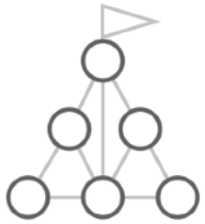


Attention Mechanisms and Transformer

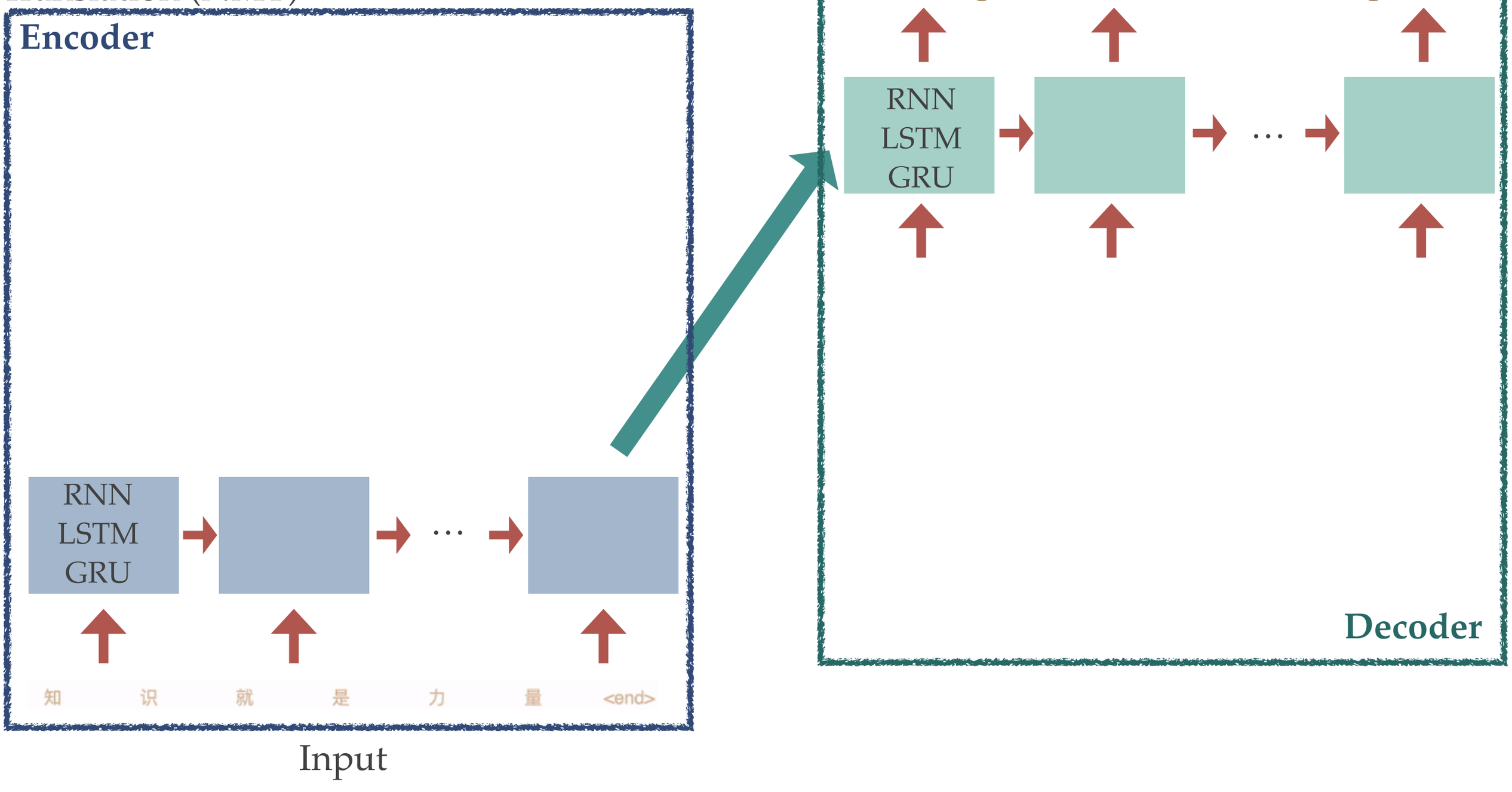
Itthi Chatnuntawech

Sequence-to-Sequence Model

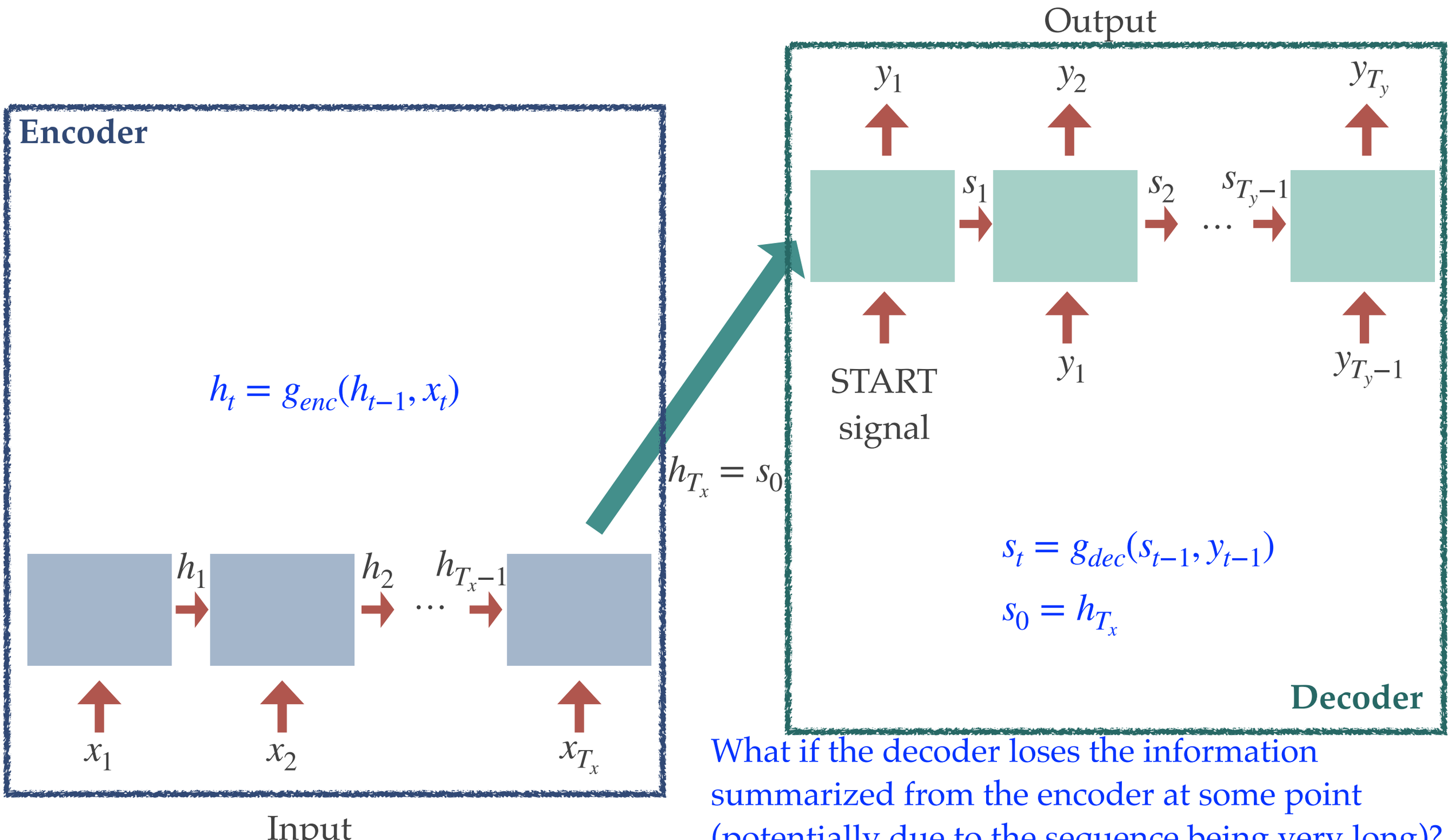
An RNN encoder-decoder model

The encoder summarizes the information from the input sequence and send it to the decoder for further processing

Task: Chinese-to-English Neural Machine Translation (NMT)

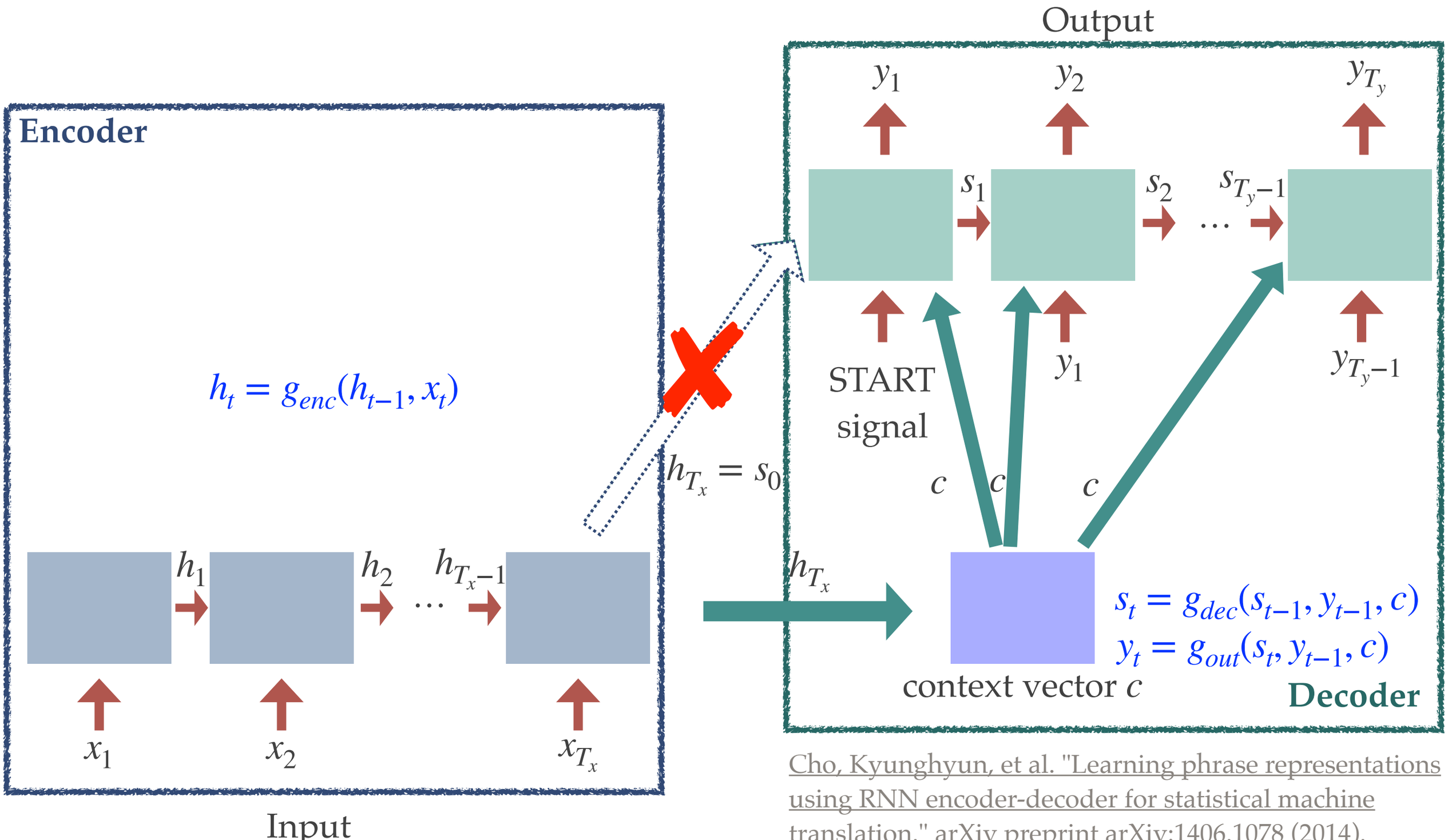


Sequence-to-Sequence Model



What if the decoder loses the information summarized from the encoder at some point (potentially due to the sequence being very long)?

