UNIVERSITY OF TECHNOLOGY
IN THE EUROPEAN CAPITAL OF CULTURE
CHEMNITZ

Neurocomputing

Attentional neural networks

Julien Vitay

Professur für Künstliche Intelligenz - Fakultät für Informatik

https://tu-chemnitz.de/informatik/KI/edu/neurocomputing

Attentional recurrent networks

- The problem with seq2seq is that it **compresses** the complete input sentence into a single state vector.

- For long sequences, the beginning of the sentence may not be present in the final state vector:
  - Truncated BPTT, vanishing gradients.
  - When predicting the last word, the beginning of the paragraph might not be necessary.
- Consequence: there is not enough information in the state vector to start translating.

https://jalammar.github.io/visualizing-neural-machine-translation-mechanics-of-seq2seq-models-with-attention/

Attentional recurrent networks

- A solution would be to concatenate the **state vectors** of all steps of the encoder and pass them to the decoder.

- **Problem 1:** it would make a lot of elements in the state vector of the decoder (which should be constant).
- **Problem 2:** the state vector of the decoder would depend on the length of the input sequence.

https://jalammar.github.io/visualizing-neural-machine-translation-mechanics-of-seq2seq-models-with-attention/

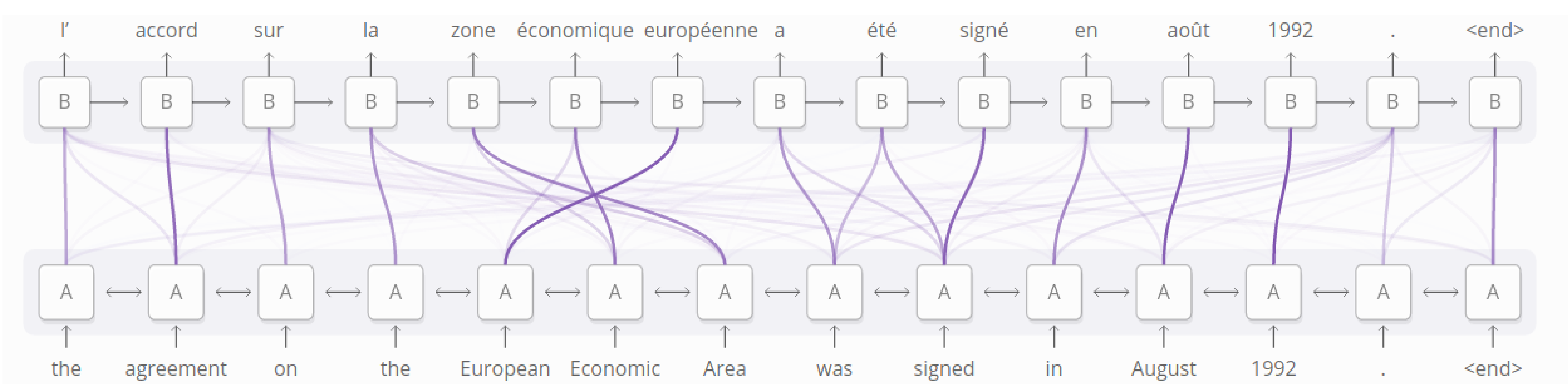
Attentional recurrent networks

- Attentional mechanisms let the decoder decide (by learning) which state vectors it needs to generate each word at each step.

- The **attentional context vector** of the decoder A_t^decoder at time t is a weighted average of all state vectors C_i^encoder of the encoder.

$$A_t^\text{decoder} = \sum_{i=0}^{T} a_i \, C_i^\text{encoder}$$

- The coefficients a_i are called the **attention scores** : how much attention is the decoder paying to each of the encoder's state vectors?



