# The mechanism of attention and transformers

Skoltech

# Machine translation: Recap

Skoltech

# Evaluations of Machine Translation: BLEU

*"the closer a machine translation is to a professional human translation, the better it is"*

- **Precision**: share of words from $\hat{y}$ that appear in $y$

| | | | | | | |
|---|---|---|---|---|---|---|
| True $y$ | the | cat | is | on | the | mat |
| Candidate $\hat{y}$ | the | the | the | the | the | the |

- Unigram precision is 6/6 = 1
- Limit with number of words from references: there are 2 "the" in $y$
- BLEU modified unigram precision is 2/6 = 0.33

Skoltech

Source: Papineni, K.; Roukos, S.; Ward, T.; Zhu, W. J. 2002. *BLEU: a method for automatic evaluation of machine translation*. ACL-2002: 40th Annual meeting of the Association for Computational Linguistics

# BLEU evaluation examples

| | | | | | | |
|---|---|---|---|---|---|---|
| True $y$ | the | cat | is | on | the | mat |
| Candidate $\hat{y}$ | the | the | the | the | the | the |

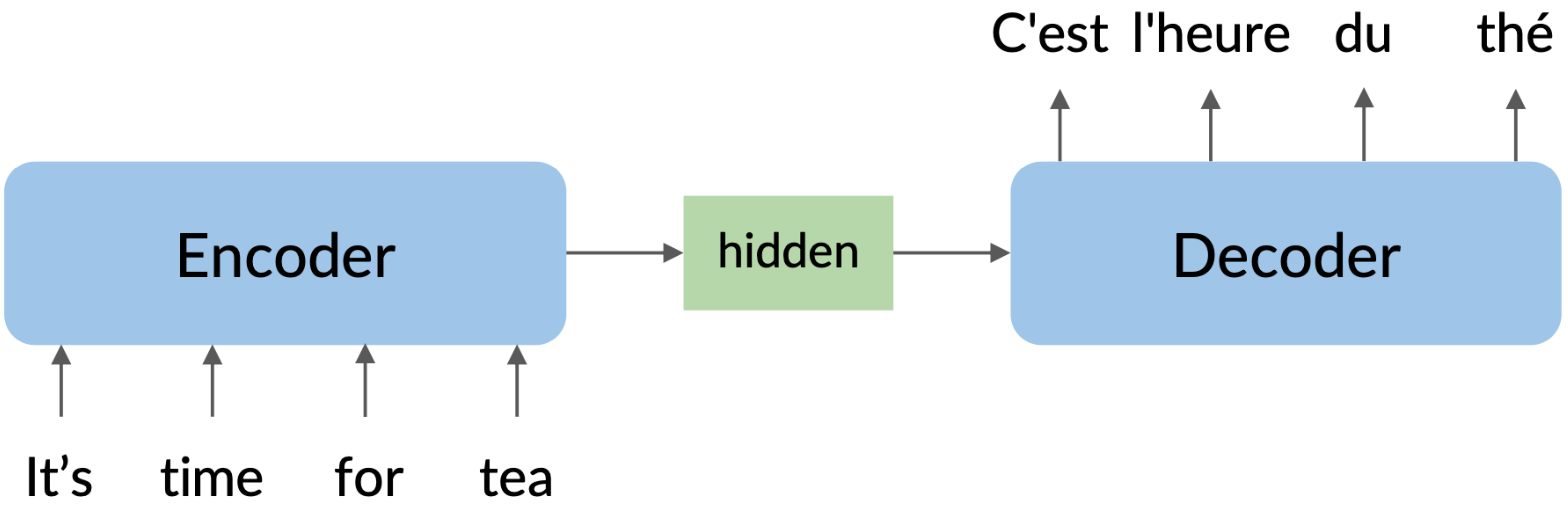
- Unigram precision is 6/6 = 1
- Bigram precision is 1/5 = 0.2
- BLEU unigram score 2/6
- BLEU bigram score is 1/5

- *Good BLEU scores is about 0.7*

Skoltech

Source: Papineni, K.; Roukos, S.; Ward, T.; Zhu, W. J. 2002. *BLEU: a method for automatic evaluation of machine translation*. ACL-2002: 40th Annual meeting of the Association for Computational Linguistics