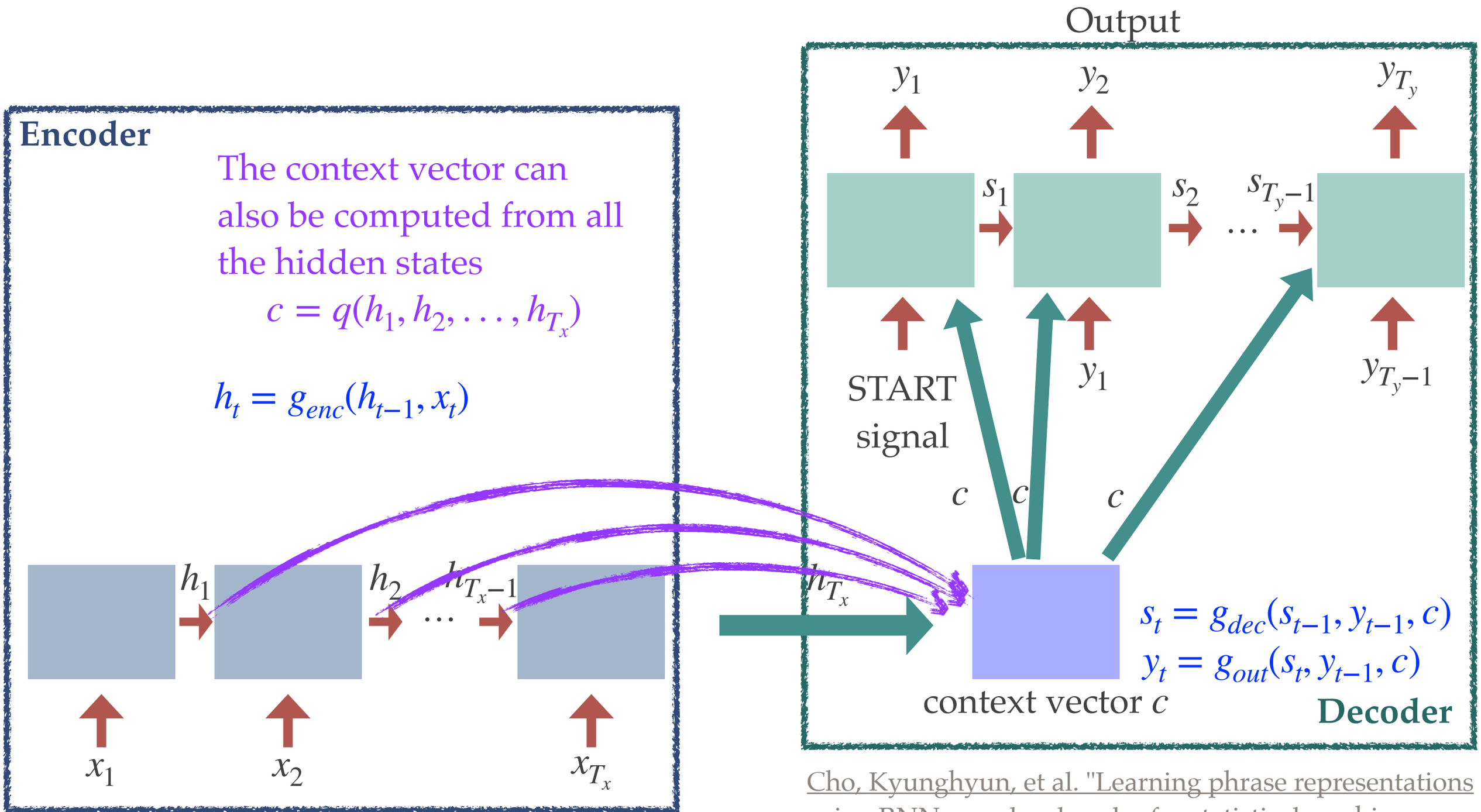
Sequence-to-Sequence Model



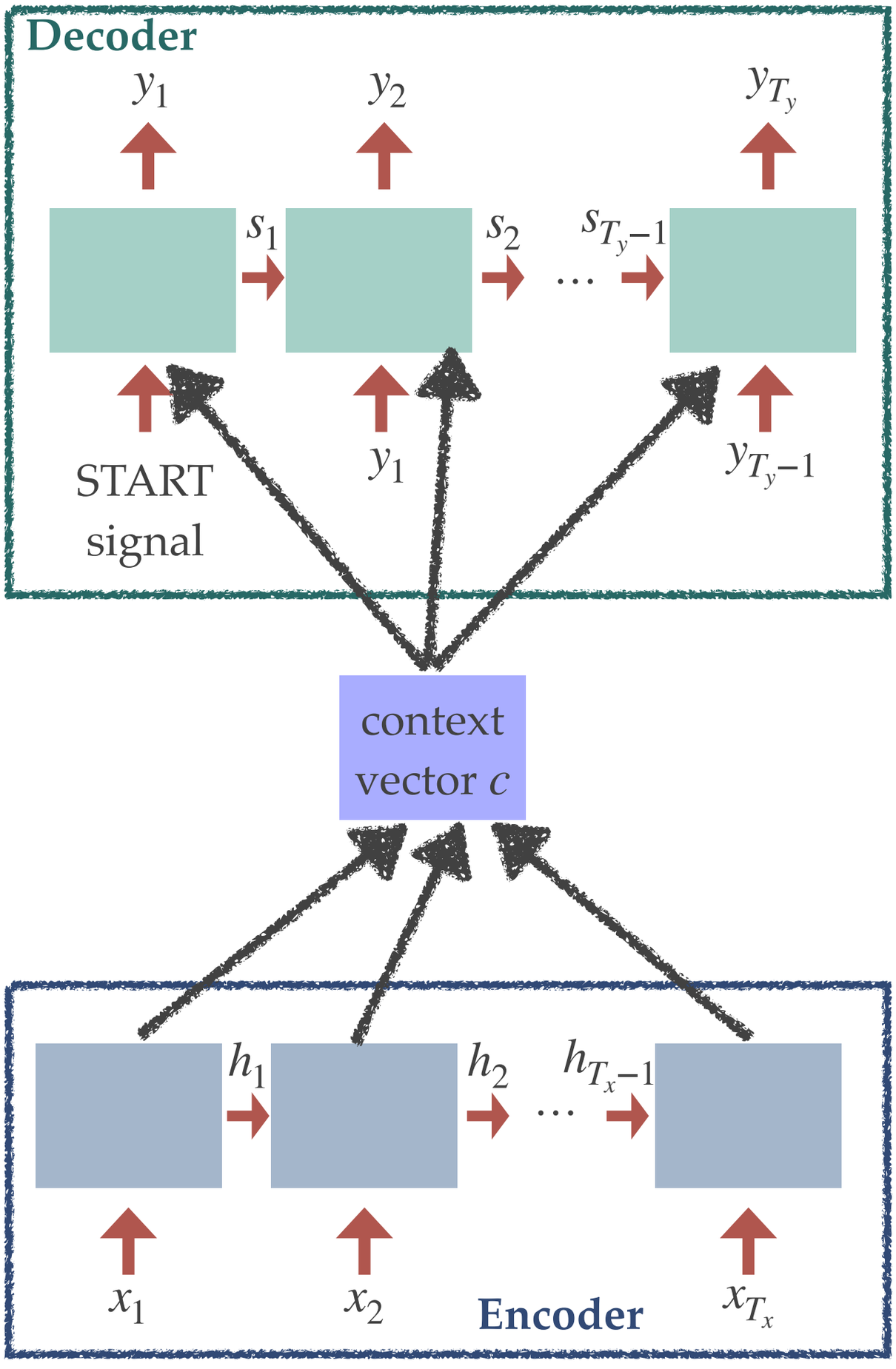
Cho, Kyunghyun, et al. "Learning phrase representations using RNN encoder-decoder for statistical machine translation." arXiv preprint arXiv:1406.1078 (2014).

Question: Is a single context vector enough to capture all important information on the input?

Answer: Most likely not → **Attention Mechanism** where each step of the decoder can access information from all hidden states of the encoder



Cho, Kyunghyun, et al. "Learning phrase representations using RNN encoder-decoder for statistical machine translation." arXiv preprint arXiv:1406.1078 (2014).



Without attention mechanism

$$s_t = g_{dec}(s_{t-1}, y_{t-1}, c)$$
$$y_t = g_{out}(s_t, y_{t-1}, c)$$
$$c = q(h_1, h_2, \ldots, h_{T_x})$$

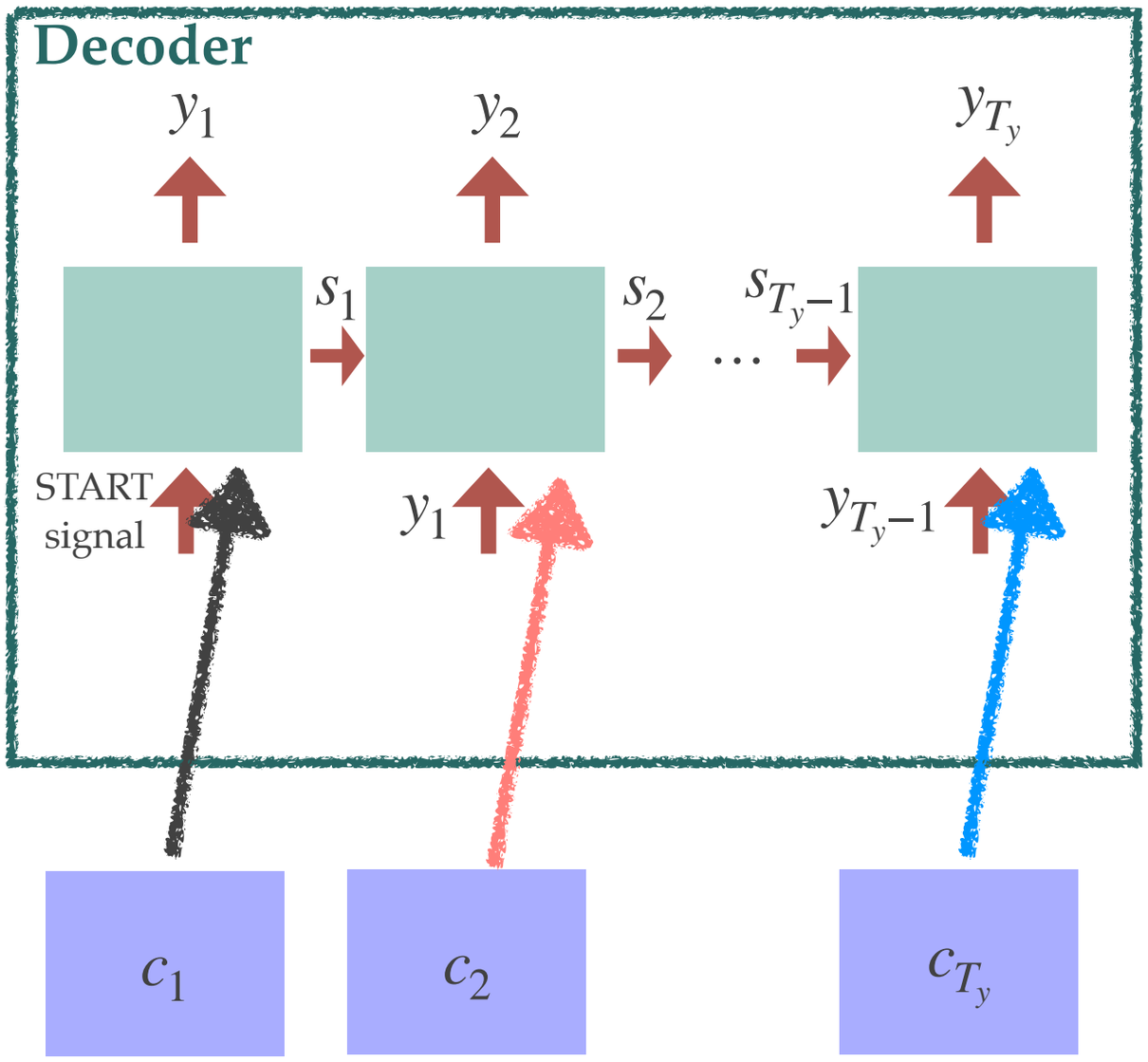
With attention mechanism

$$s_t = g_{dec}(s_{t-1}, y_{t-1}, c_t)$$
$$y_t = g_{out}(s_t, y_{t-1}, c_t)$$
$$c_t = q_t(h_1, h_2, \ldots, h_{T_x}, s_{t-1})$$

This is just one of many ways to include attention mechanisms (simplified from Bahdanau et al. ICLR 2015)

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

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." arXiv preprint arXiv:1409.0473 (2014).



Without attention mechanism

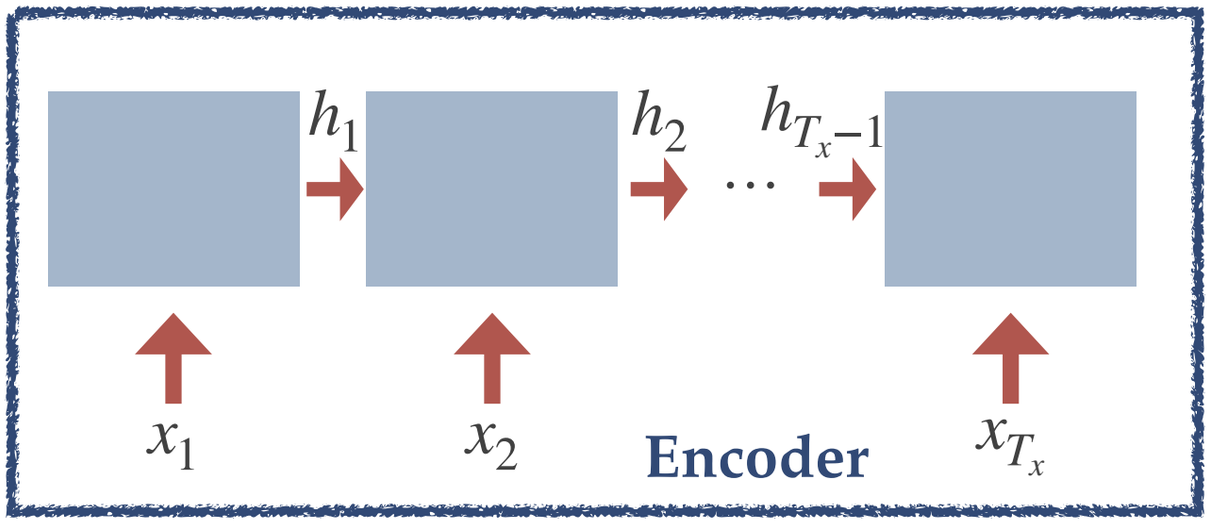
$$s_t = g_{dec}(s_{t-1}, y_{t-1}, c)$$
$$y_t = g_{out}(s_t, y_{t-1}, c)$$
$$c = q(h_1, h_2, \ldots, h_{T_x})$$

With attention mechanism

$$s_t = g_{dec}(s_{t-1}, y_{t-1}, c_t)$$
$$y_t = g_{out}(s_t, y_{t-1}, c_t)$$
$$c_t = q_t(h_1, h_2, \ldots, h_{T_x}, s_{t-1})$$