Dzmitry Bahdanau, Kyunghyun Cho, Yoshua Bengio (2014). Neural Machine Translation by Jointly Learning to Align and Translate. arXiv:1409.0473

Attentional recurrent networks

- The attention scores a_i are computed as a **softmax** over the scores e_i (in order to sum to 1):

$$a_i = \frac{\exp e_i}{\sum_j \exp e_j} \Rightarrow A_t^\text{decoder} = \sum_{i=0}^T \frac{\exp e_i}{\sum_j \exp e_j} \, C_i^\text{encoder}$$

- The score e_i is computed using:
  - the previous output of the decoder $\mathbf{h}_{t-1}^\text{decoder}$.
  - the corresponding state vector C_i^encoder of the encoder at step i.
  - attentional weights W_a.

$$e_i = \tanh(W_a \, [\mathbf{h}_{t-1}^\text{decoder}; C_i^\text{encoder}])$$

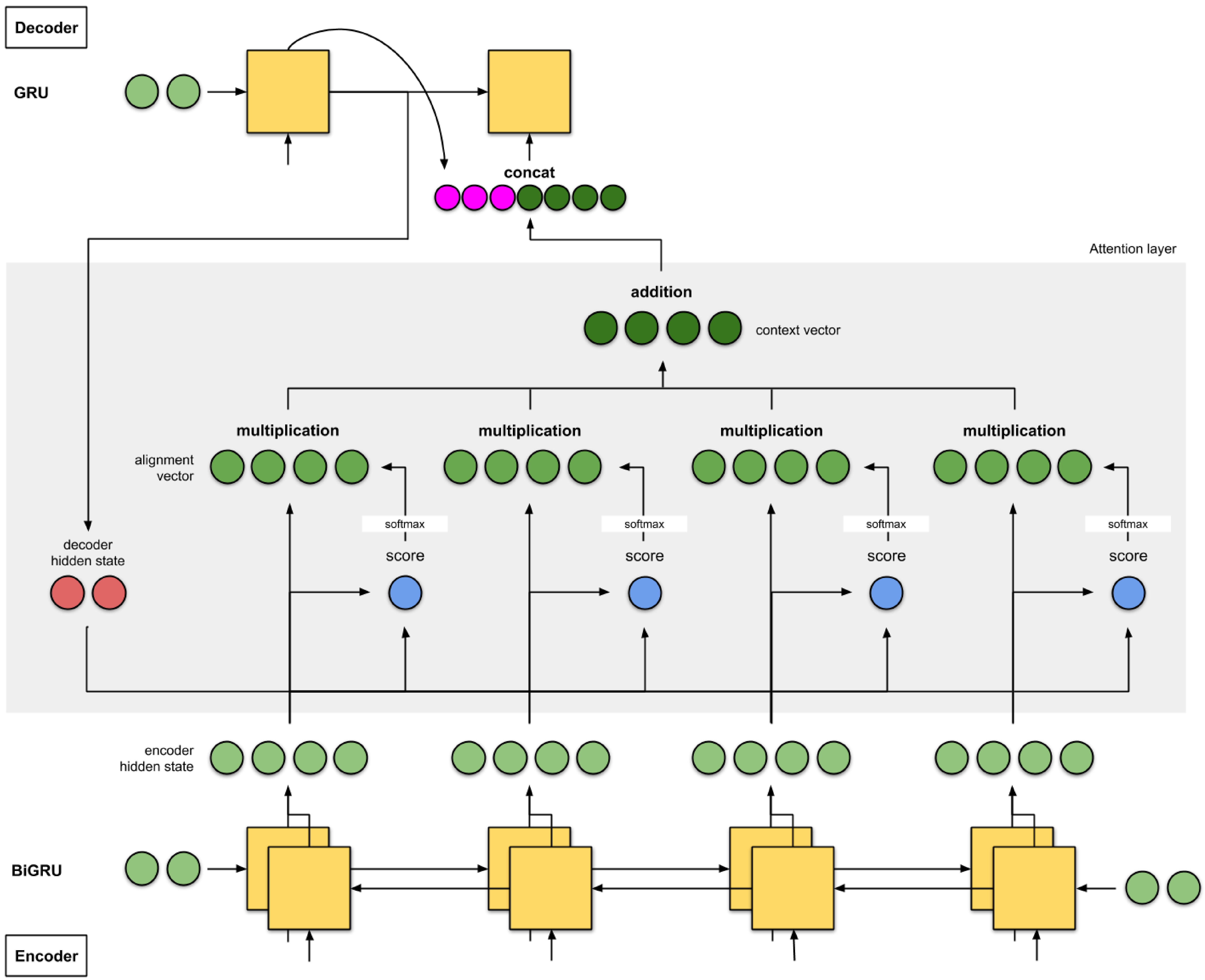
- Everything is differentiable, these attentional weights can be learned with BPTT.

