ATTENTIVE LANGUAGE MODELS

disclaimer:
the content is Hossein’s understanding of methods

BERT: Bidirectional Encoder Representations from Transformers

We finish the course without explaining BERT!

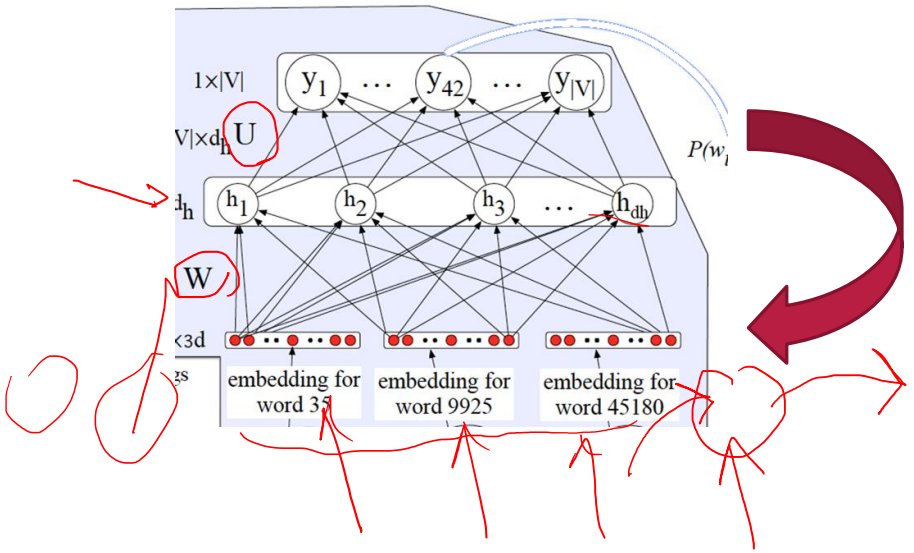
Recurrent Neural LM

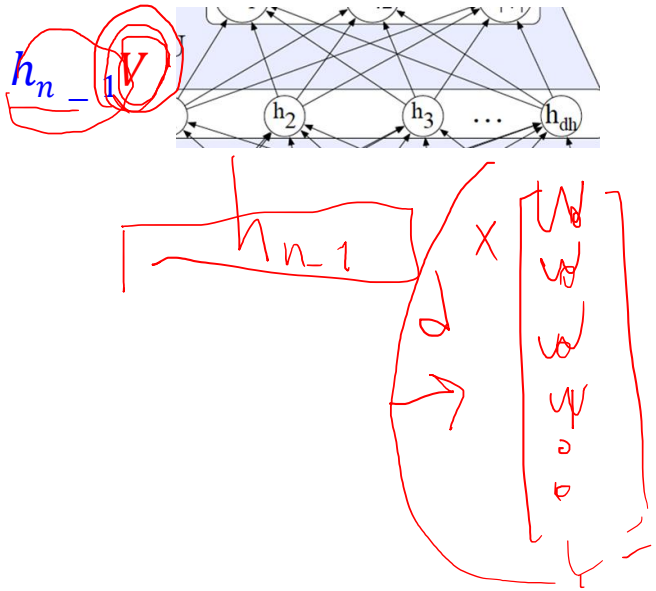
$$Y_n = softmax(hnU)$$

$$h_n = \tanh(Xn\,W + b + h_{n\_1}V)$$

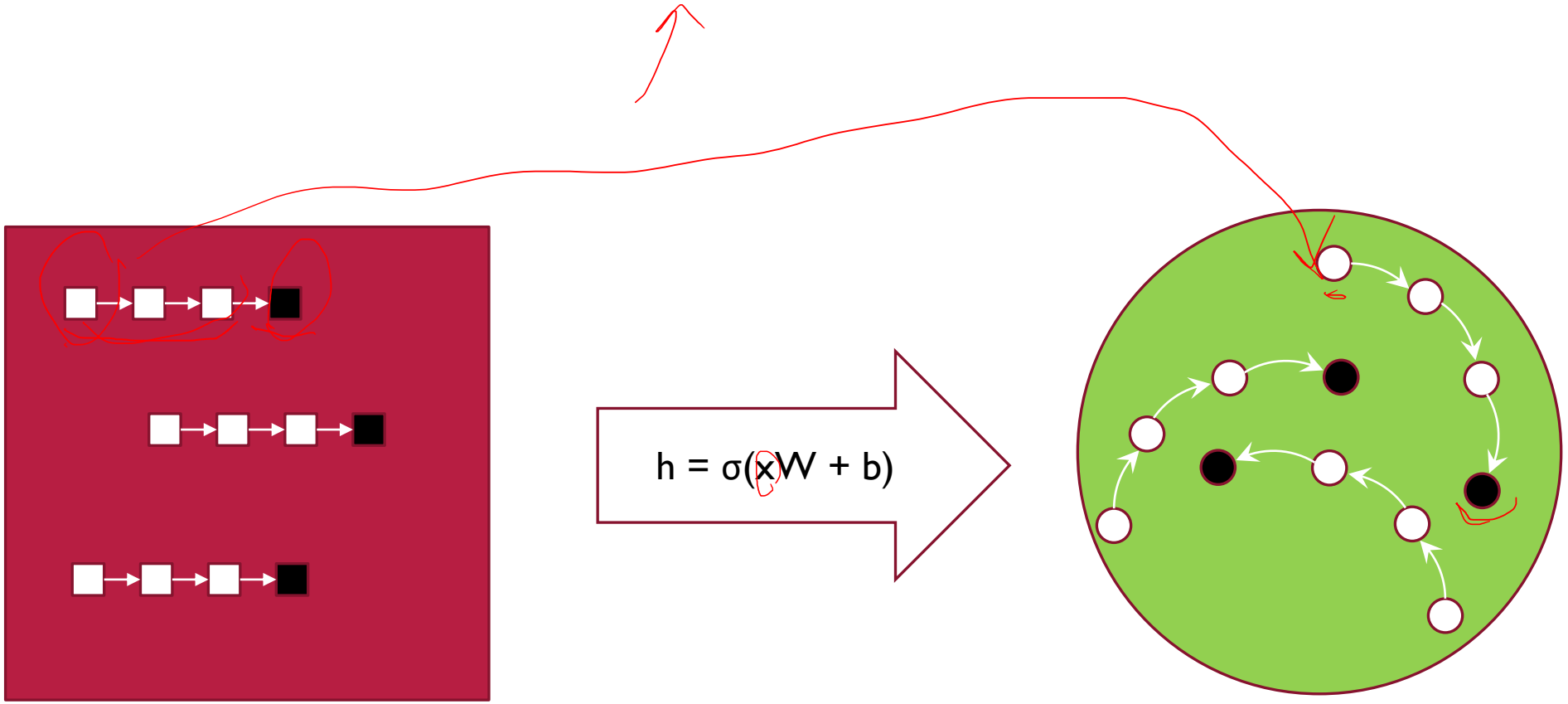
$$X_n\,W + b$$

$$X_n$$

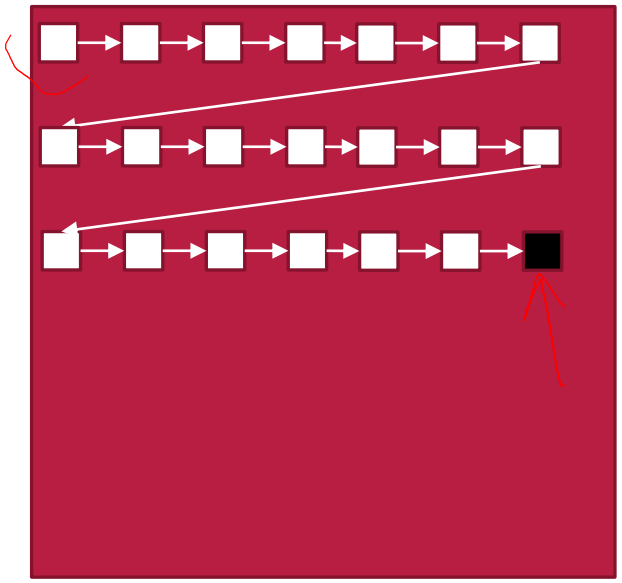




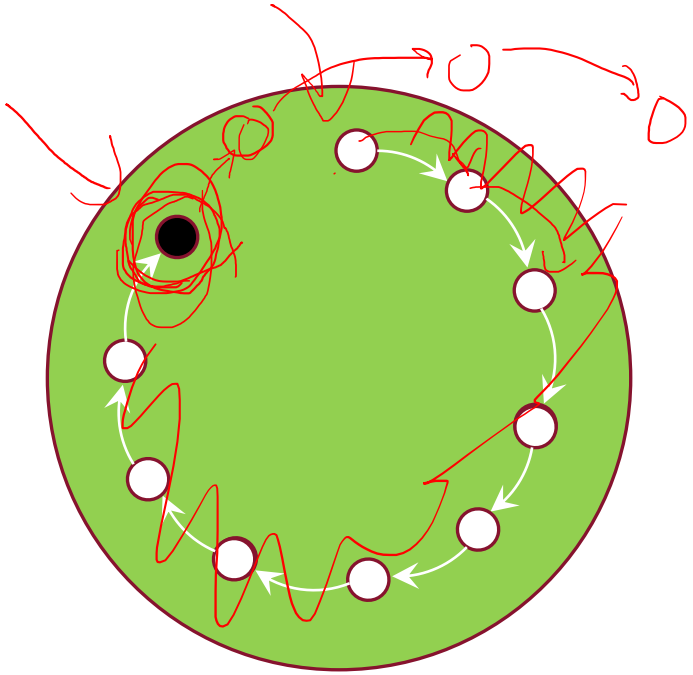
Neural LM



Recurrent Neural LM



$$h_n = \sigma(xW + b + h_{n-1}V)$$



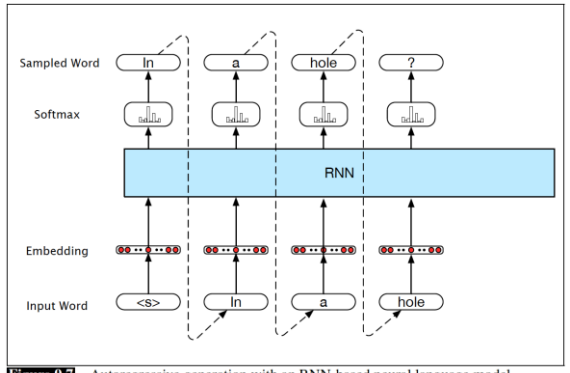
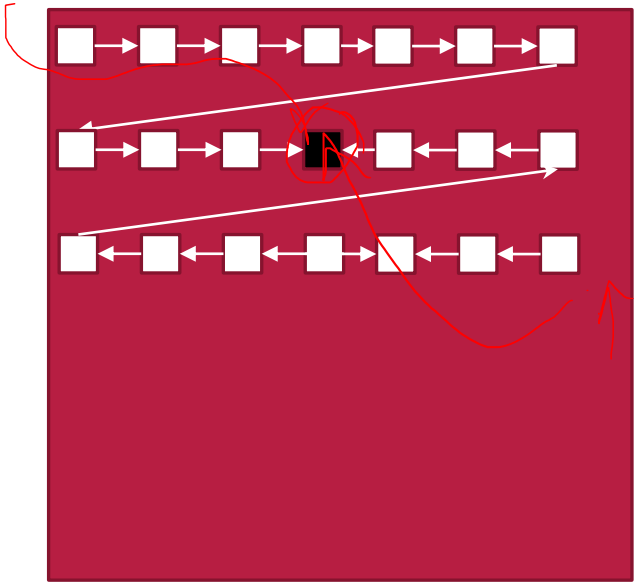


Figure 9.7 Autoregressive generation with an RNN-based neural language model.

Recurrent Neural LM: Bidirection



$$h_i = \sigma(xW + b + h_{i-1}V)$$
$$h_i = \sigma(xW' + b' + h_{i+1}V)$$

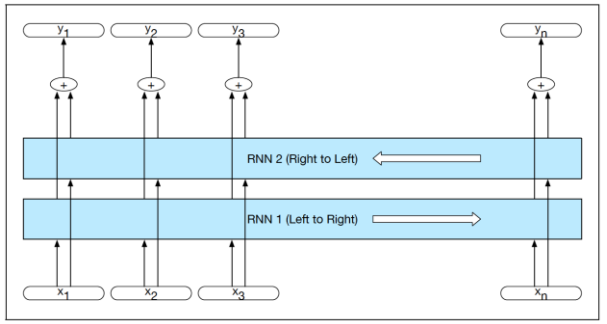
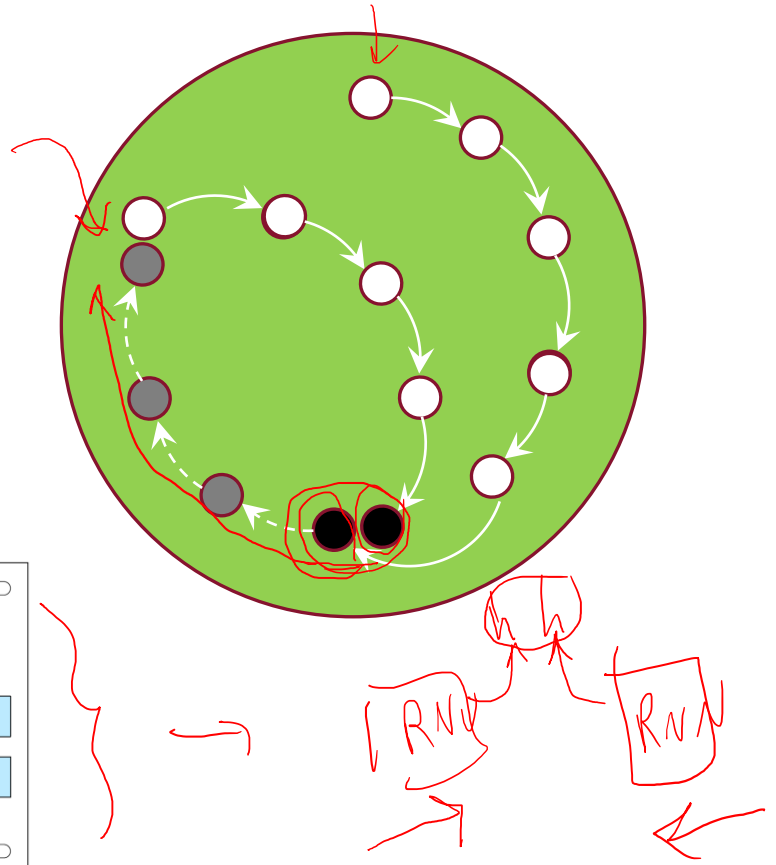
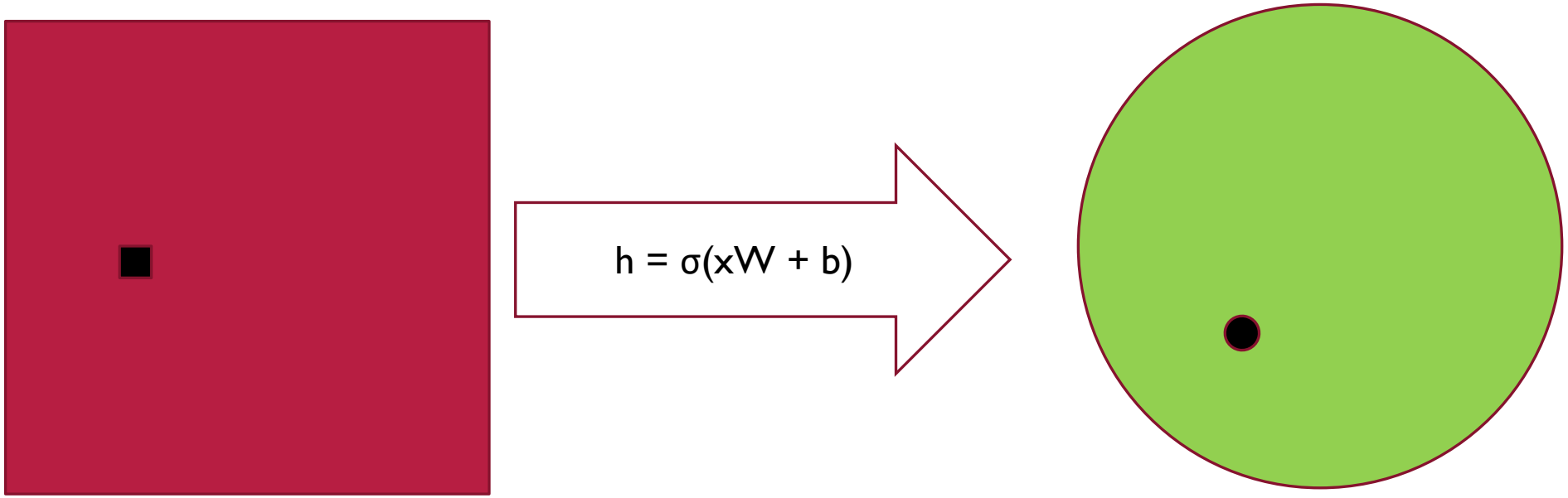


Figure 9.11 A bidirectional RNN. Separate models are trained in the forward and backward directions with the output of each model at each time point concatenated to represent the state of affairs at that point in time. The box wrapped around the forward and backward network emphasizes the modular nature of this architecture.

