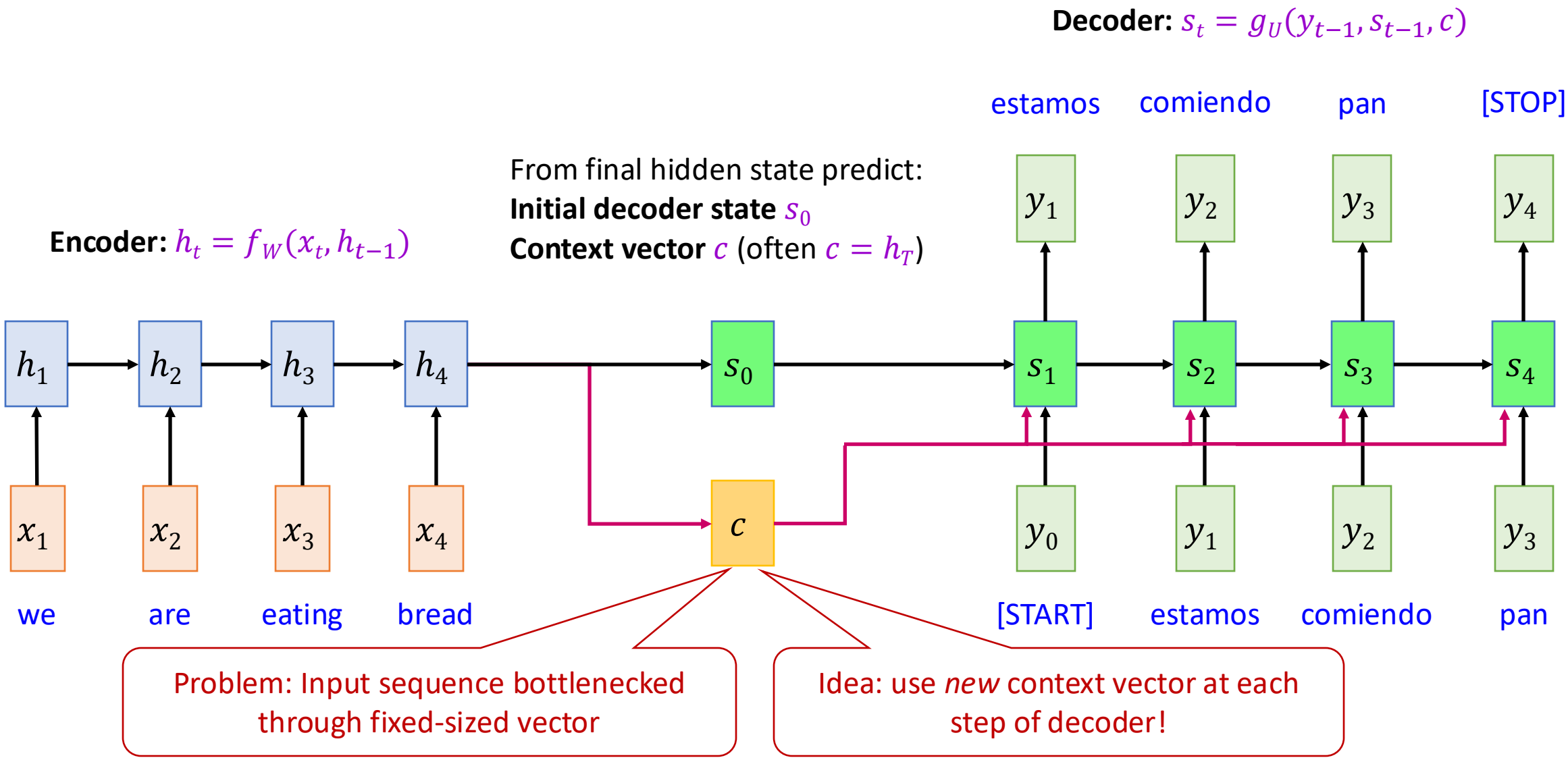


https://arxiv.org/abs/1910.10683

Sequence-to-sequence with RNNs



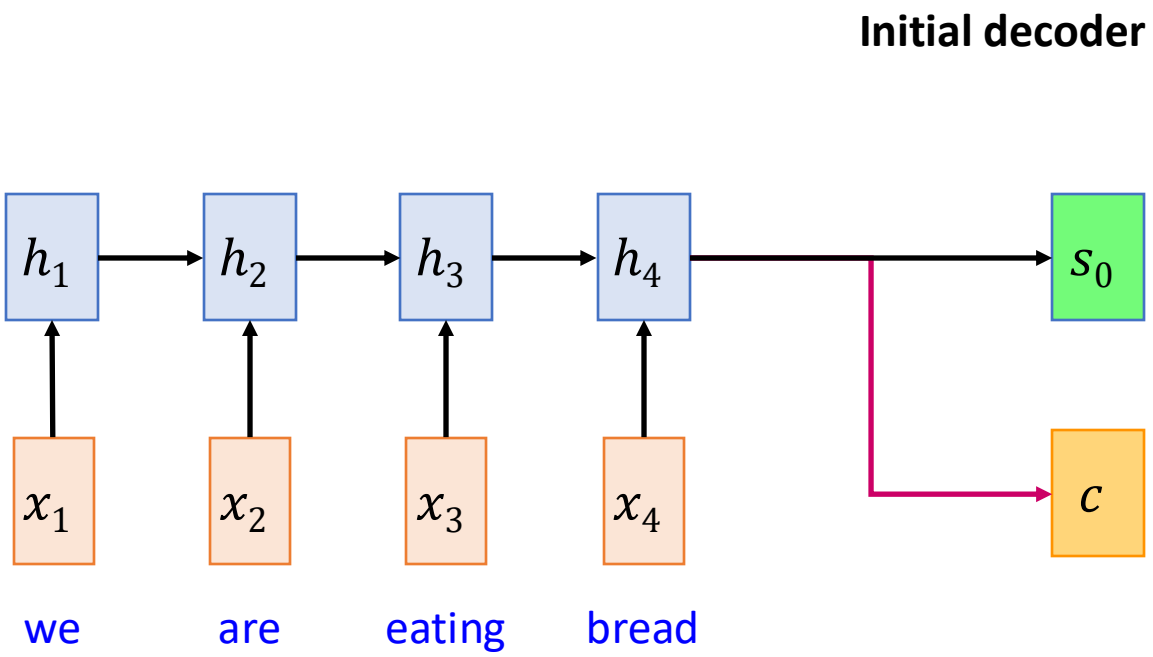
Sequence-to-sequence with RNNs



A. Sutskever, O. Vinyals, Q. Le, Sequence to sequence learning with neural networks, NeurIPS 2014

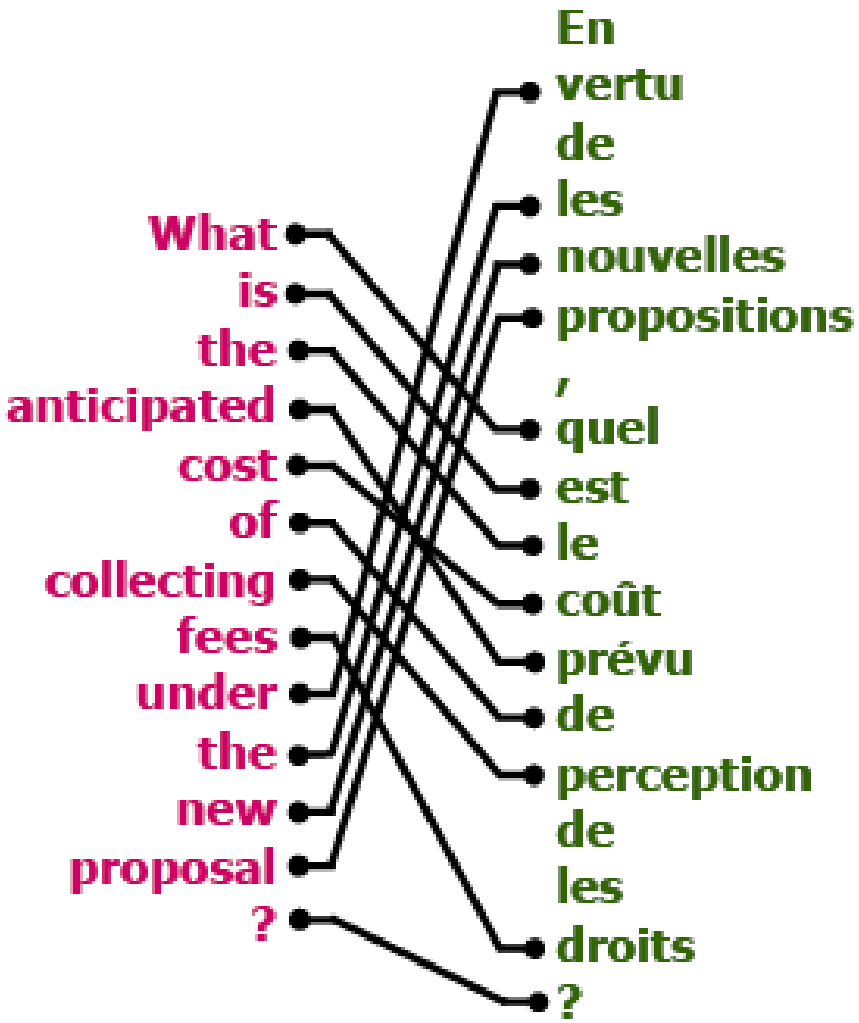
Sequence-to-sequence with RNNs and attention

- At each timestep of decoder, context vector “looks at” different parts of the input sequence

