2022

Deep Learning based Text Processing

Lec 12: Sequence to Sequence Model with RNN

hsyi@kisti.re.kr

Hongsuk Yi (이홍석)

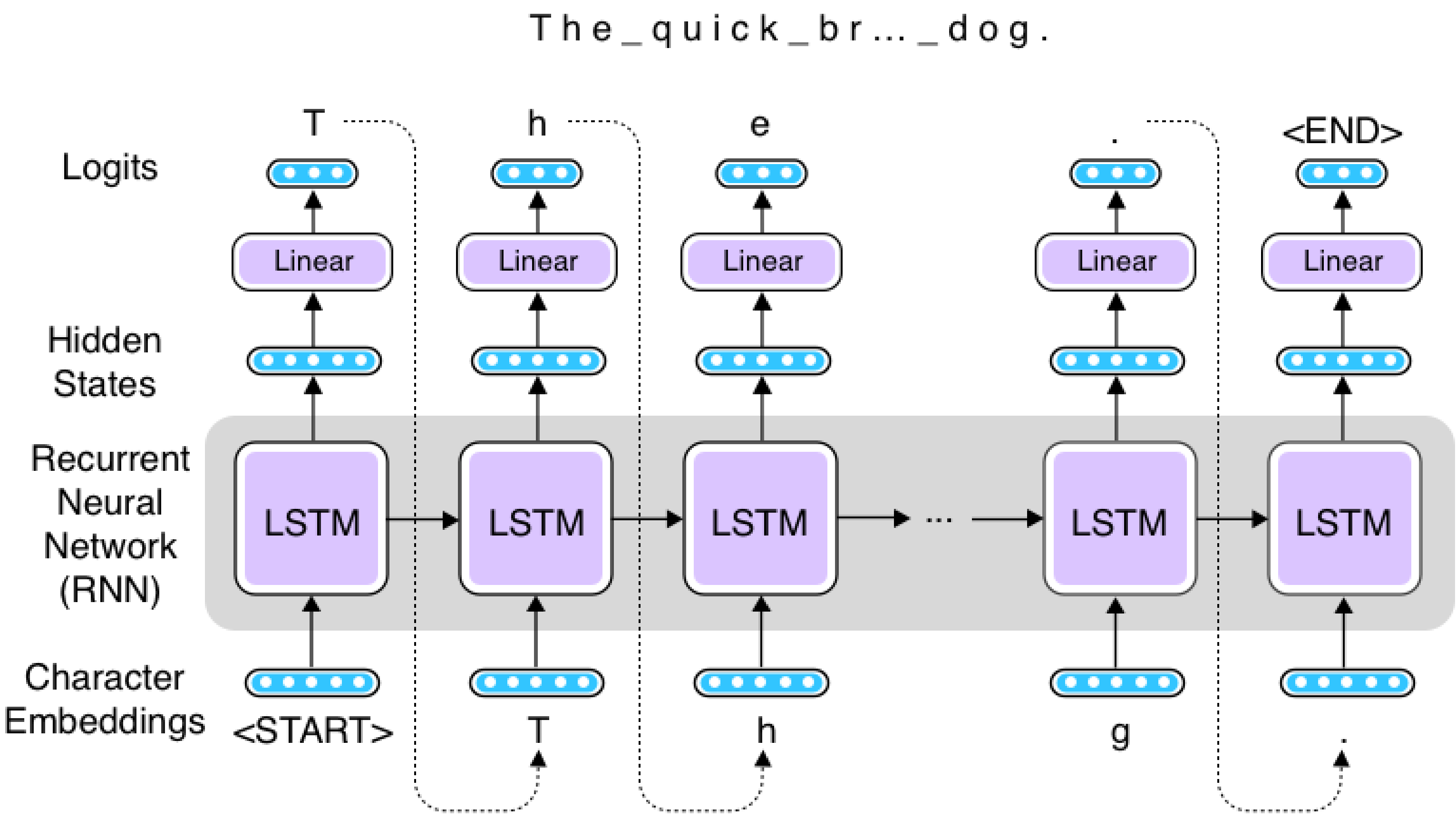
Overview of Course (III)

- Introduction to Recurrent Neural Network
  - ✓ Simple RNN, BPTT, Memory Cell
  - ✓ Code: Implementing an RNN with Keras
- Introduction to Long-Short Term Memroy
  - ✓ Cell state, LSTM, and GRU, and Applications
  - ✓ A Visual Guide to Recurrent Layers in Keras
  - ✓ Code: A simple LSTM layers
- Text generation with RNN
  - ✓ Tokenizer, Character-Level Language model
  - ✓ Code: Alice's Adventures in Wonderland
- **Sequence to Sequence Learning model with RNN**
  - ✓ Introduction to Seq2Seq and Attention model
  - ✓ Code: Character-Level Neural Machine Translation

Review the last class:

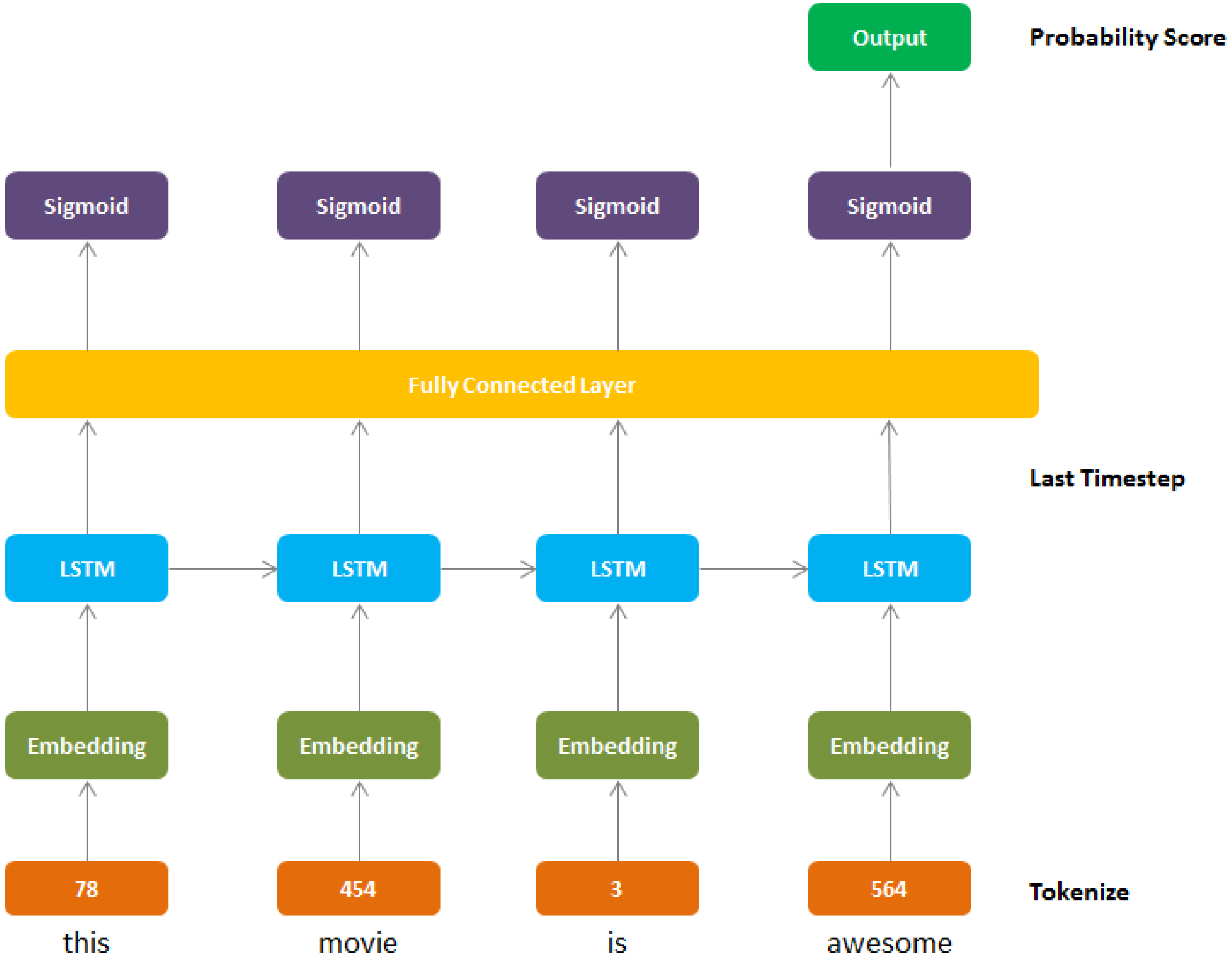
Character-level language model

Last time: Text Generation using RNN



<source> http://www.realworldnlpbook.com/blog/training-a-shakespeare-reciting-monkey-using-rl-and-seqgan.html

	review	sentiment
0	One of the other reviewers has mentioned that ...	positive
1	A wonderful little production. The filming tec...	positive
2	I thought this was a wonderful way to spend ti...	positive
3	Basically there's a family where a little boy ...	negative
4	Petter Mattei's "Love in the Time of Money" is...	positive
5	Probably my all-time favorite movie, a story o...	positive
6	I sure would like to see a resurrection of a u...	positive
7	This show was an amazing, fresh & innovative i...	negative
8	Encouraged by the positive comments about this...	negative



KISTI SINCE 1962

"Modeling word probabilities is really difficult"

	Supervised learning	Sequence modelling
Data	$\{x, y\}_i$	$\{x\}_i$
Model	$y \approx f_\theta(x)$	$p(x) \approx f_\theta(x)$
Loss	$\mathcal{L}(\theta) = \sum_{i=1}^{N} l(f_\theta(x_i), y_i)$	$\mathcal{L}(\theta) = \sum_{i=1}^{N} \log p(f_\theta(x_i))$
Optimisation	$\theta^* = \arg\min_\theta \mathcal{L}(\theta)$	$\theta^* = \arg\max_\theta \mathcal{L}(\theta)$

Simplest model:
Assume independence of words

$$p(\mathbf{x}) = \prod_{t=1}^{T} p(x_t)$$

p("modeling") × p("word") × p("probabilities") × p("is") × p("really") × p("difficult")

Word	$p(x_i)$
the	0.049
be	0.028
...	...
really	0.0005
...	...

More realistic model:

Assume conditional dependence of words

$$p(x_T) = p(x_T | x_1, \ldots, x_{T-1})$$

Modeling word probabilities is really ?

Context — Target

Target	p(x|context)
difficult	0.01
hard	0.009
fun	0.005
...	...
easy	0.00001

KISTI SINCE 1962

The chain rule

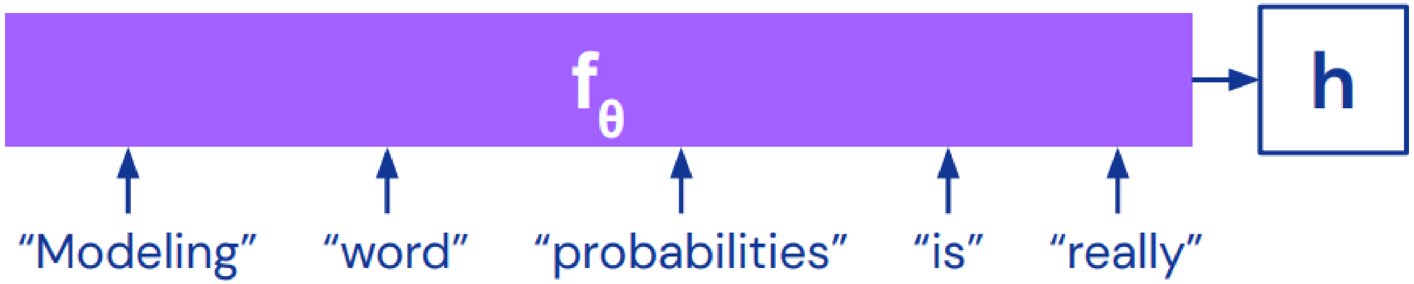
Computing the joint p(**x**) from conditionals

$$p(\mathbf{x}) = \prod_{t=1}^{T} p(x_t | x_1, \ldots, x_{t-1})$$

Modeling	$p(x_1)$
Modeling **word**	$p(x_2 \mid x_1)$
Modeling word **probabilities**	$p(x_3 \mid x_2, x_1)$
Modeling word probabilities **is**	$p(x_4 \mid x_3, x_2, x_1)$
Modeling word probabilities is **really**	$p(x_5 \mid x_4, x_3, x_2, x_1)$
Modeling word probabilities is really **difficult**	$p(x_6 \mid x_5, x_4, x_3, x_2, x_1)$

Learning to model word probabilities

✓Vectorising the context



f_θ summarises the context in $\boxed{h}$ such that:

$$p(x_t|x_1, \ldots, x_{t-1}) \approx p(x_t|h)$$

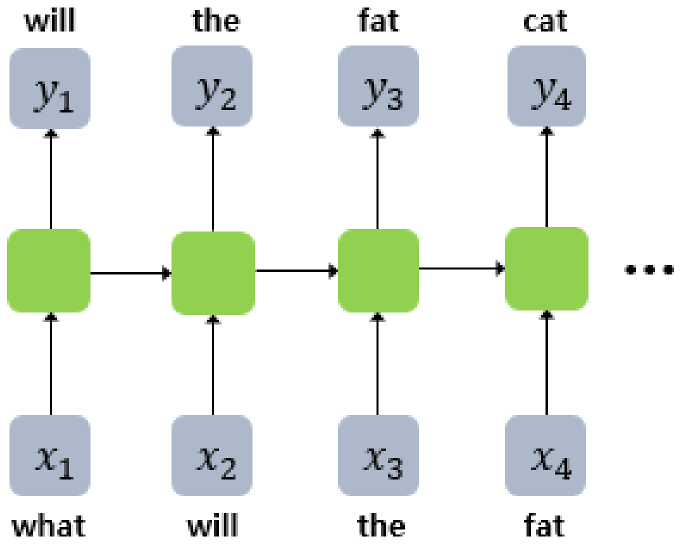
Desirable properties for f_θ:

