University of Ljubljana, Faculty of Computer and Information Science

Attention mechanism

Prof Dr Marko Robnik-Šikonja

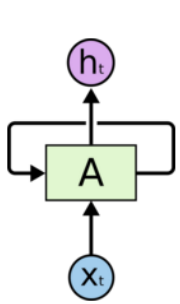
Natural Language Processing, Edition 2023

Contents

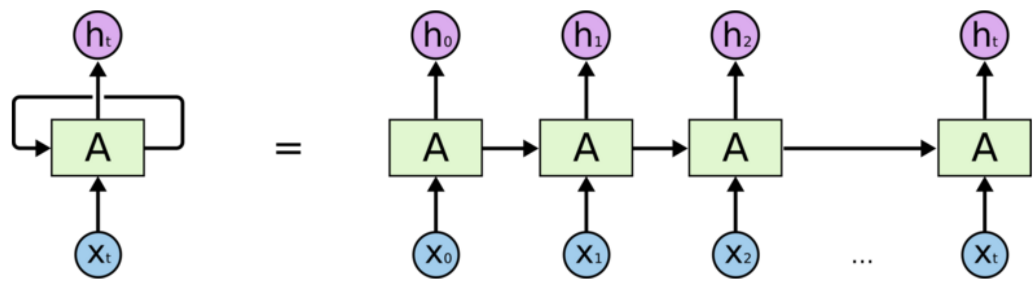
- encoder decoder networks
- attention mechanism

Recurrent Neural Networks

- Recurrent Neural Networks are networks with loops in them, allowing information to persist.



Recurrent Neural Networks have loops.

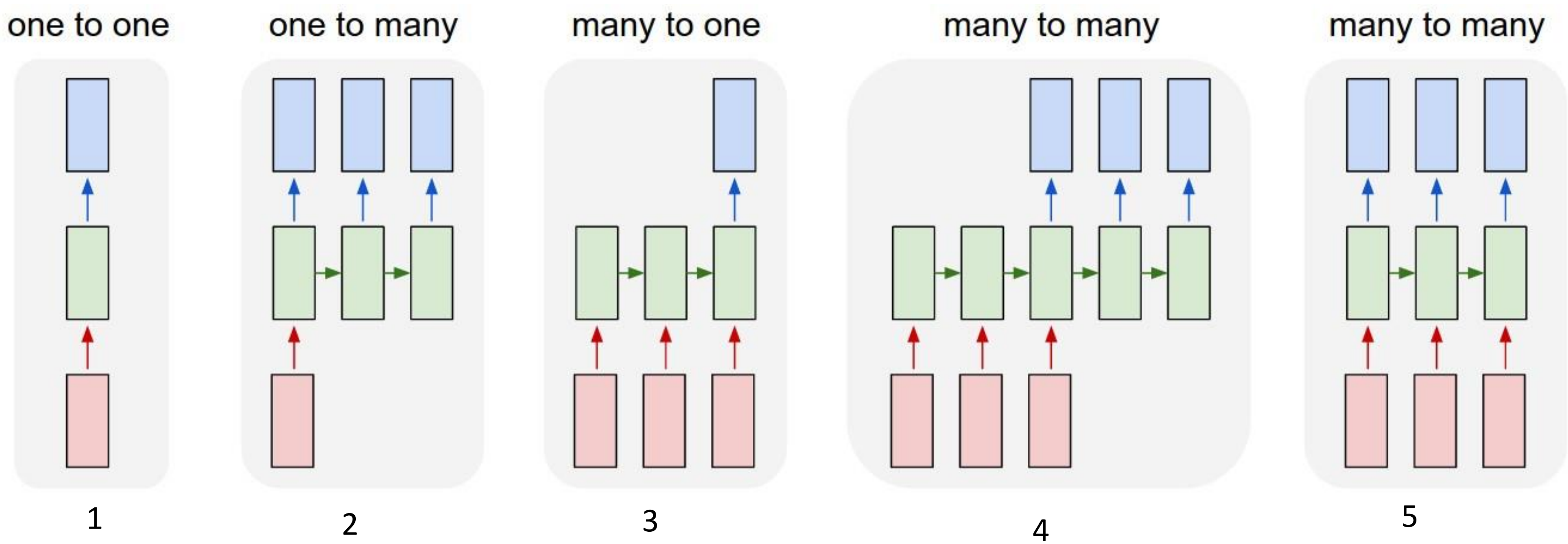


An unrolled recurrent neural network.

In the above diagram, a chunk of neural network, ***A***, looks at some input $\boldsymbol{x_t}$ and outputs a value $\boldsymbol{h_t}$.
A loop allows information to be passed from one step of the network to the next.