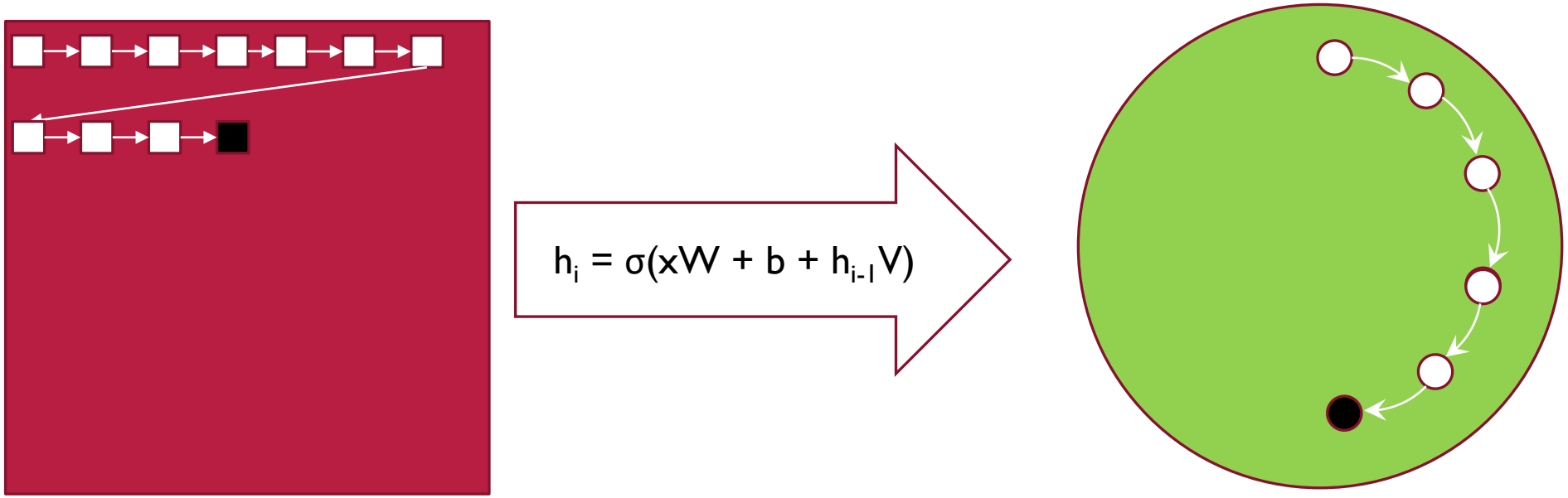


Seq2Seq

Neural LM



Recurrent Neural LM



Recurrent Neural LM: Encoder-Decoder

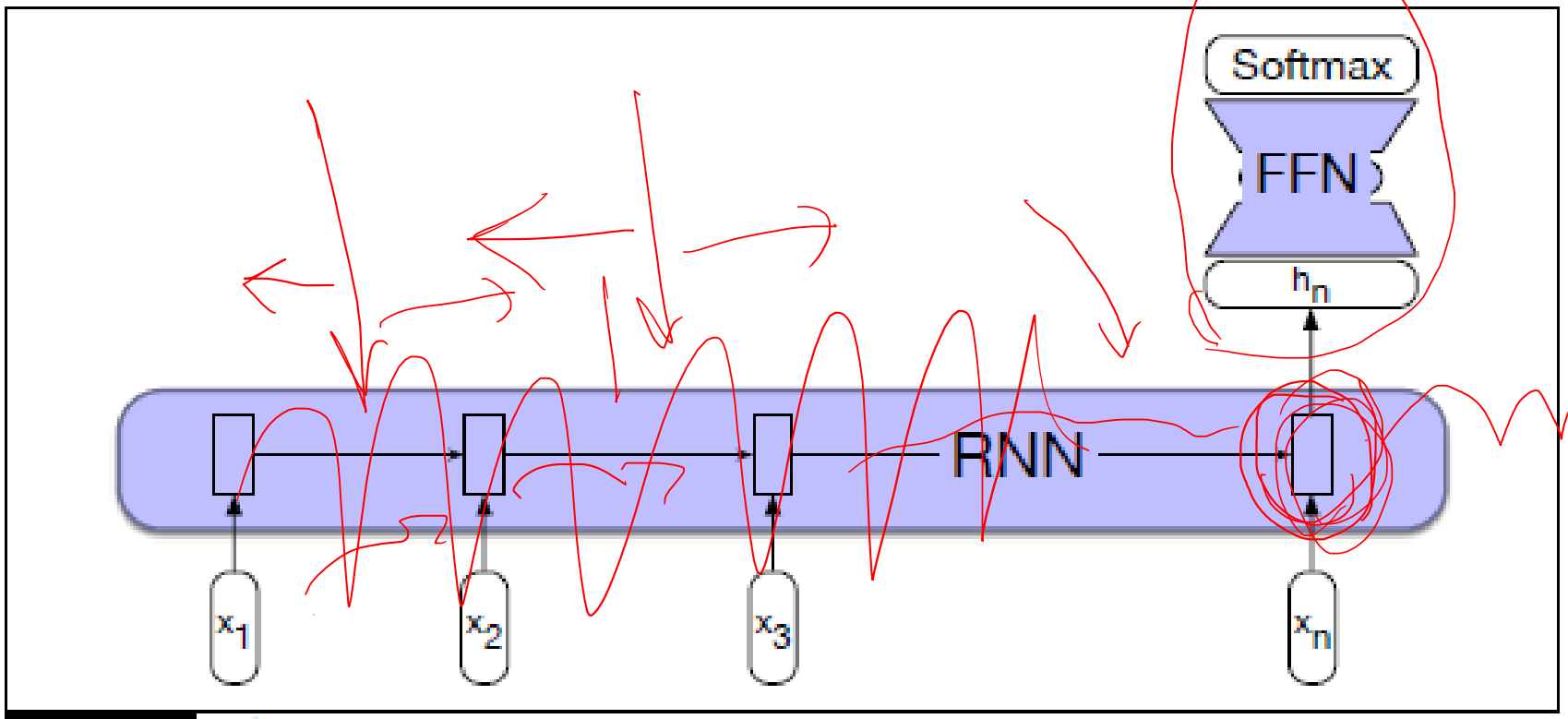
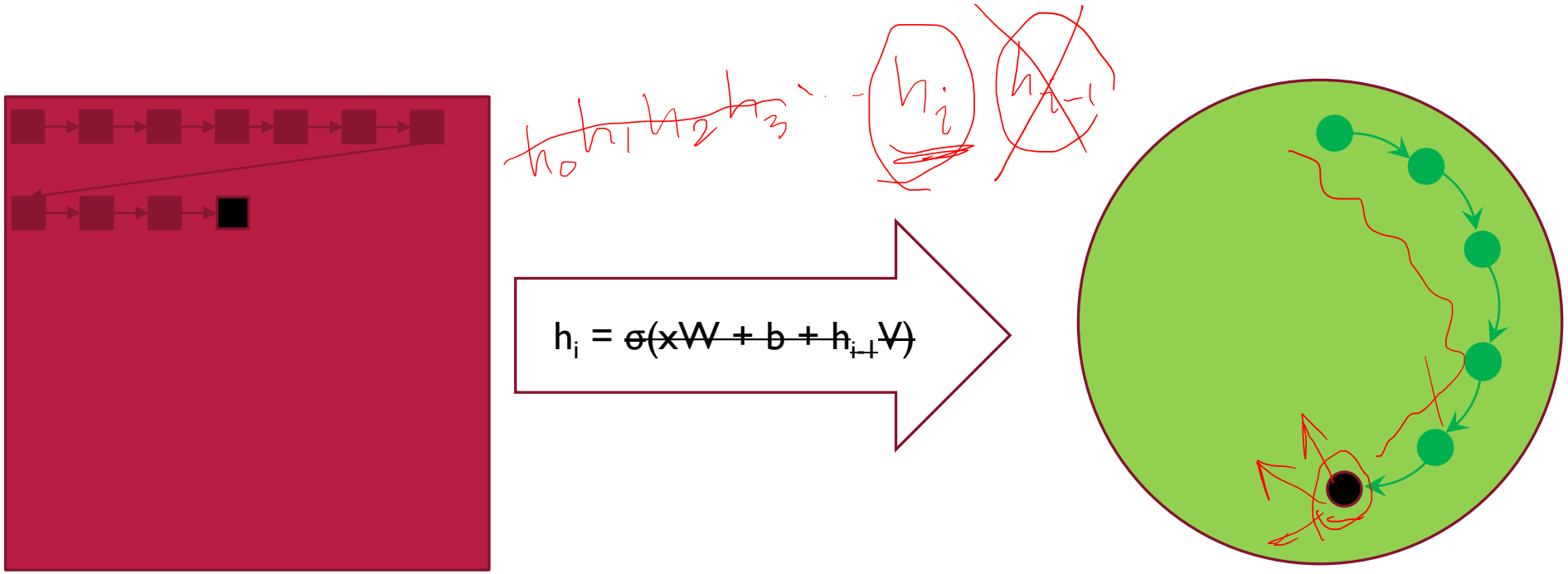


Figure 9.8 Sequence classification using a simple RNN combined with a feedforward network. The final hidden state from the RNN is used as the input to a feedforward network that performs the classification.

Recurrent Neural LM: Encoder-Decoder

A representation for the whole history



All history is encoded in h_i. We don't need history!
h_i is enough to decode the whole future.

Encoder-RNN

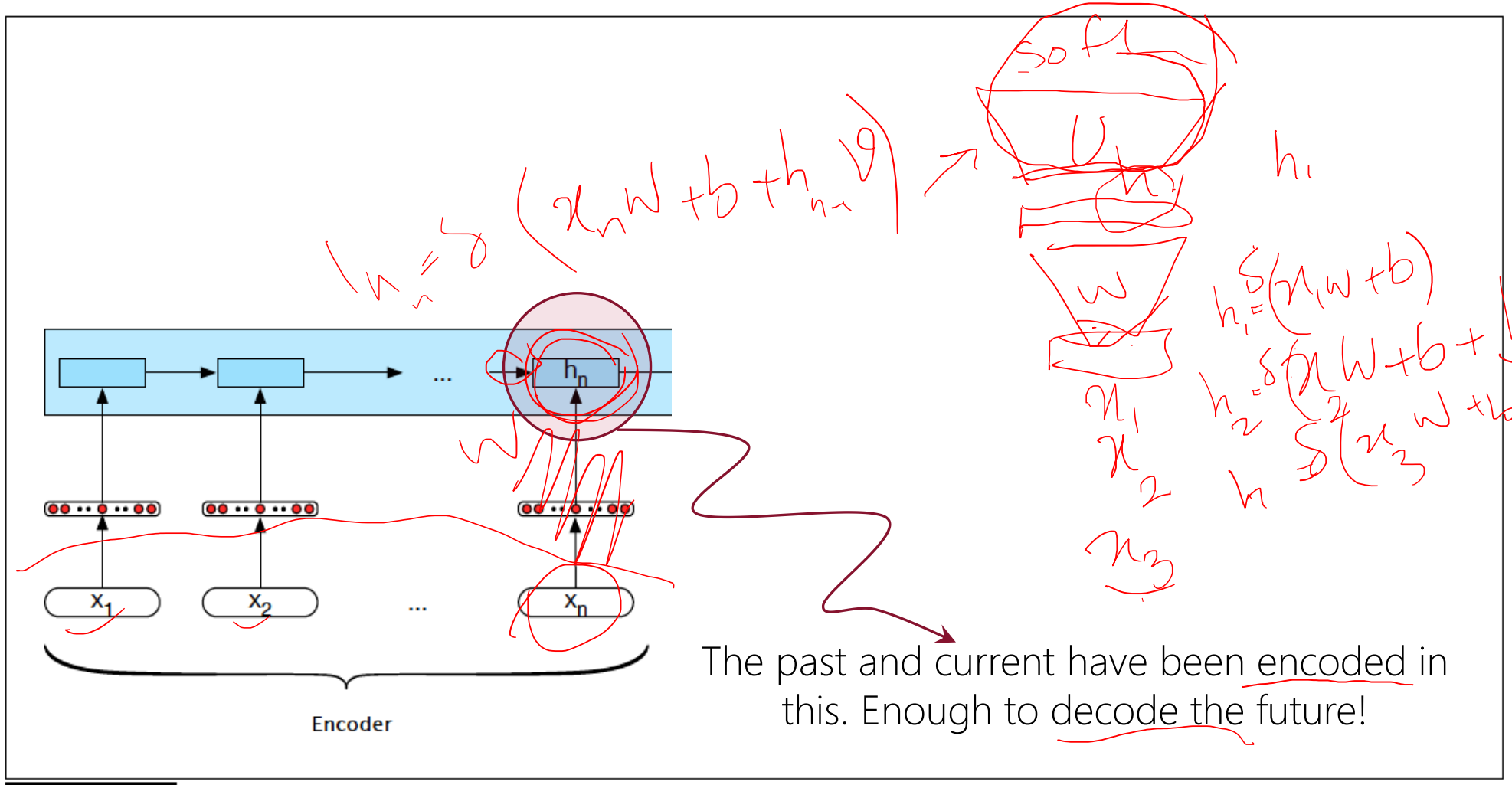
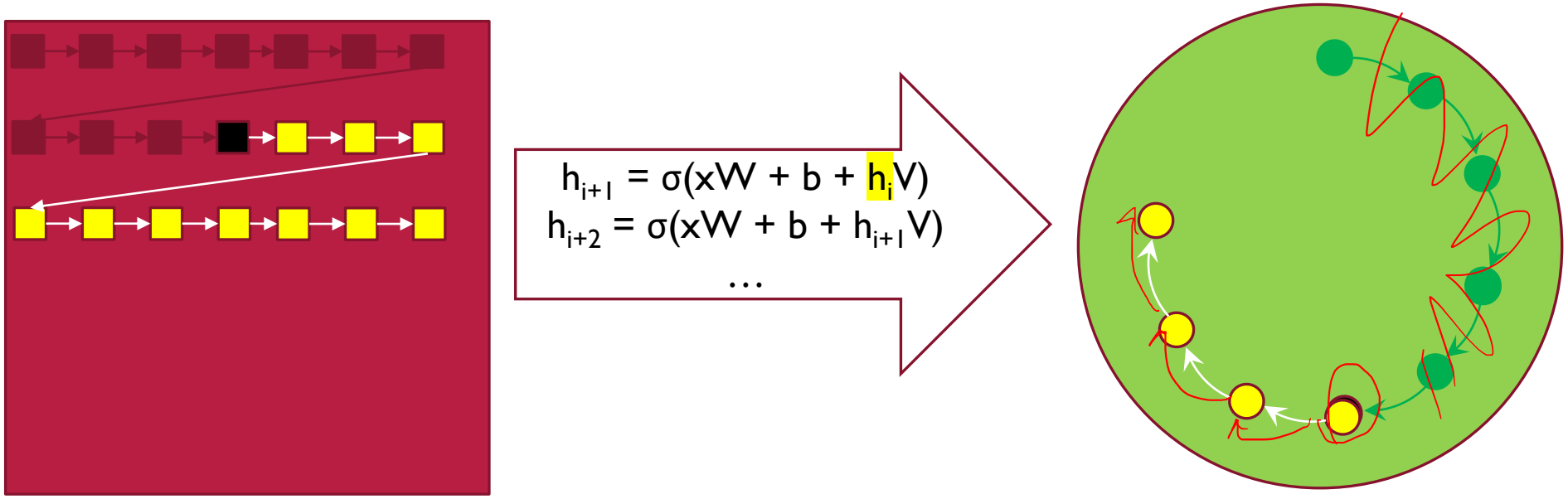
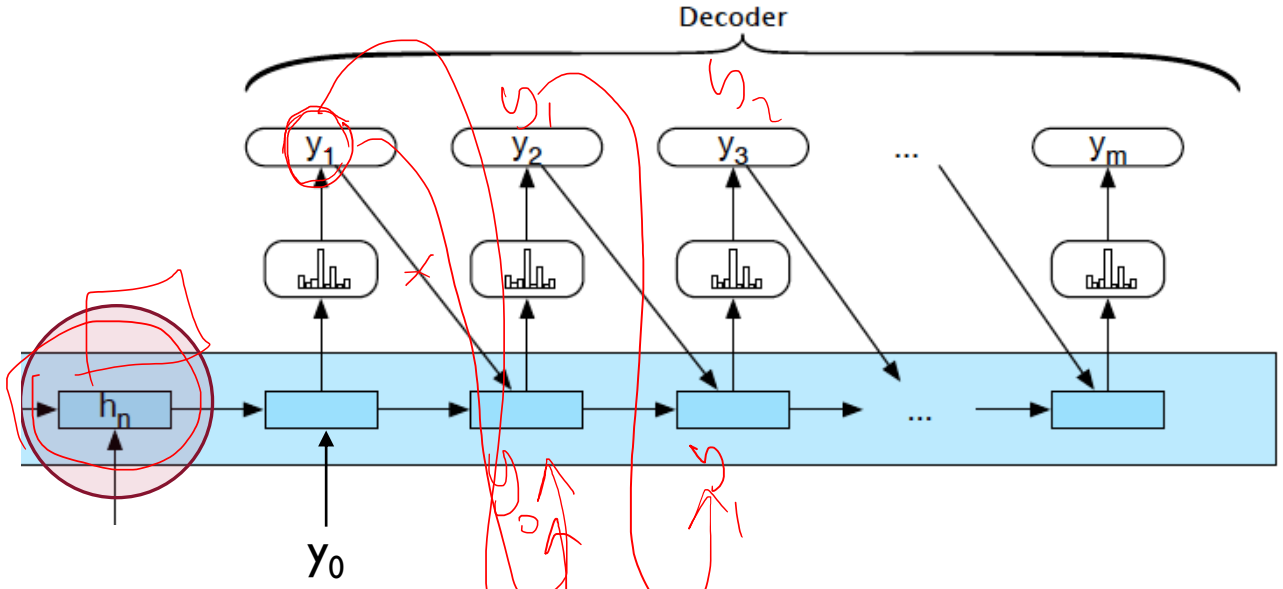


Figure 10.3 Basic RNN-based encoder-decoder architecture. The final hidden state of the encoder RNN serves as the context for the decoder in its role as h_0 in the decoder RNN.