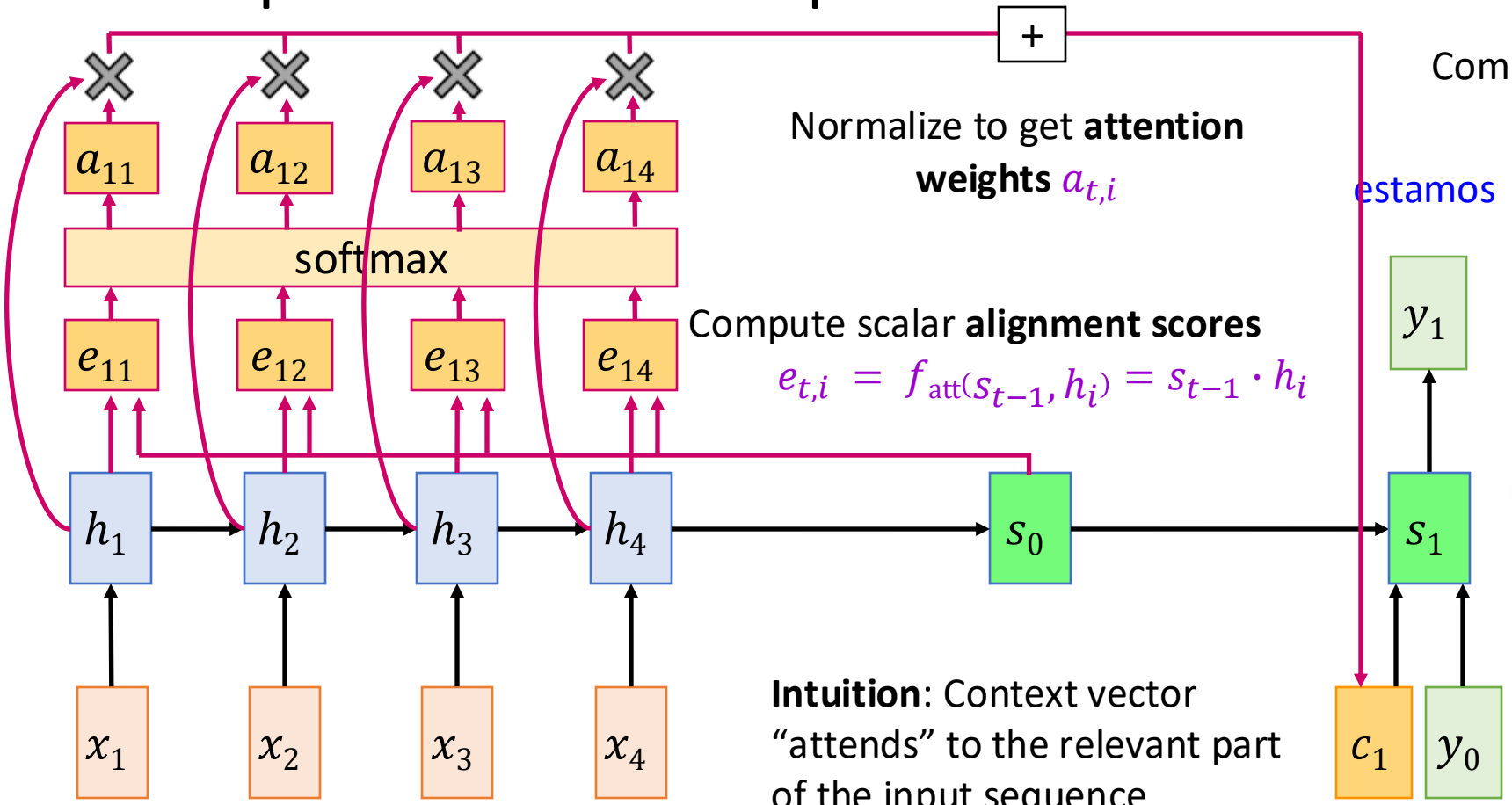


Sequence-to-sequence with RNNs and attention

- Intuition: translation requires *alignment*



Sequence-to-sequence with RNNs and attention



Normalize to get **attention weights** $a_{t,i}$

Compute scalar **alignment scores**

$$e_{t,i} = f_{\text{att}}(s_{t-1}, h_i) = s_{t-1} \cdot h_i$$

Compute **context vector** as

$$c_t = \sum_i a_{t,i} h_i$$

Use context vector in decoder:

$$s_t = g_U(y_{t-1}, s_{t-1}, c_t)$$

Intuition: Context vector "attends" to the relevant part of the input sequence

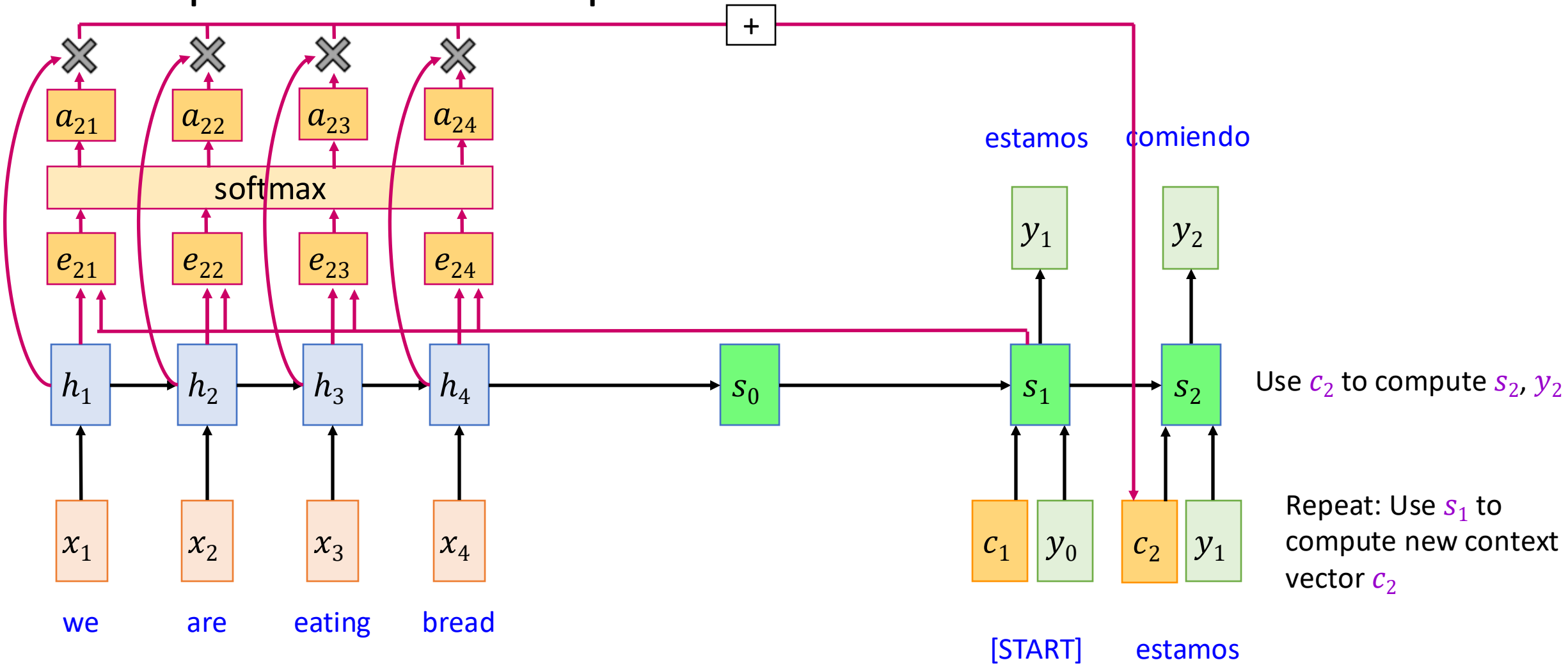
"estamos" = *"we are"*

so maybe $a_{11} = a_{12} = 0.45$,

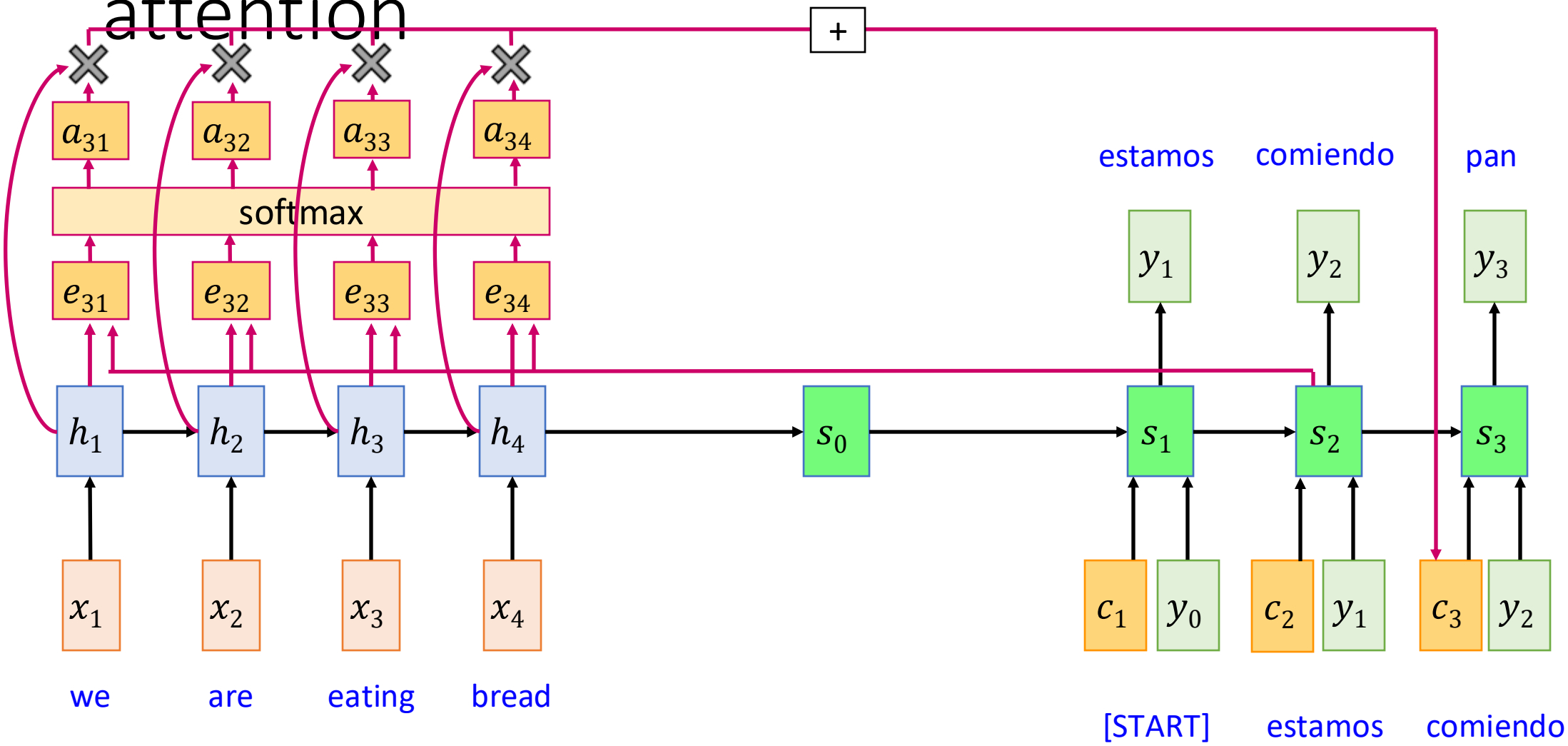
$a_{13} = a_{14} = 0.05$

D. Bahdanau, K. Cho, Y. Bengio, Neural Machine Translation by Jointly Learning to Align and Translate, ICLR 2015

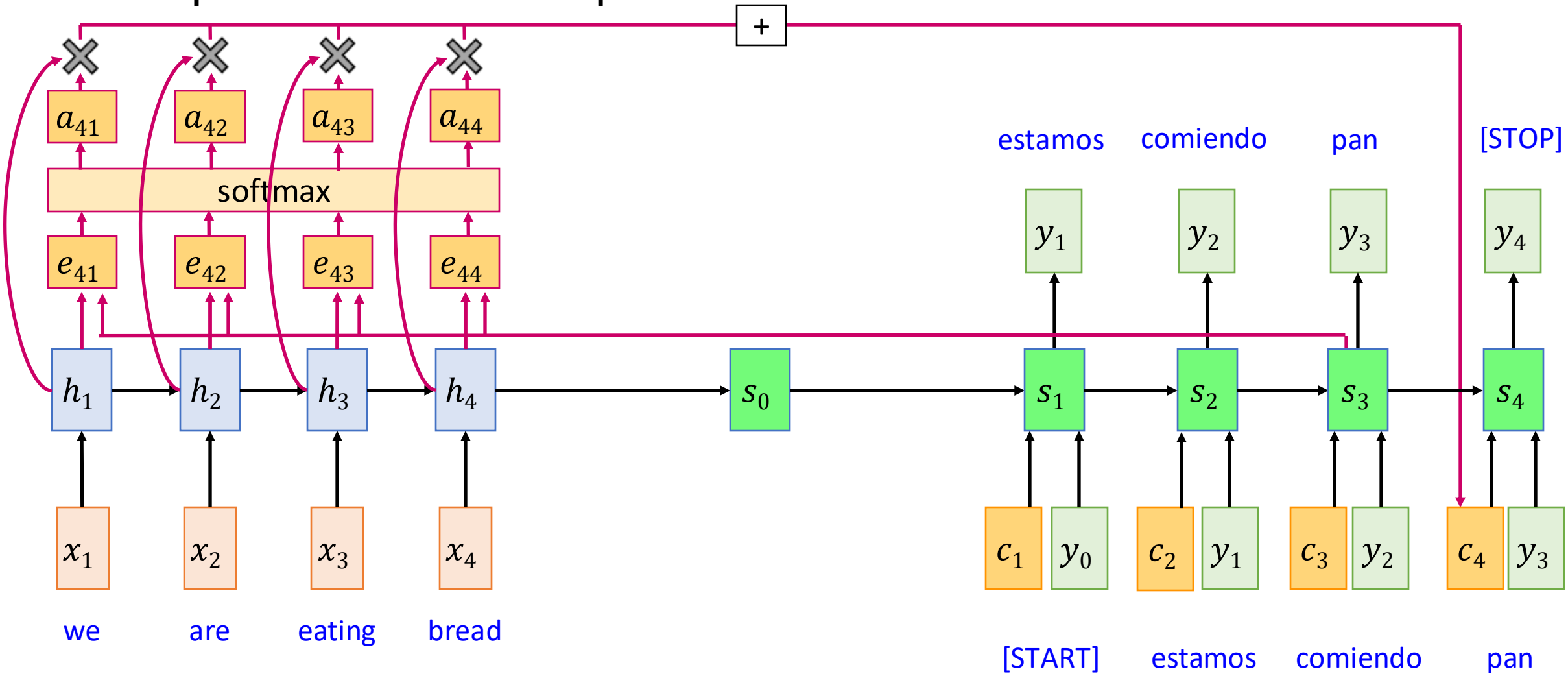
Sequence-to-sequence with RNNs and attention



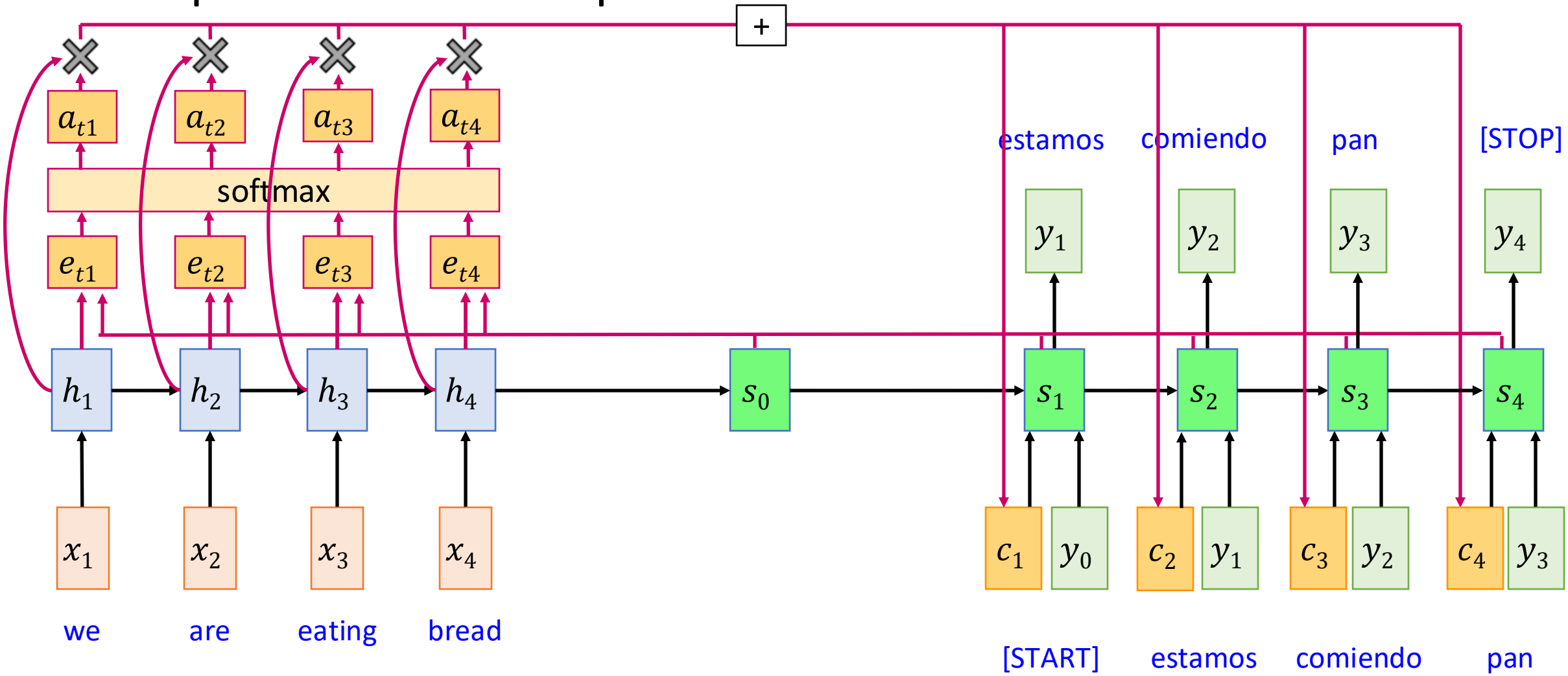
Sequence-to-sequence with RNNs and attention