# seq2seq model




Source: DeepLearning.AI

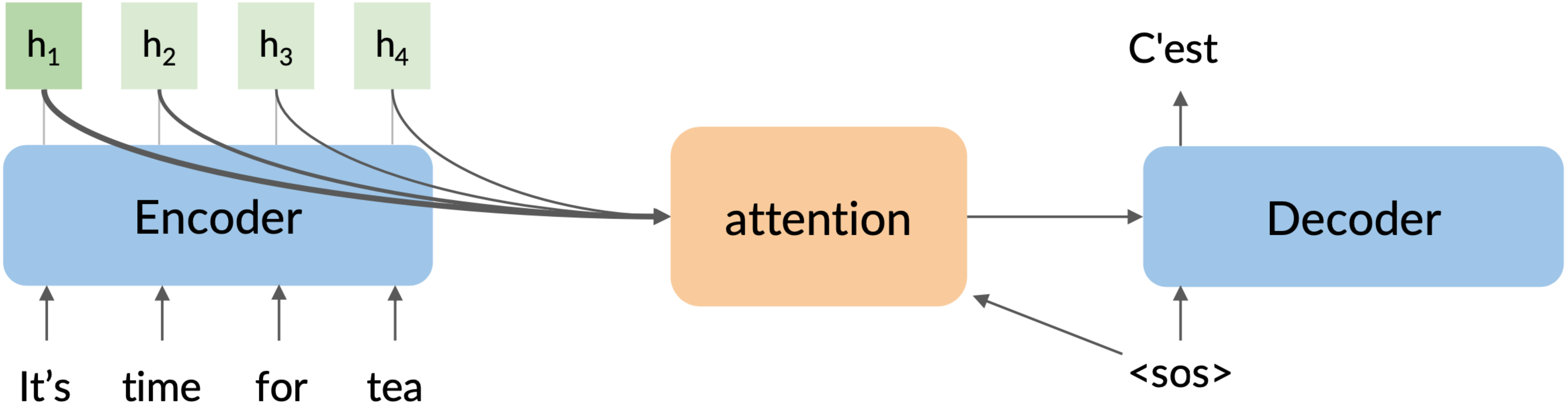
Skoltech

Main issue: bottleneck, the network will have time to forget everything

# Attention model

Skoltech

# Attention focusing

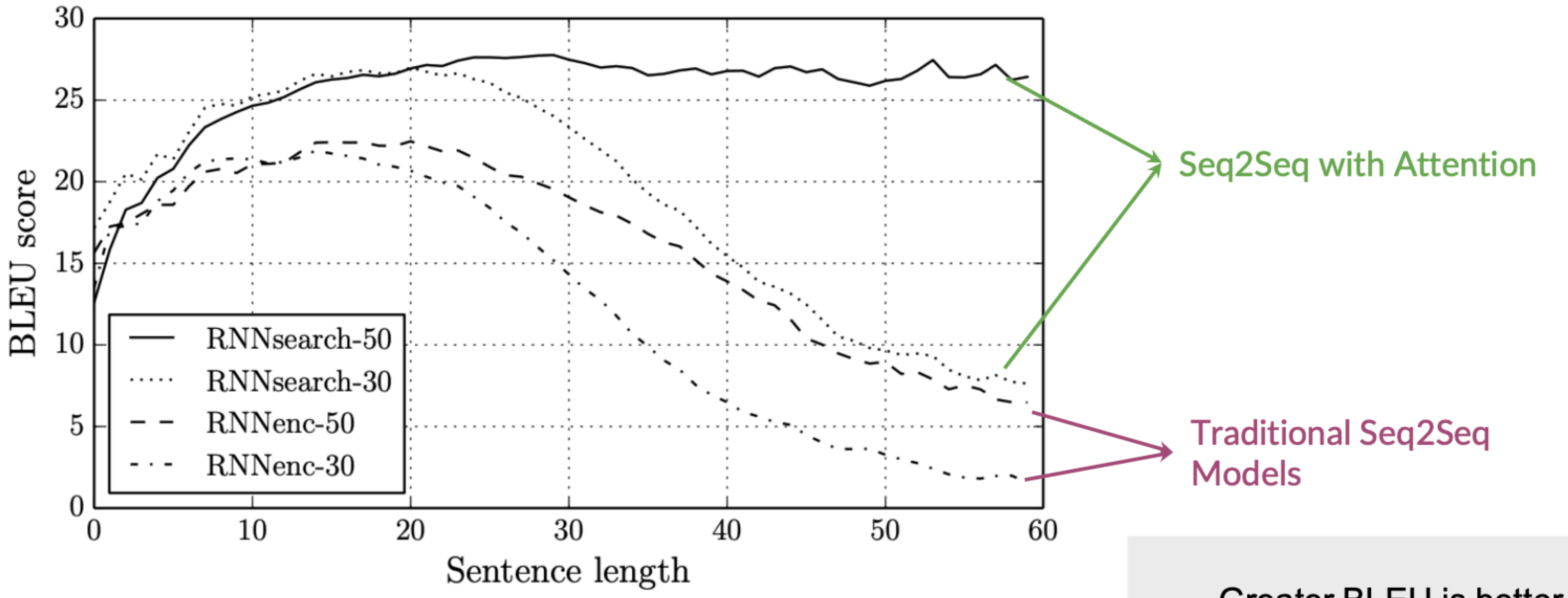



Source: DeepLearning.AI