Recurrent Neural LM: Encoder-Decoder



Decoder-RNN (Test: Autoregressive)



Decoder-RNN is a Recurrent LM, but extra condition h_n

Figure 10.3 Basic RNN-based encoder-decoder architecture. The final hidden state of the encoder RNN serves as the context for the decoder in its role as h_0 in the decoder RNN.

$$p(w_n \mid w_1 \cdots w_{n-1})$$

$$p(w_i \mid w_{i-2} w_{i-1}) = p(w_i \mid w_{i-1}, w_{i-2})$$

Conditional Language Modeling

The probability of the next word, given the previous words and conditioning context x:

$$p(w_i \mid x, w_1, w_2, \ldots, w_{i-1})$$

Conditional Language Modeling

Condition x	Language Model
An author	A document written by that author
A topic label	An article about that topic
{SPAM, NOT_SPAM}	An email
A sentence in French	Its English translation
An image	A text description of the image
A document	Its summary
Meterological measurements	A weather report
A question	Its answer
Acoustic signal	Transcription of speech
X-Ray	Medical report
Code	Documentation
Code in C	Code in Java

Decoder-RNN (Train: Teacher Forcing)

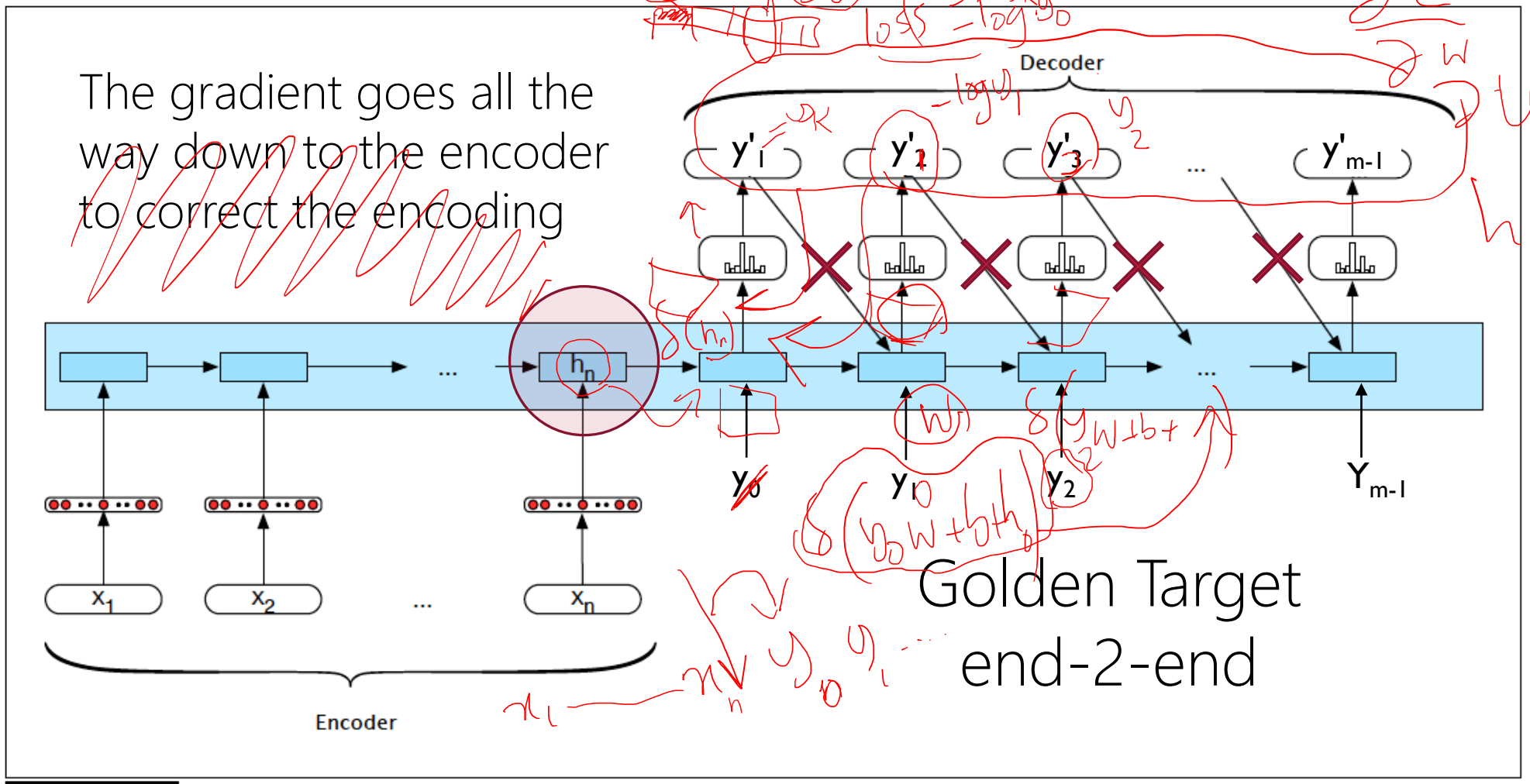


Figure 10.3 Basic RNN-based encoder-decoder architecture. The final hidden state of the encoder RNN serves as the context for the decoder in its role as h_0 in the decoder RNN.

Decoder-RNN (Train: Teacher Forcing)

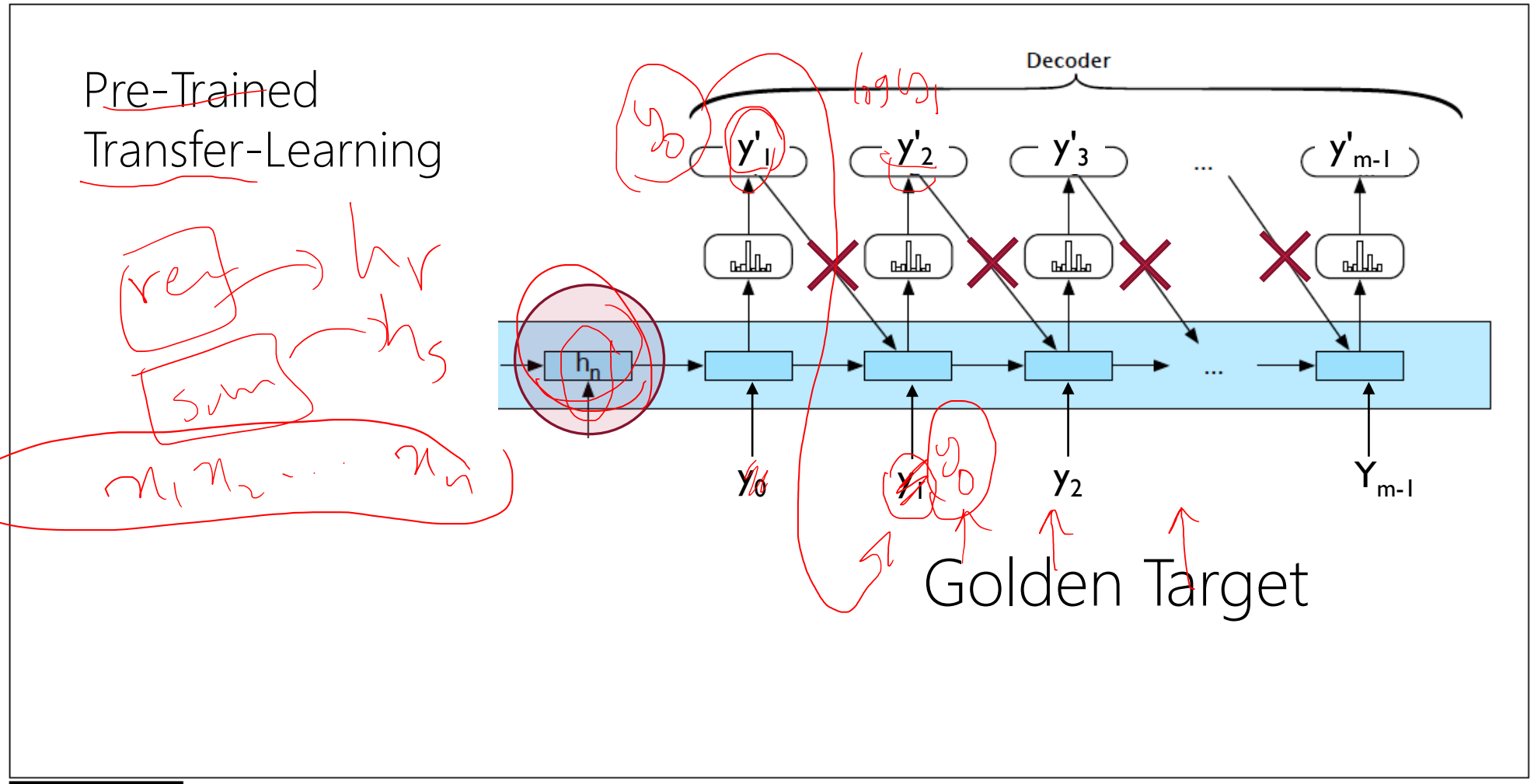


Figure 10.3 Basic RNN-based encoder-decoder architecture. The final hidden state of the encoder RNN serves as the context for the decoder in its role as h_0 in the decoder RNN.

Sequence2Sequence

Sutskever, I., Vinyals, O., & Le, Q.V. (2014). Sequence to sequence learning with neural networks. In NIPS.

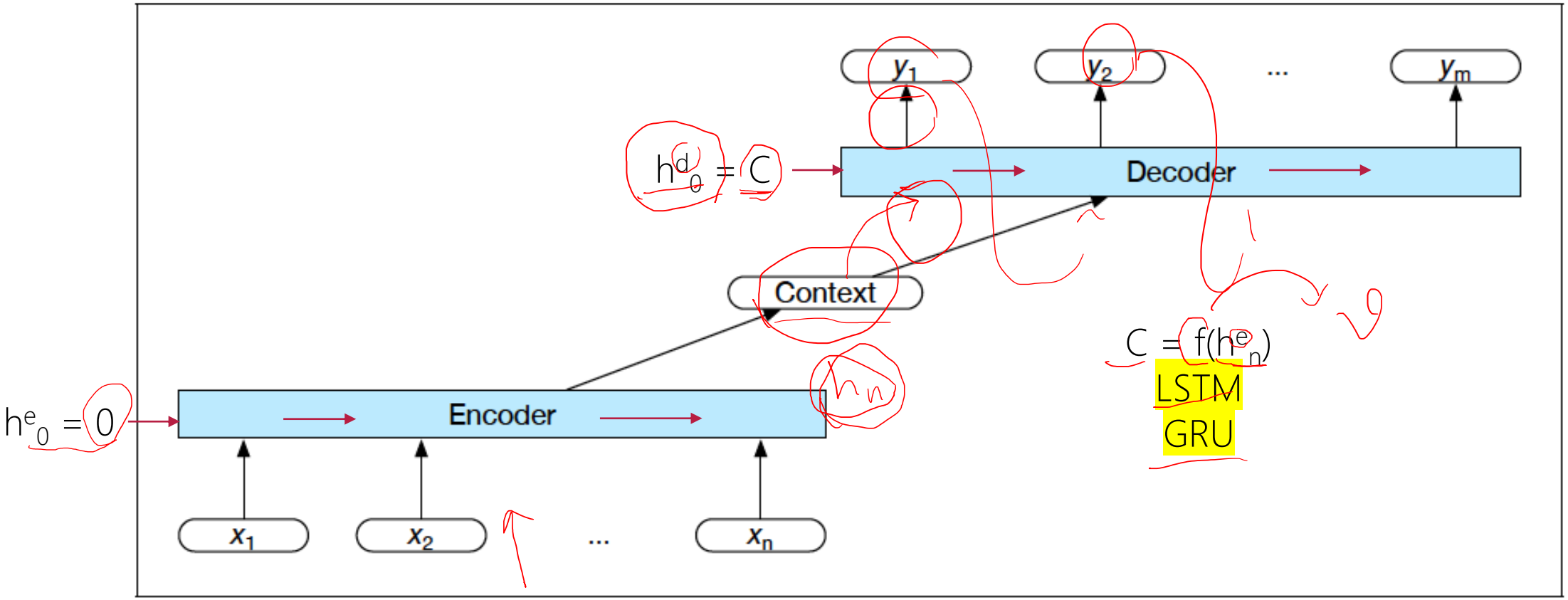
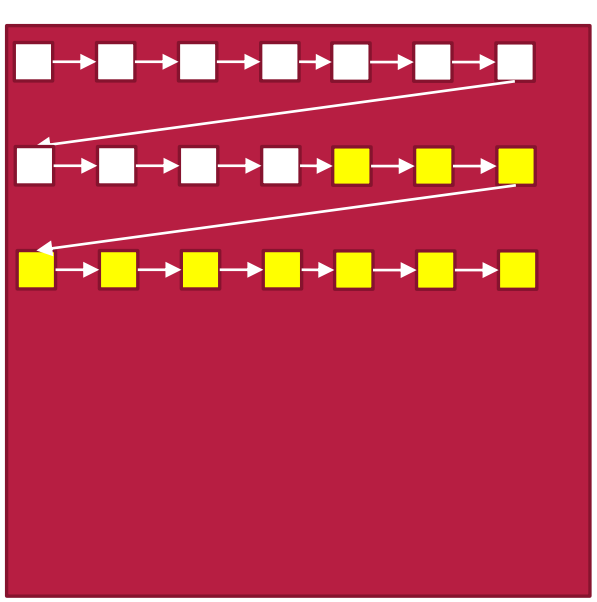


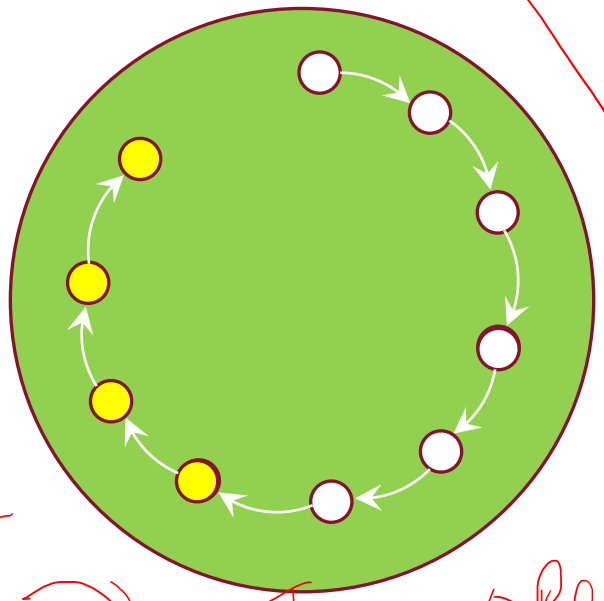
Figure 10.4 Basic architecture for an abstract encoder-decoder network. The context is a function of the vector of contextualized input representations and may be used by the decoder in a variety of ways.