Sequence-to-sequence with RNNs and attention

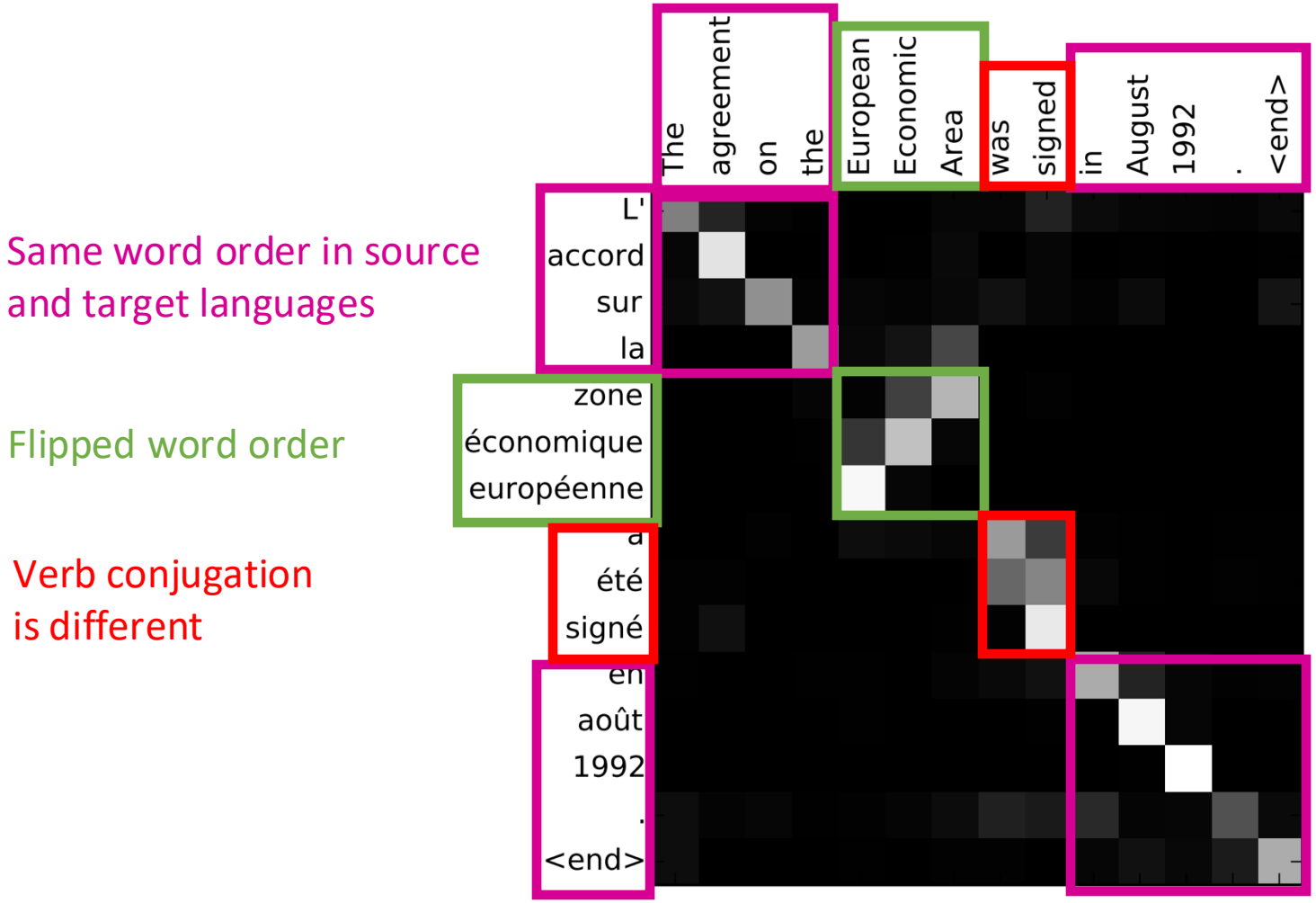


Sequence-to-sequence with RNNs and attention



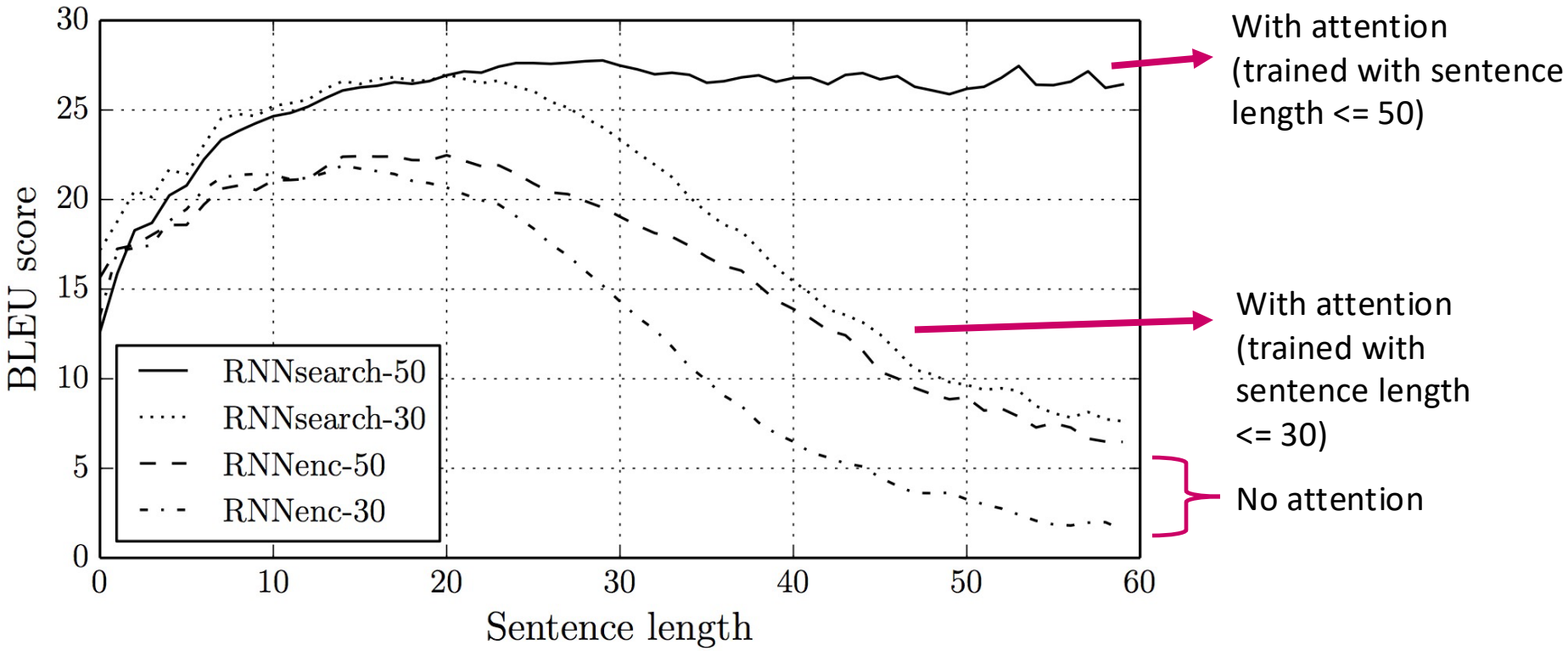
Sequence-to-sequence with RNNs and attention

- Visualizing attention weights (English source, French target):



D. Bahdanau, K. Cho, Y. Bengio, Neural Machine Translation by Jointly Learning to Align and Translate, ICLR 2015

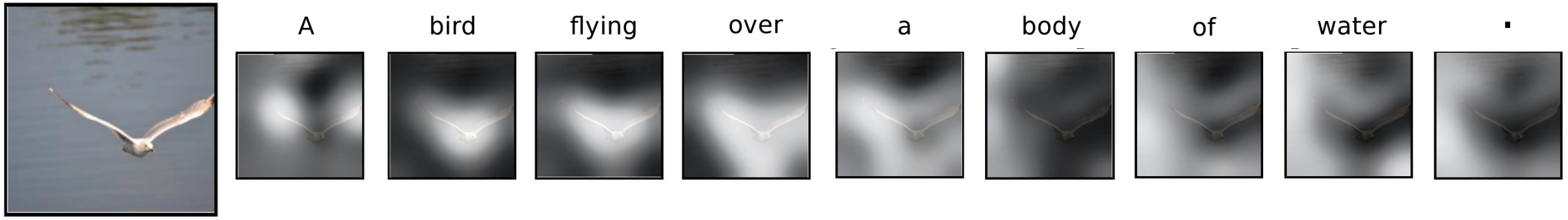
Quantitative evaluation



D. Bahdanau, K. Cho, Y. Bengio, Neural Machine Translation by Jointly Learning to Align and Translate, ICLR 2015

Image captioning with RNNs and attention

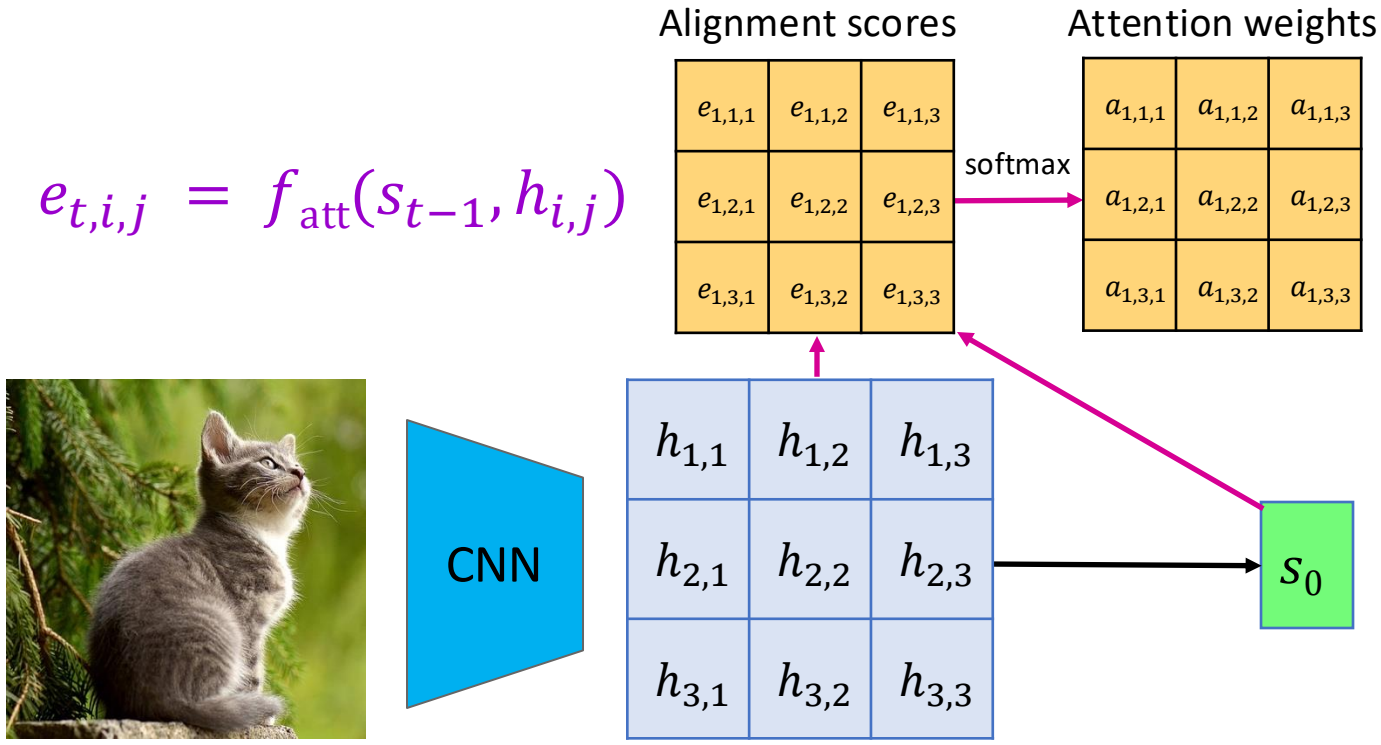
- Idea: pay attention to different parts of the image when generating different words
- Automatically learn this *grounding* of words to image regions without direct supervision