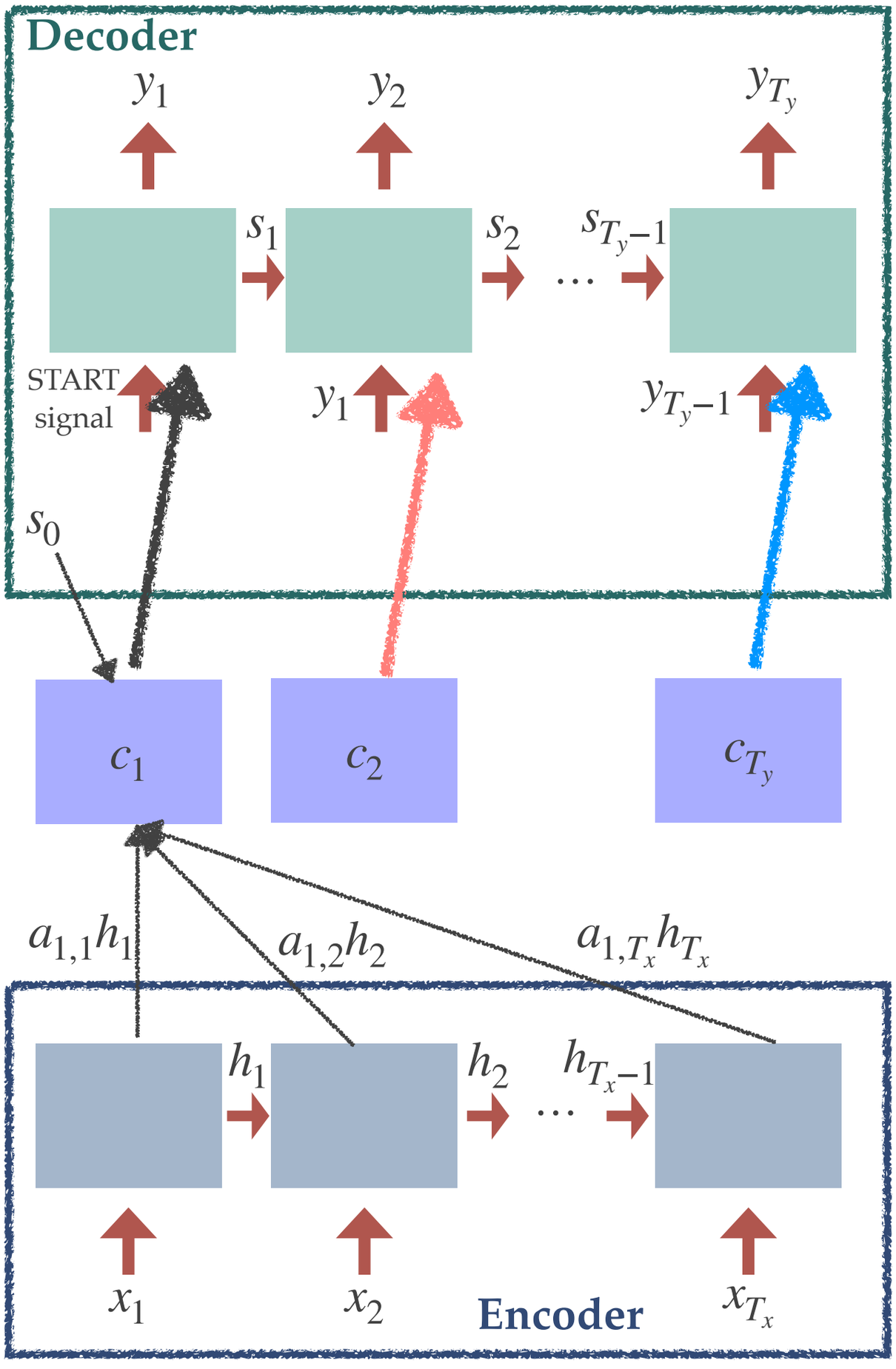
Different c_t's for different time points

The context vector is a weighted sum of the encoder hidden states



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

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." arXiv preprint arXiv:1409.0473 (2014).



Without attention mechanism

$$s_t = g_{dec}(s_{t-1}, y_{t-1}, c)$$
$$y_t = g_{out}(s_t, y_{t-1}, c)$$
$$c = q(h_1, h_2, \ldots, h_{T_x})$$

With attention mechanism

$$s_t = g_{dec}(s_{t-1}, y_{t-1}, c_t)$$
$$y_t = g_{out}(s_t, y_{t-1}, c_t)$$
$$c_t = q_t(h_1, h_2, \ldots, h_{T_x}, s_{t-1})$$

Different c_t's for different time points

The context vector is a weighted sum of the encoder hidden states

$$c_1 = a_{1,1}h_1 + a_{1,2}h_2 + \ldots + a_{1,T_x}h_{T_x}$$

$$1 = a_{1,1} + a_{1,2} + \ldots + a_{1,T_x}$$
$$0 \le a_{1,i}$$

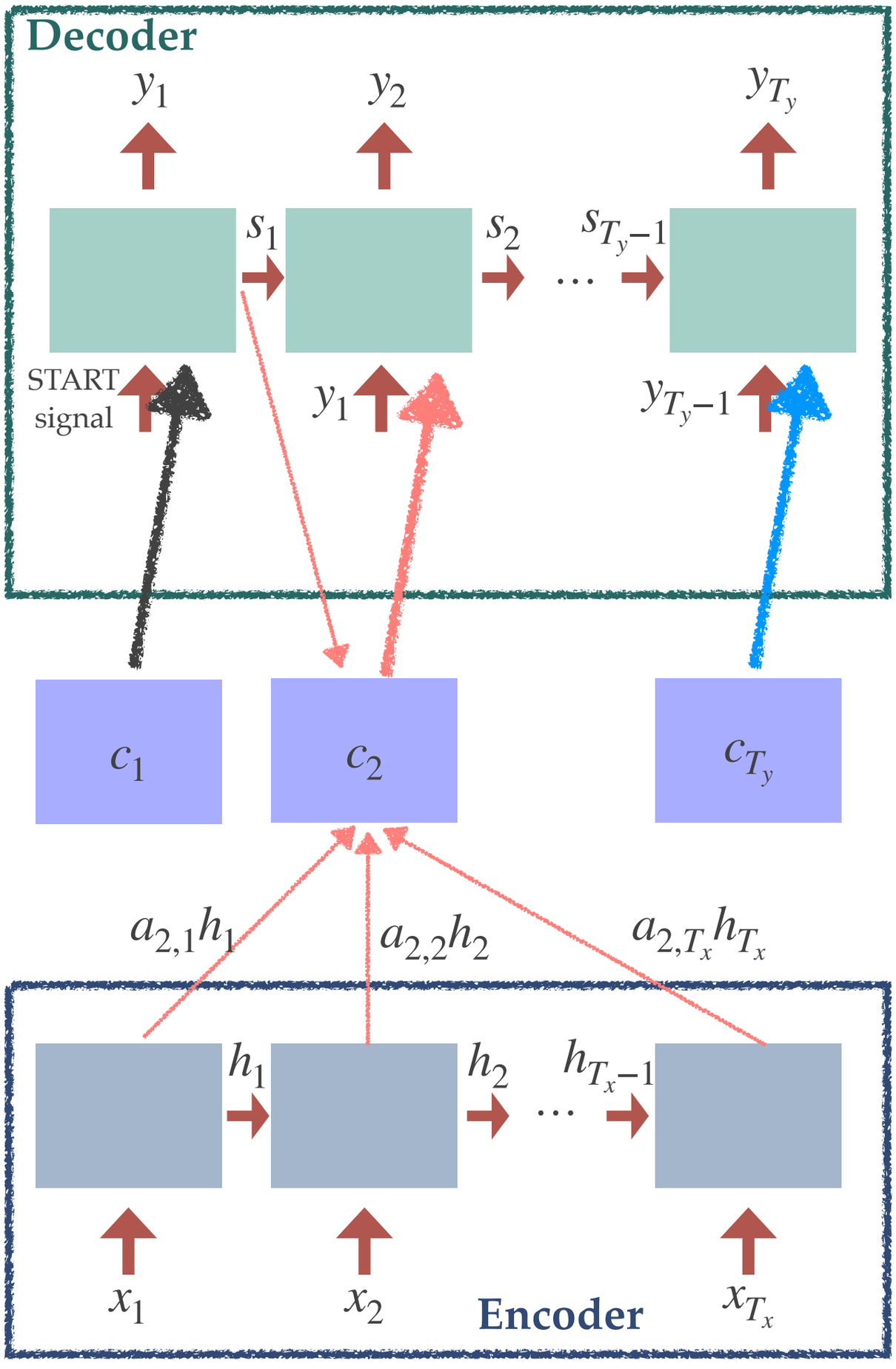
These constraints can be enforced using softmax

$$a_{1,i} = score(s_0, h_i)$$

A function that computes a "similarity" score between the two inputs (e.g., inner product)

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

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." arXiv preprint arXiv:1409.0473 (2014).



Without attention mechanism

$$s_t = g_{dec}(s_{t-1}, y_{t-1}, c)$$
$$y_t = g_{out}(s_t, y_{t-1}, c)$$
$$c = q(h_1, h_2, \ldots, h_{T_x})$$

With attention mechanism

$$s_t = g_{dec}(s_{t-1}, y_{t-1}, c_t)$$
$$y_t = g_{out}(s_t, y_{t-1}, c_t)$$
$$c_t = q_t(h_1, h_2, \ldots, h_{T_x}, s_{t-1})$$

Different c_t's for different time points

The context vector is a weighted sum of the encoder hidden states

$$c_2 = a_{2,1}h_1 + a_{2,2}h_2 + \ldots + a_{2,T_x}h_{T_x}$$

$$1 = a_{2,1} + a_{2,2} + \ldots + a_{2,T_x}$$
$$0 \le a_{2,i}$$

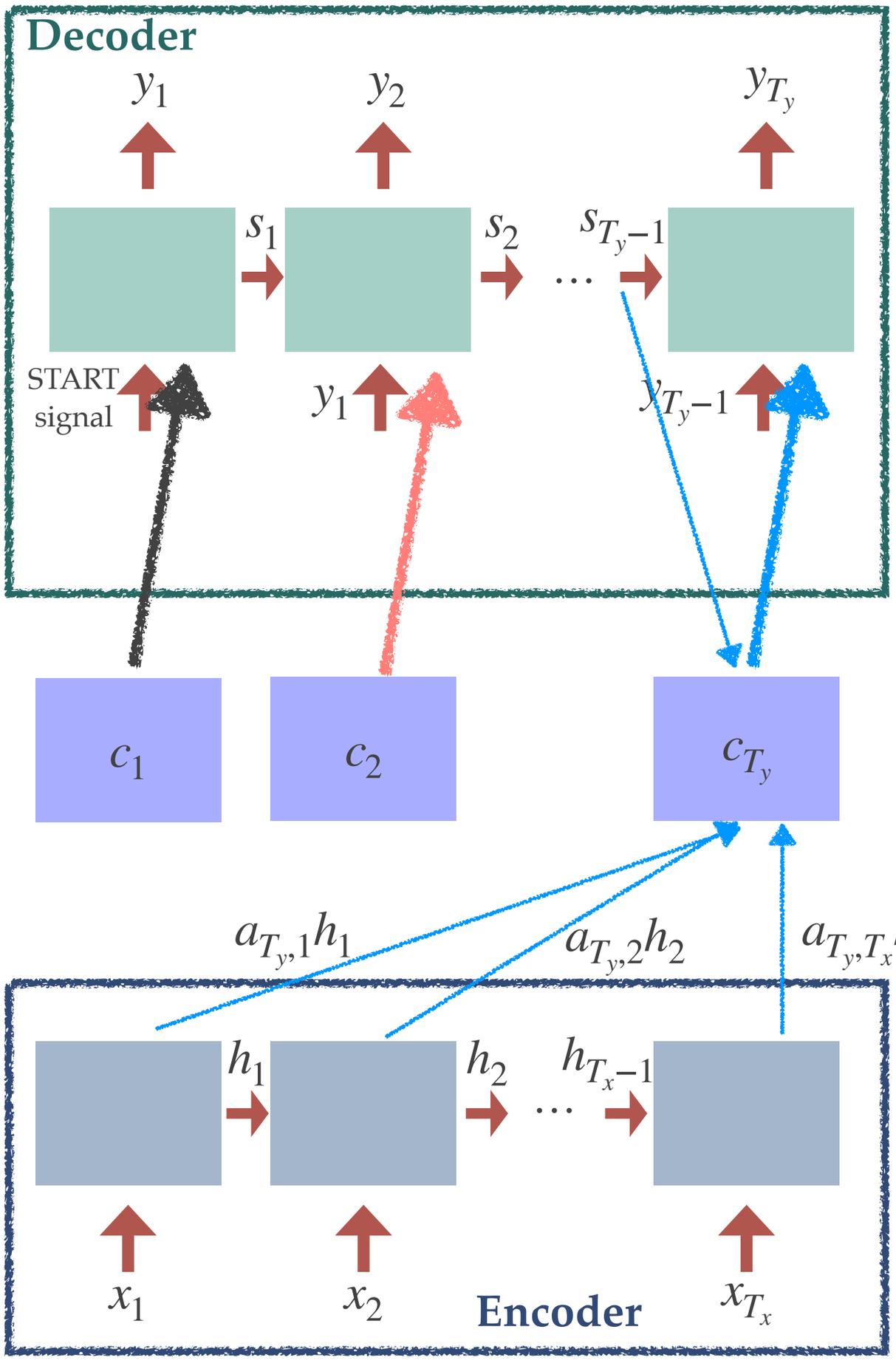
These constraints can be enforced using softmax

$$a_{2,i} = score(s_1, h_i)$$

A function that computes a "similarity" score between the two inputs (e.g., inner product)

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

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." arXiv preprint arXiv:1409.0473 (2014).