A recurrent neural network can be thought of as multiple copies of the same network, each passing a message to a successor.
The diagram above shows what happens if we unroll the loop.

Examples of Recurrent Neural Networks



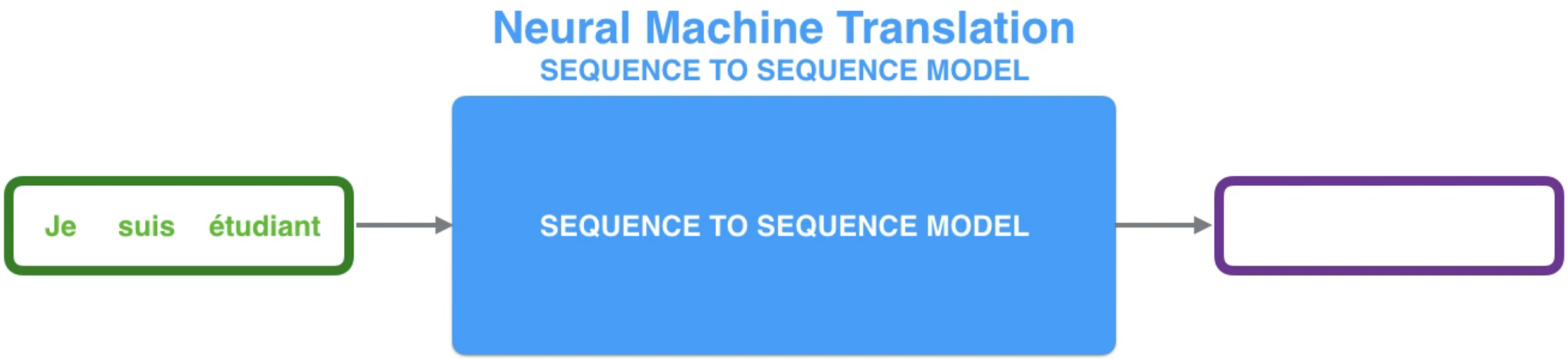
- Each rectangle is a vector and arrows represent functions (e.g. matrix multiply).
- Input vectors are in red, output vectors are in blue and green vectors hold the RNN's state

1. Standard mode of processing without RNN, from fixed-sized input to fixed-sized output (e.g. image classification).
2. Sequence output (e.g. image captioning takes an image and outputs a sentence of words).
3. Sequence input (e.g. sentiment analysis where a given sentence is classified as expressing positive or negative sentiment).
4. Sequence input and sequence output (e.g. Machine Translation: an RNN reads a sentence in English and then outputs a sentence in French).
5. Synced sequence input and output (e.g. video classification where we wish to label each frame of the video).

Seq2Seq model

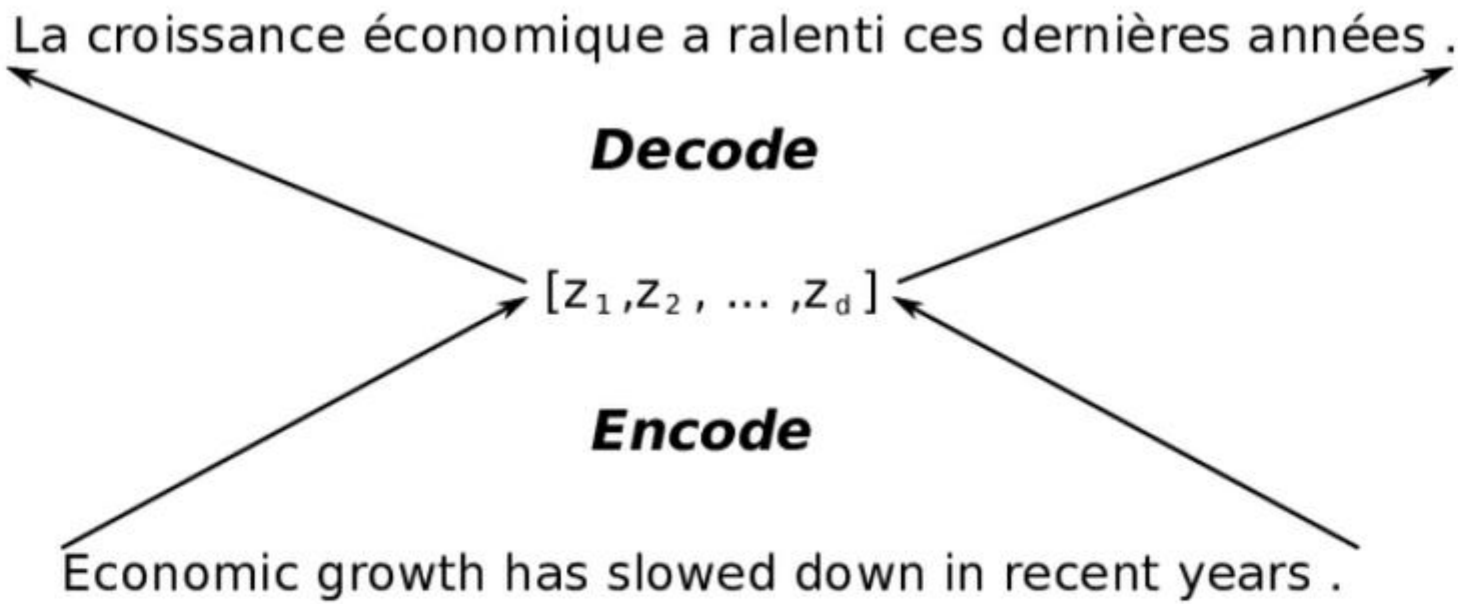
Videos by Jay Alammar: [Visualizing A Neural Machine Translation Model (Mechanics of Seq2seq Models With Attention)](), 2018