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

Image captioning with RNNs and attention

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

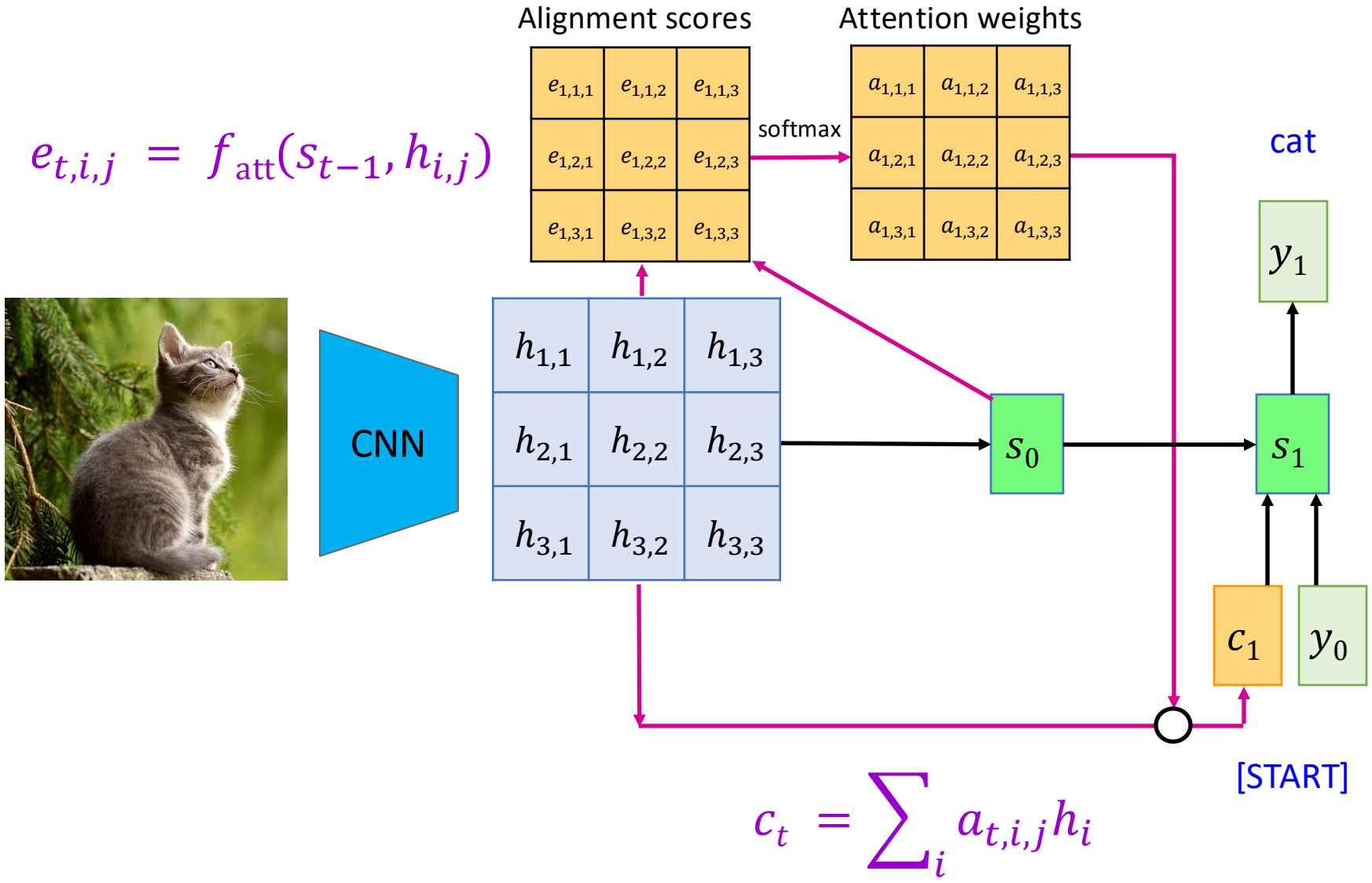


Use CNN to extract a grid of features

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

Image captioning with RNNs and attention

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

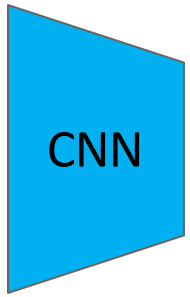


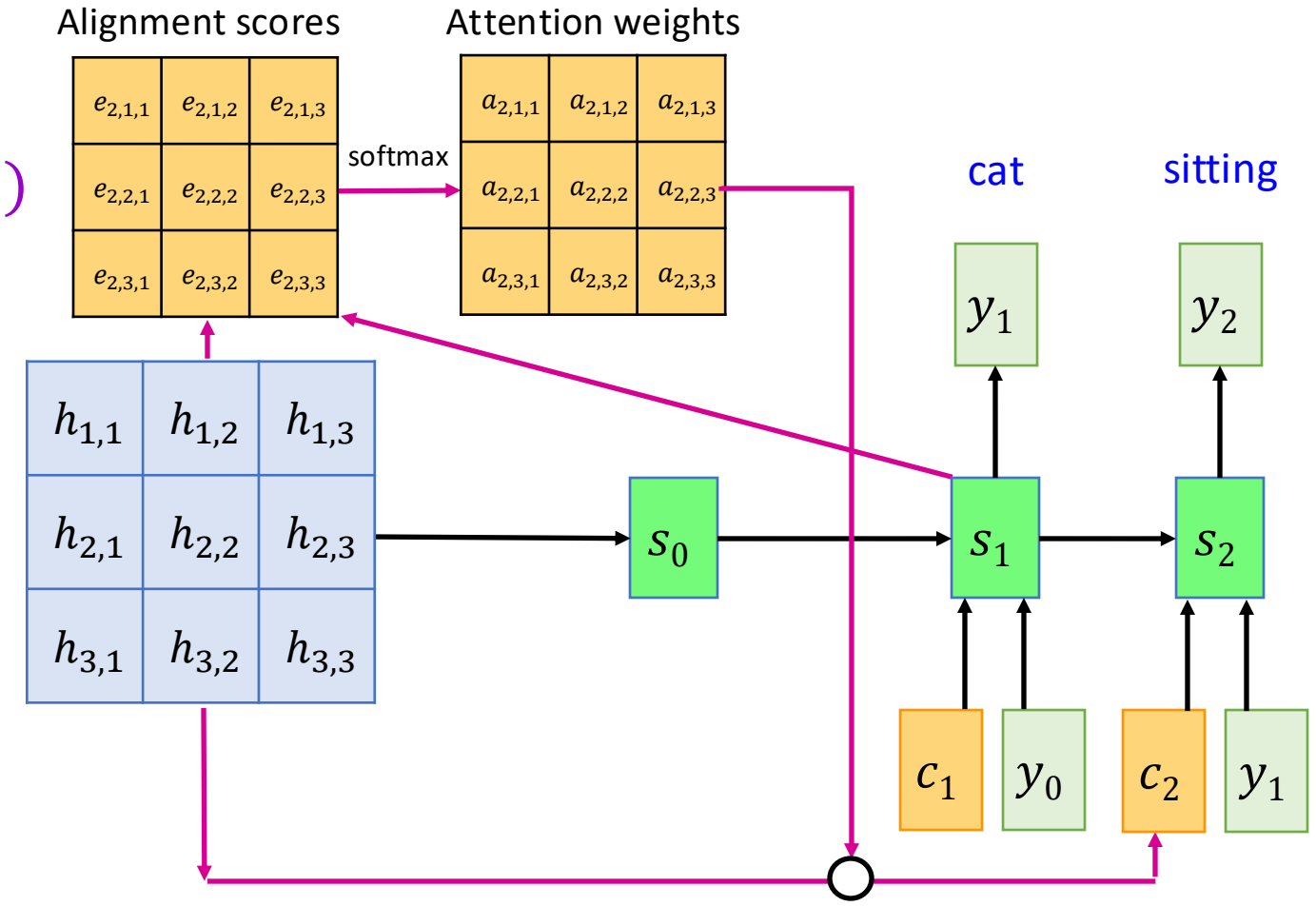
$$c_t = \sum_i a_{t,i,j} h_i$$

Image captioning with RNNs and attention

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

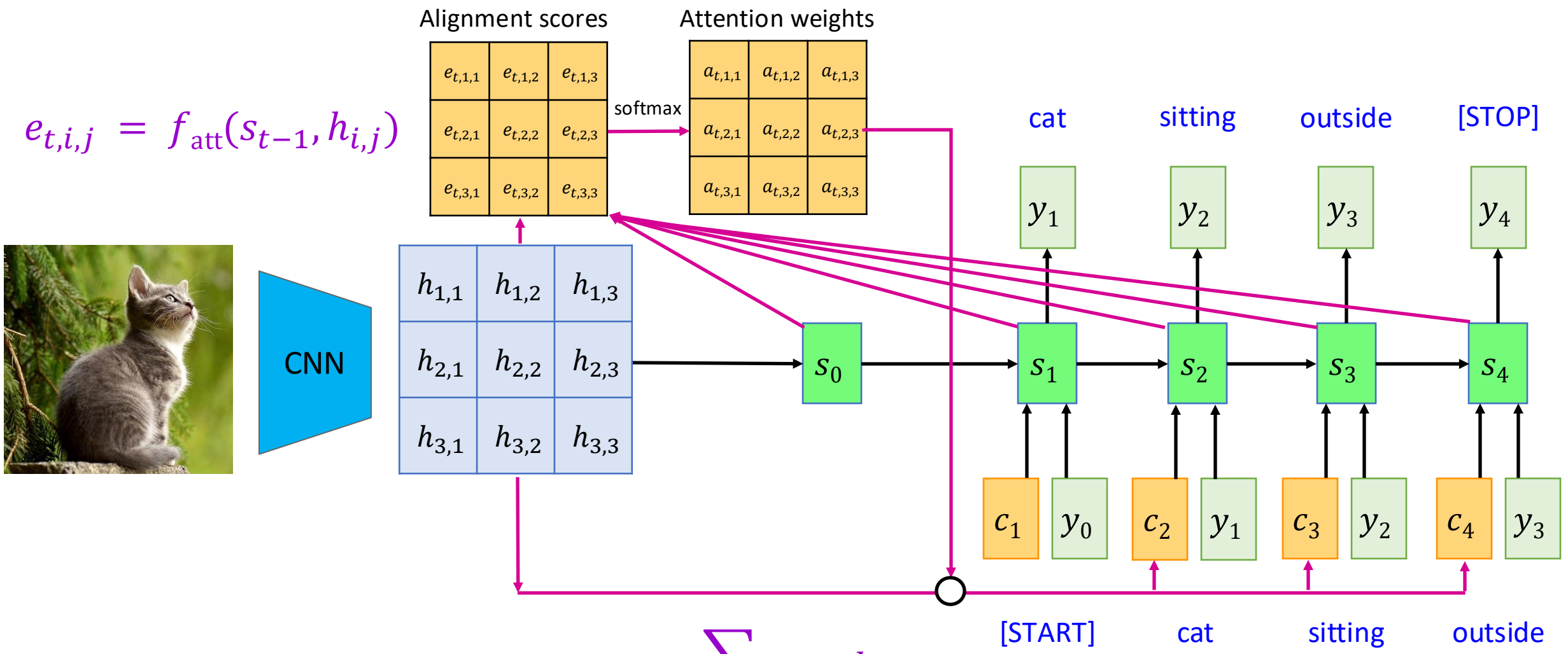
Use a CNN to compute a grid of features for an image





$$c_t = \sum_i a_{t,i,j} h_i$$

Image captioning with RNNs and attention



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

Example results

- Good captions

A woman is throwing a frisbee in a park.