Without attention mechanism

$$s_t = g_{dec}(s_{t-1}, y_{t-1}, c)$$
$$y_t = g_{out}(s_t, y_{t-1}, c)$$
$$c = q(h_1, h_2, \ldots, h_{T_x})$$

With attention mechanism

$$s_t = g_{dec}(s_{t-1}, y_{t-1}, c_t)$$
$$y_t = g_{out}(s_t, y_{t-1}, c_t)$$
$$c_t = q_t(h_1, h_2, \ldots, h_{T_x}, s_{t-1})$$

Different c_t's for different time points

The context vector is a weighted sum of the encoder hidden states

$$c_{T_y} = a_{T_y,1}h_1 + a_{T_y,2}h_2 + \ldots + a_{T_y,T_x}h_{T_x}$$

$$1 = a_{T_y,1} + a_{T_y,2} + \ldots + a_{T_y,T_x}$$
$$0 \le a_{T_y,i}$$

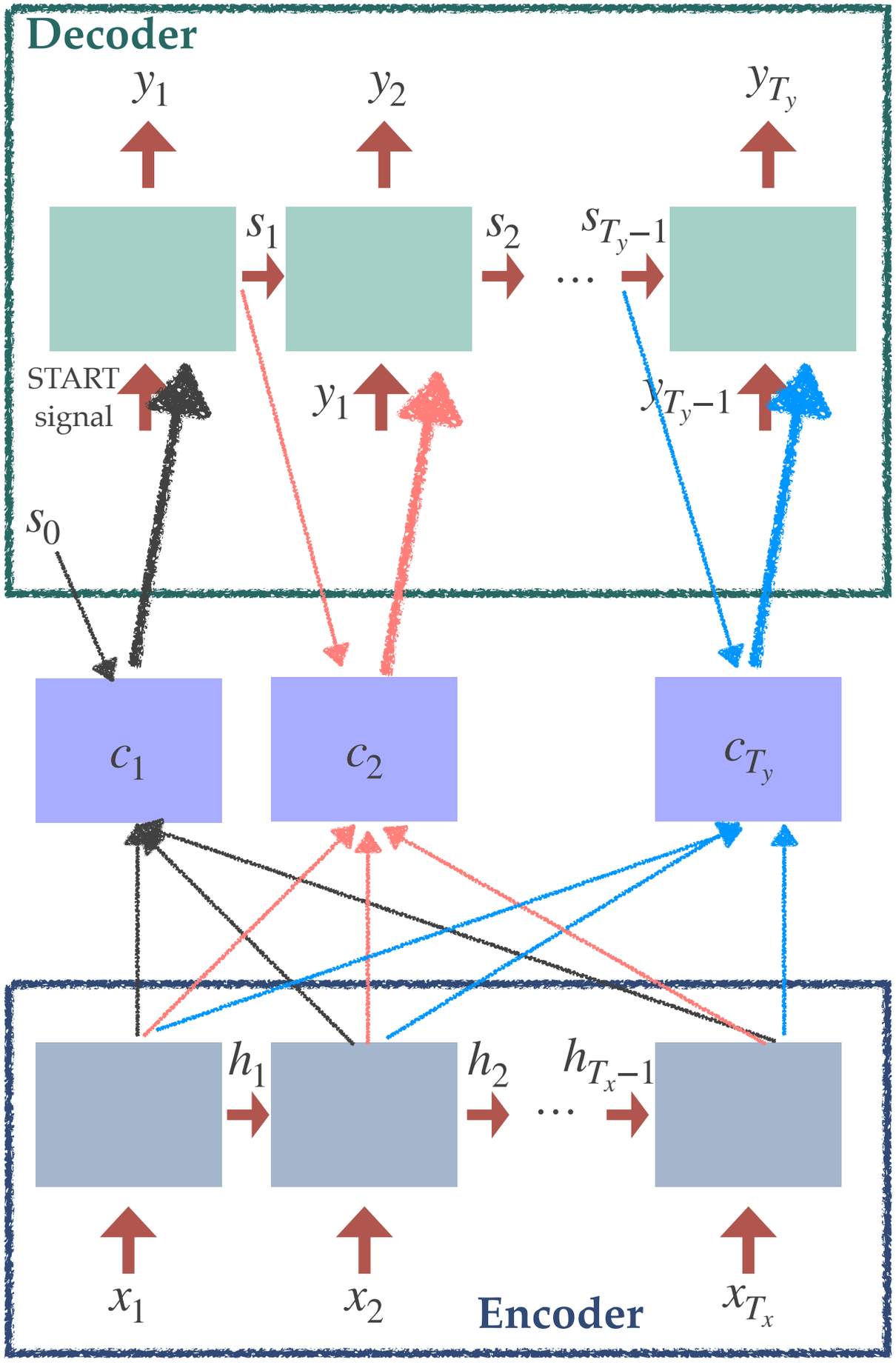
These constraints can be enforced using softmax

$$a_{T_y,i} = score(s_{T_y-1}, h_i)$$

A function that computes a "similarity" score between the two inputs (e.g., inner product)

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

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." arXiv preprint arXiv:1409.0473 (2014).



Without attention mechanism

$$s_t = g_{dec}(s_{t-1}, y_{t-1}, c)$$
$$y_t = g_{out}(s_t, y_{t-1}, c)$$
$$c = q(h_1, h_2, \ldots, h_{T_x})$$

With attention mechanism

$$s_t = g_{dec}(s_{t-1}, y_{t-1}, c_t)$$
$$y_t = g_{out}(s_t, y_{t-1}, c_t)$$
$$c_t = q_t(h_1, h_2, \ldots, h_{T_x}, s_{t-1})$$

Different c_t's for different time points

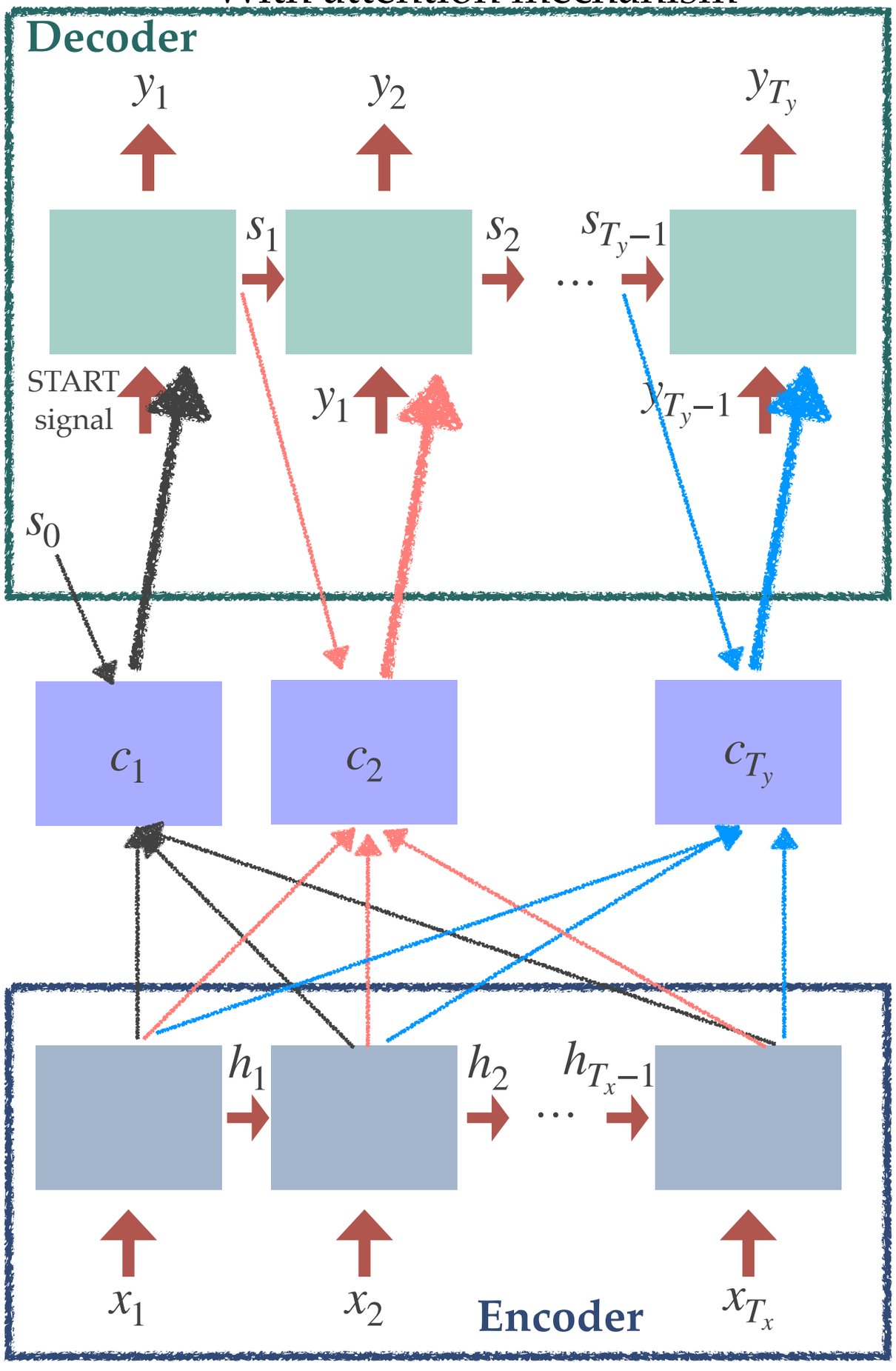
The context vector is a weighted sum of the encoder hidden states

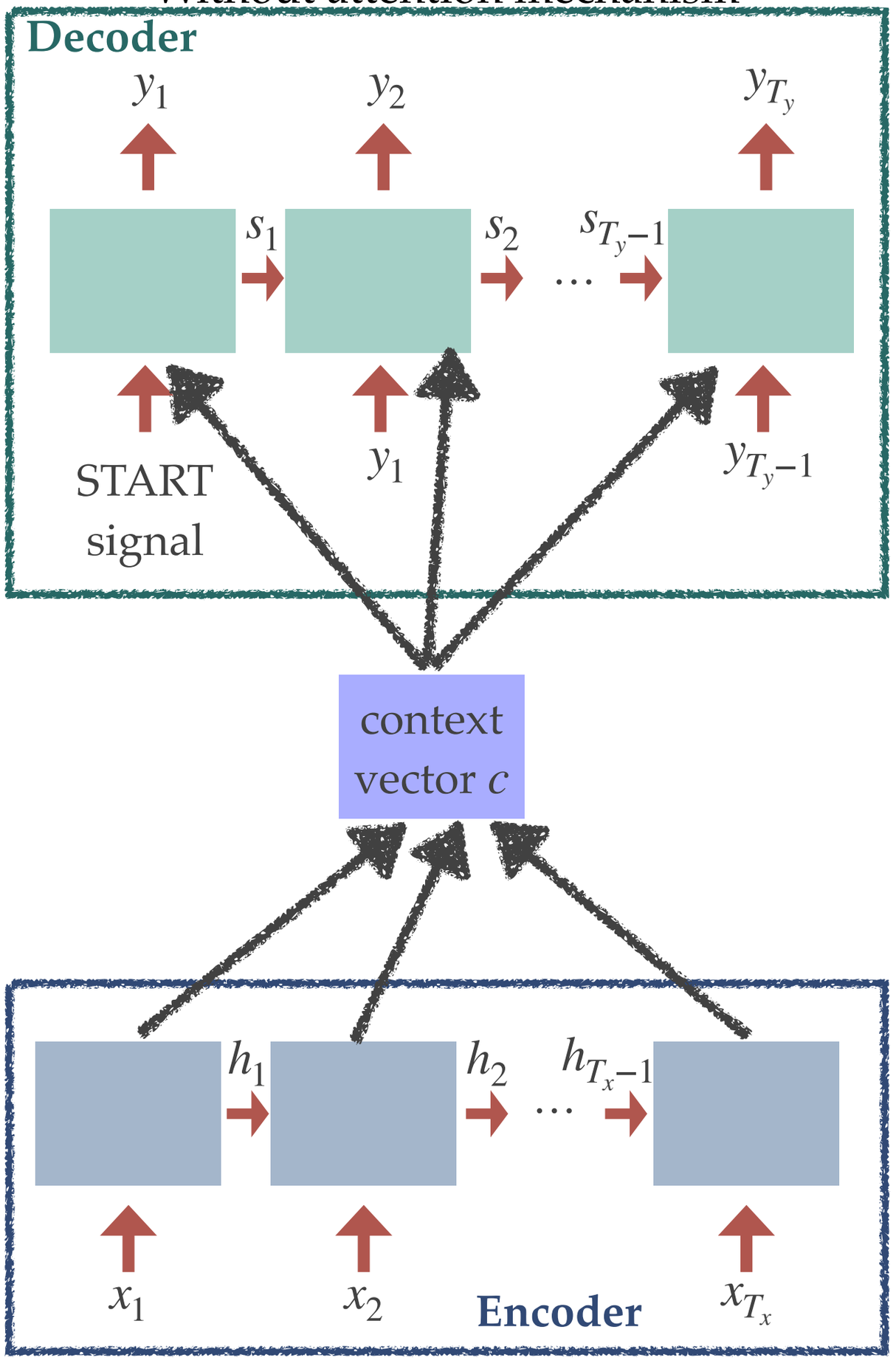
$$h_t = g_{enc}(h_{t-1}, x_t)$$

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." arXiv preprint arXiv:1409.0473 (2014).