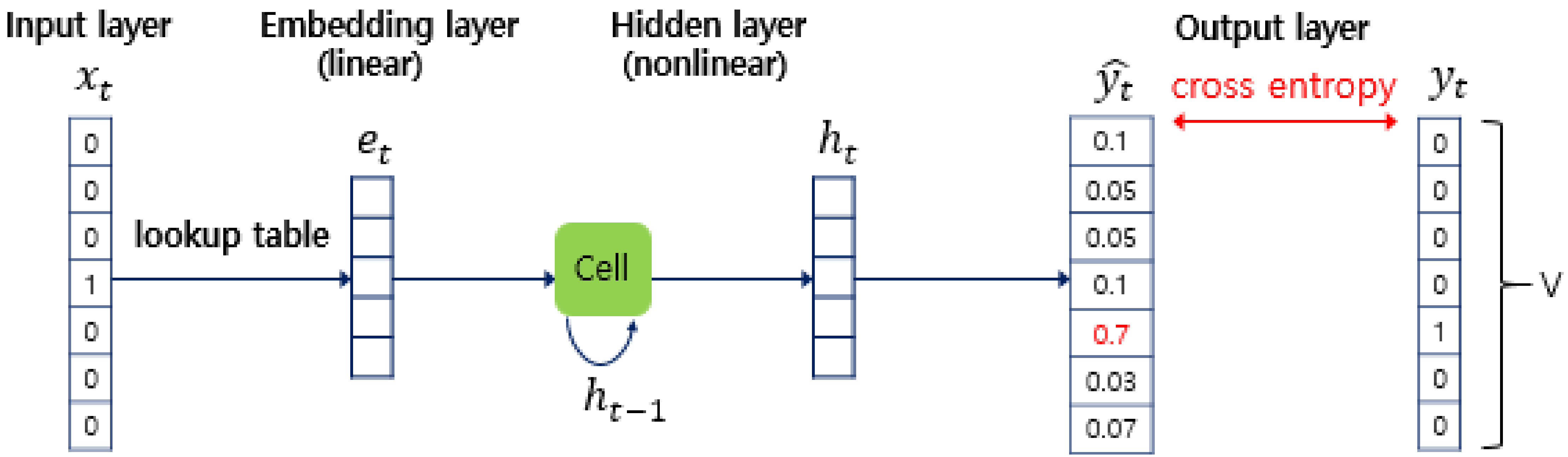
- Order matters
- Variable length
- Learnable (differentiable)

Recurrent Neural Network Language Model (RNNLM)

- Persistent state “h” stores information from the context observed so far
- (Example)
  - ✓ Predicts the next word from a word sequence

“What will the fat cat sit on”



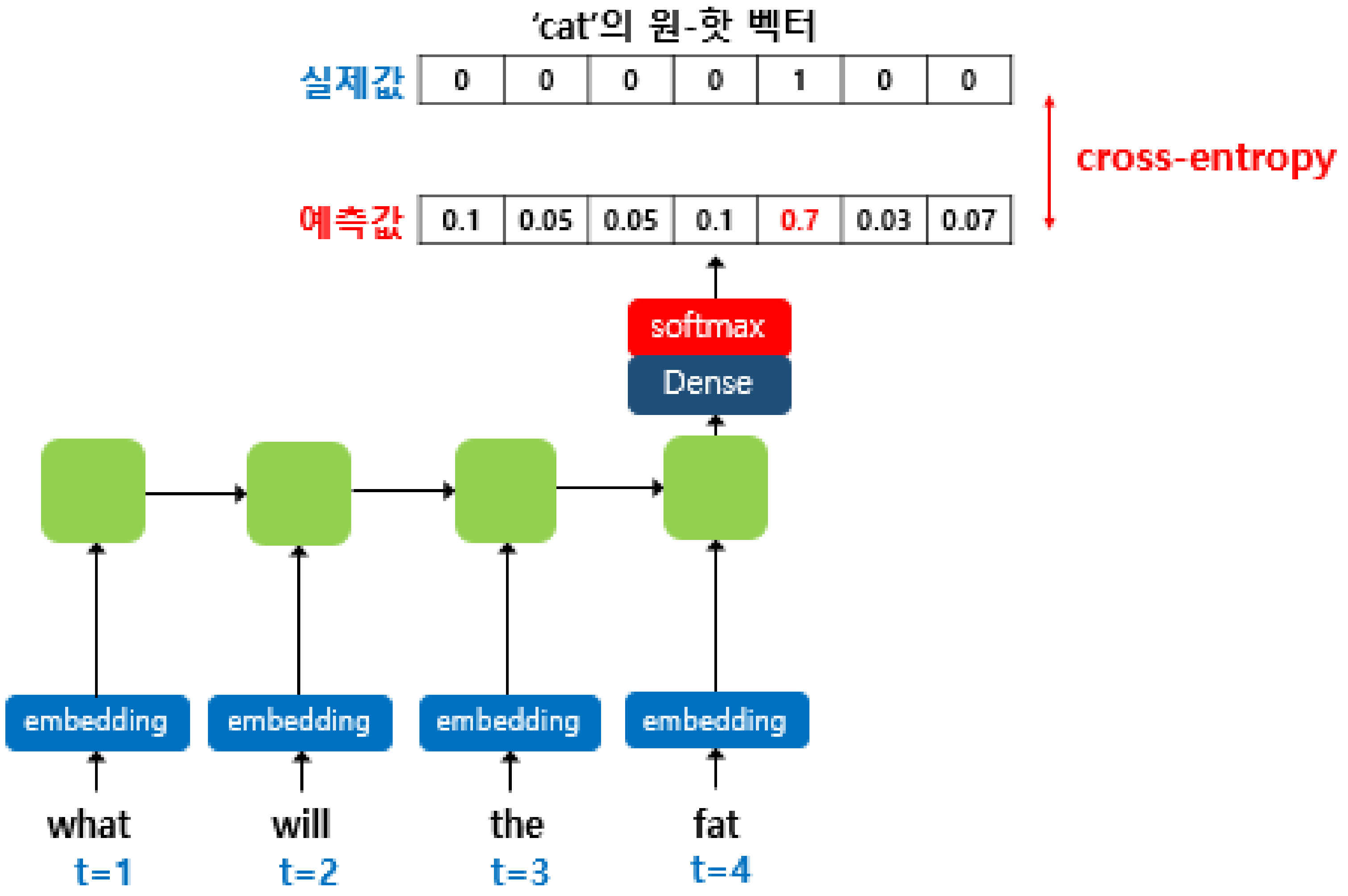


$$\mathbf{h}_t = \tanh(\mathbf{W}_h \mathbf{h}_{t-1} + \mathbf{W}_x \mathbf{x}_t)$$

$$p(\mathbf{x}_{t+1}) = softmax(\mathbf{W}_y \mathbf{h}_t)$$

$$\mathcal{L}_\theta(\mathbf{y}, \hat{\mathbf{y}})_t = -\mathbf{y}_t \log \hat{\mathbf{y}}_t$$