Skoltech

Solution: focus attention in the right place

# New state of the art: attention is all we need [1]

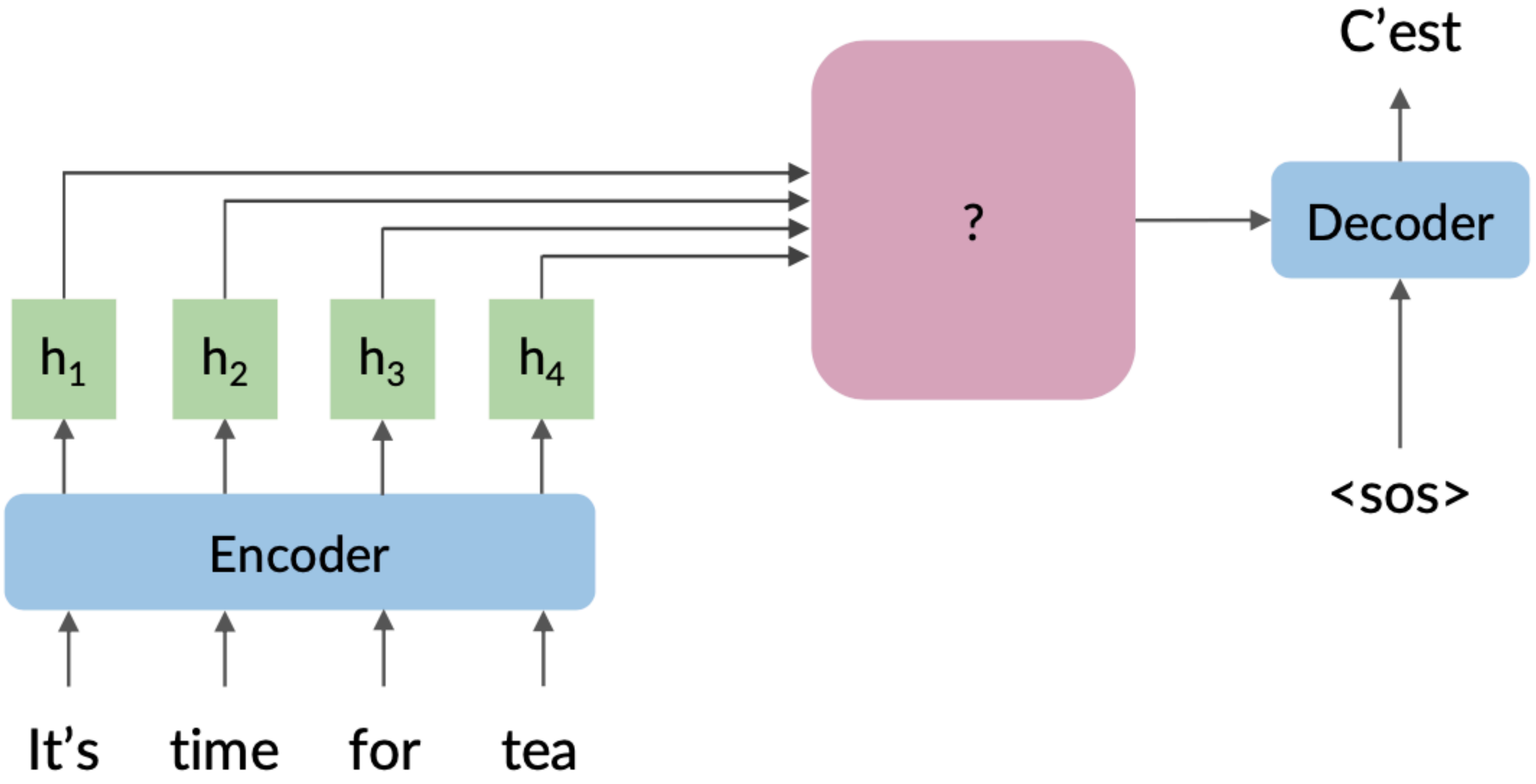



Greater BLEU is better

Source: DeepLearning.AI

Skoltech

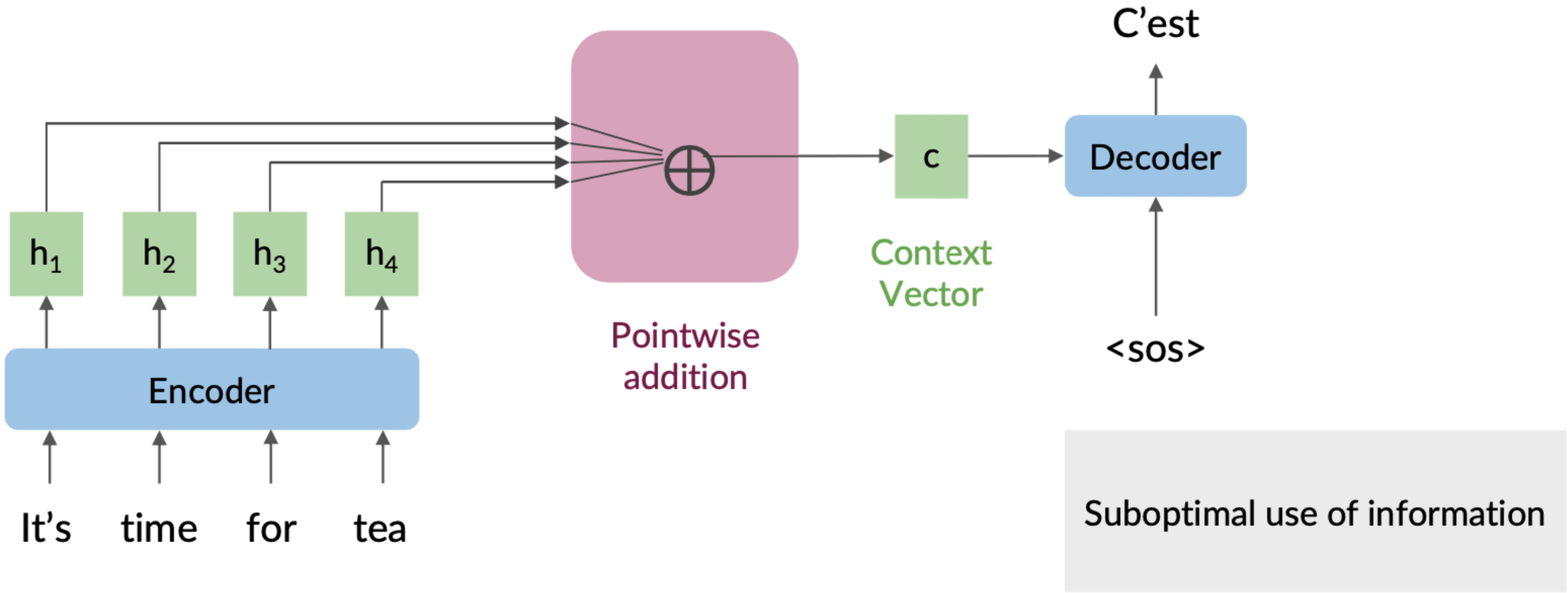
# How to use all the hidden states?