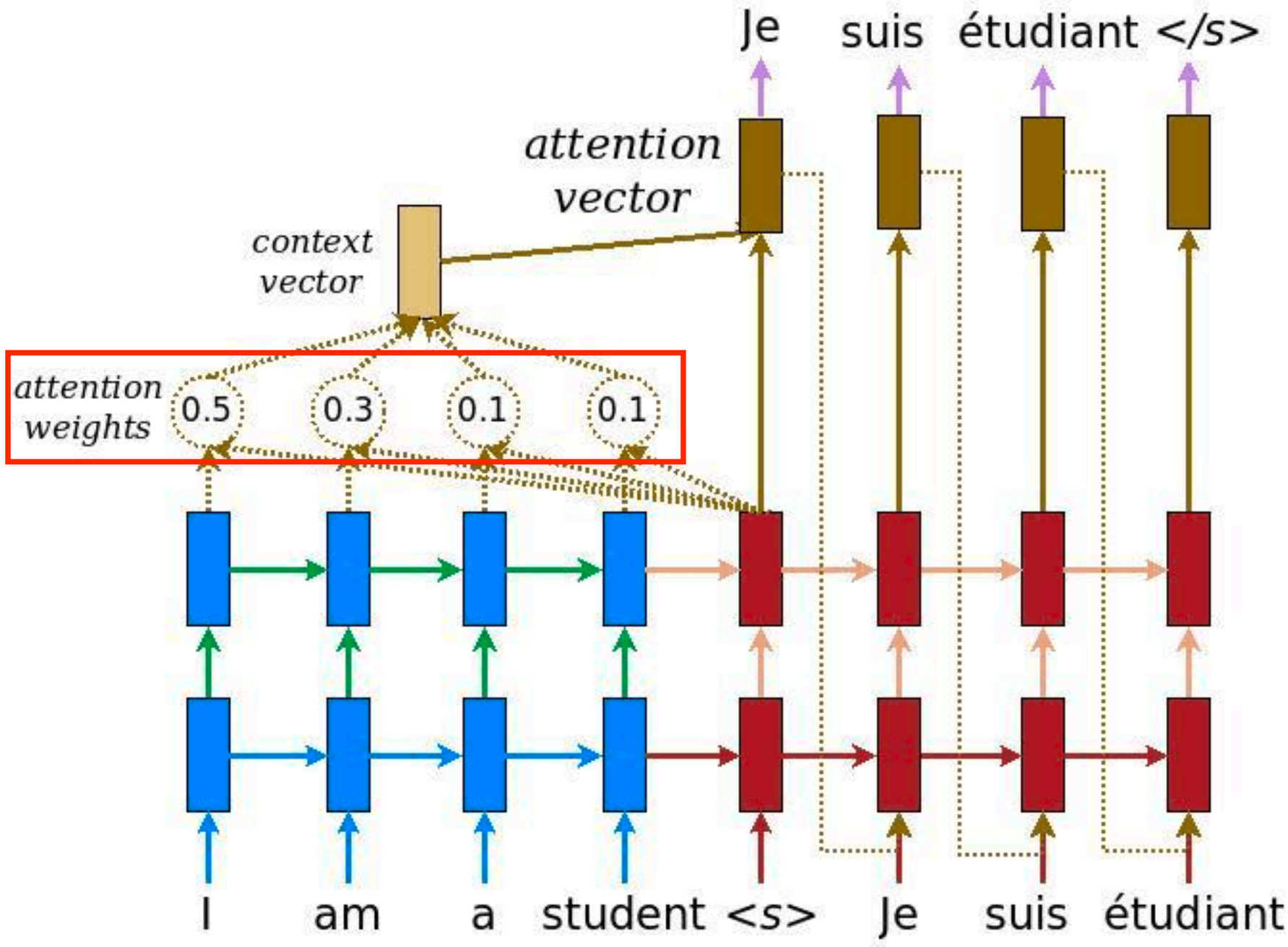
With attention mechanism

Without attention mechanism





Example: English to French with an Attention Mechanism



Tato, Ange, and Roger Nkambou. "Infusing Expert Knowledge Into a Deep Neural Network Using Attention Mechanism for Personalized Learning Environments." Frontiers in Artificial Intelligence (2022): 128.

Example: English to French with an Attention Mechanism

Example: English to French translation

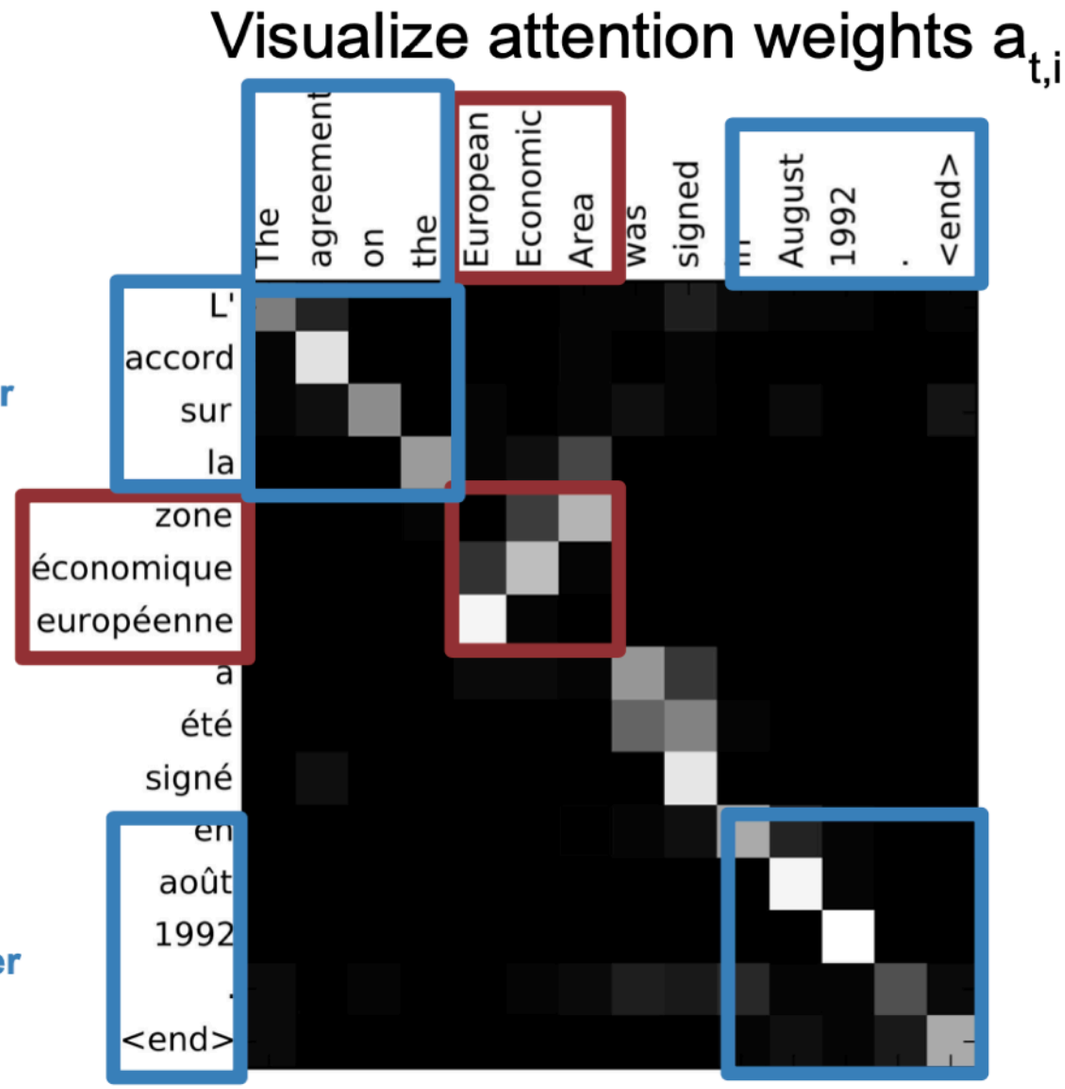
Input: "The agreement on the European Economic Area was signed in August 1992."

Output: "L'accord sur la zone économique européenne a été signé en août 1992."

Diagonal attention means words correspond in order

Attention figures out different word orders

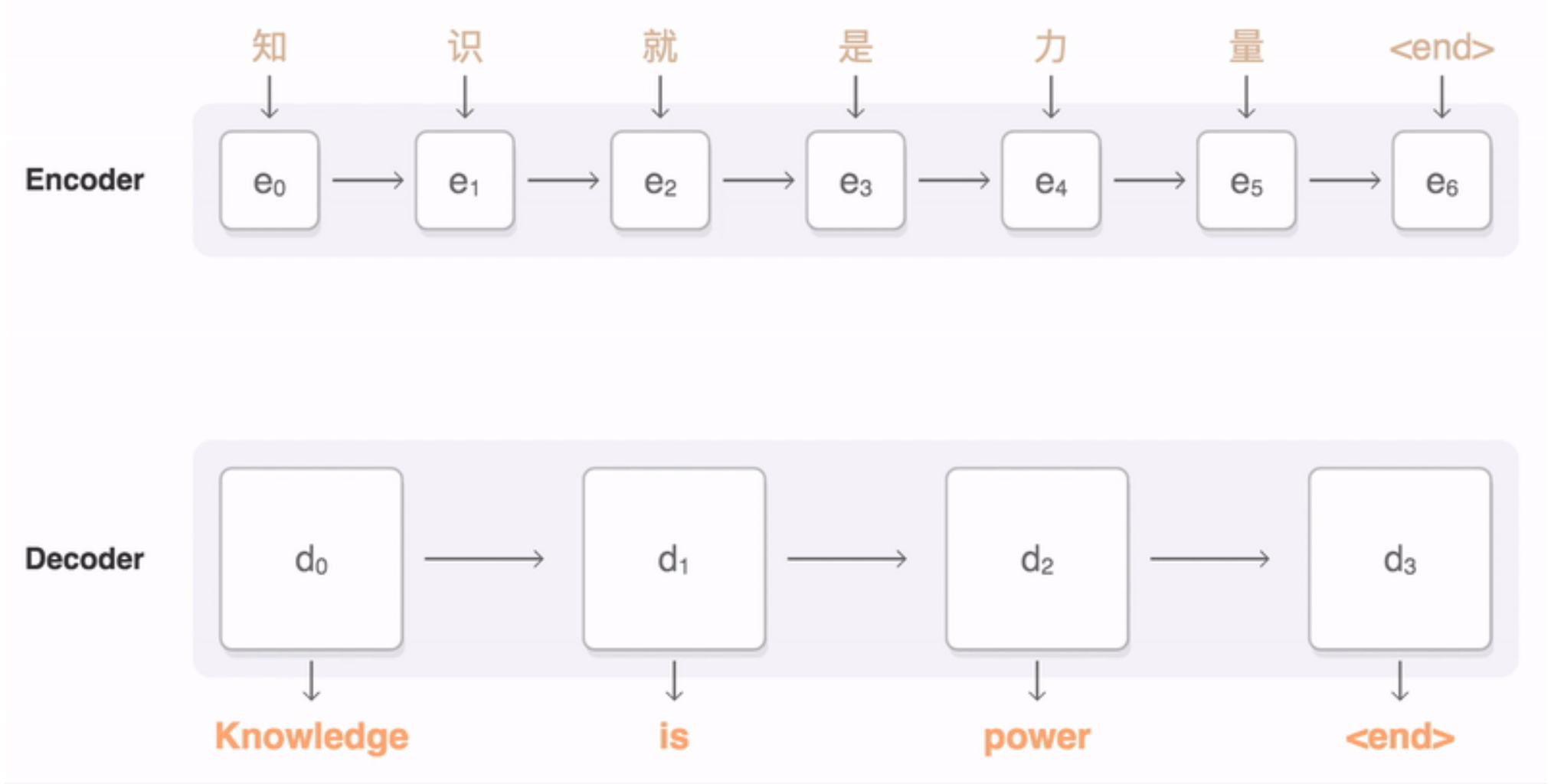
Diagonal attention means words correspond in order



Stanford CS231n (Spring 2022): Lecture 11.

Tato, Ange, and Roger Nkambou. "Infusing Expert Knowledge Into a Deep Neural Network Using Attention Mechanism for Personalized Learning Environments." Frontiers in Artificial Intelligence (2022): 128.

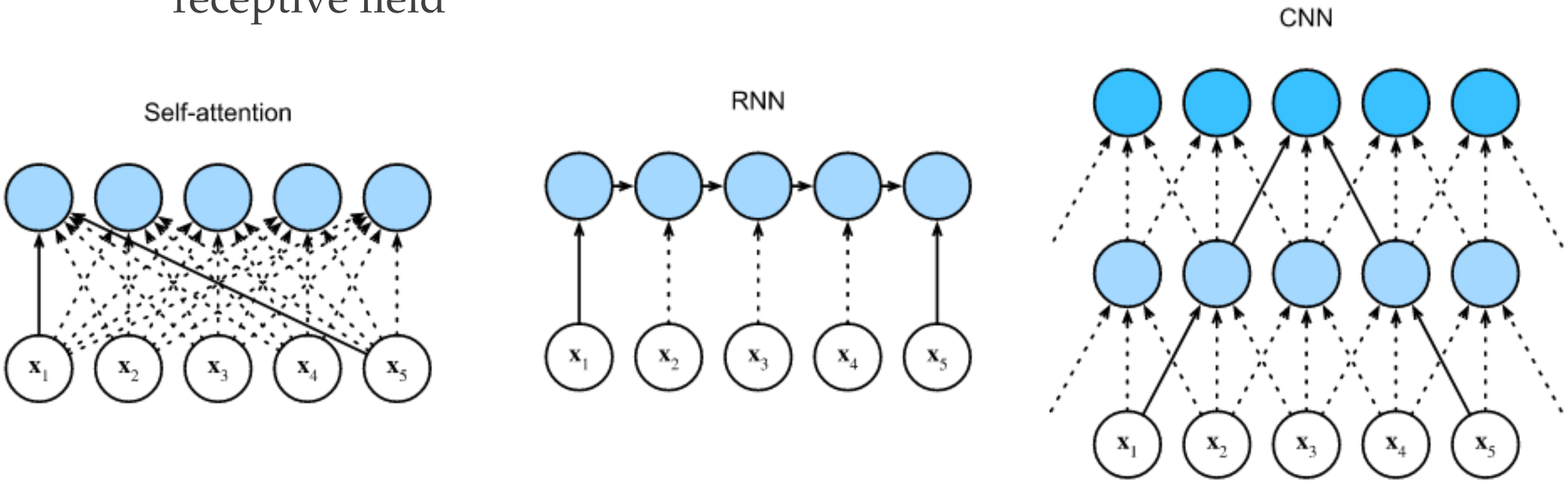
Example: Chinese to English with an Attention Mechanism