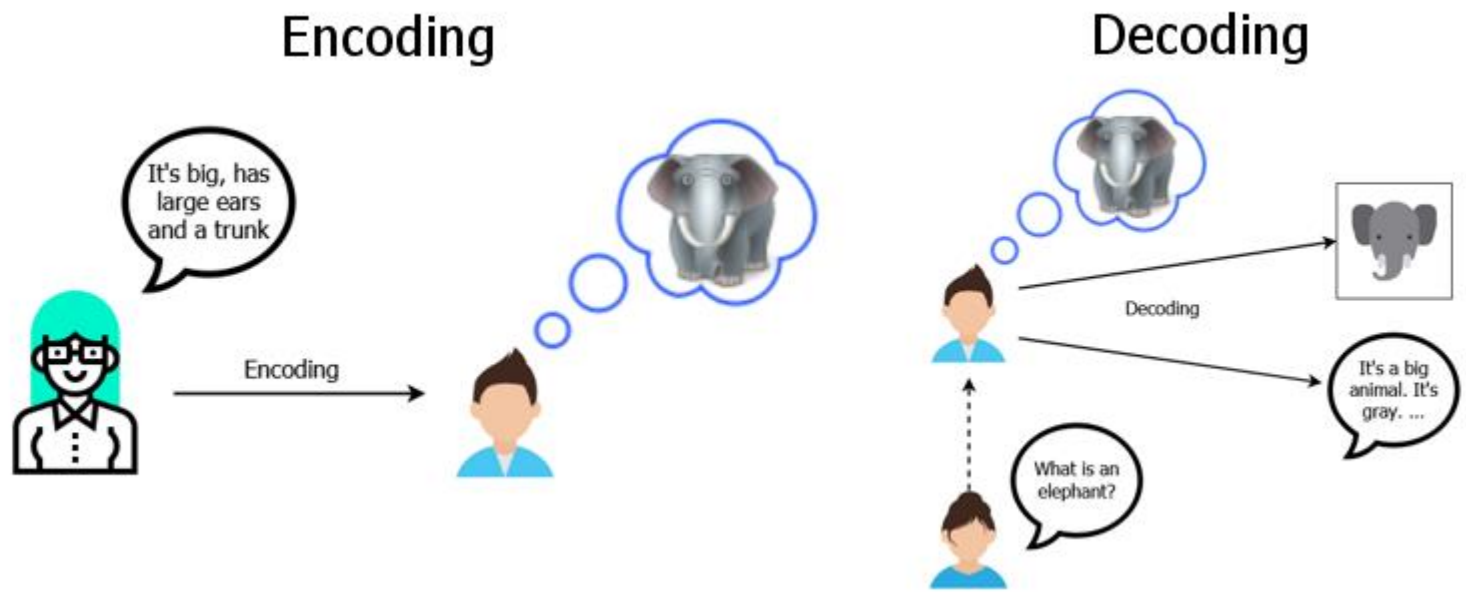
RNNLM: Teacher Forcing Learning



Sequence to Sequence (Seq2Seq)
Neural Machine Translation

❖ Seq2Seq

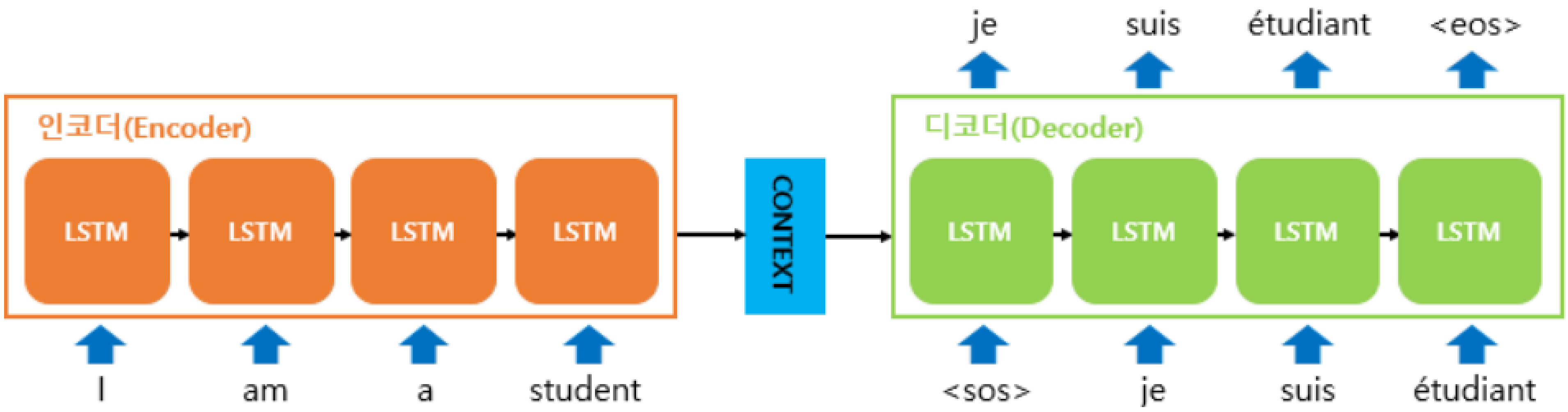
- ✓The encoding process is analogous to a teacher explaining to you what an elephant looks like and you create a mental image of that.
- ✓The decoding process takes place if a friend of yours ask what an elephant looks like.