Seq2Seq: Bitext Translation

For machine, they are just sequences of tokens!



$$h_i = \sigma(xW + b + h_{i-1}V)$$



SMT → NMT → Open NMT

<s>Prague Stock Market falls to minus by the end of the trading day</s> <s>Die Prager Börse stürzt gegen Geschäftsschluss ins Minus</s>

Seq2Seq: Bitext Translation

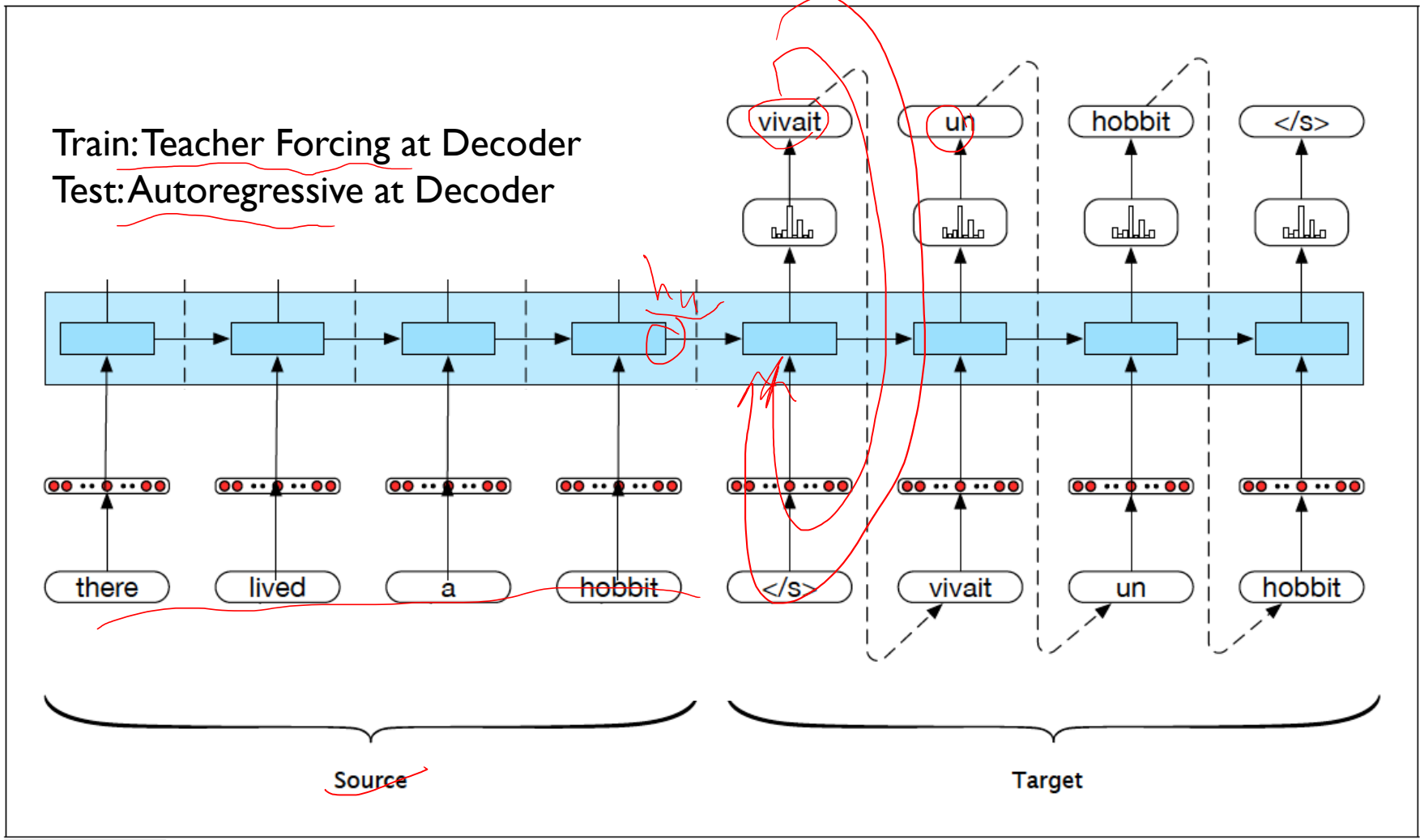
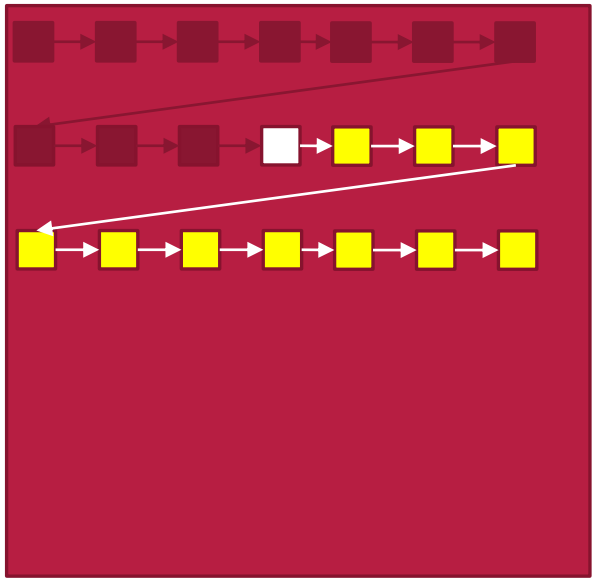


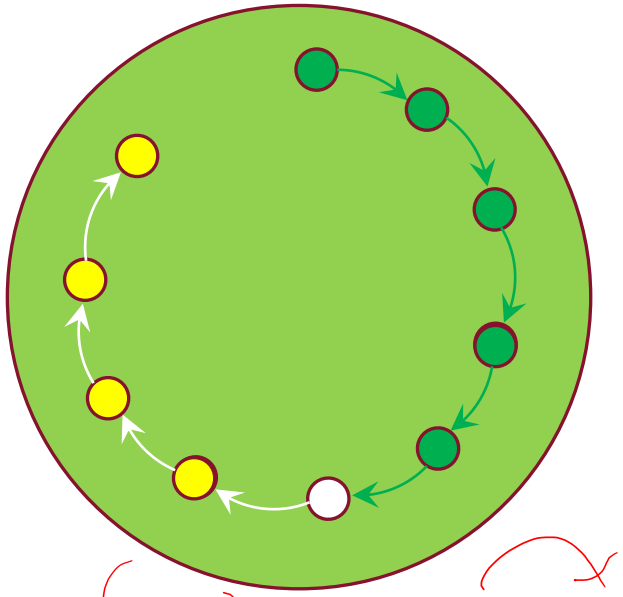
Figure 10.2 Training setup for a neural language model approach to machine translation. Source-target bi-texts are concatenated and used to train a language model.

Seq2Seq: Bitext Translation

For machine, they are just sequences of tokens!



$h_i = \sigma(xW + b + h_{i-1}V)$



<s>Prague Stock Market falls to minus by the end of the trading day</s><s>Die Prager Börse stürzt gegen Geschäftsschluss ins Minus</s>

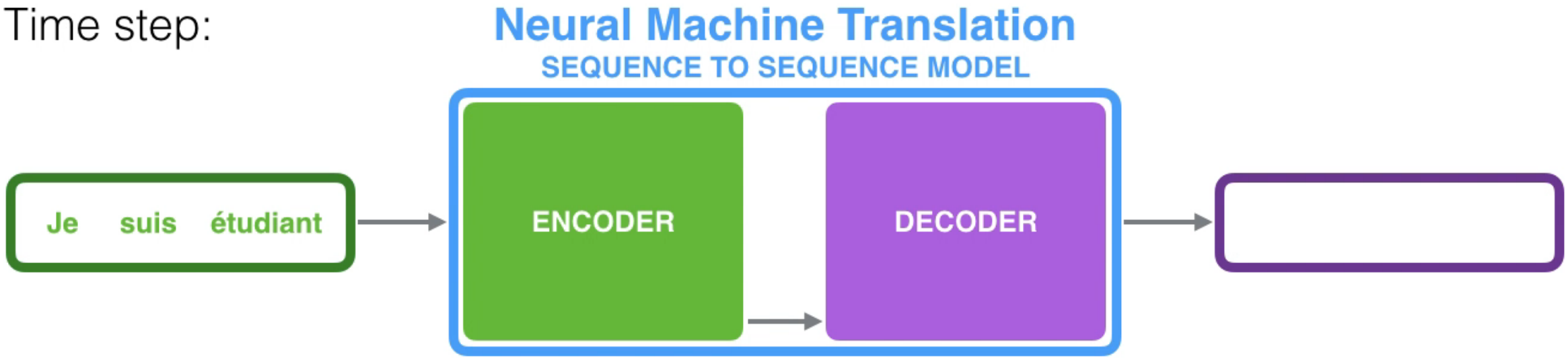
Jay Alammar

Visualizing machine learning one concept at a time.

@JayAlammar on Twitter. YouTube Channel

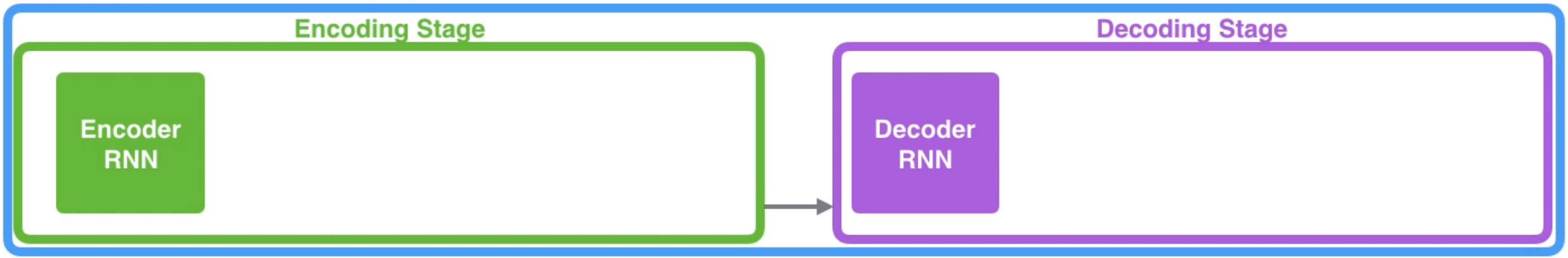
Blog About

Time step:



Neural Machine Translation

SEQUENCE TO SEQUENCE MODEL



Je suis étudiant

Question to Answers

Anything to Sequence

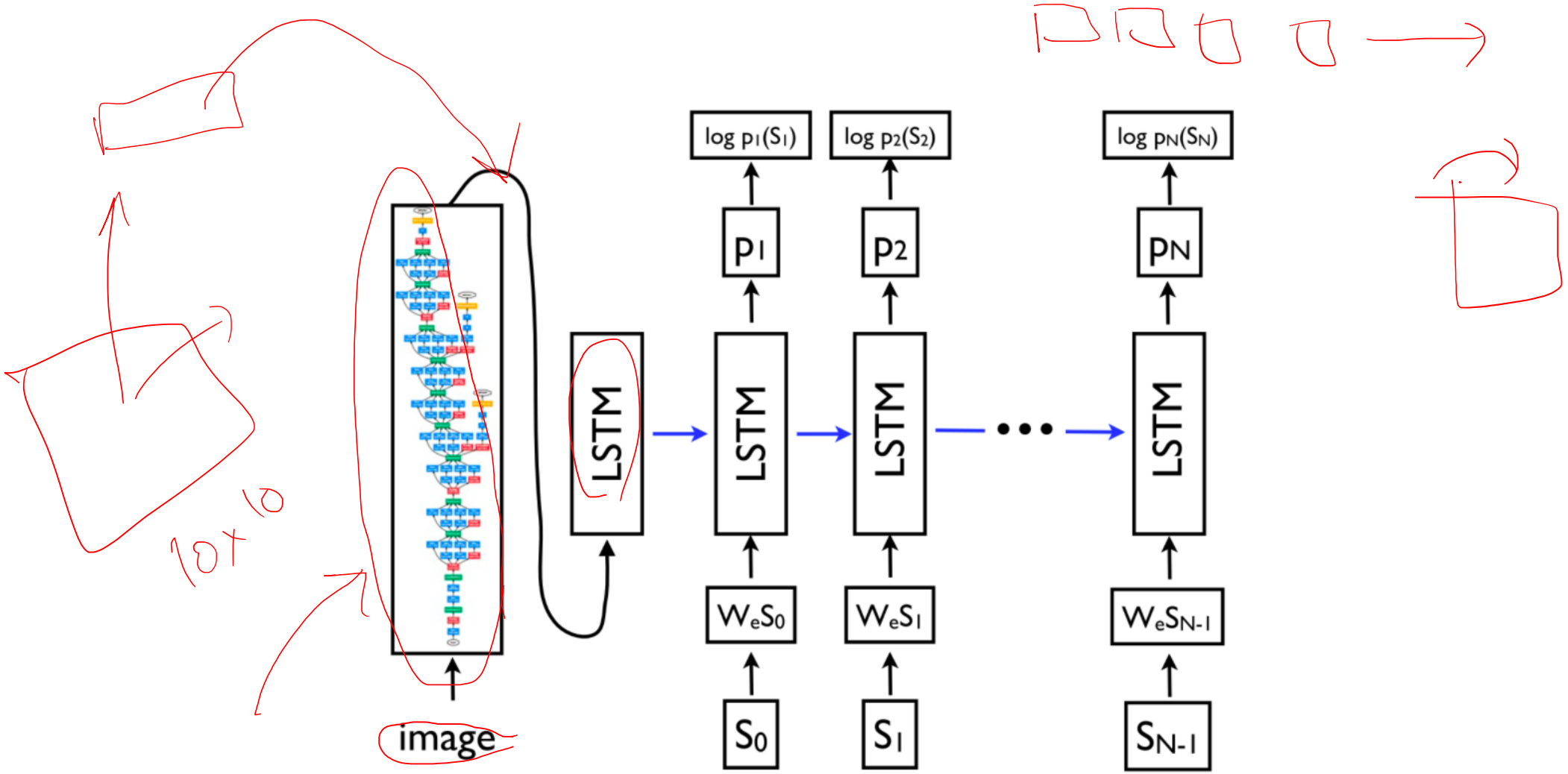


Figure 9: Generating a caption based on an image (Vinyals et al., 2015)

Seq2Seq: Information Bottleneck

Although we have LSTM, GRU, h_n falls short to encode *all* the past to decode all the future when the sequence becomes long!

Too much pressure to a single vector!

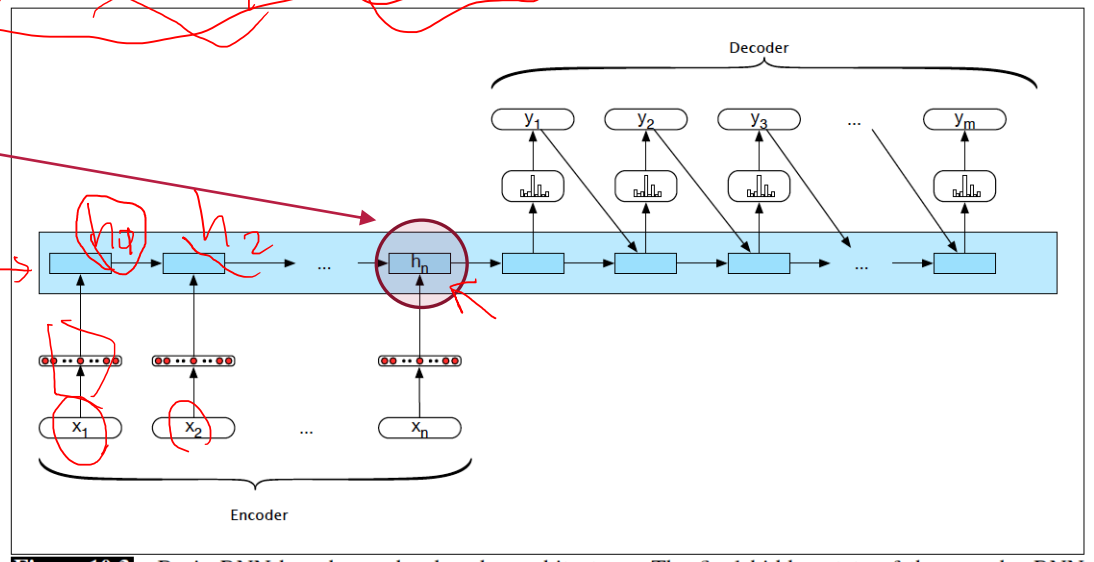


Figure 10.3 Basic RNN-based encoder-decoder architecture. The final hidden state of the encoder RNN serves as the context for the decoder in its role as h_0 in the decoder RNN.

Look into the Past based on future!

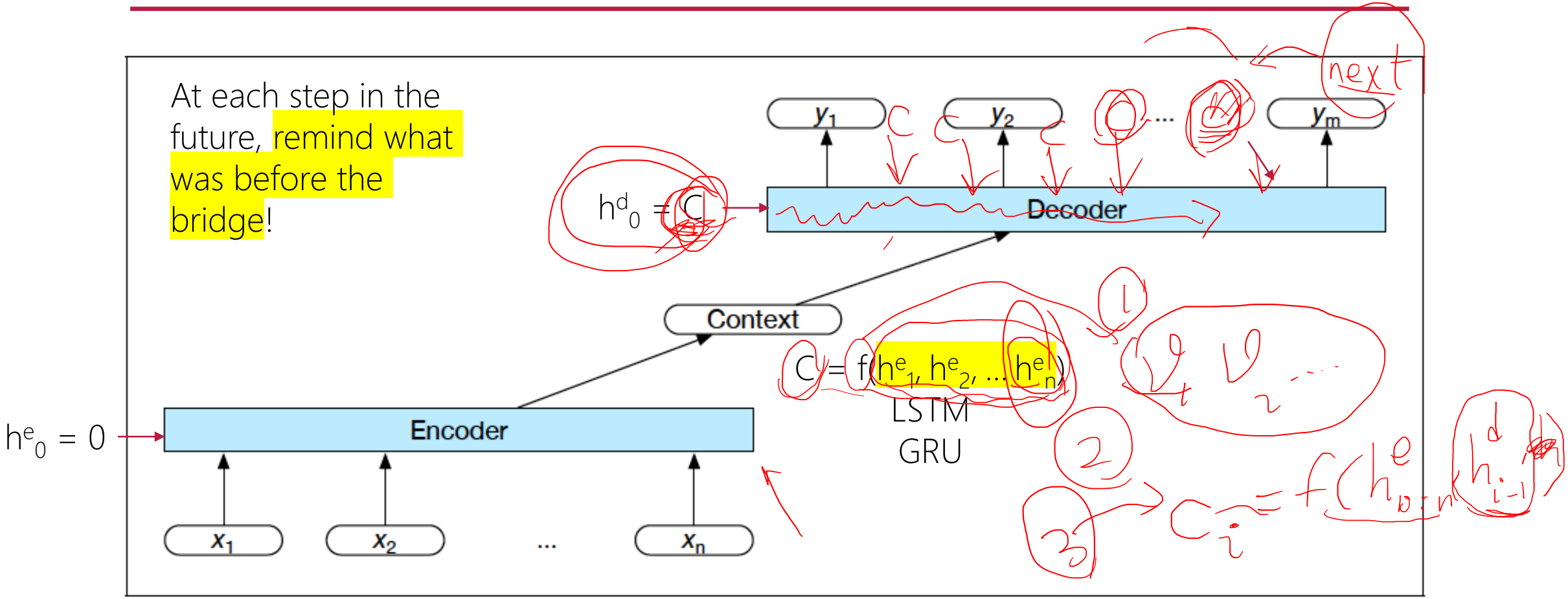


Figure 10.4 Basic architecture for an abstract encoder-decoder network. The context is a function of the vector of contextualized input representations and may be used by the decoder in a variety of ways.