Google's tf-seq2seq

Attention vs RNN/CNN

- We have seen how the attention mechanism helps with neural machine translation
  - Allow for both short and long range dependencies
    - RNN has no explicit modeling of short-/long-range dependencies
    - CNN exploits short-range dependencies and the long-range dependencies can be captured by combining many layers sequentially to get larger receptive field

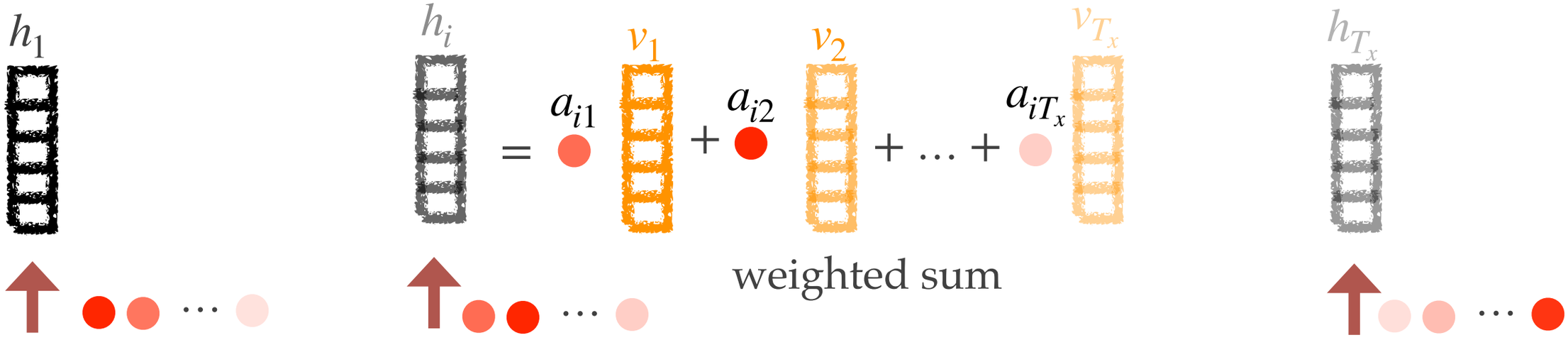


Dive Into Deep Learning: Self-Attention and Positional Encoding

Attention vs RNN/CNN

- We have seen how the attention mechanism helps with neural machine translation
  - Allow for both short and long range dependencies
    - RNN has no explicit modeling of short-/long-range dependencies
    - CNN exploits short-range dependencies and the long-range dependencies can be captured by combining many layers sequentially to get larger receptive field
  - Allow parallelization (per layer)
    - RNN relies on sequential computations (wait for previous time point)
    - CNN can also be parallelized
- We can use attention for data representation as well

Self-Attention



step 2 and step 3: compute the attention distributions and use them to calculate the outputs (weighted sum of the values): $softmax(QK^T)V$

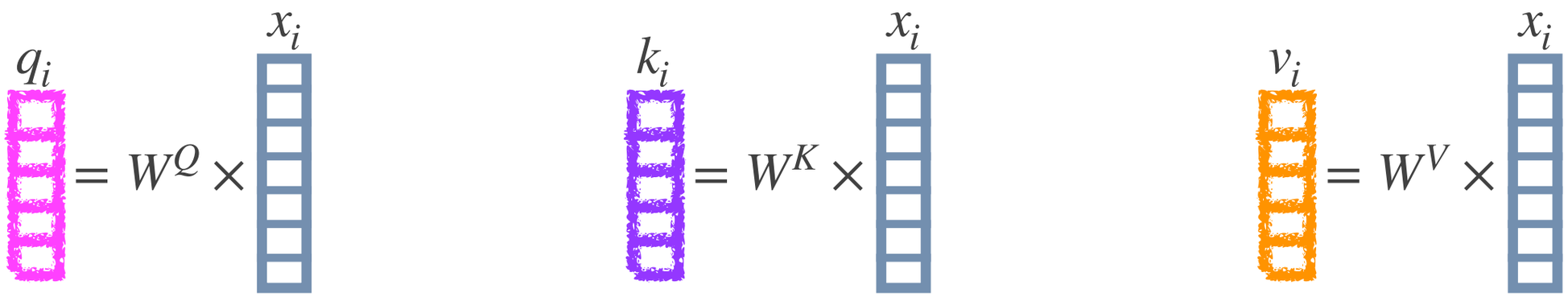
step 1: compute queries, keys, and values

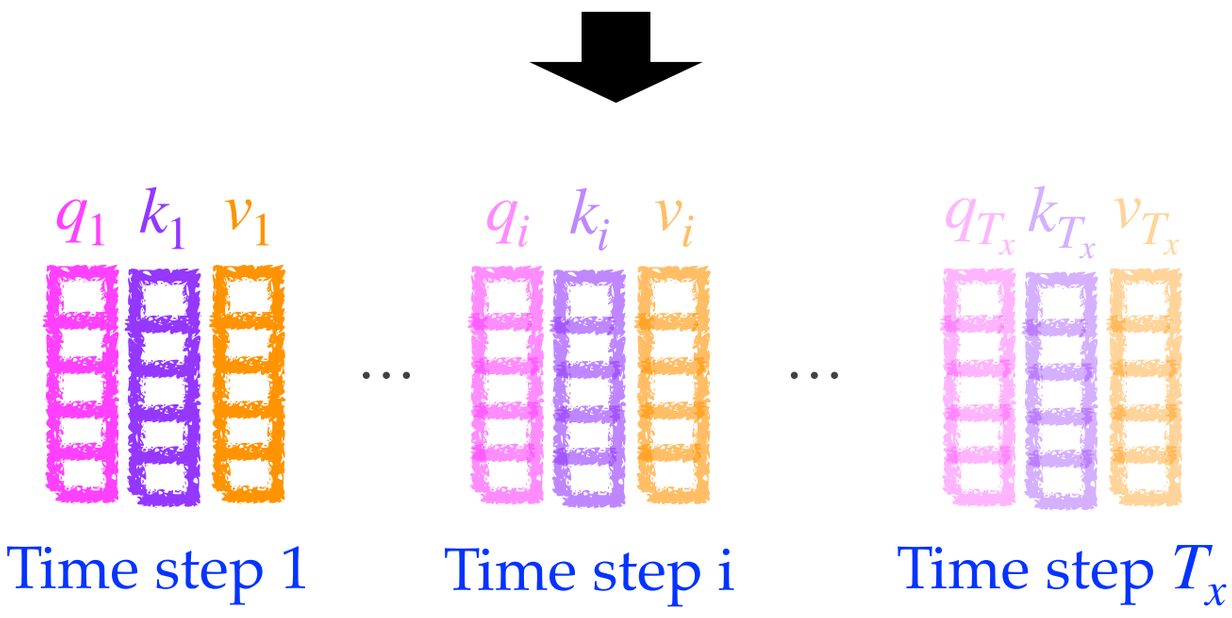
Self-Attention

For each input x_i $(i = 1,2,...,T_x)$

1. Calculate its query, key, and value

$$q_i = W^Q x_i \qquad k_i = W^K x_i \qquad v_i = W^V x_i$$

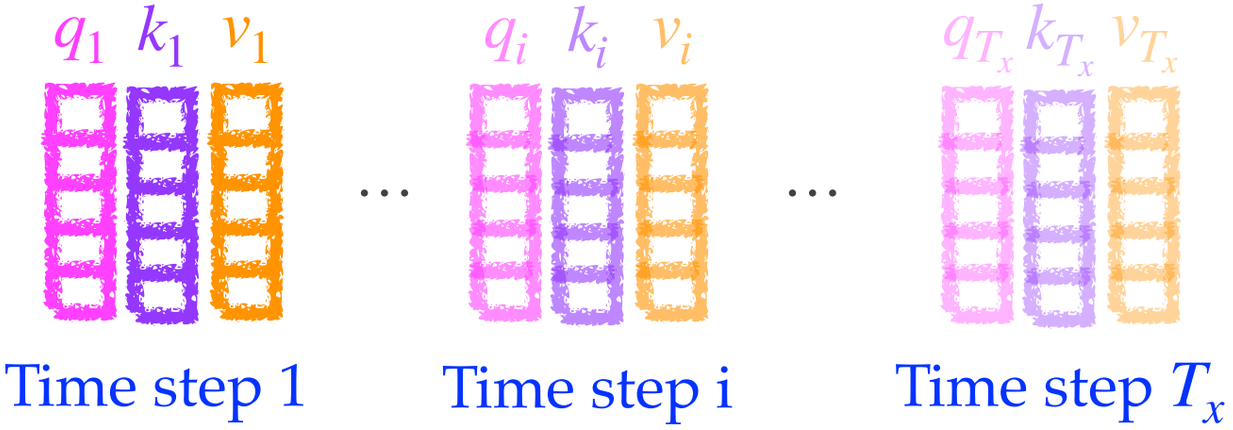




Self-Attention

For each input x_i $(i = 1,2,...,T_x)$

1. Calculate its query, key, and value



2. Calculate attention scores e's between a query and keys, and turn them into the corresponding attention distribution a using softmax

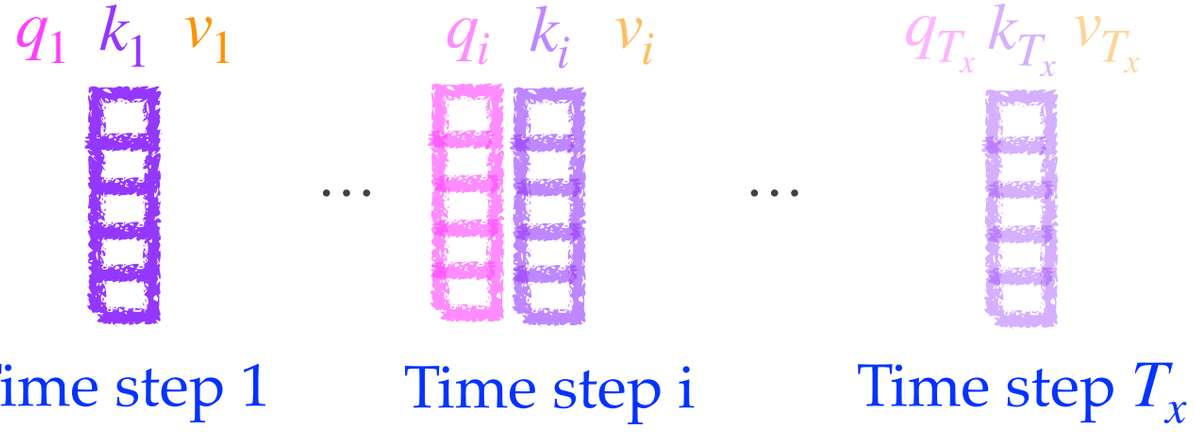
$$e_{ij} = q_i^T k_j$$

The inner product between the query and key
Just one way to compute an attention score

Self-Attention

For each input x_i $(i = 1,2,...,T_x)$

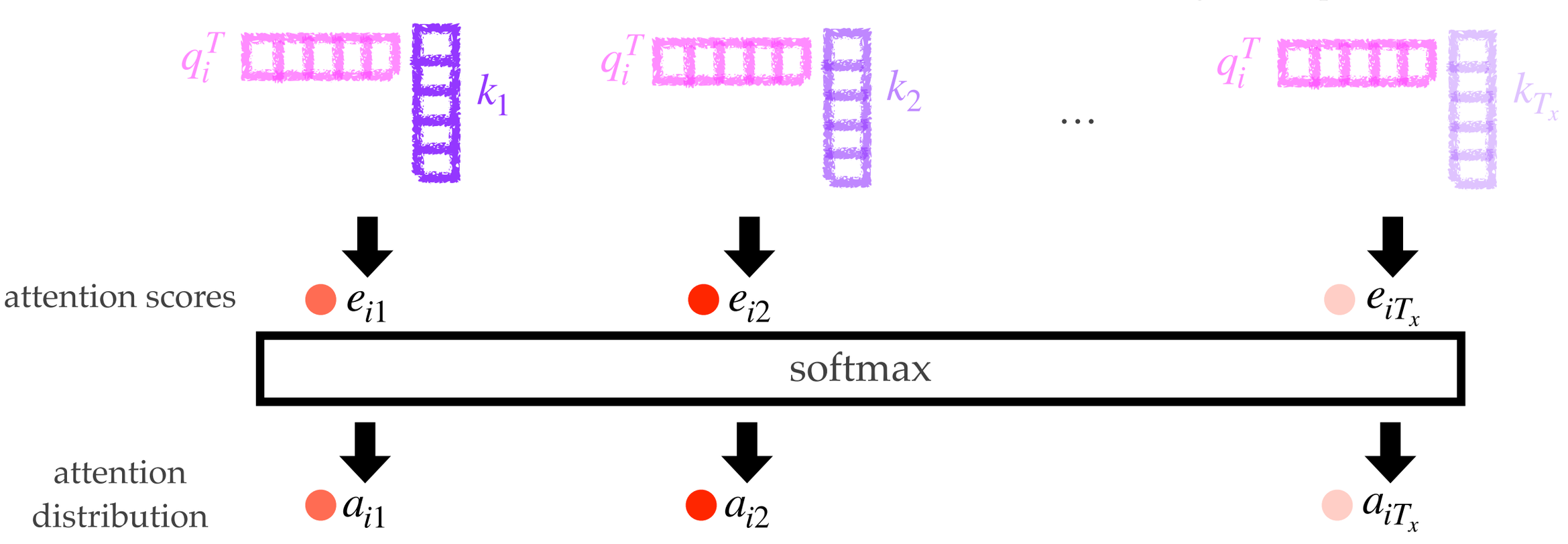
1. Calculate its query, key, and value



2. Calculate attention scores e's between a query and keys, and turn them into the corresponding attention distribution a using softmax

$$e_{ij} = q_i^T k_j$$

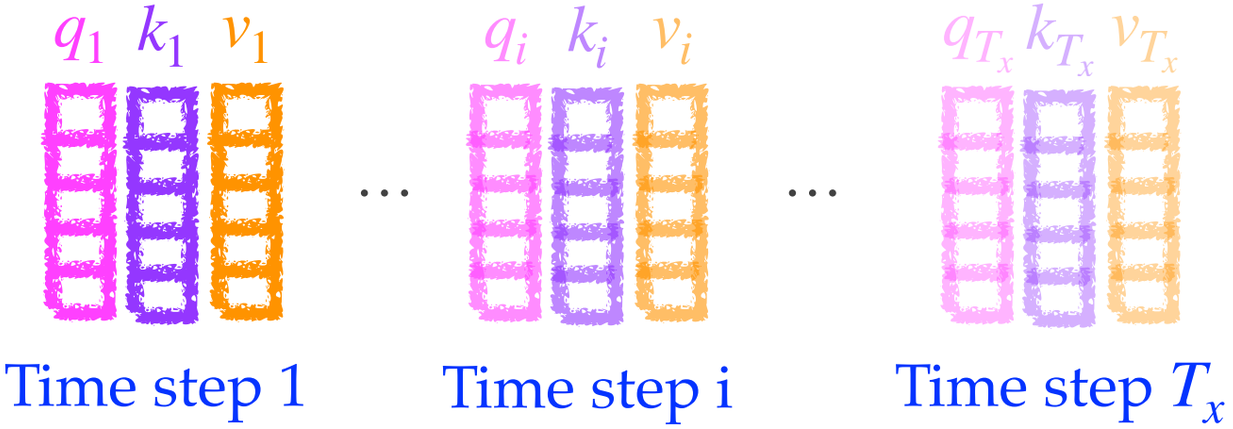
The inner product between the query and key
Just one way to compute an attention score