Source: DeepLearning.AI

Skoltech

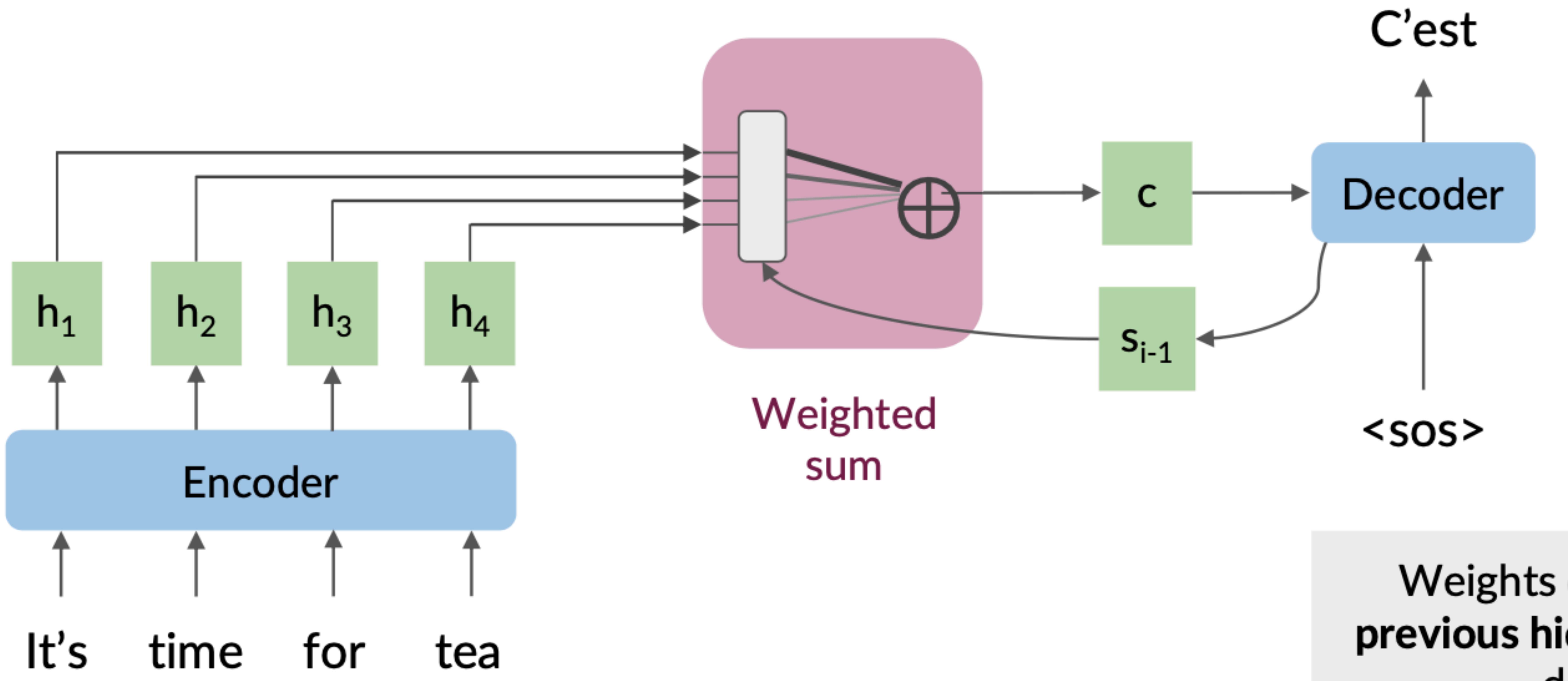
# How to use all the hidden states?




Source: DeepLearning.AI

Skoltech

# How to use all the hidden states?




Weights depend on the **previous hidden state** in the decoder

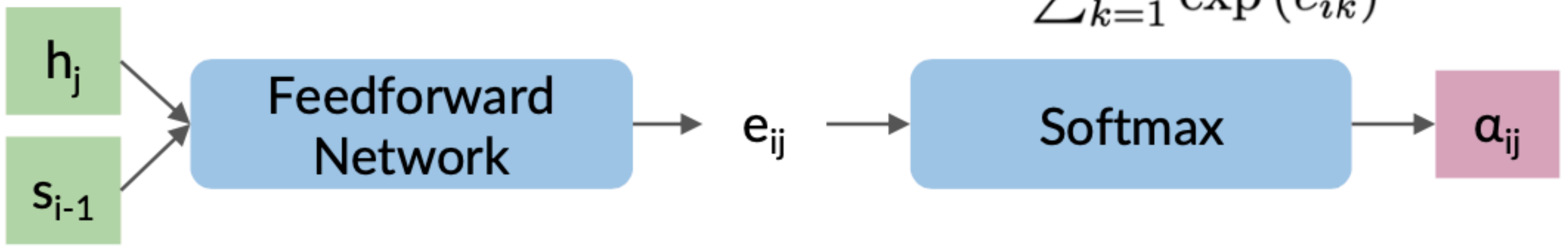
Source: DeepLearning.AI

Skoltech

# The attention layer in more depth

$$e_{ij} = a(s_{i-1}, h_j)$$

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})}$$




Weights used for the sum of hidden states

Learnable parameters

$$c_i = \sum_{j=1}^{T_x} \boxed{\alpha_{ij}}\boxed{h_j}$$

$$\alpha_{i1}h_1 + \alpha_{i2}h_2 + \alpha_{i3}h_3 + \cdots + \alpha_{iM}h_M \longrightarrow c_i$$