Neural Machine Translation

SEQUENCE TO SEQUENCE MODEL WITH ATTENTION

Look into the Past based on future!

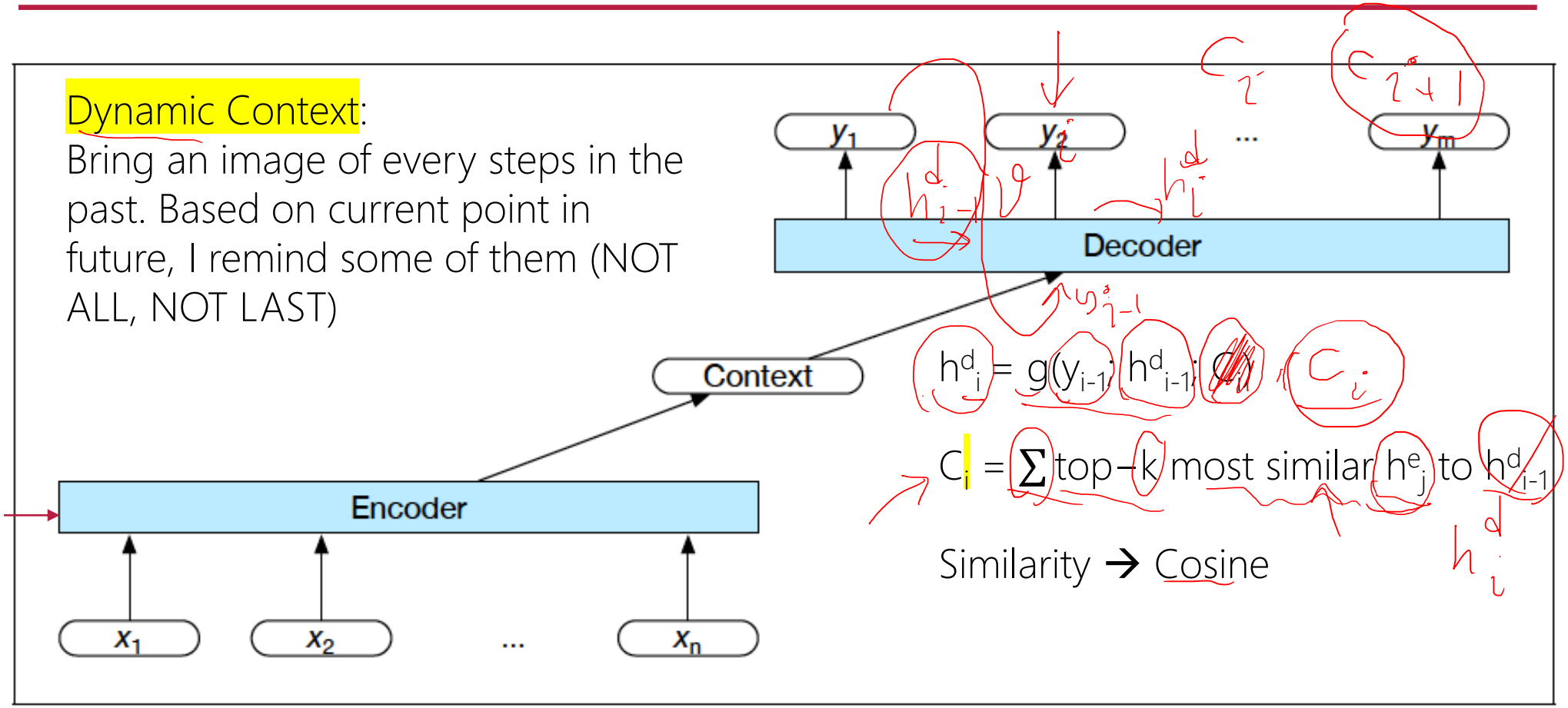


Figure 10.4 Basic architecture for an abstract encoder-decoder network. The context is a function of the vector of contextualized input representations and may be used by the decoder in a variety of ways.

Look into the Past based on future!

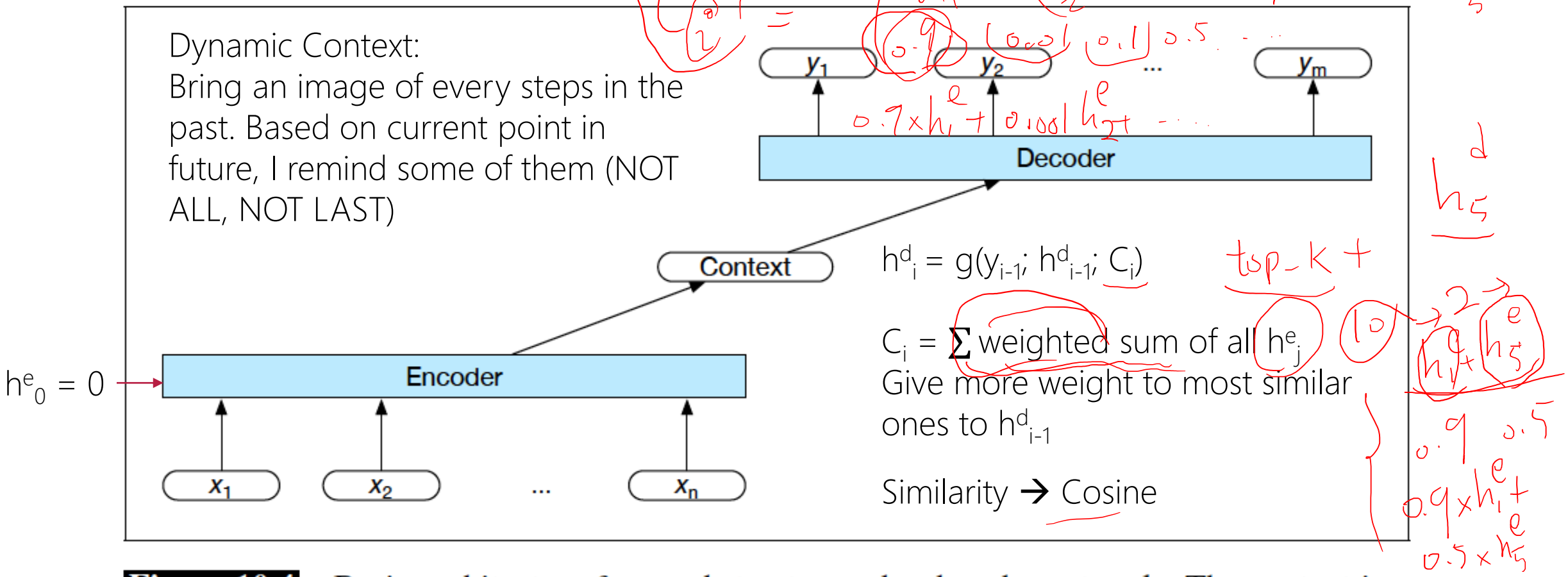
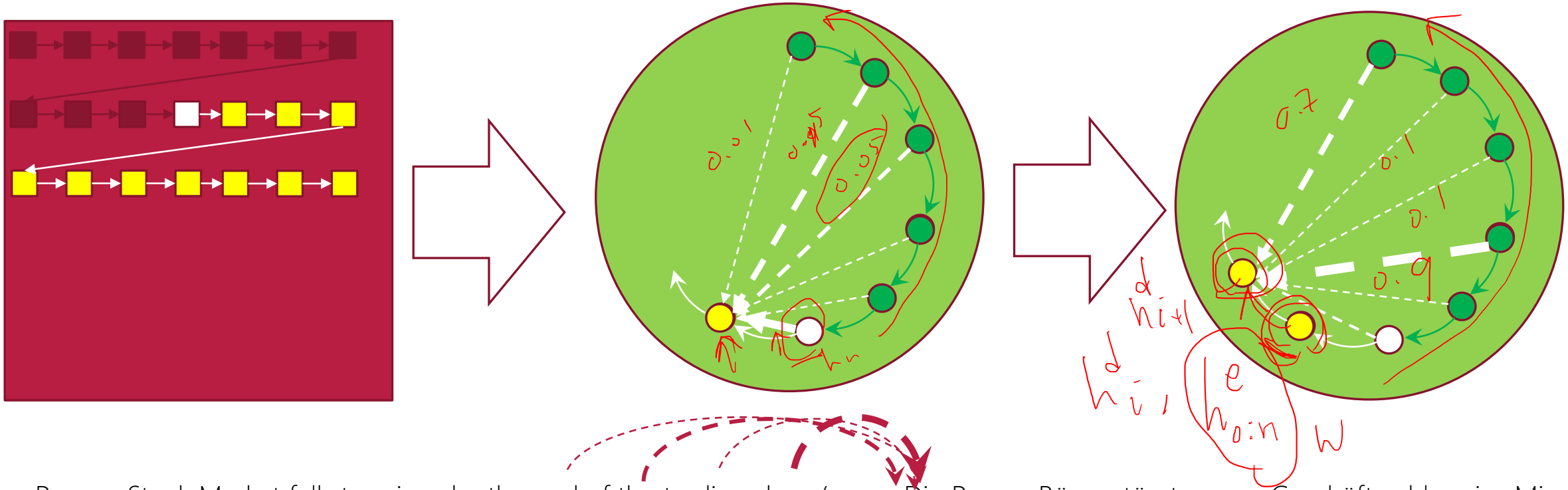


Figure 10.4 Basic architecture for an abstract encoder-decoder network. The context is a function of the vector of contextualized input representations and may be used by the decoder in a variety of ways.

Recurrent + Attention!

For machine, they are just sequences of tokens!



<s>Prague Stock Market falls to minus by the end of the trading day</s> <s>Die Prager Börse stürzt gegen Geschäftsschluss ins Minus</s>

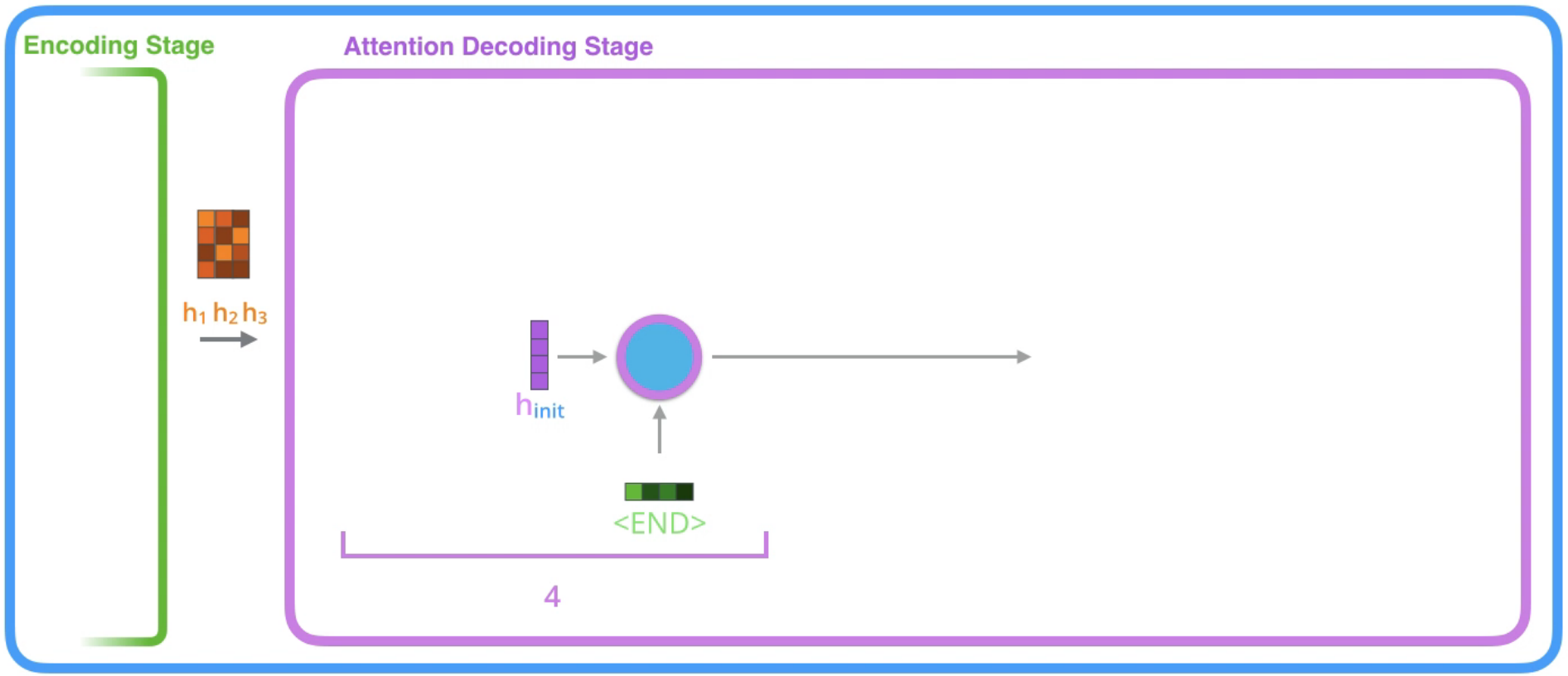
Attention at time step 4

Jay Alammar

Visualizing machine learning one concept at a time.
@JayAlammar on Twitter. YouTube Channel

Neural Machine Translation

SEQUENCE TO SEQUENCE MODEL WITH ATTENTION



Jay Alammar

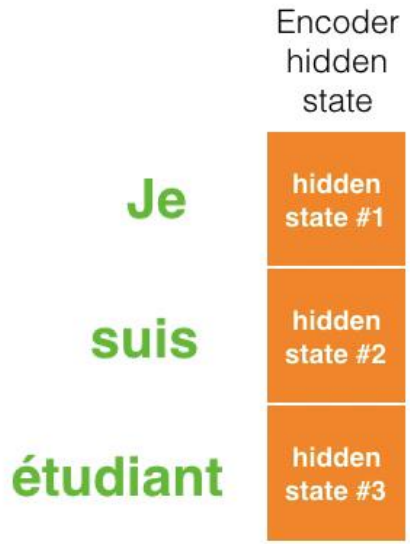
Visualizing machine learning one concept at a time.