Self-Attention

For each input x_i $(i = 1,2,...,T_x)$

1. Calculate its query, key, and value



2. Calculate attention scores e's between a query and keys, and turn them into the corresponding attention distribution a using softmax

$$e_{ij} = q_i^T k_j$$

3. Compute the weighted sum of the values

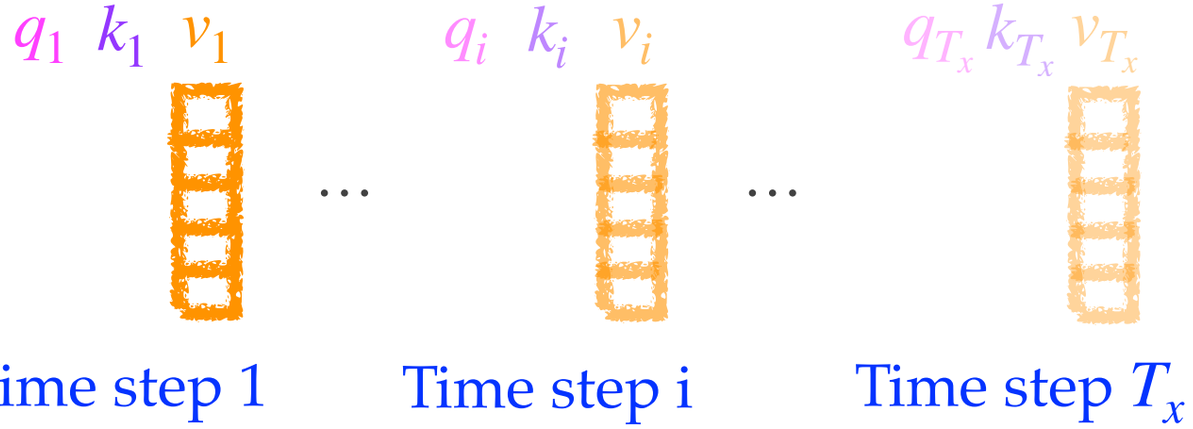
$$h_i = \sum_j a_{ij} v_j$$

attention distribution

Self-Attention

For each input x_i $(i = 1,2,...,T_x)$

1. Calculate its query, key, and value

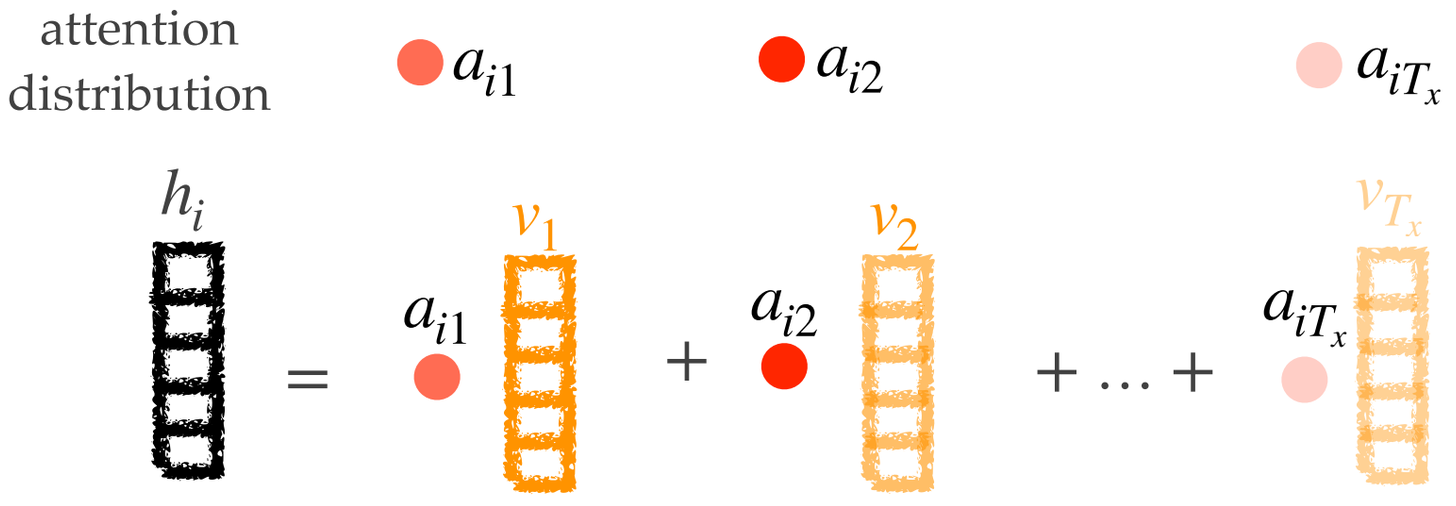


2. Calculate attention scores e's between a query and keys, and turn them into the corresponding attention distribution a using softmax

$$e_{ij} = q_i^T k_j$$

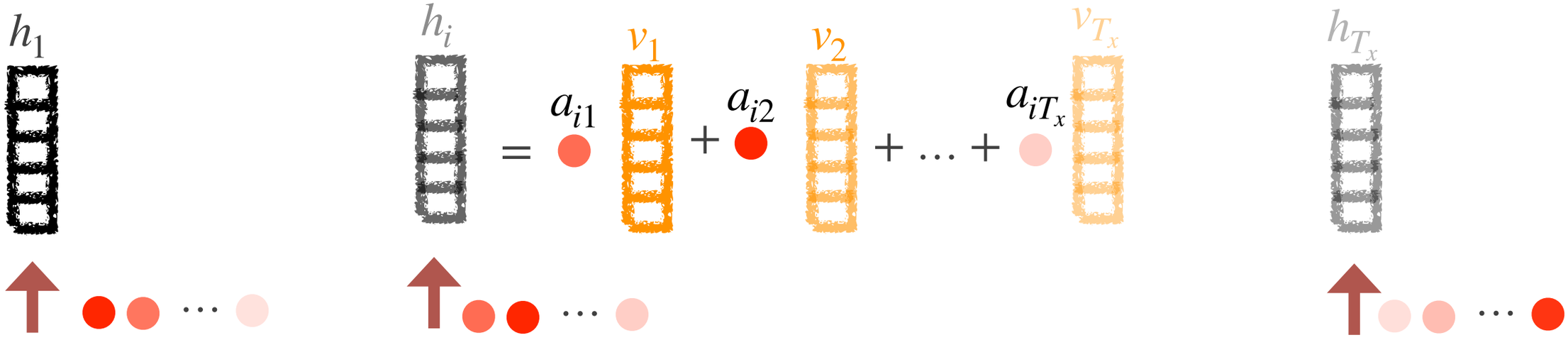
3. Compute the weighted sum of the values

$$h_i = \sum_j a_{ij} v_j$$



Self-Attention

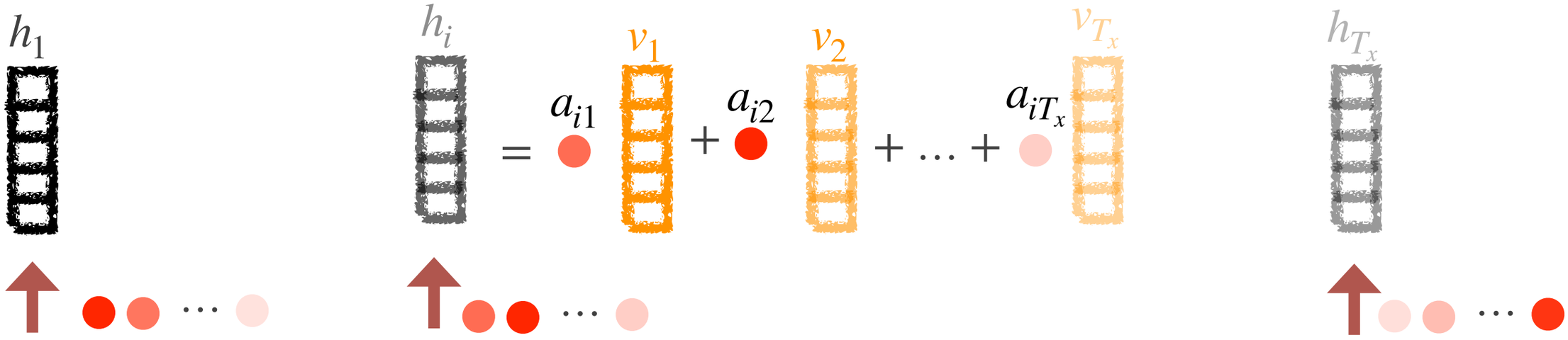
This version doesn't take into account the order of the representations



step 2 and step 3: compute the attention distributions and use them to calculate the outputs (weighted sum of the values): $softmax(QK^T)V$

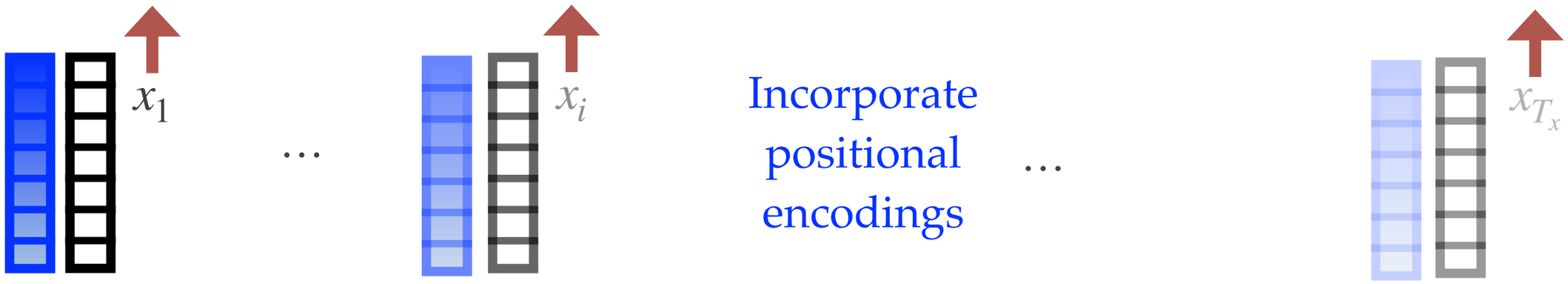
step 1: compute queries, keys, and values

Self-Attention

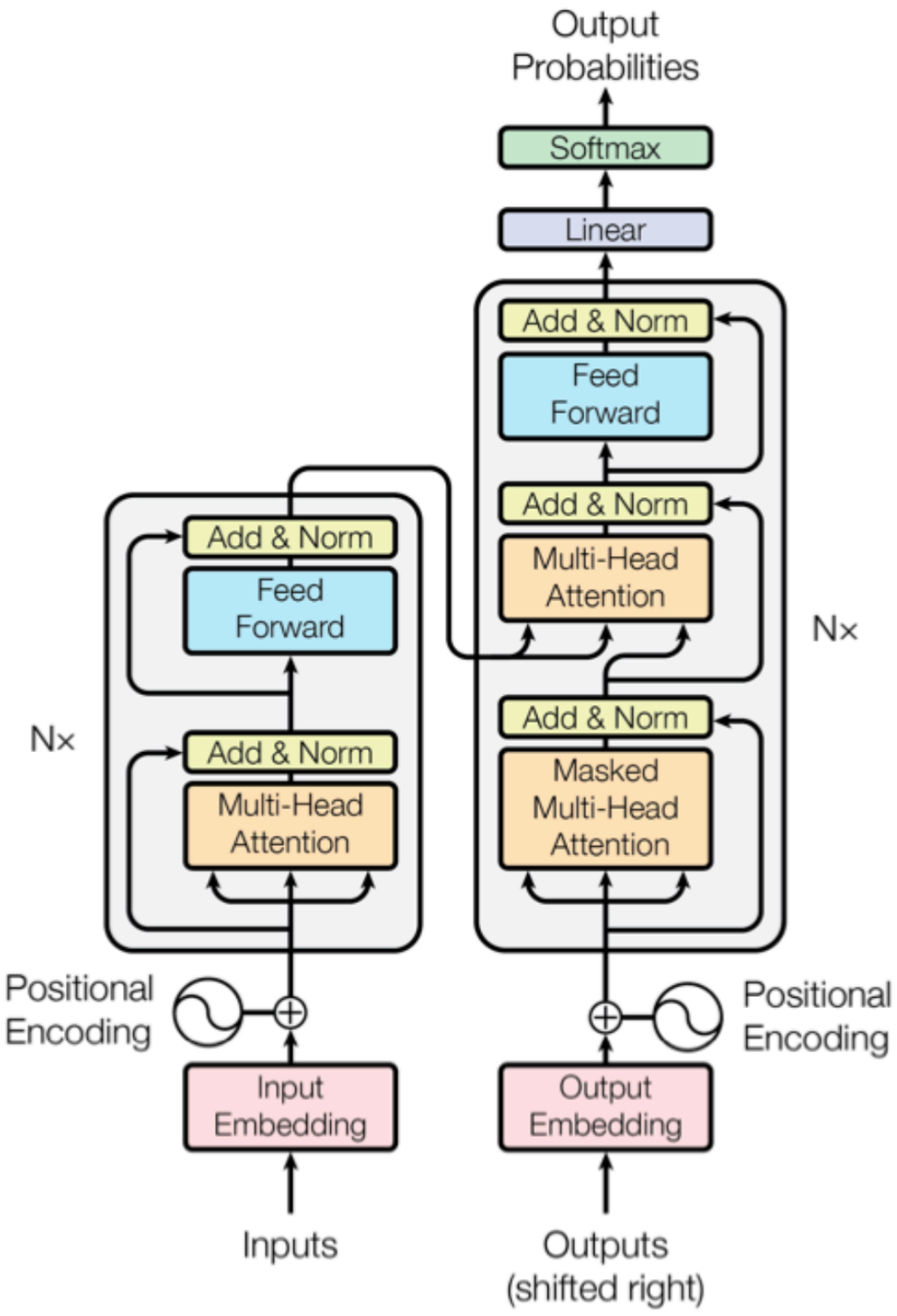


step 2 and step 3: compute the attention distributions and use them to calculate the outputs (weighted sum of the values): $softmax(QK^T)V$