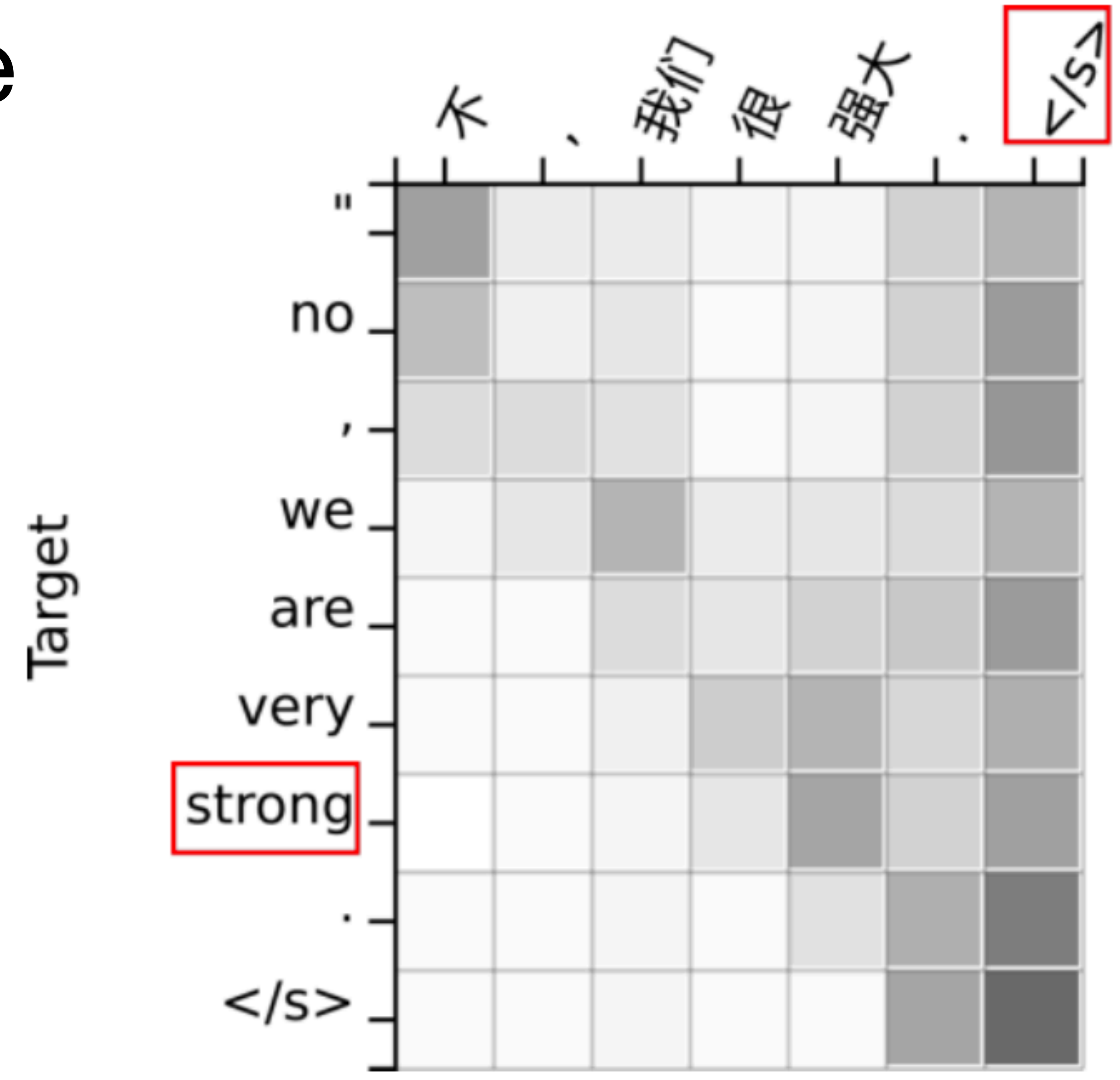
Context Vector is an expected value

Source: DeepLearning.AI

Skoltech

# Attention is just great

- Solves the bottleneck problem: all encoder tokens are connected to all decoder tokens
- No more vanishing gradients : all to all connection
- Provides some interpretability: see alignment figure




- Similar to RNN seq2seq, but greater!

# Transformers

Skoltech

# Queries, Keys, and Values

| Query | Key | Value |
|---|---|---|
| l'heure | It's | [0.5, 0.2, -1.2, ..., ] |
| | time | [0.2, -0.7, 0.9, ..., ] |
| | for | [1.3, 0.3, 0.8, ..., ] |
| | tea | [-0.4, 0.6, -1.1, ..., ] |