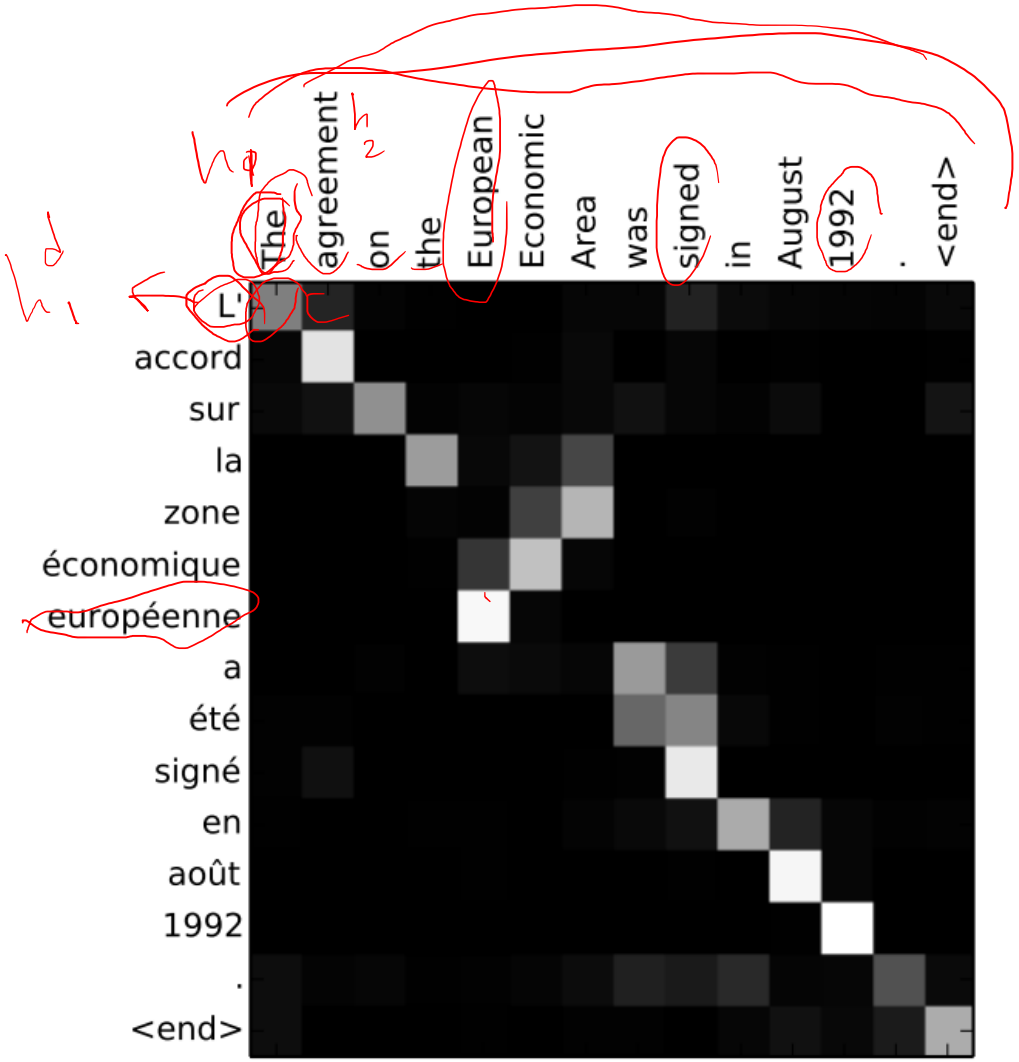
@JayAlammar on Twitter. YouTube Channel

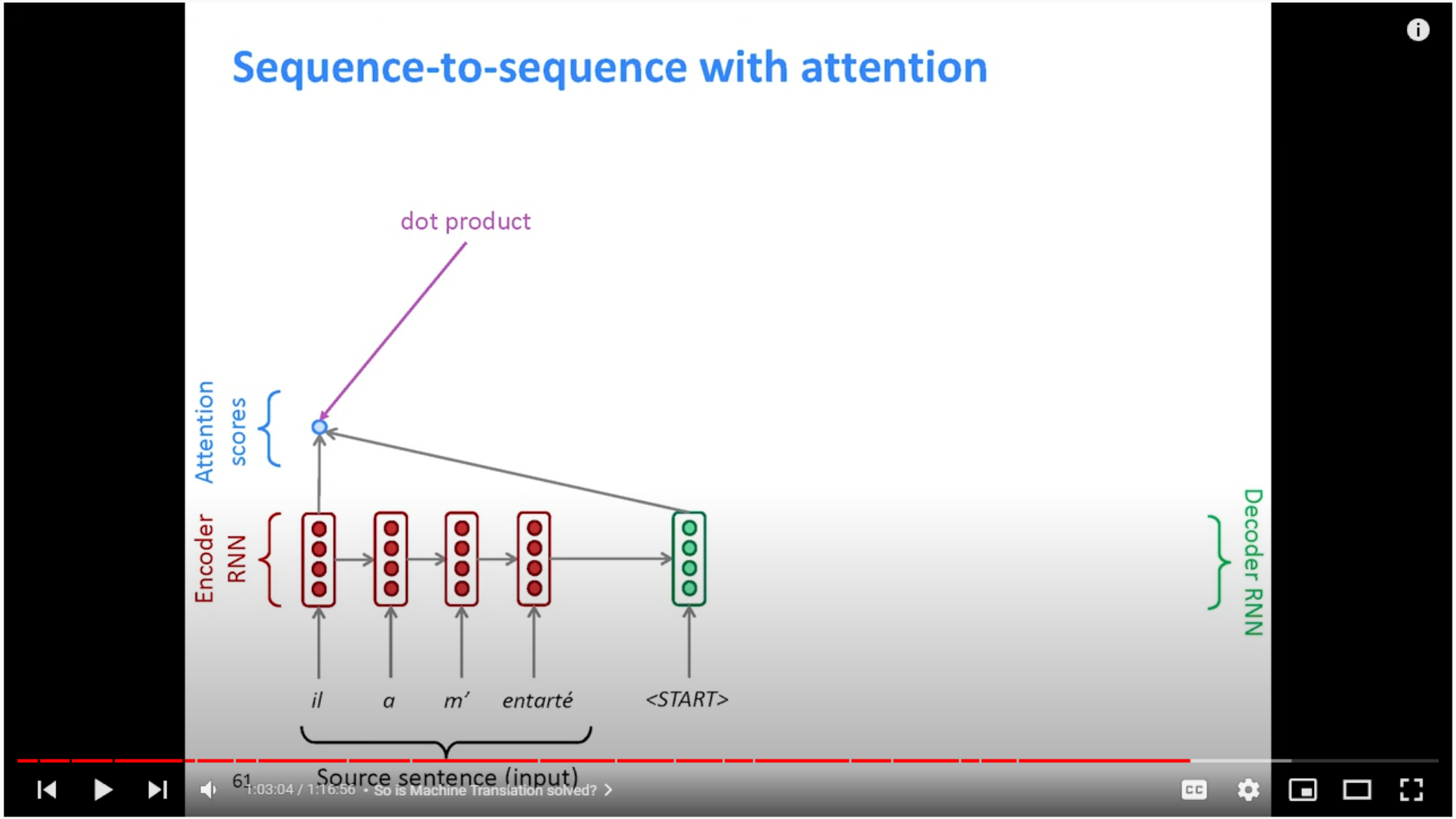
Blog About

RNN

The ag- - - - - -.







Stanford CS224N: NLP with Deep Learning | Winter 2019 | Lecture 8 – Translation, Seq2Seq, Attention

100,227 views • May 3, 2019

1.3K DISLIKE SHARE CLIP SAVE ...

Stanford Online
346K subscribers

SUBSCRIBE

Recurrent + Attention!

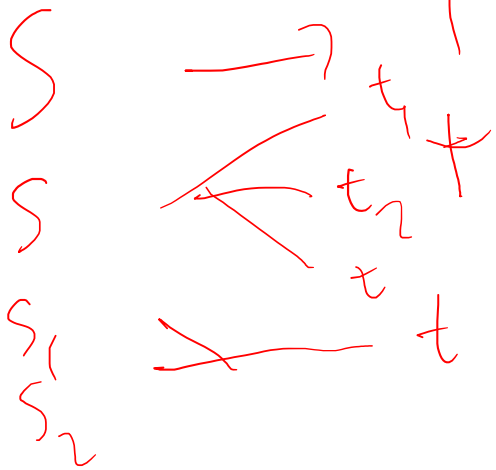
Published as a conference paper at ICLR 2015

seq2seq => MT

NEURAL MACHINE TRANSLATION BY JOINTLY LEARNING TO ALIGN AND TRANSLATE

Dzmitry Bahdanau
Jacobs University Bremen, Germany

KyungHyun Cho **Yoshua Bengio***
Université de Montréal



Recurrent + Attention!

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

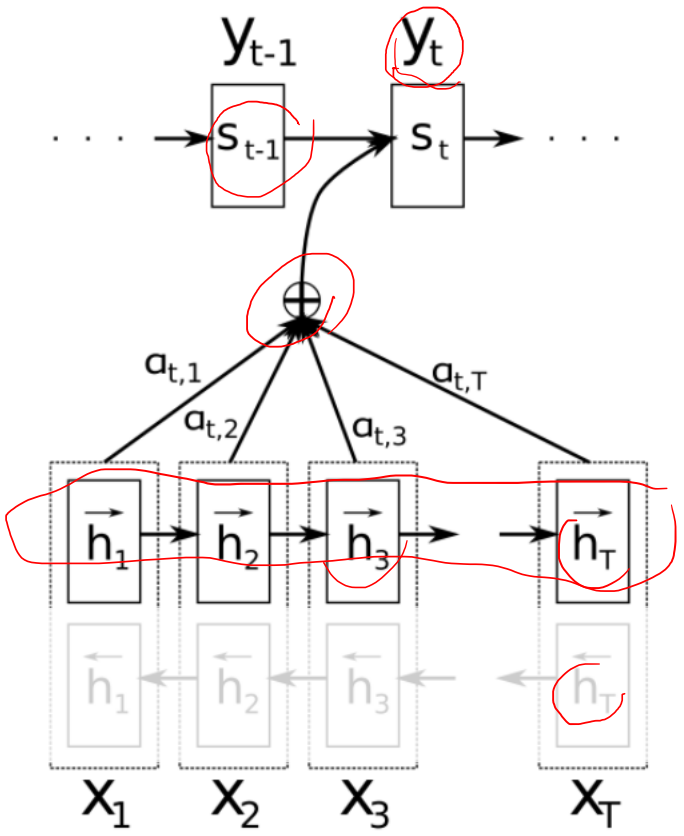


Figure 11: Attention (Bahdanau et al., 2015)

Similarity → Cosine → Learn Similarity as a functions of inputs!!

$\text{score}(h^d_{i-1}; h^e_j) = h^d_{i-1} W_s h^e_j$

$a_{ij} = \text{softmax}(\text{score}(h^d_{i-1}, h^e_j))$ for all j in encoder

$C_i = \sum_{j=1}^{n} a_{ij} h^e_j$

The point in future make a query to find out how much attention should put to what part of past (values)!

Recurrent + Attention!

A woman is throwing a frisbee in a park.

Figure 12: Visual attention in an image captioning model indicating what the model is attending to when generating the word "frisbee". (Xu et al., 2015)

~~Recurrent +~~ Attention!

Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com

Noam Shazeer* Google Brain noam@google.com

Niki Parmar* Google Research nikip@google.com

Jakob Uszkoreit* Google Research usz@google.com

Llion Jones* Google Research llion@google.com

Aidan N. Gomez* † University of Toronto aidan@cs.toronto.edu

Łukasz Kaiser* Google Brain lukaszkaiser@google.com

Illia Polosukhin* ‡ illia.polosukhin@gmail.com

Abstract

The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. Experiments on two machine translation tasks show these models to be superior in quality while being more parallelizable and requiring significantly less time to train. Our model achieves 28.4 BLEU on the WMT 2014 English-to-German translation task, improving over the existing best results, including ensembles, by over 2 BLEU. On the WMT 2014 English-to-French translation task, our model establishes a new single-model state-of-the-art BLEU score of 41.0 after training for 3.5 days on eight GPUs, a small fraction of the training costs of the best models from the literature.

Recurrent + Attention!

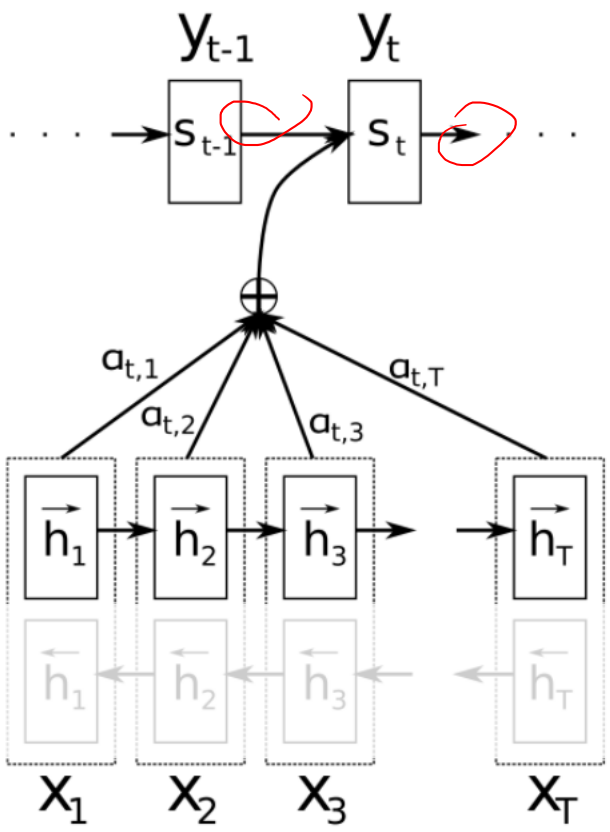


Figure 11: Attention (Bahdanau et al., 2015)

Similarity → Cosine → Learn Similarity as a functions of inputs!!

$\text{score}(h^d_{i-1}; h^e_j) = h^d_{i-1} W_s h^e_j$

$a_{ij} = \text{softmax}(\text{score}(h^d_{i-1}; h^e_j))$ for all j in encoder

$C_i = \sum_{j=1}^{n} a_{ij} h^e_j$

This is not parallelizable for all $[h^d_i]$
Why?

Recurrent + Attention!

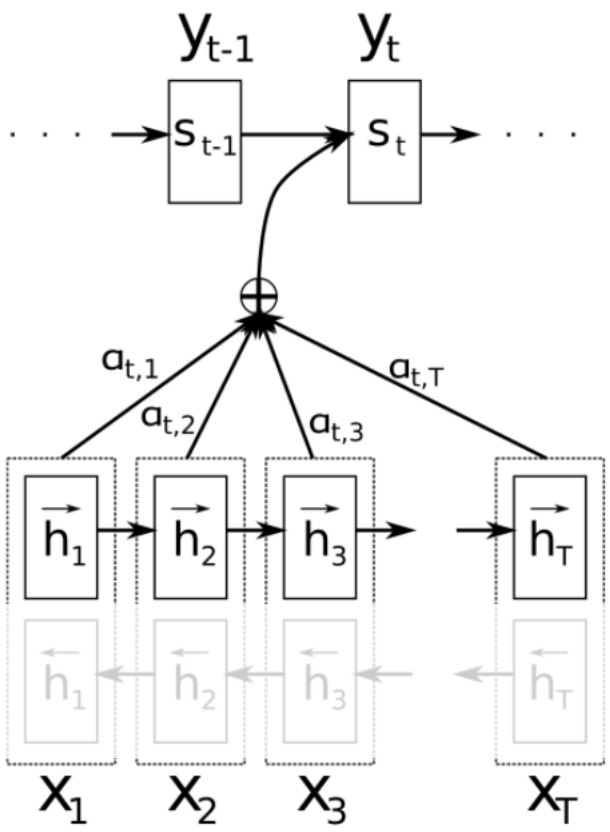


Figure 11: Attention (Bahdanau et al., 2015)

Similarity → Cosine → Learn Similarity as a functions of inputs!!

$\text{score}(h^d_{i-1}; h^e_j) = h^d_{i-1} W_s h^e_j$

$a_{ij} = \text{softmax}(\text{score}(h^d_{i-1}; h^e_j))$ for all j in encoder

$C_i = \sum_{j=1}^{n} a_{ij} h^e_j$

This is not parallelizable for all $[h^d_i]$

For $[h^d_i]$, we need to know $[h^d_{i-1}]$

For $[h^d_{i-1}]$, we need to know $[h^d_{i-2}]$

…

~~Recurrent +~~ Attention!

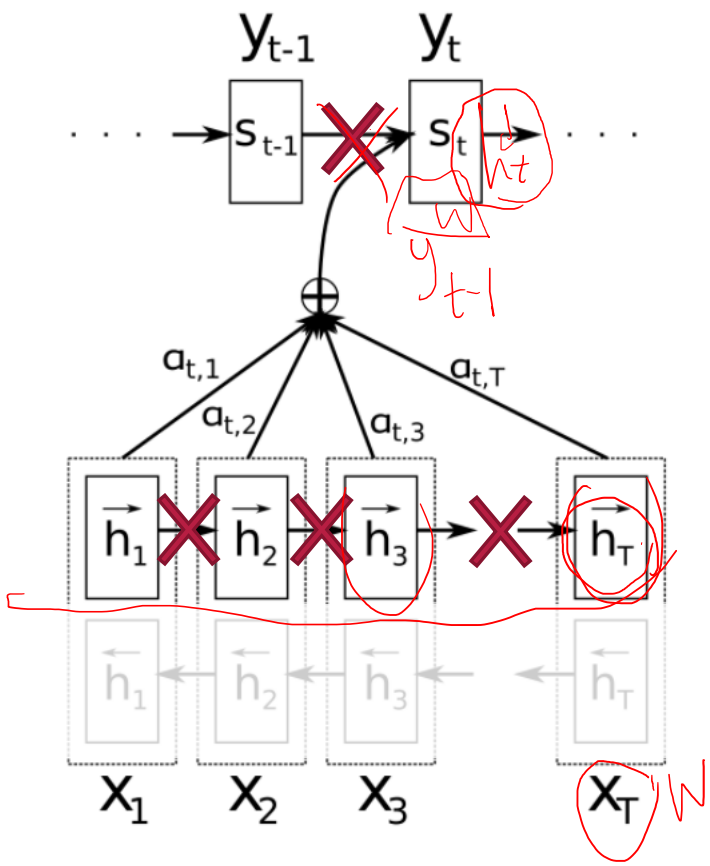


Figure 11: Attention (Bahdanau et al., 2015)

Similarity → Cosine → Learn Similarity as a functions of inputs!!

$\text{score}(h^d_{i-1}; h^e_j) = h^d_{i-1} W_s h^e_j$

$a_{ij} = \text{softmax}(\text{score}(h^d_{i-1}; h^e_j))$ for all j in encoder

$C_i = \sum_{j=1}^{n} a_{ij} h^e_j$

Drop the recurrence!

Both at encode and decoder.

~~Recurrent +~~ Attention!