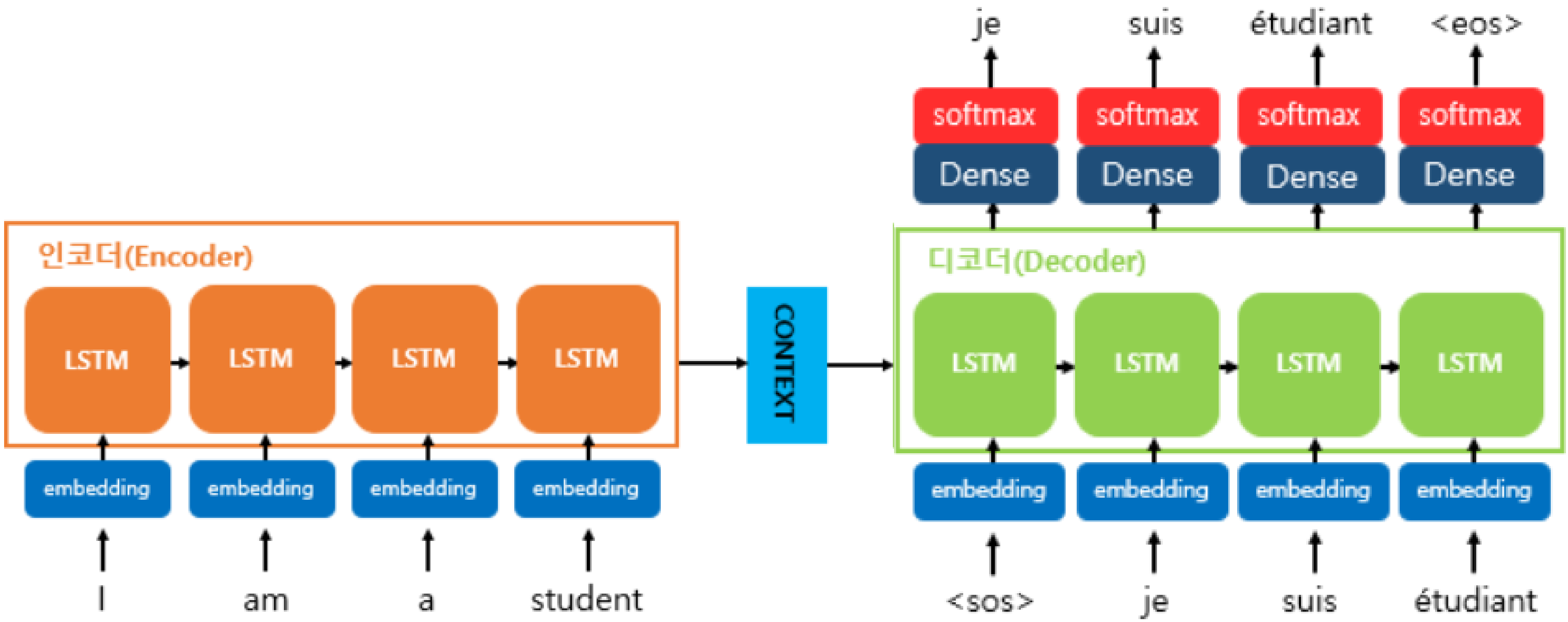
What is a Seq2Seq model?

❖ Sequence-to-sequence (Seq2Seq) for machine translation

- ✓ Encoder is “forced” to send only a single vector, regardless of the length of our input
- ✓ The last **hidden state** becomes the content vector that is sent to the decoder



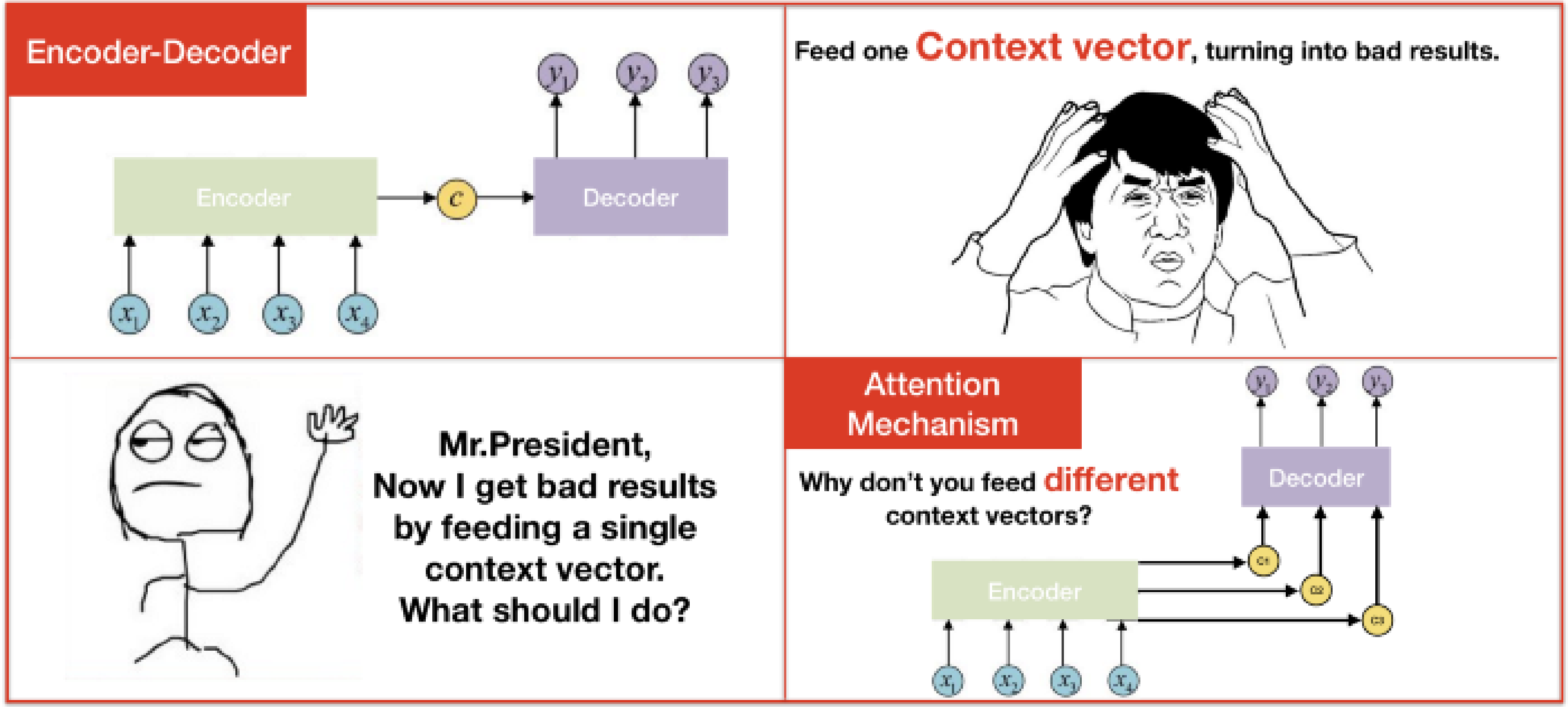
❖ Decoder is essentially an RNNLM (RNN Language Model)
  - ✓ Teacher Forcing Learning
  - ✓ Softmax for next prediction word