Similarity is used in for weighted sum

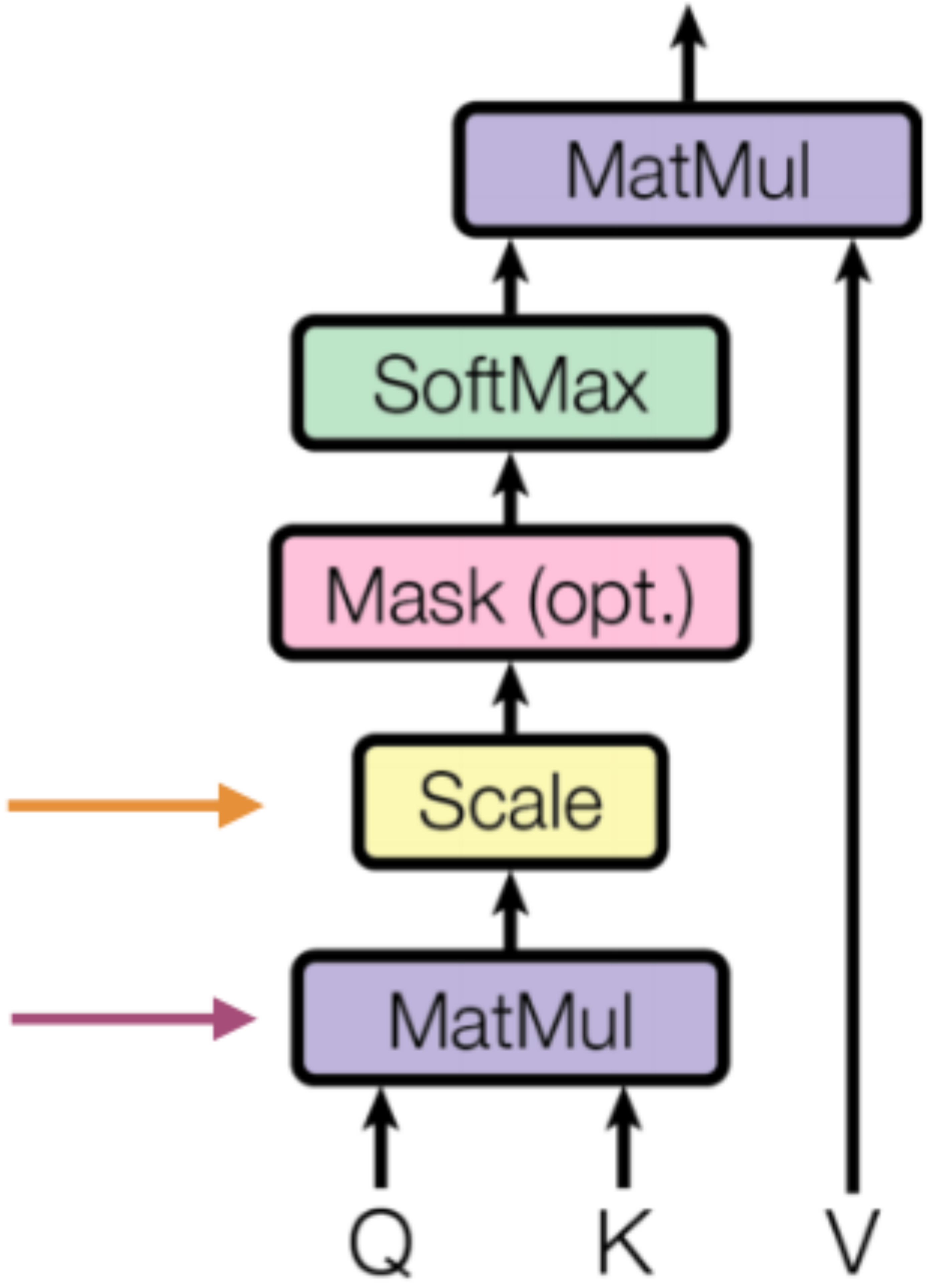
Source: DeepLearning.AI

# Scaled dot-product attention

Similarity Between
Q and K

$$\mathrm{Attention}(Q, K, V) = \mathrm{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

Scale using the root of
the key vector size




Source: Vaswani et al., 2017

- We produce queries, keys, and values using initial word embeddings:

$$\mathrm{Q} = \mathrm{XW^Q}, \dim(\mathrm{W^Q}) = \mathrm{d_x} \times \mathrm{d}_q$$

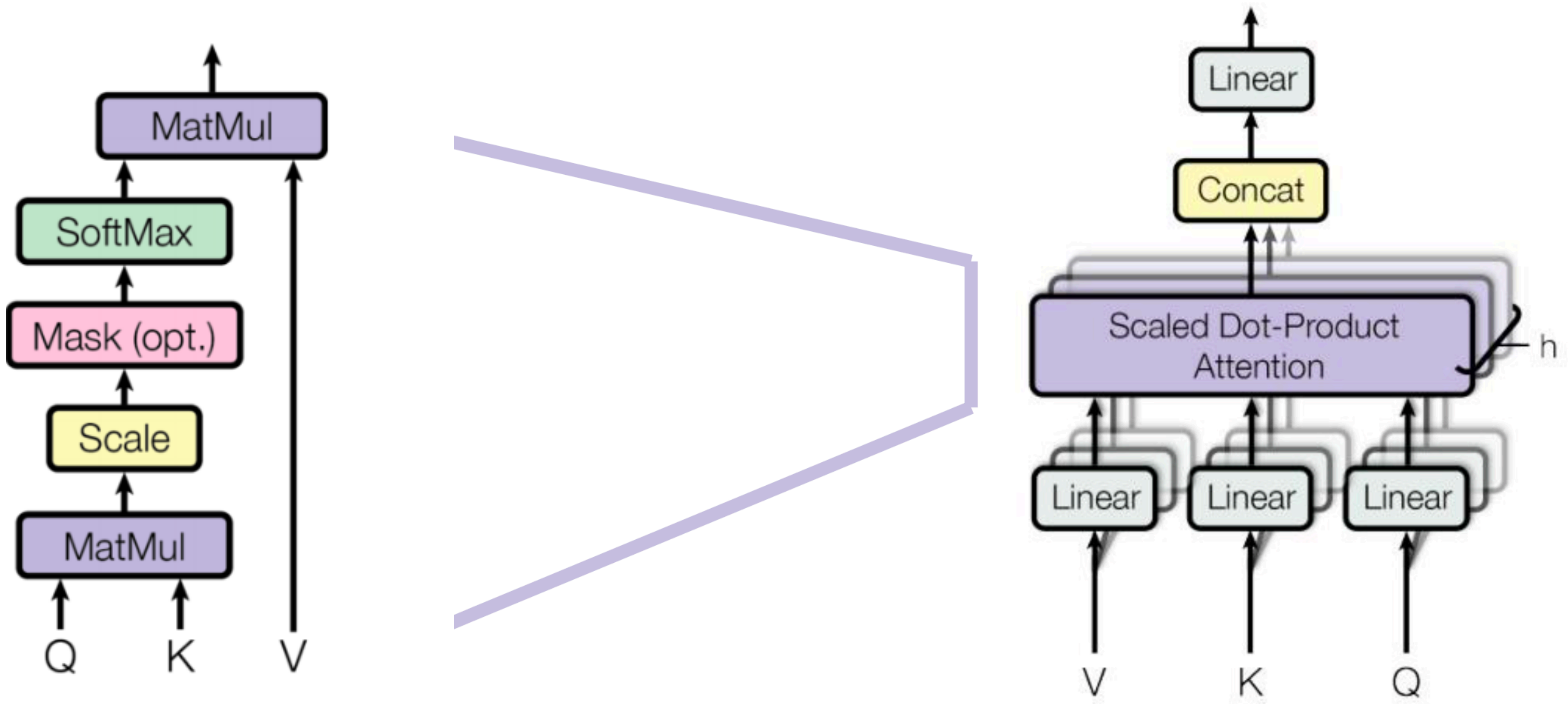
$$\mathrm{K} = \mathrm{XW}^K, \dim(\mathrm{W^Q}) = \mathrm{d_x} \times \mathrm{d}_k$$

$$\mathrm{V} = \mathrm{XW}^V, \dim(\mathrm{W^Q}) = \mathrm{d_x} \times \mathrm{d}_v$$