A dog is standing on a hardwood floor.

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

A little girl sitting on a bed with a teddy bear.

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

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

Readings: X, Attend, and Y

"**Show, attend, and tell**" *(Xu et al, ICML 2015)*
Look at image, attend to image regions, produce question

"**Ask, attend, and answer**" *(Xu and Saenko, ECCV 2016)*
"**Show, ask, attend, and answer"** *(Kazemi and Elqursh, 2017)*
Read text of question, attend to image regions, produce answer

"**Listen, attend, and spell**" *(Chan et al, ICASSP 2016)*
Process raw audio, attend to audio regions while producing text

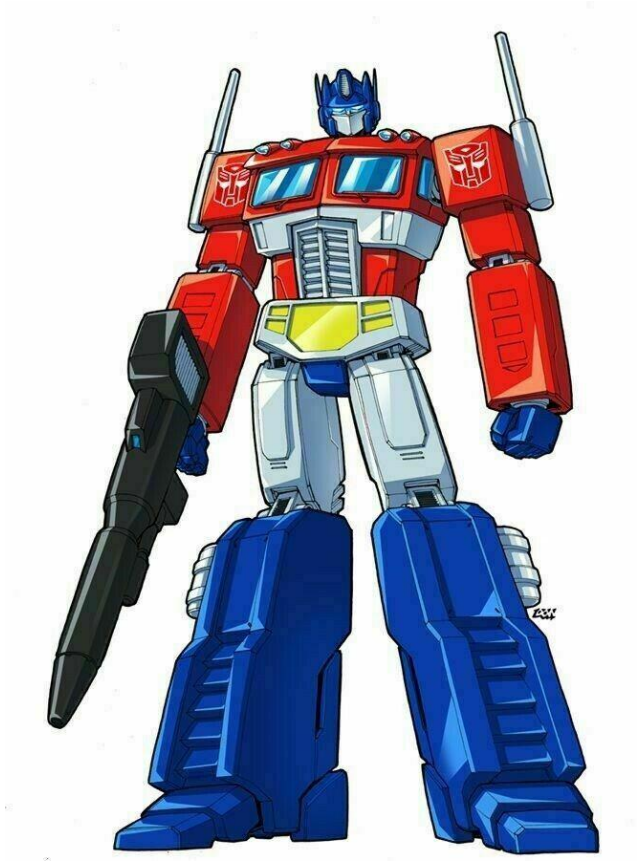
"**Listen, attend, and walk**" *(Mei et al, AAAI 2016)*
Process text, attend to text regions, output navigation commands

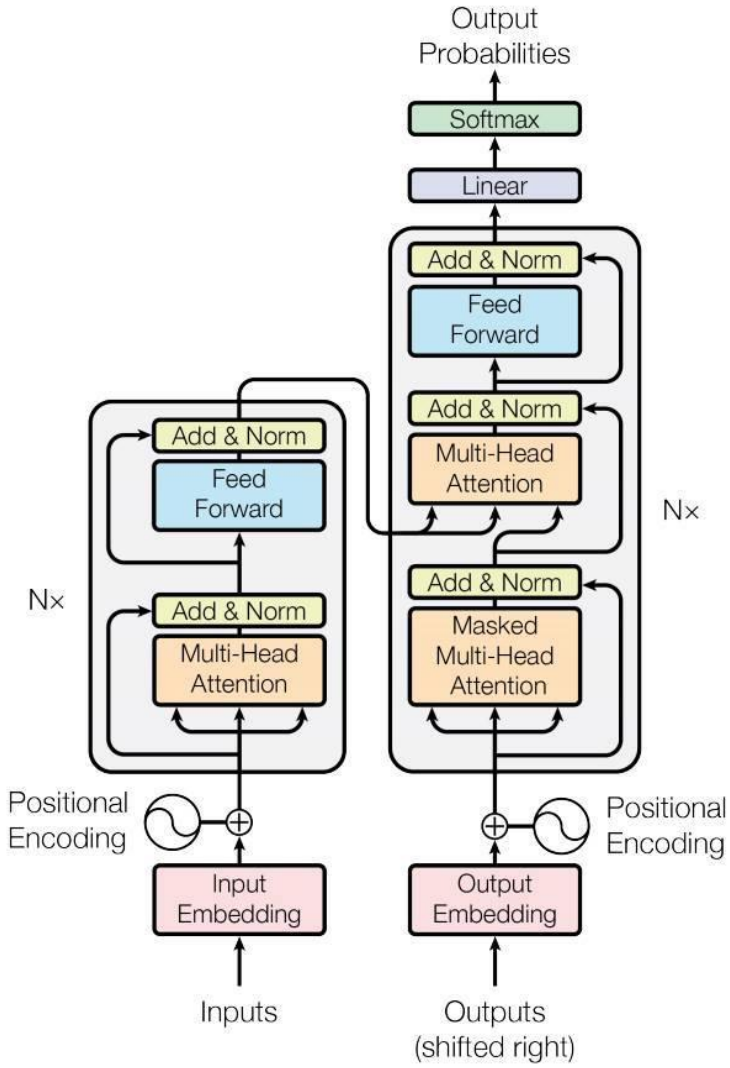
"**Show, attend, and interact**" *(Qureshi et al, ICRA 2017)*
Process image, attend to image regions, output robot control commands

"**Show, attend, and read**" *(Li et al, AAAI 2019)*
Process image, attend to image regions, output text