Limitation of Seq2Seq model

❖ A single Context vector

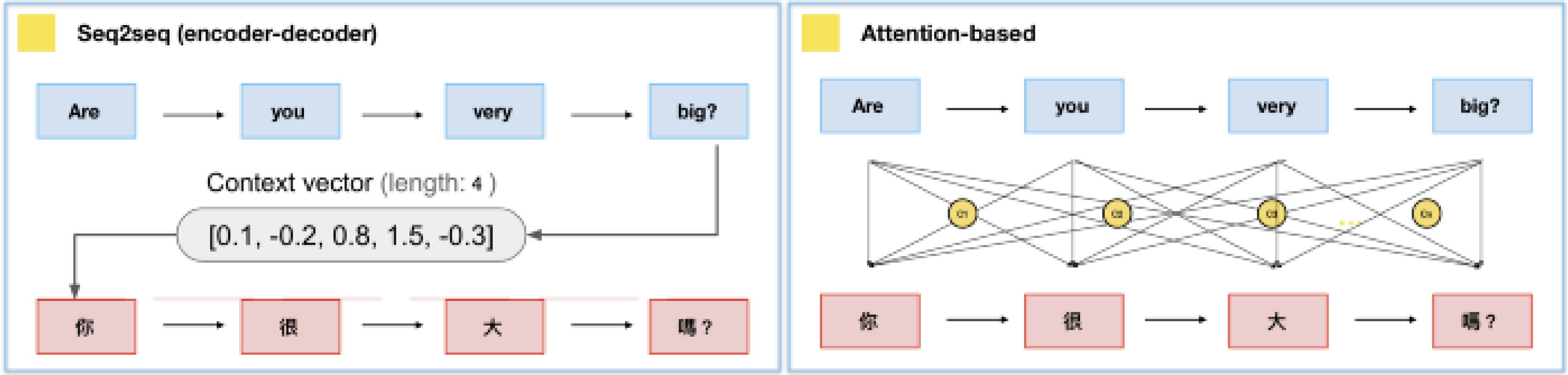
✓ compress all the information into one fixed-size vector results in information loss.



<source> https://bgg.medium.com/seq2seq-pay-attention-to-self-attention-part-1-d332e85e9aad

Attention in RNNs

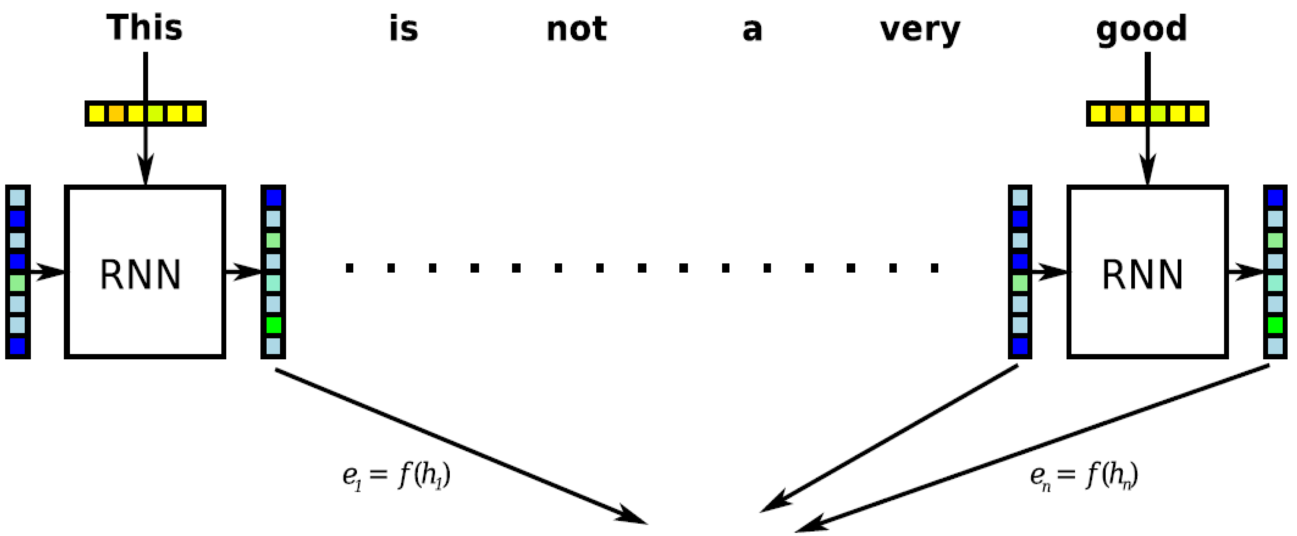
- The difference between Seq2seq model and attention model is the calculation of context vector



How do I translate the sentence by machine?

❖ Attention mechanism proposed by Bahdanau et al. (2015)

✓ We compute a "context vector" (weighted average) of the states which correspond to some notion of "importance"

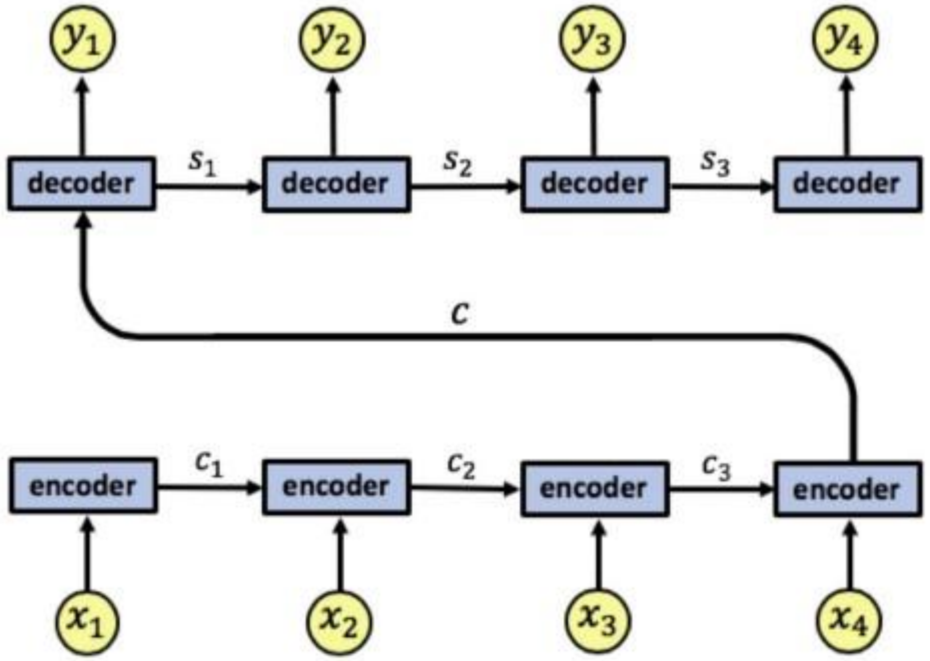


$$\alpha_i = \frac{\exp e_i}{\sum_{j=1}^{n} \exp e_j}$$

image borrowed from Richard Johansson (*Chalmers Technical University and University of Gothenburg)*