# Multi-Head attention

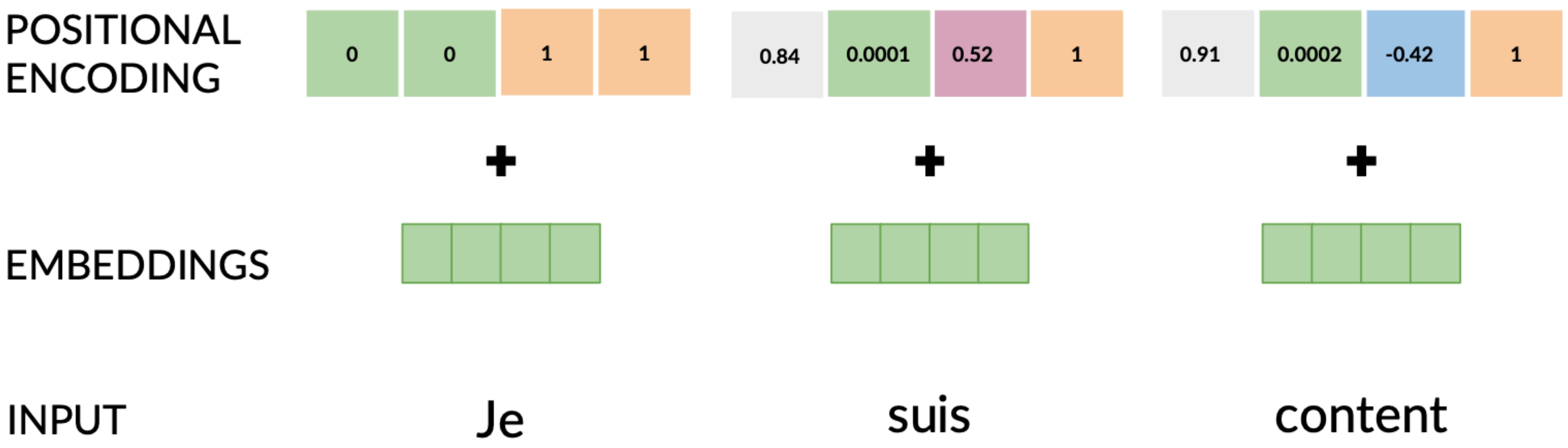
$$\mathrm{MultiHead}(Q, K, V) = \mathrm{Concat}(\mathrm{head_1}, \ldots, \mathrm{head_h})W^O$$

$$\text{where } \mathrm{head_i} = \mathrm{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