https://towardsdatascience.com/day-1-2-attention-seq2seq-models-65df3f49e263

Attentional recurrent networks

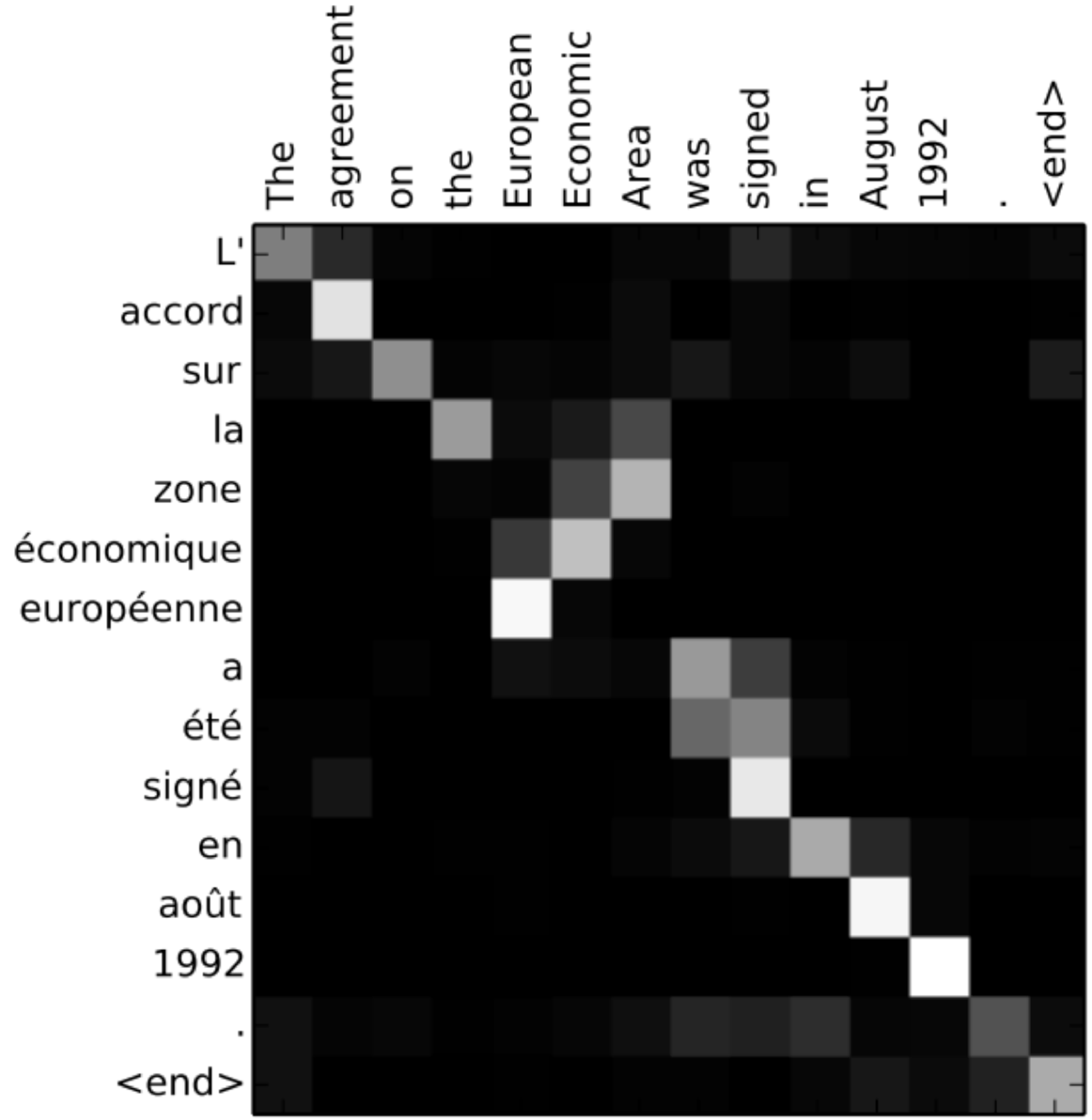
- The attentional context vector A_t^{decoder} is concatenated with the previous output $\mathbf{h}_{t-1}^{\text{decoder}}$ and used as the next input $\mathbf{x}_t^{\text{decoder}}$ of the decoder:

$$\mathbf{x}_t^{\text{decoder}} = [\mathbf{h}_{t-1}^{\text{decoder}}; A_t^{\text{decoder}}]$$


Decoder
GRU
concat
Attention layer
addition
context vector
multiplication
multiplication
multiplication
multiplication
alignment vector
softmax
softmax
softmax
softmax
score
score
score
score
decoder hidden state
encoder hidden state
BiGRU
Encoder

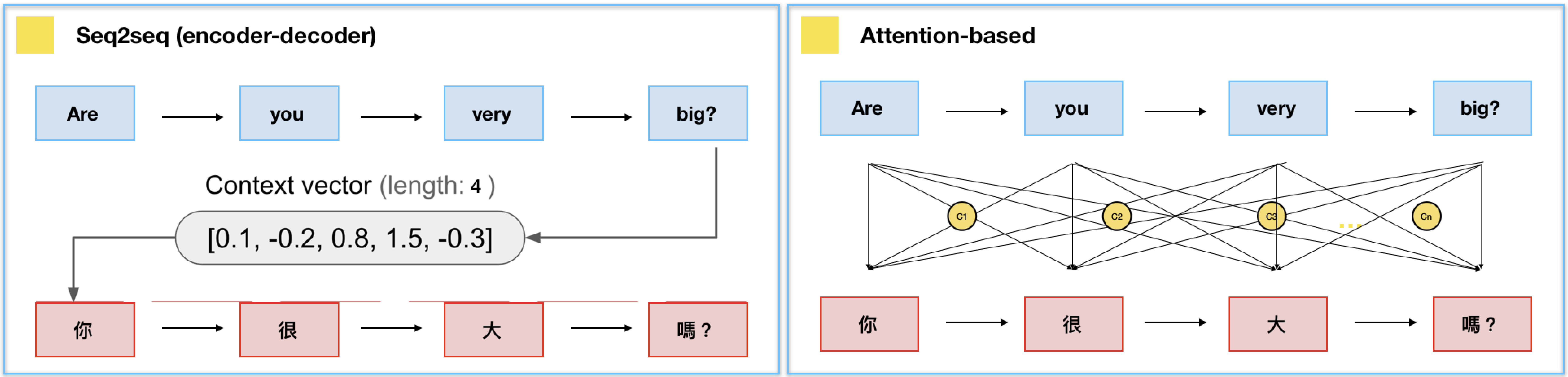
Attentional recurrent networks

- The attention scores or **alignment scores** a_i are useful to interpret what happened.
- They show which words in the original sentence are the most important to generate the next word.