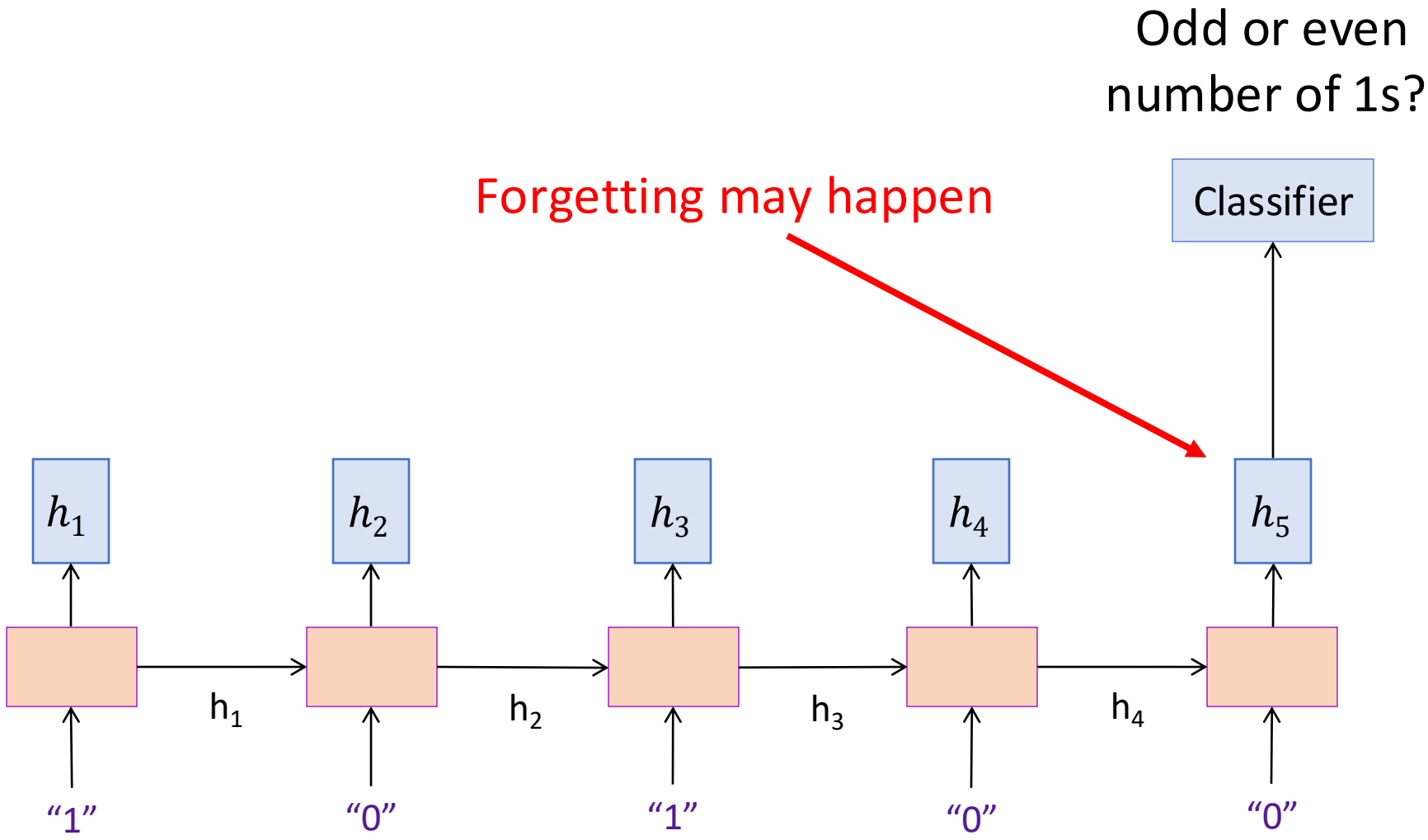
Source: J. Johnson

Transformer

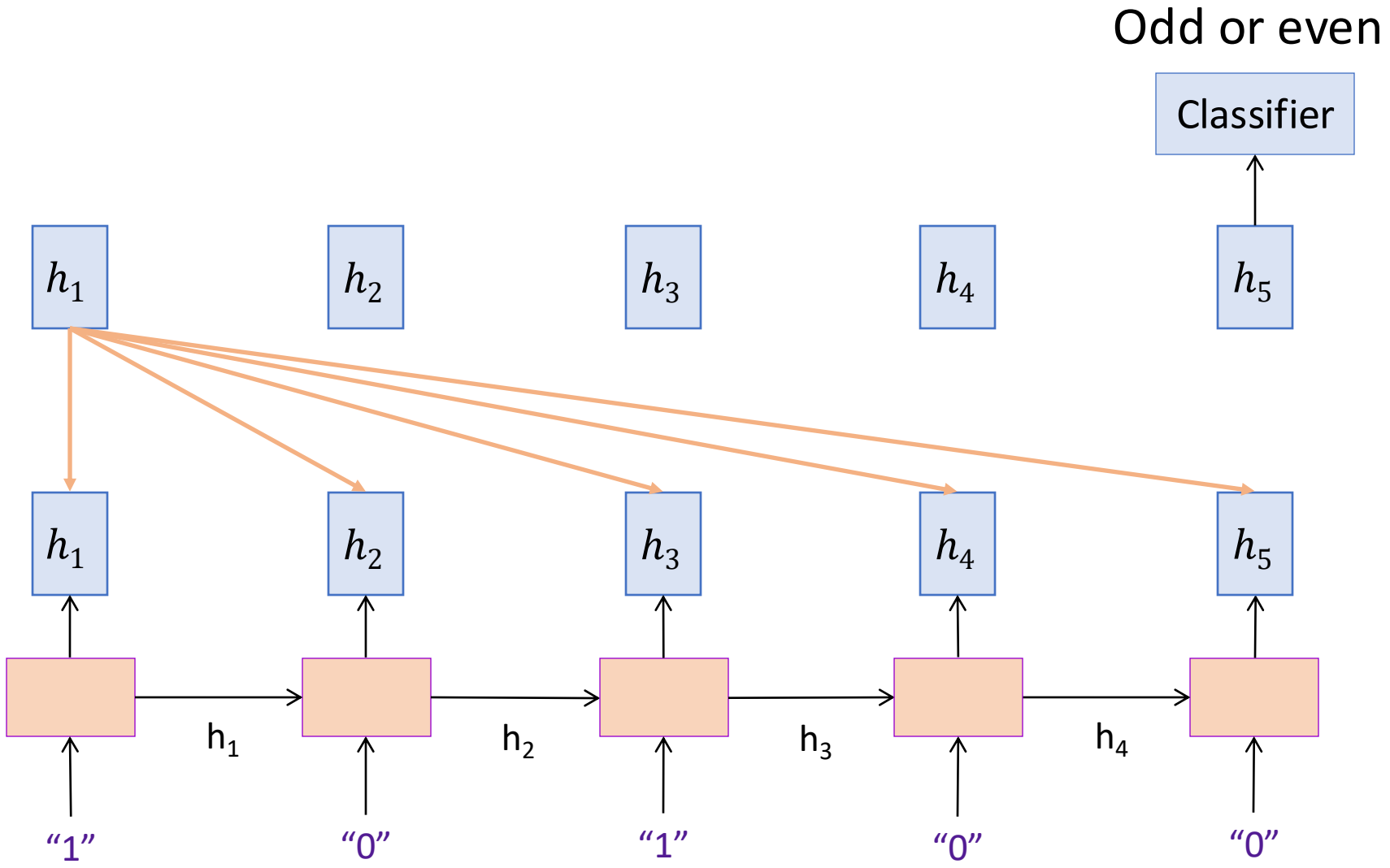




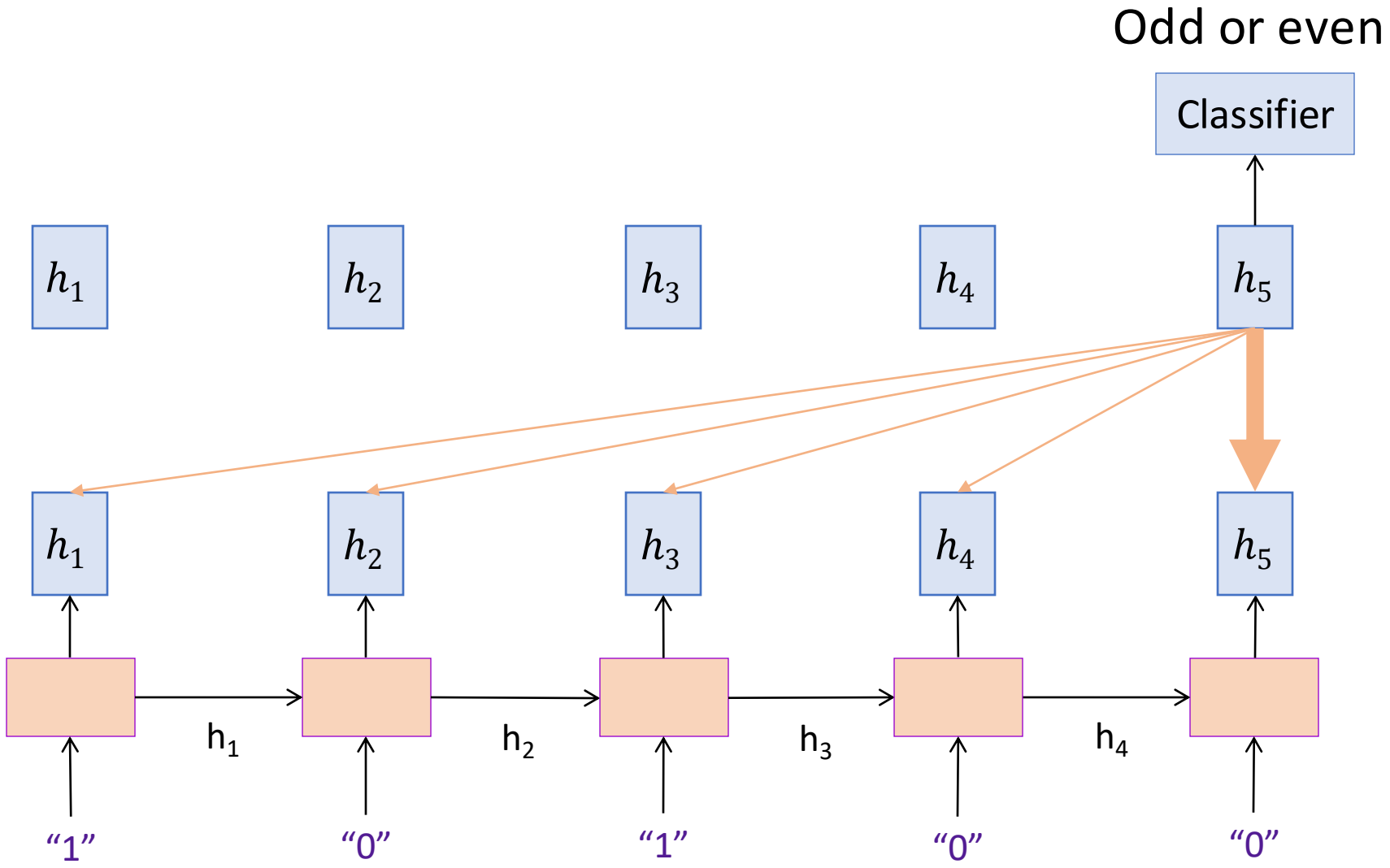
Intuition of Transformer: Self-attention



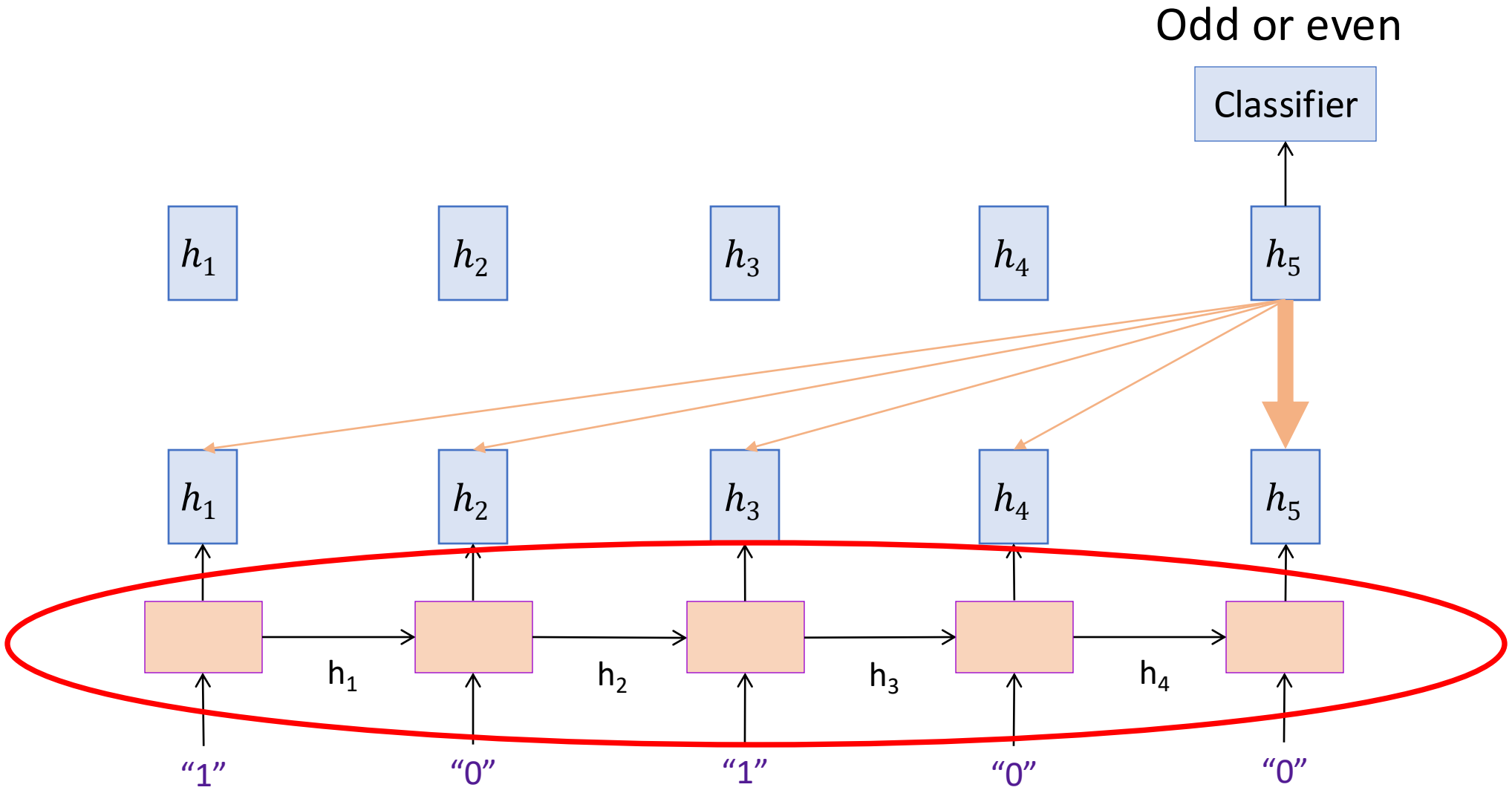
Intuition of Transformer: Self-attention