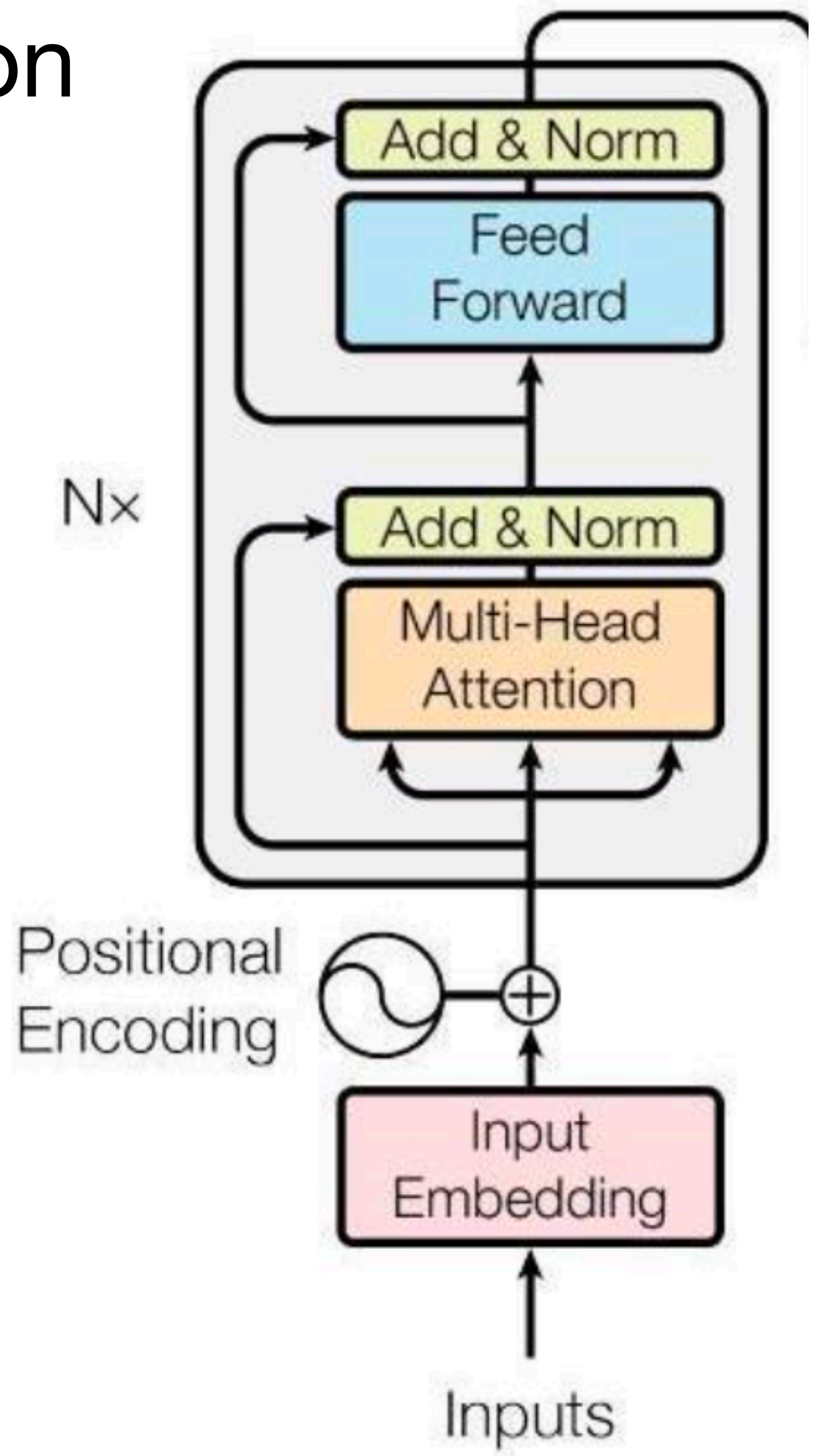


# Position encoding

- In addition to usual embeddings of inputs we use position encoding to capture position
- They are not one-hot vectors, as we want to handle various-length sequences

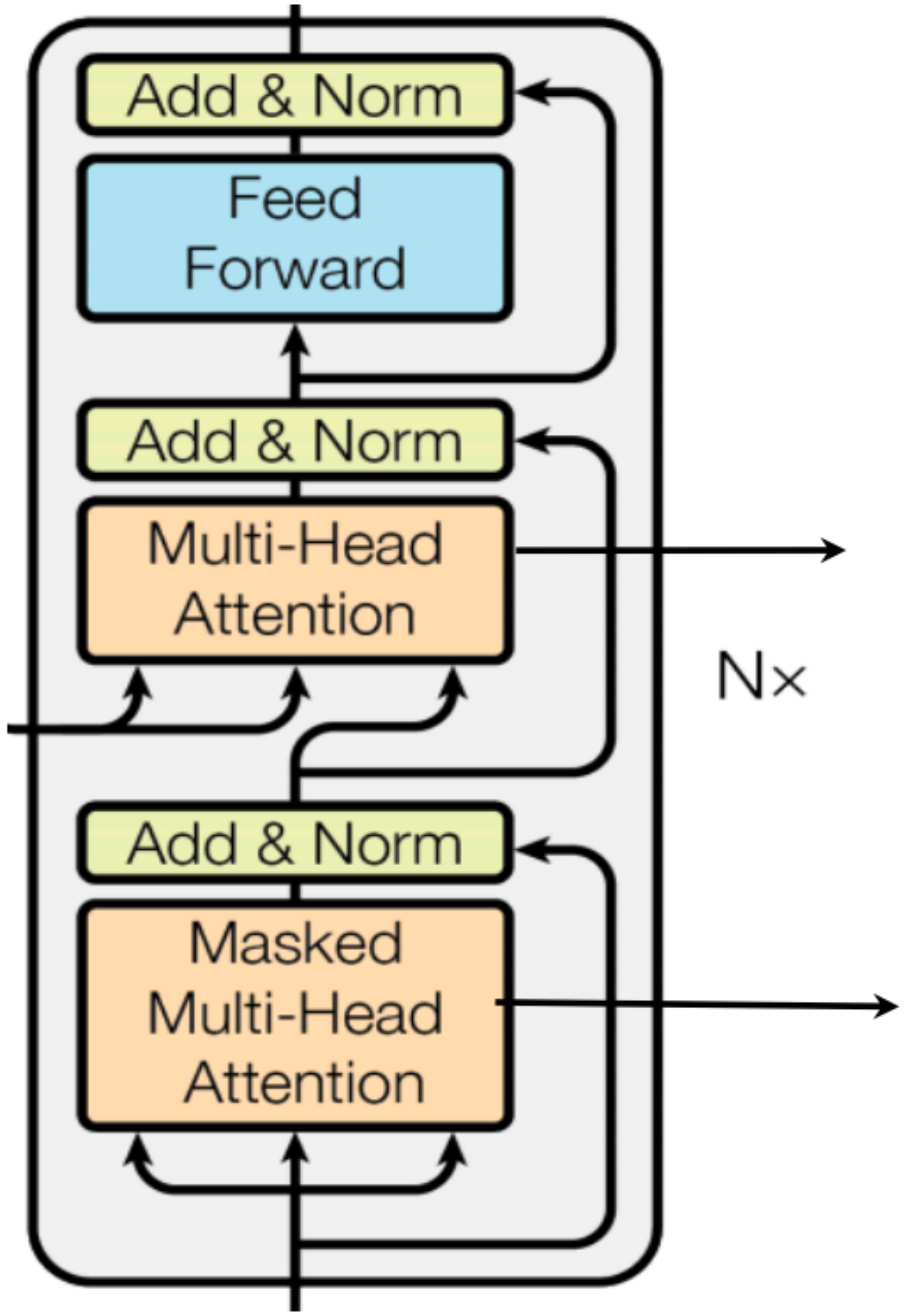



Source: DeepLearning.AI




Skoltech

# Decoder is similar with another N blocks




**Encoder-Decoder Attention**

Every position from the decoder attends to the outputs from the encoder

**Masked Self-Attention**

Every position attends to **previous** positions

Source: DeepLearning.AI

# Conclusion

- For long sequences in RNNs there is loss of information
- In RNNs there is the problem of vanishing gradient
- Transformers help with all of the above

*Next lecture: Hawke's neural process*

Skoltech