Attentional recurrent networks

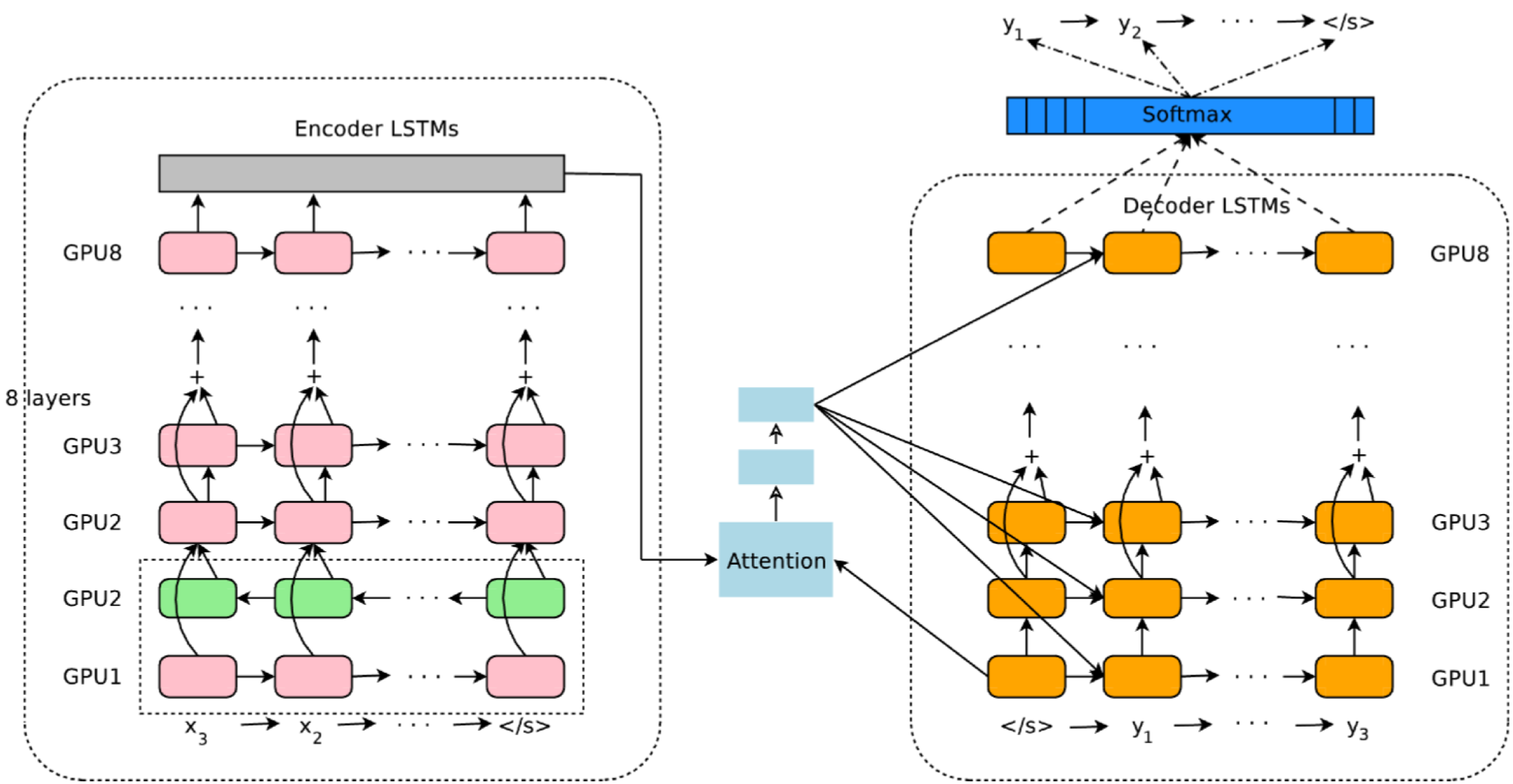
- **Attentional mechanisms** are now central to NLP.



- The whole **history** of encoder states is passed to the decoder, which learns to decide which part is the most important using **attention**.
- This solves the bottleneck of seq2seq architectures, at the cost of much more operations.
- They require to use fixed-length sequences (generally 50 words).

Google Neural Machine Translation (GNMT)

- Google Neural Machine Translation (GNMT) uses an attentional recurrent NN, with bidirectional GRUs, 8 recurrent layers on 8 GPUs for both the encoder and decoder.



Wu et al. (2016). Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. arXiv:1609.08144v2