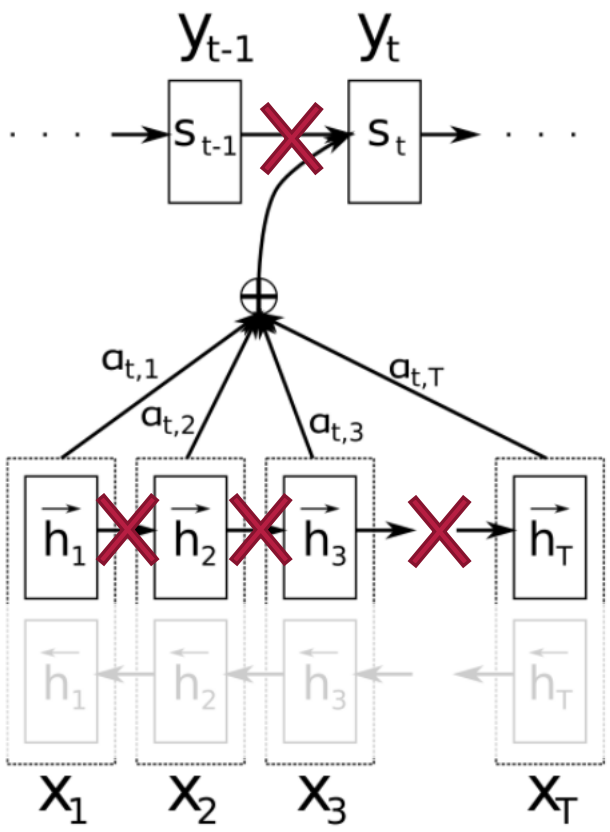


Figure 11: Attention (Bahdanau et al., 2015)

Similarity → Cosine

$score(h^d_i; h^e_j) = h^d_i \cdot h^e_j$

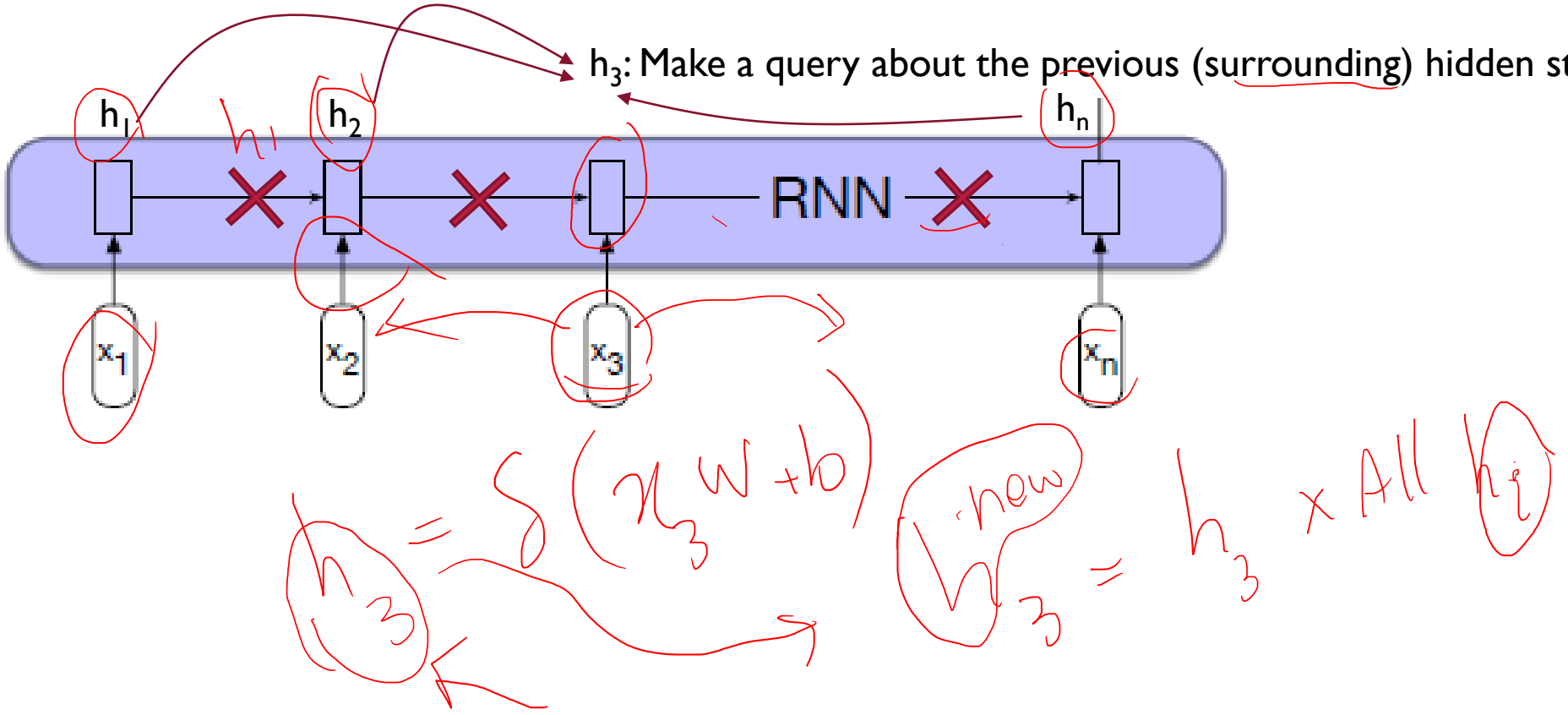
$a_{ij} = softmax(score(h^d_i; h^e_j))$ for all j in encoder

$C_i = \sum_{j=1}^{n} a_{ij} h^e_j$

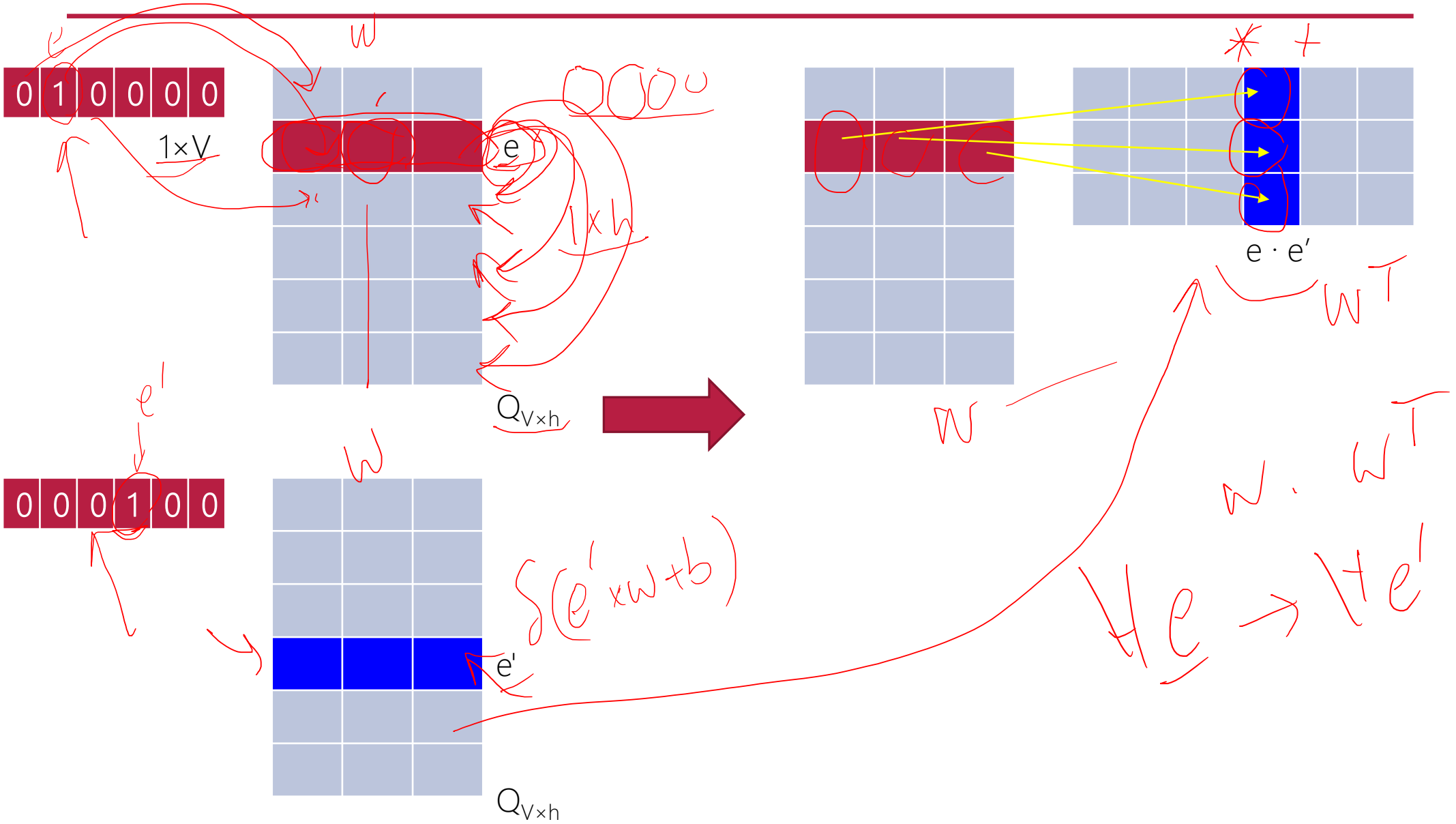
Back to Cosine!

Self-Attention

Back to Recurrent LM

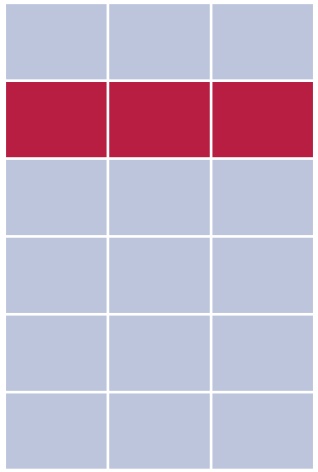


Self-Attention



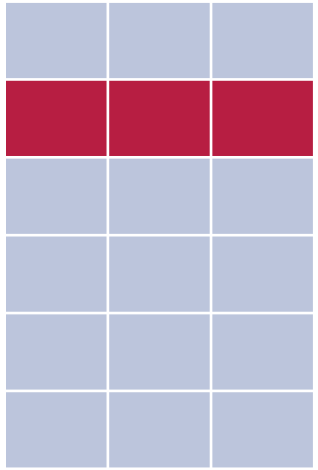
Self-Attention

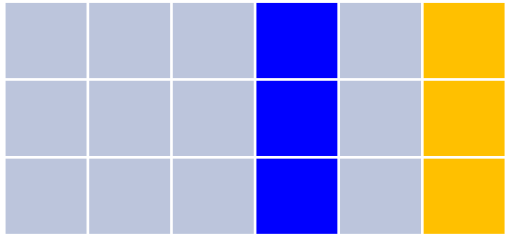
1×V



e

$Q_{V\times h}$

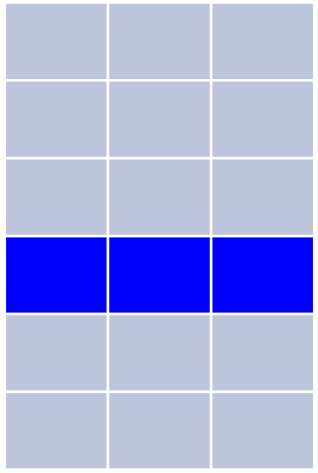




$e \cdot e'$ $e \cdot e''$

$Q_{V\times h}\, Q^{T}_{V\times h} = Q_{V\times h}\, Q_{h\times V}$

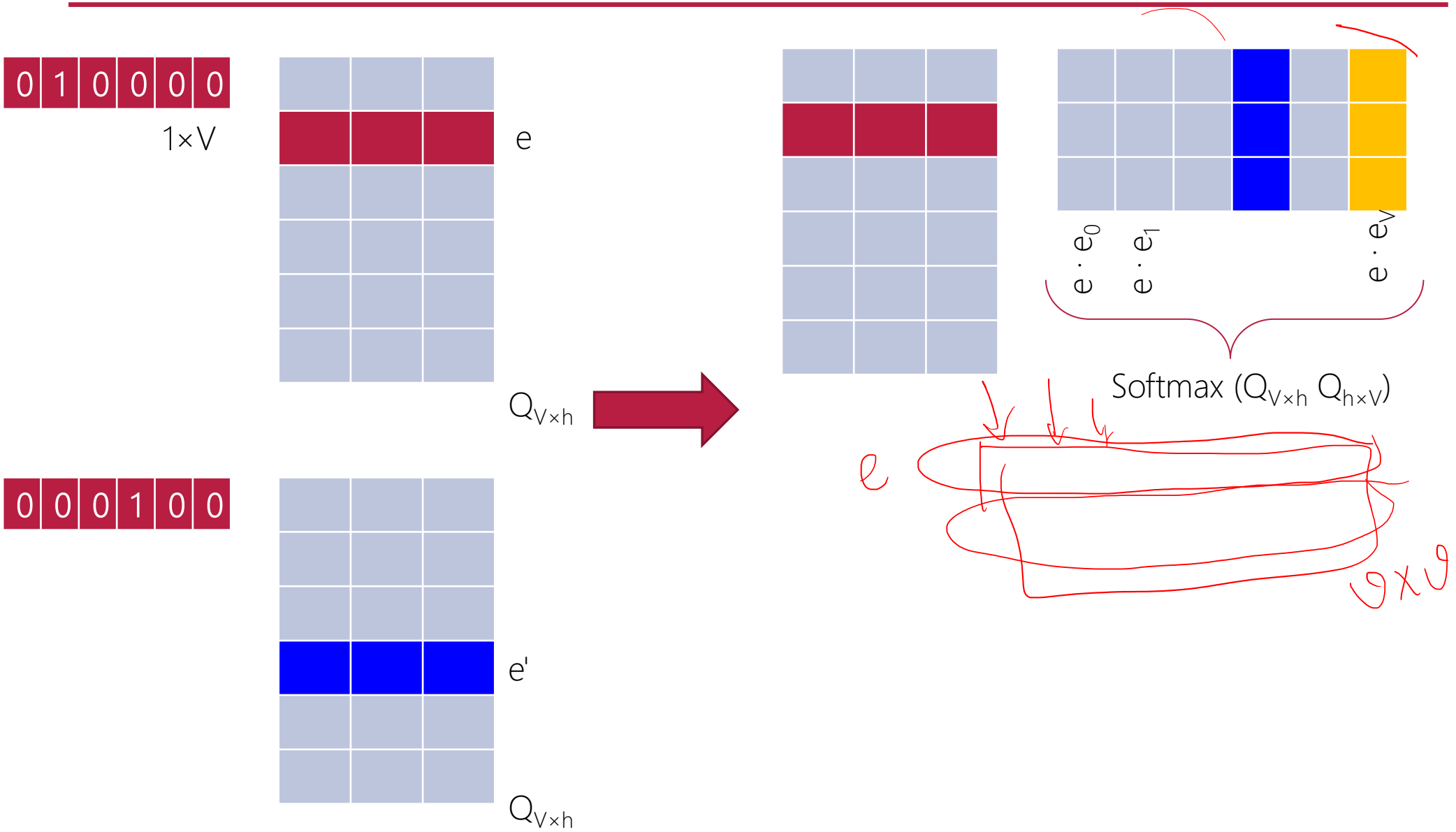
Dot Products of all word vectors with each other



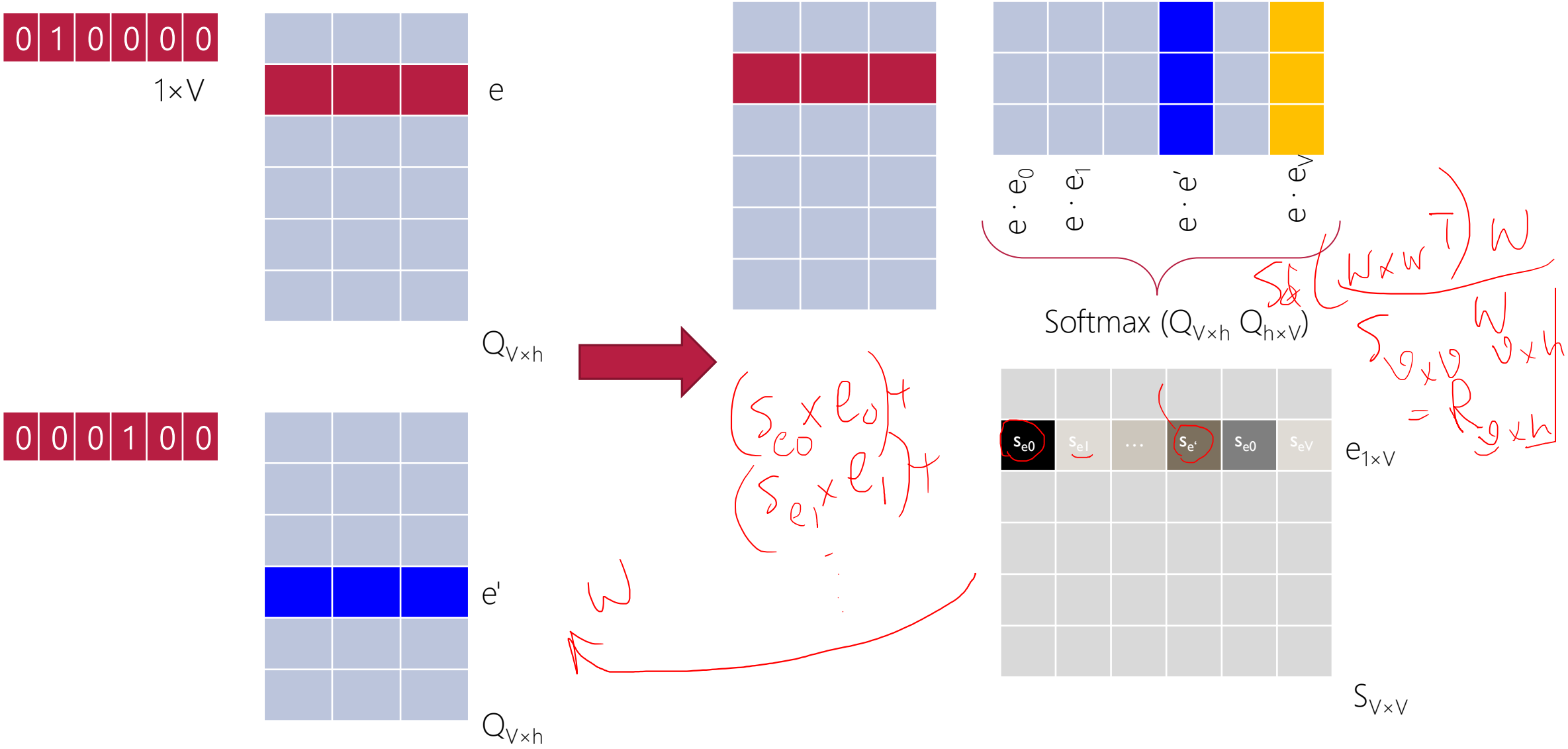
e'