Seq2Seq for NMT

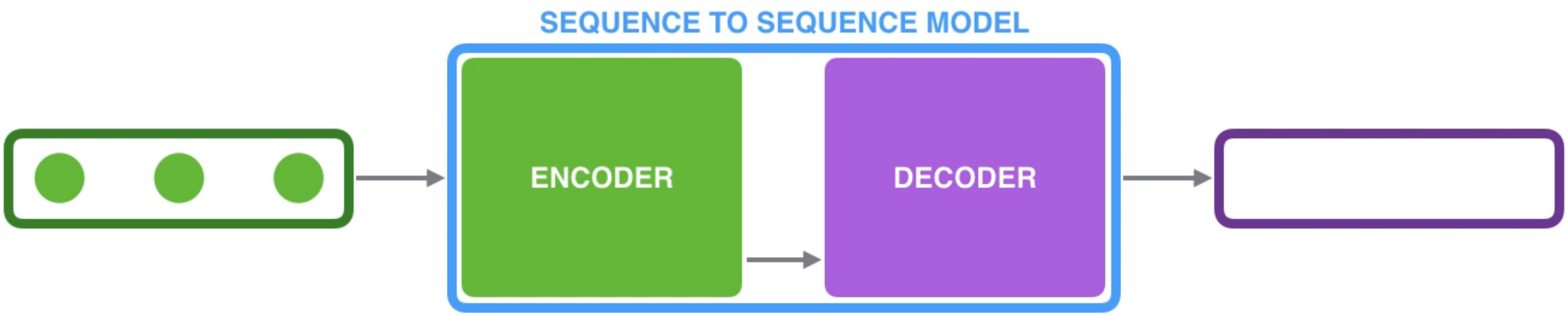


Encoder-Decoder model

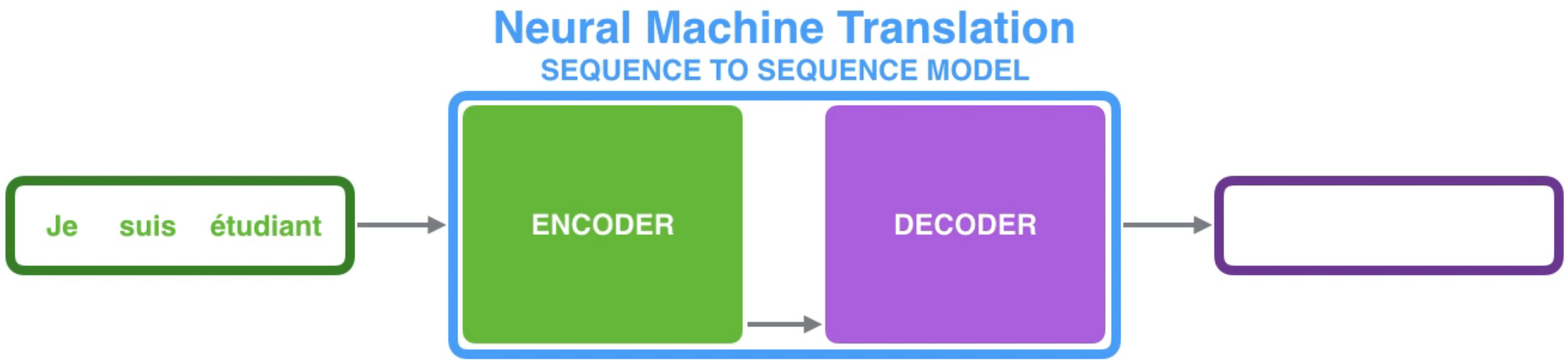
- encode into a latent space

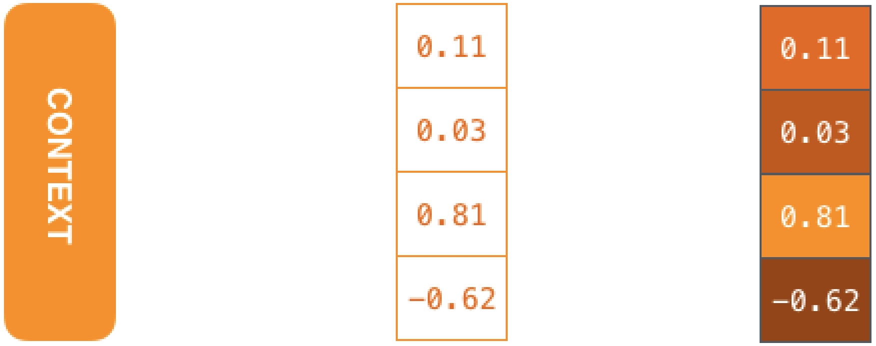


Encoder-decoder for sequences

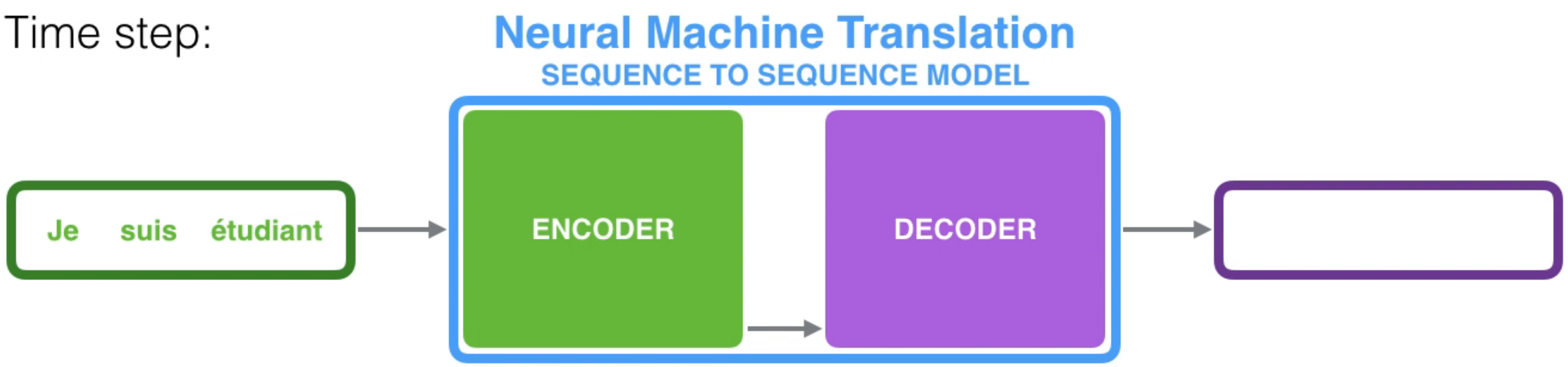


Encoder-decoder for NMT

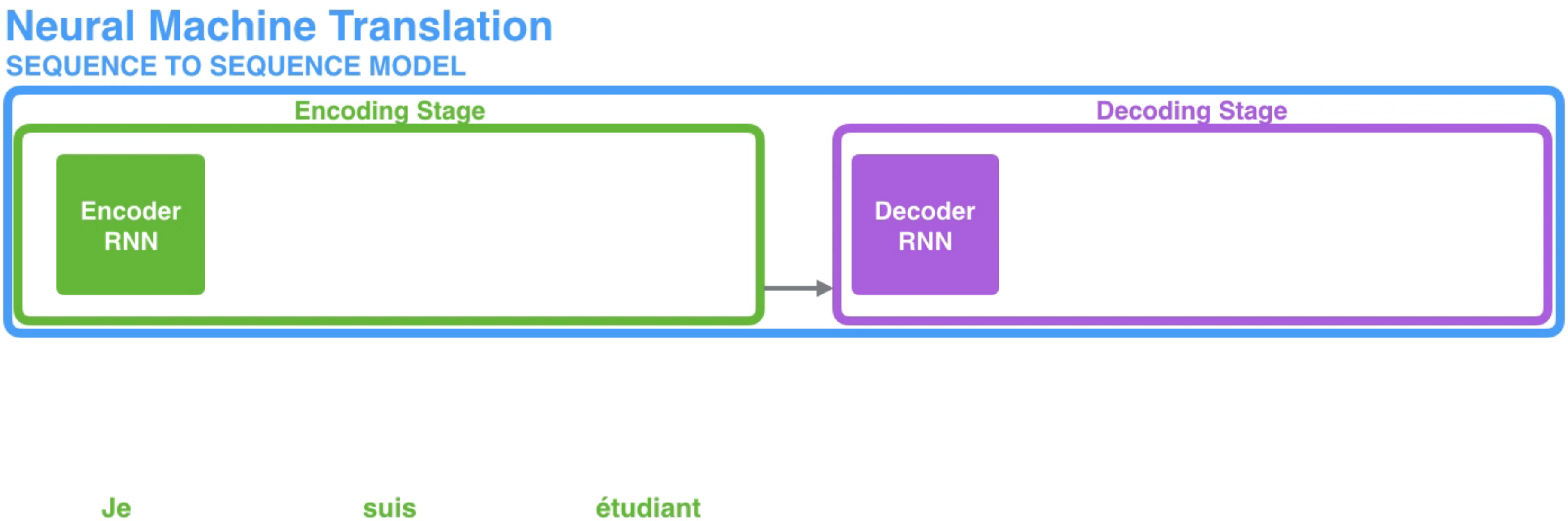




Encoder-decoder hidden states



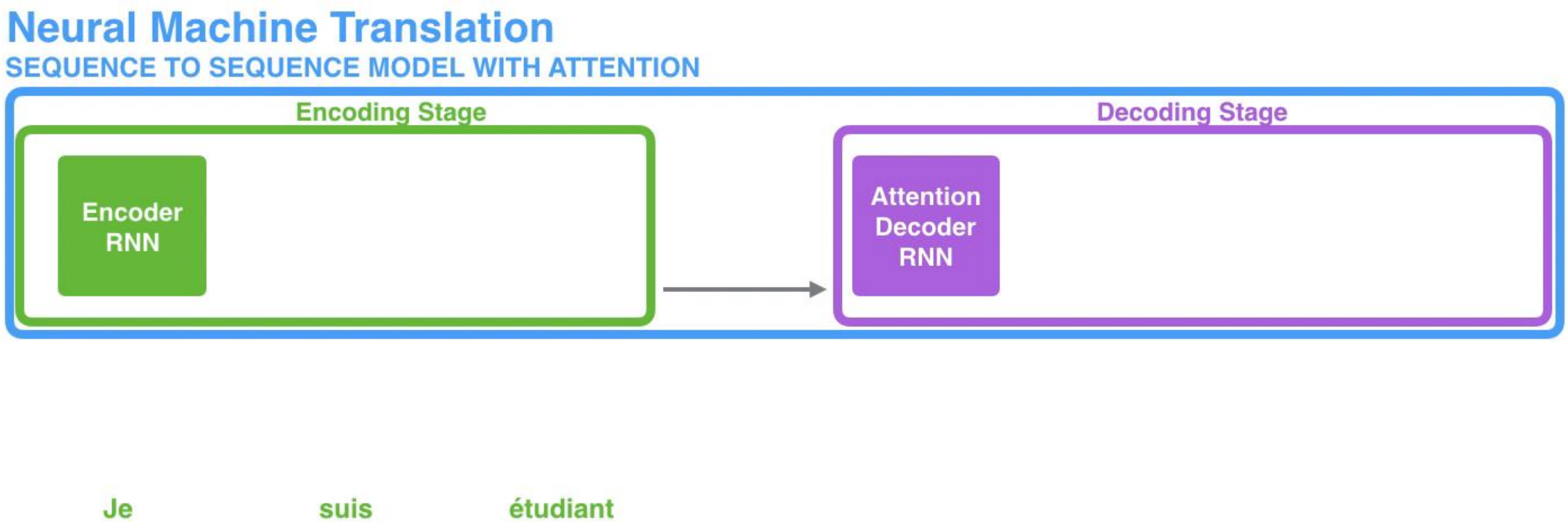
Unrolled encoder-decoder



Problems of encoder-decoder models

- long dependencies that would require larger networks and many more training data
- the information of different length sentences is stored in the fixed length hidden layer (migh be too long or to short)
- solution: attention mechanism

NMT with attention



Attention mechanism implementation for RNNs 1/2

- for all input words, we store their hidden layer weights
- during decoding, we add these vectors to the decoder input
- we use bidirectional encoding (forward and backward LM) and concatenate both weight vectors
- vectors are stored into a matrix

$$\overrightarrow{\boldsymbol{h}}_j^{(f)} = \mathrm{RNN}(\mathrm{embed}(f_j), \overrightarrow{\boldsymbol{h}}_{j-1}^{(f)})$$
$$\overleftarrow{\boldsymbol{h}}_j^{(f)} = \mathrm{RNN}(\mathrm{embed}(f_j), \overleftarrow{\boldsymbol{h}}_{j+1}^{(f)}).$$

$$\boldsymbol{h}_j^{(f)} = [\overleftarrow{\boldsymbol{h}}_j^{(f)}; \overrightarrow{\boldsymbol{h}}_j^{(f)}].$$

$$H^{(f)} = \mathrm{concat\_col}(\boldsymbol{h}_1^{(f)}, \ldots, \boldsymbol{h}_{|F|}^{(f)}).$$