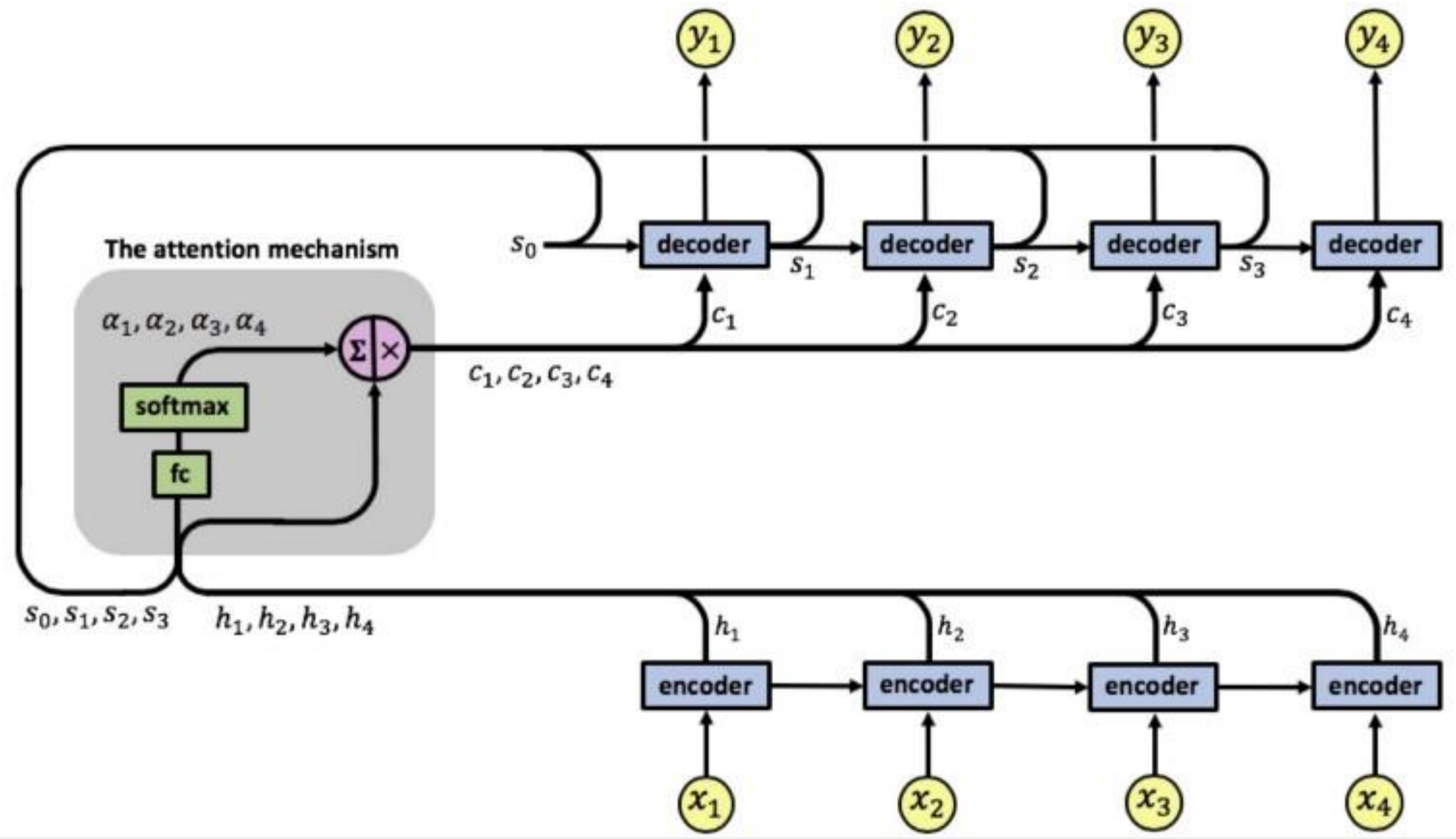
❖ An RNN encoder-decoder architecture

- ✓ we take an architecture with 4 time steps for simplicity
- ✓ a single context vector, c, which can casue information loss