Intuition of Transformer: Self-attention

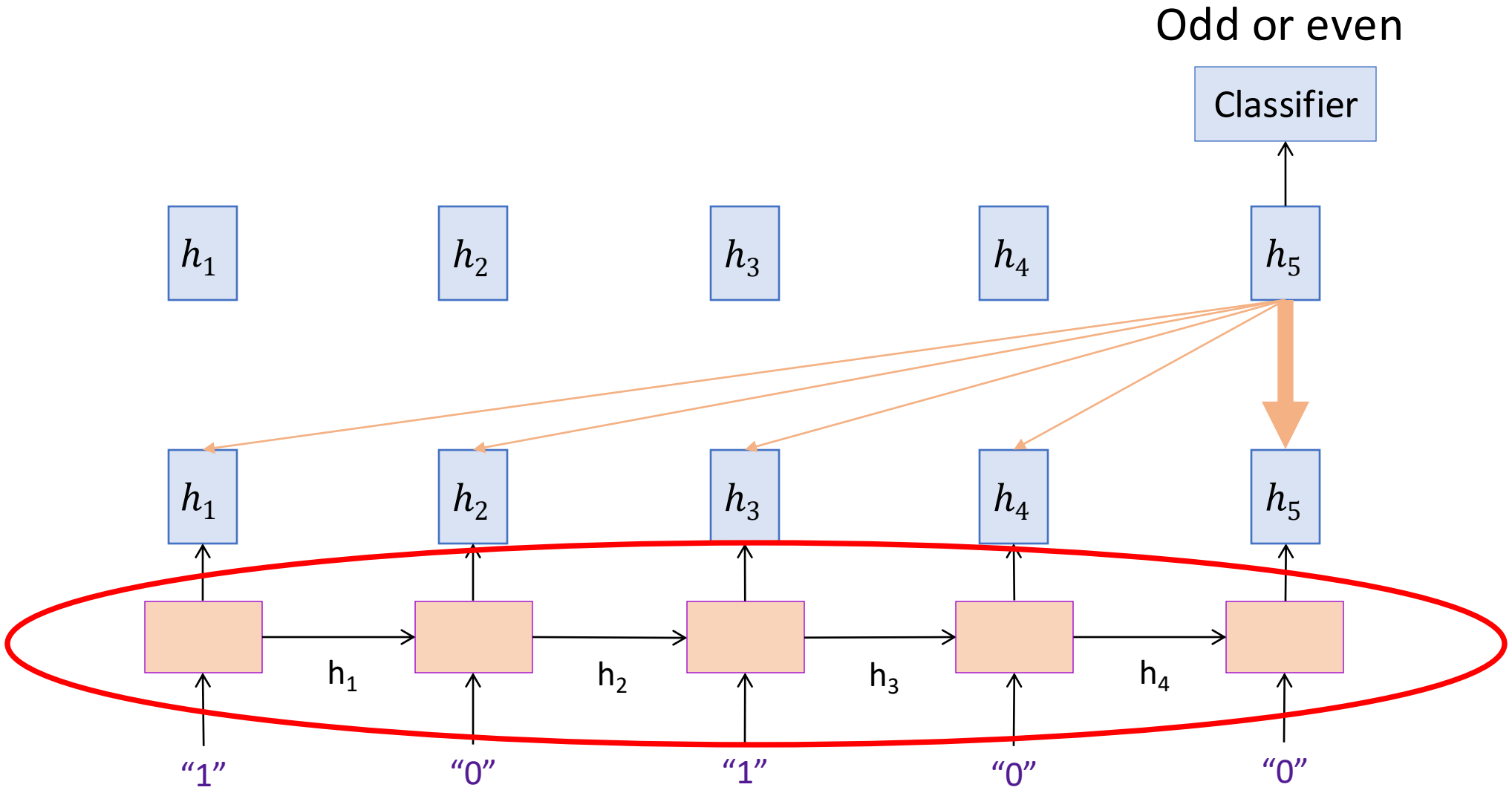


Intuition of Transformer: Self-attention



Do we still need the Recurrent Neural Network to encode the input?

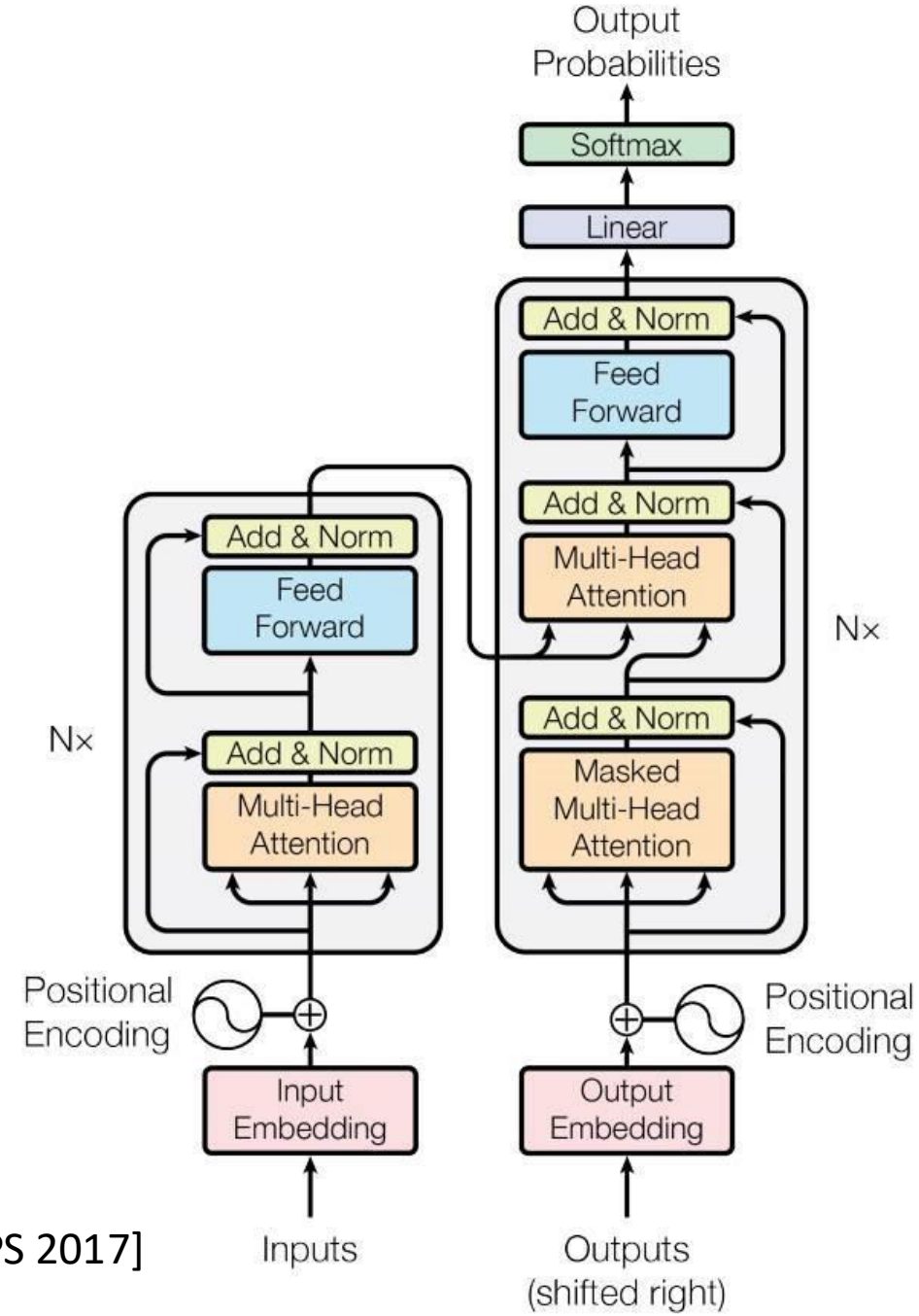
Intuition of Transformer: Self-attention



Do we still need the Recurrent Neural Network to encode the input?

Overview of Transformer

- Encoder (left part)
  - multi-head self attention
  - Position embedding
  - Feedforward network
  - Non-linearity and normalization in-between
- Decoder (right part), optional
  - Multi-head cross attention
  - And others in the encoder



[Vaswani et al., Attention is all you need. NeurIPS 2017]

Next Class

Transformer in detail