Attention mechanism implementation for RNNs 2/2

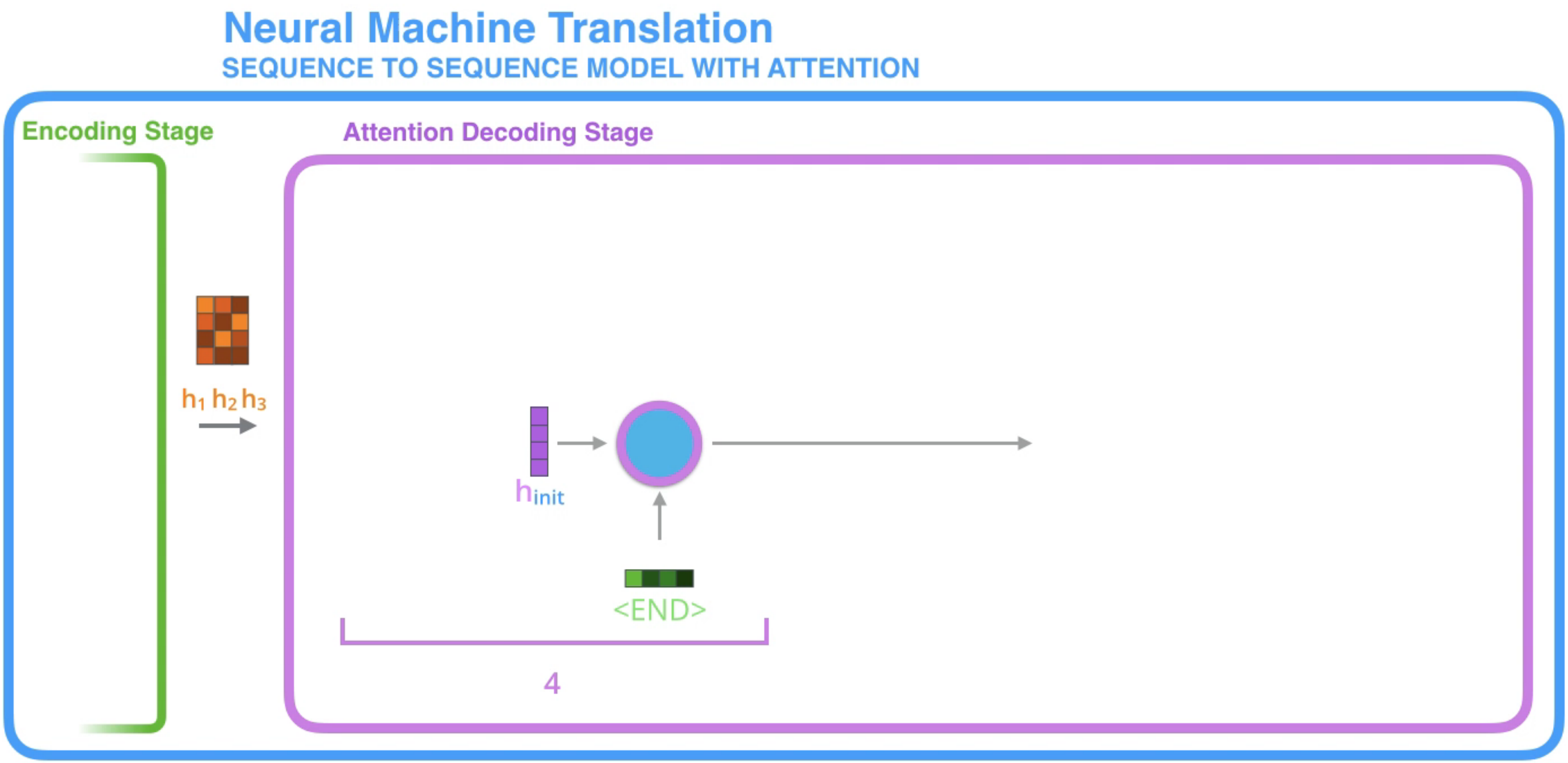
- we train the attention – which stored vectors are more or less important for decoding certain words
- the importance is determined with the attention vector α_t (between 0 and 1, sums to 1), applied to stored hidden weights and given as additional input to the decoder