$Q_{V\times h}$

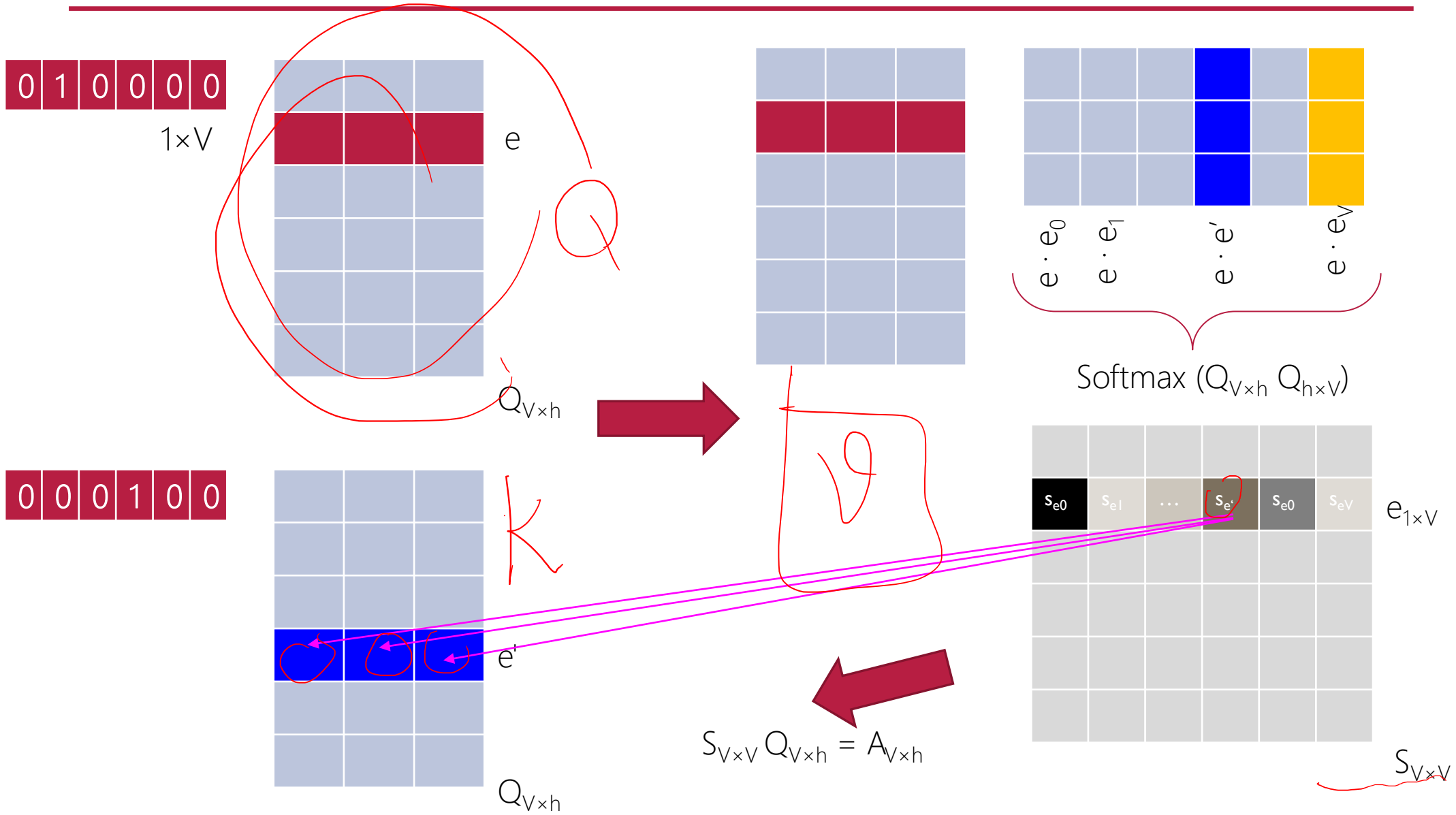
Self-Attention



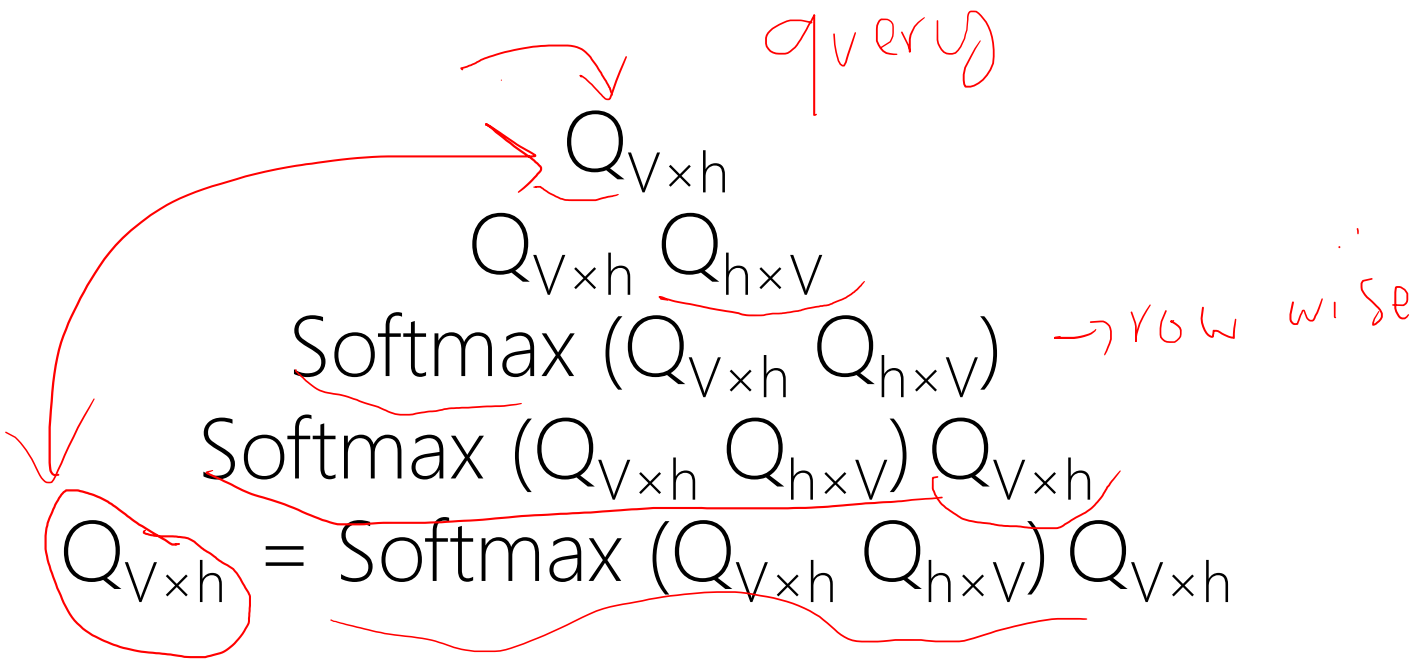
Self-Attention



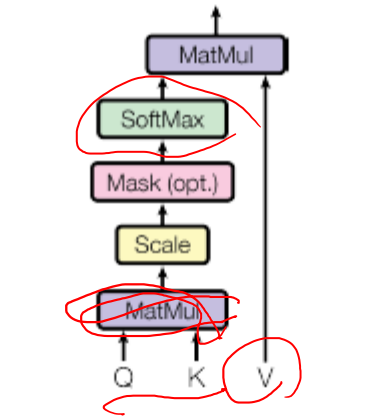
Self-Attention



Self-Attention! Renew Representation



Self-Attention! Renew Representation



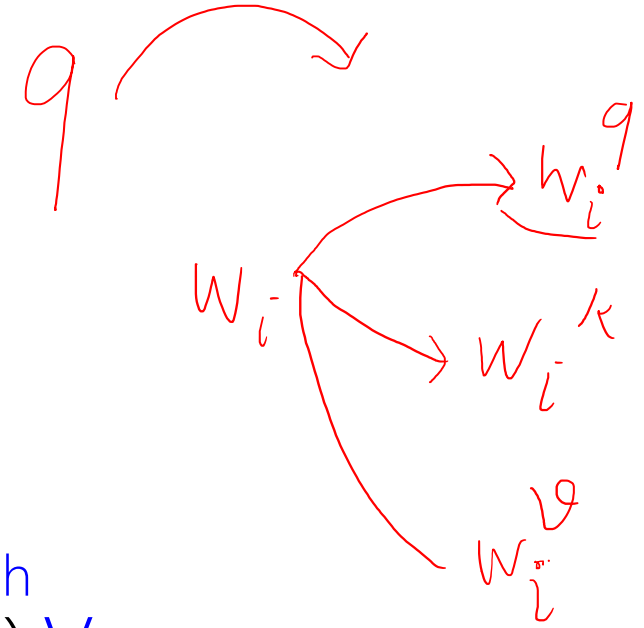
$$Q_{V\times h}$$

$$Q_{V\times h} K_{h\times V}$$

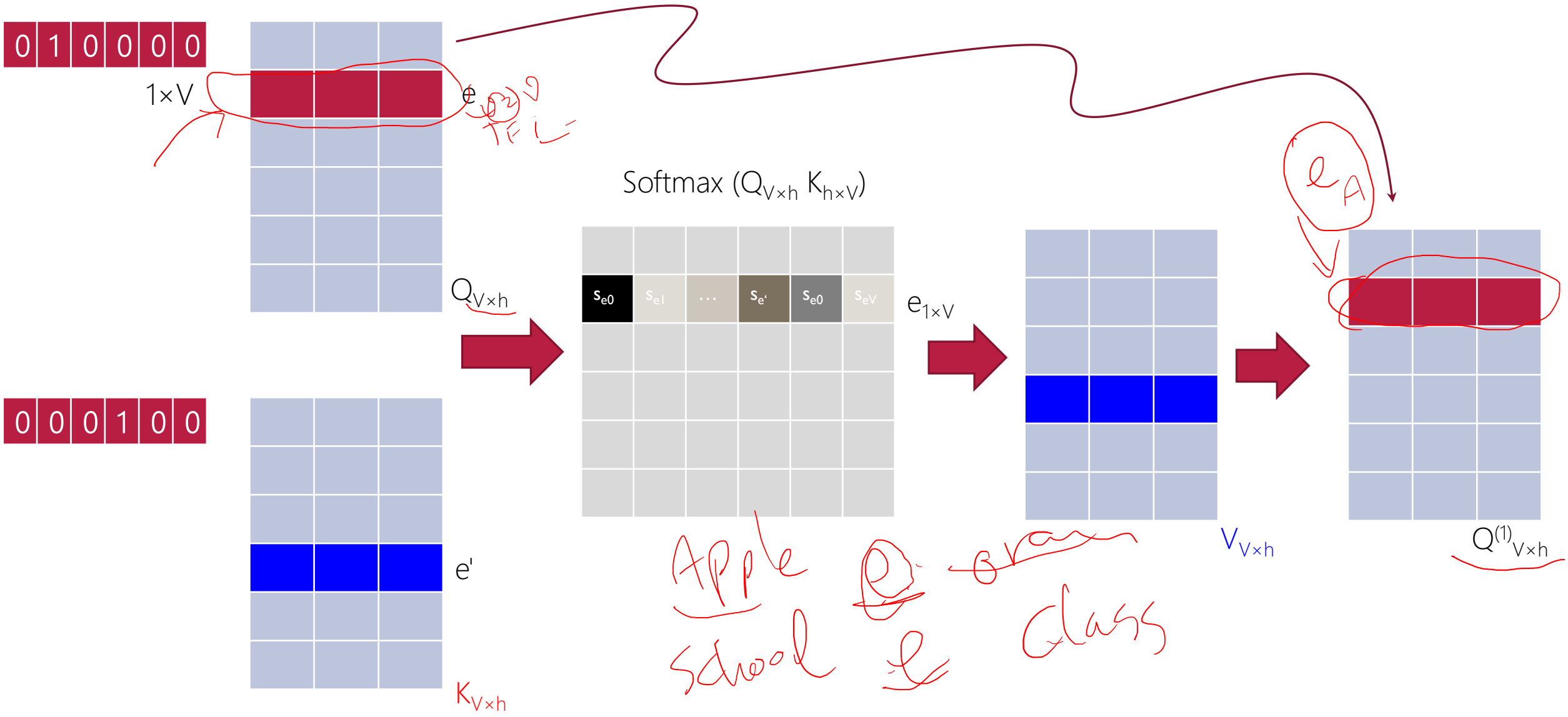
$$\text{Softmax}(Q_{V\times h} K_{h\times V})$$

$$\text{Softmax}(Q_{V\times h} K_{h\times V}) V_{V\times h}$$

$$Q^{(1)}{}_{V\times h} = \text{Softmax}(Q_{V\times h} K_{h\times V}) V_{V\times h}$$



Self-Attention



The Transformer

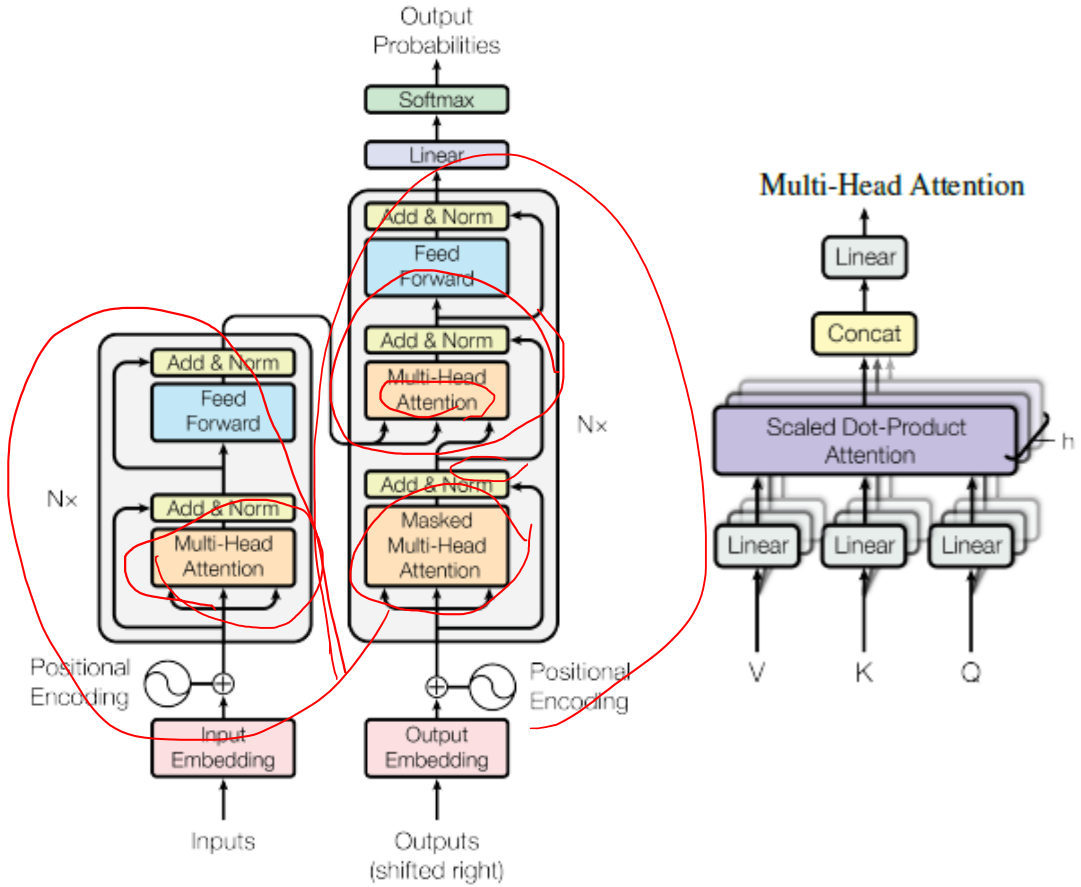


Figure 1: The Transformer - model architecture.

Contextualized Word Embedding

Positional Embedding

BERT: Bidirectional Encoder Representations from Transformers