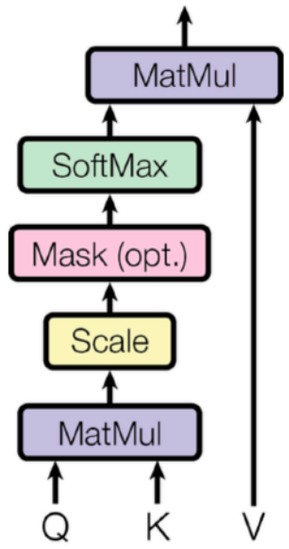
step 1: compute queries, keys, and values



Transformer



Scaled Dot-Product Attention

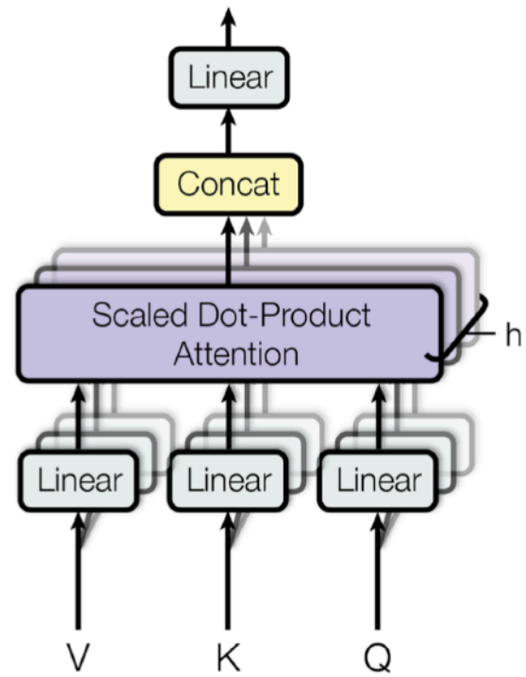


$$softmax(QK^T)V$$

$$\Downarrow$$

$$softmax(QK^T/\sqrt{d_k})V$$

Multi-Head Attention



Each attention head performs attention independently

Feed Forward Add more expressive power

Add & Norm Add residual connections & Normalize the data within each layer

Vaswani, Ashish, et al. "Attention is all you need." Advances in neural information processing systems 30 (2017).

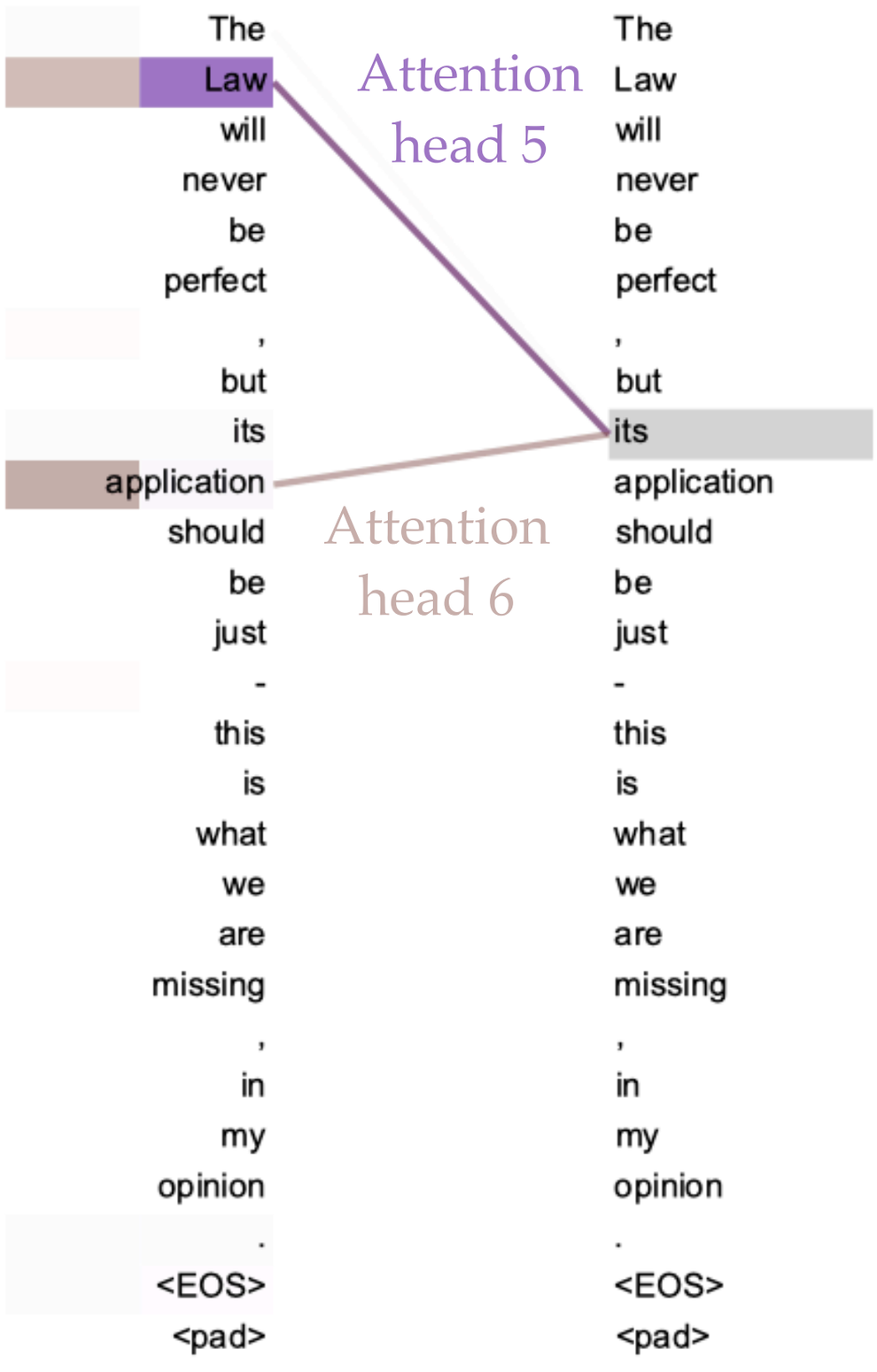
Transformer

Table 2: The Transformer achieves better BLEU scores than previous state-of-the-art models on the English-to-German and English-to-French newstest2014 tests at a fraction of the training cost.

Model	BLEU		Training Cost (FLOPs)	
	EN-DE	EN-FR	EN-DE	EN-FR
ByteNet [18]	23.75			
Deep-Att + PosUnk [39]		39.2		$1.0 \cdot 10^{20}$
GNMT + RL [38]	24.6	39.92	$2.3 \cdot 10^{19}$	$1.4 \cdot 10^{20}$
ConvS2S [9]	25.16	40.46	$9.6 \cdot 10^{18}$	$1.5 \cdot 10^{20}$
MoE [32]	26.03	40.56	$2.0 \cdot 10^{19}$	$1.2 \cdot 10^{20}$
Deep-Att + PosUnk Ensemble [39]		40.4		$8.0 \cdot 10^{20}$
GNMT + RL Ensemble [38]	26.30	41.16	$1.8 \cdot 10^{20}$	$1.1 \cdot 10^{21}$
ConvS2S Ensemble [9]	26.36	**41.29**	$7.7 \cdot 10^{19}$	$1.2 \cdot 10^{21}$
Transformer (base model)	27.3	38.1	**$3.3 \cdot 10^{18}$**	
Transformer (big)	**28.4**	**41.8**	$2.3 \cdot 10^{19}$	

Vaswani, Ashish, et al. "Attention is all you need." Advances in neural information processing systems 30 (2017).

Transformer

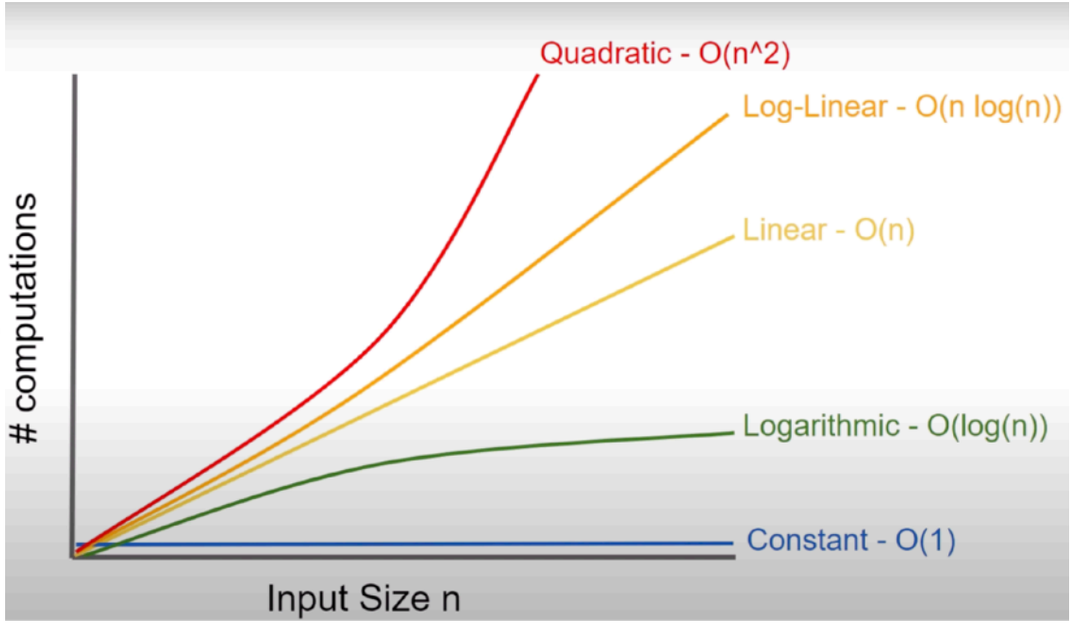


Vaswani, Ashish, et al. "Attention is all you need." Advances in neural information processing systems 30 (2017).

Transformer

- The computation of self-attention grows quadratically with the sequence length
  - For RNNs, the computation grows linearly
  - Recent models can alleviate the problem
    - Set Transformer (2019)
    - Linformer (2020)
    - Big Bird (2020)
    - Perceiver (2021)/Perceiver IO (2021)

Algorithms Explained: Computational Complexity



- Stanford CS224n (Winter 2022): Lecture 9.
- Vaswani, Ashish, et al. "Attention is all you need." Advances in neural information processing systems 30 (2017).
- Lee, Juho, et al. "Set transformer: A framework for attention-based permutation-invariant neural networks." International conference on machine learning. PMLR, 2019.
- Wang, Sinong, et al. "Linformer: Self-attention with linear complexity." arXiv preprint arXiv:2006.04768 (2020).
- Zaheer, Manzil, et al. "Big bird: Transformers for longer sequences." Advances in Neural Information Processing Systems 33 (2020): 17283-17297.
- Jaegle, Andrew, et al. "Perceiver io: A general architecture for structured inputs & outputs." arXiv preprint arXiv:2107.14795 (2021).

SYSTEM PROMPT (HUMAN-WRITTEN)	*In a shocking finding, scientist discovered a herd of unicorns living in a remote, previously unexplored valley, in the Andes Mountains. Even more surprising to the researchers was the fact that the unicorns spoke perfect English.*
MODEL COMPLETION (MACHINE-WRITTEN, 10 TRIES)	The scientist named the population, after their distinctive horn, Ovid's Unicorn. These four-horned, silver-white unicorns were previously unknown to science. Now, after almost two centuries, the mystery of what sparked this odd phenomenon is finally solved. Dr. Jorge Pérez, an evolutionary biologist from the University of La Paz, and several companions, were exploring the Andes Mountains when they found a small valley, with no other animals or humans. Pérez noticed that the valley had what appeared to be a natural fountain, surrounded by two peaks of rock and silver snow. Pérez and the others then ventured further into the valley. "By the time we reached the top of one peak, the water looked blue, with some crystals on top," said Pérez. Pérez and his friends were astonished to see the unicorn herd. These creatures could be seen from the air without having to move too much to see them – they were so close they could touch their horns. While examining these bizarre creatures the scientists discovered that the creatures also spoke some fairly regular English. Pérez stated, "We can see, for example, that they have a common 'language,' something like a dialect or dialectic."

GPT-2: A large transformer-based language model with 1.5 billion parameters