$$\boldsymbol{c}_t = H^{(f)} \boldsymbol{\alpha}_t$$

Illustration of attention

Attention at time step 4

Decoder with attention



Attention produces alignments

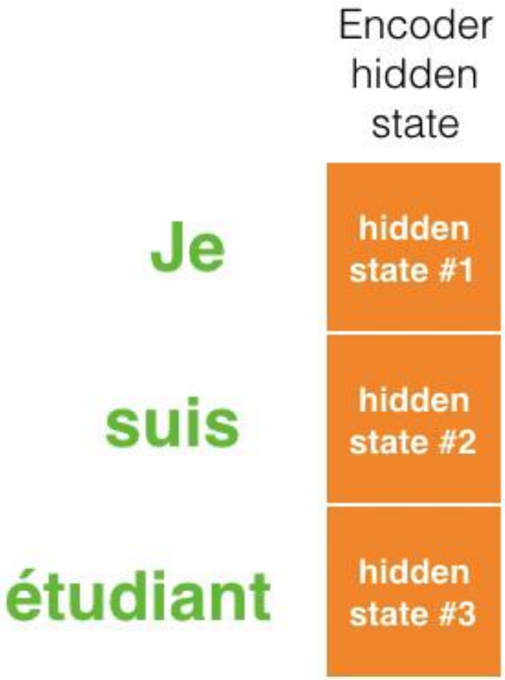
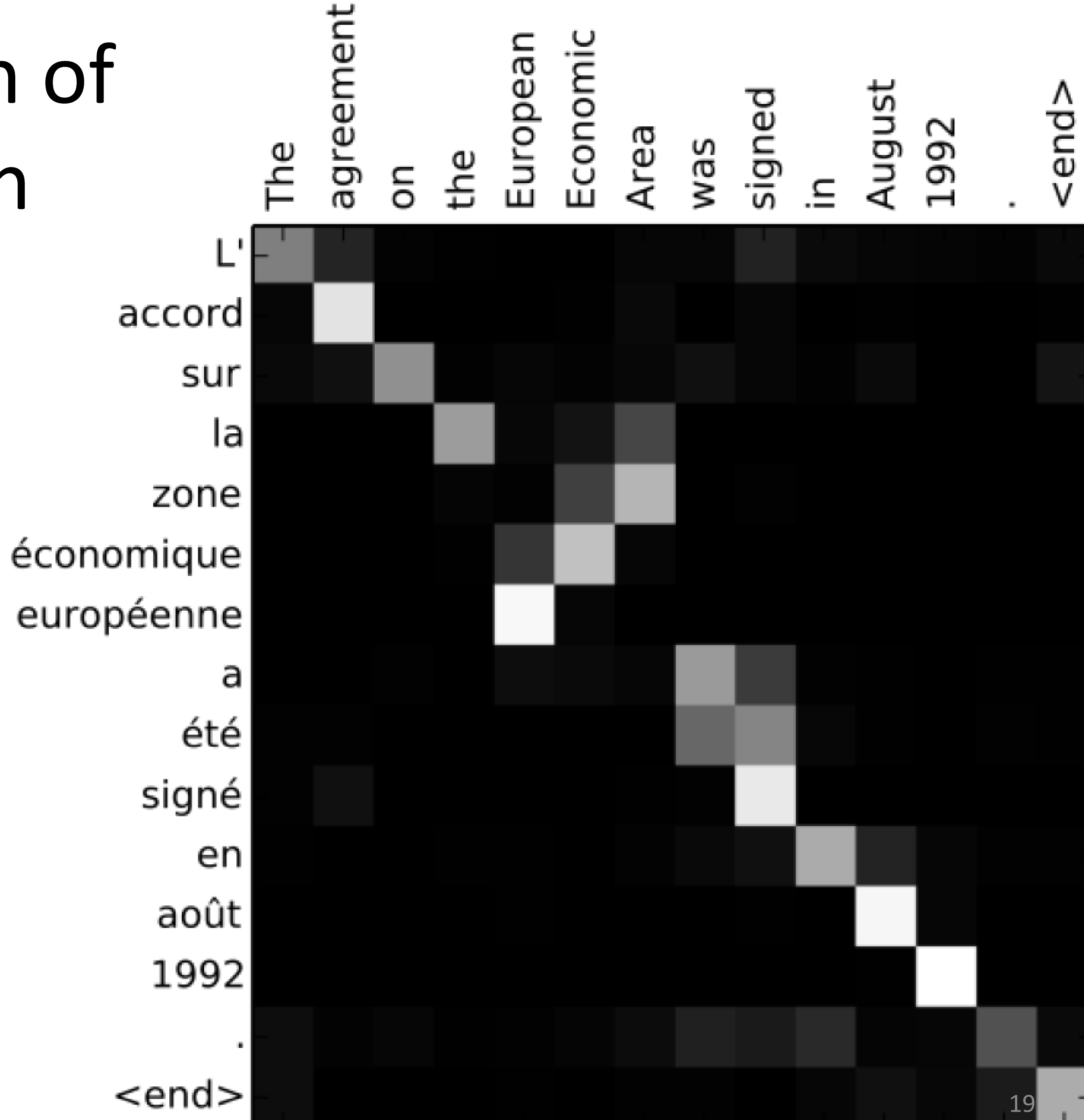
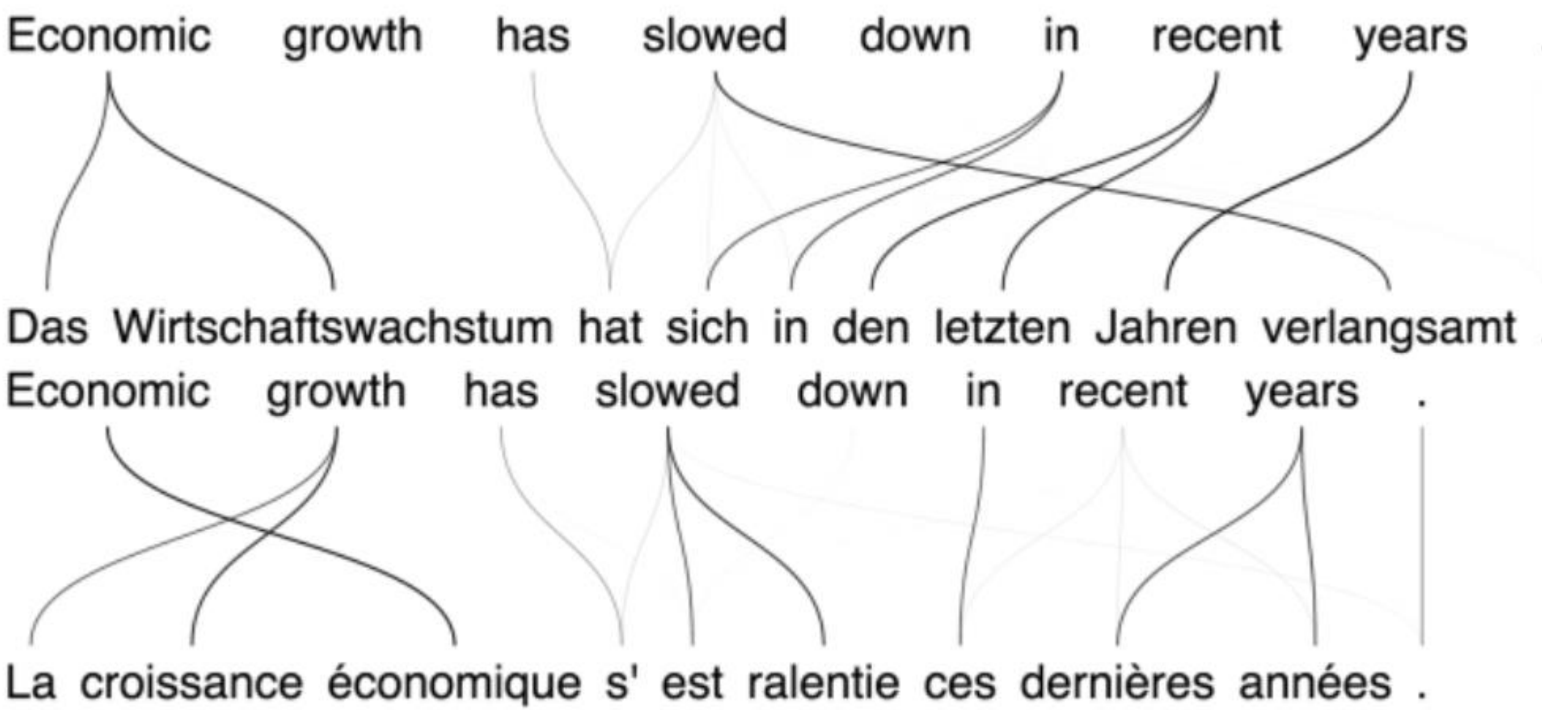


Illustration of attention



Attention illustration



Problems with RNNs

- We want parallelization but RNNs are inherently sequential
- Despite GRUs and LSTMs, RNNs still need attention mechanism to deal with long range dependencies – path length between states grows with sequence
- If attention gives us access to any state… maybe we can just use attention and don't need the RNN?