<source> https://medium.datadriveninvestor.com/attention-in-rnns-321fbcd64f05

RNNs with an attention mechanism



<source> https://medium.datadriveninvestor.com/attention-in-rnns-321fbcd64f05

Attention Mechanism

- "importance score",

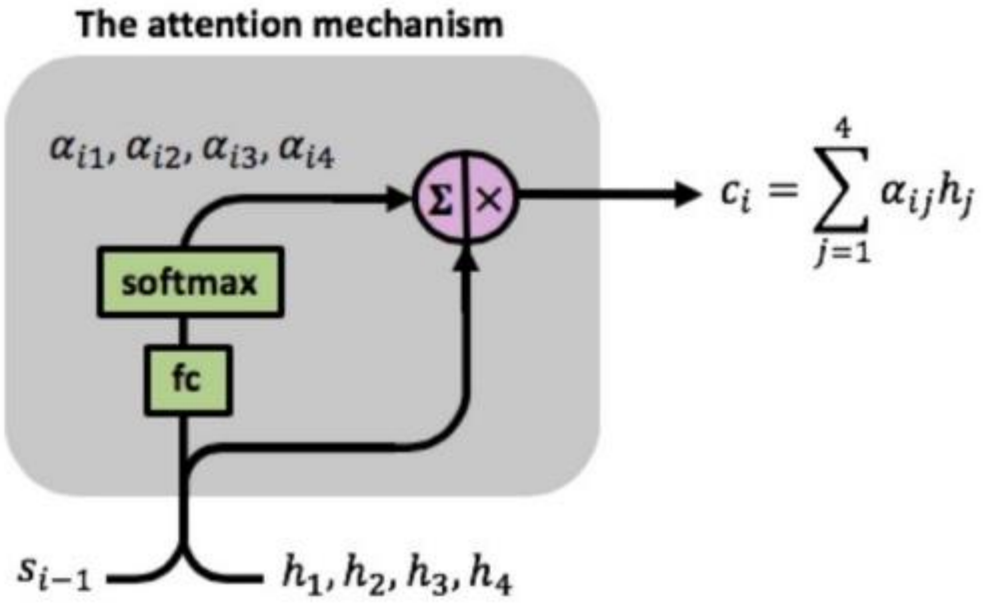
$$e_{ij} = \mathrm{fc}(s_{i-1}, h_j)$$

- "attention weight",

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

- "context vectors",

$$c_i = \sum_j \alpha_{ij} h_j$$

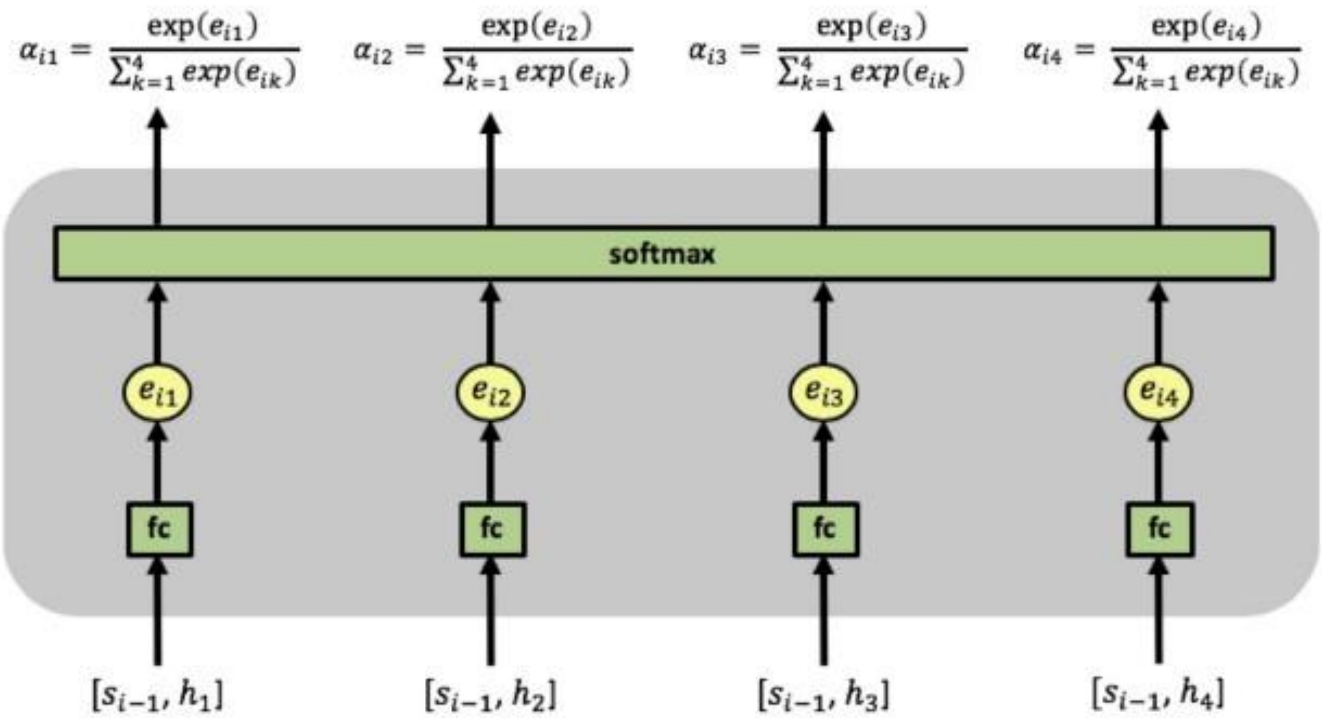


<source> https://medium.datadriveninvestor.com/attention-in-rnns-321fbcd64f05

Attention Mechanism

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

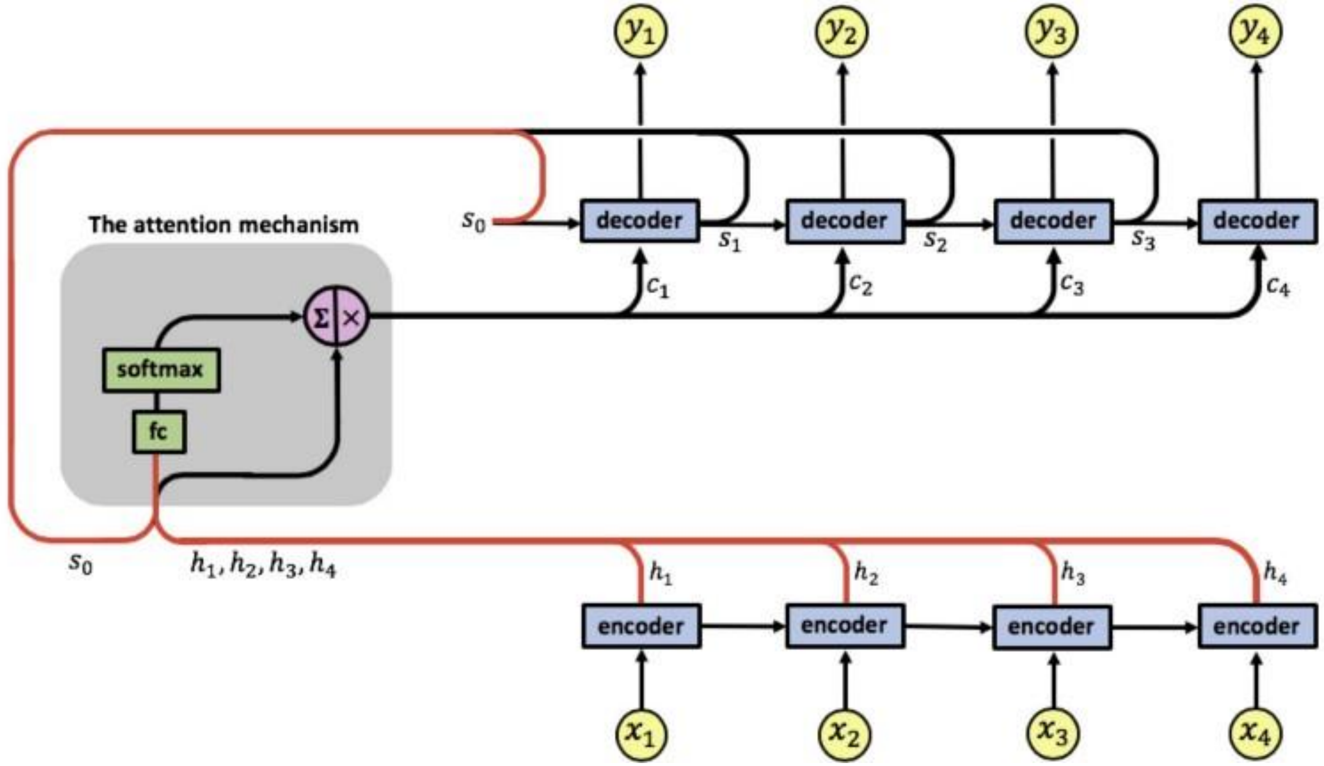
$$e_{ij} = \mathrm{fc}(s_{i-1}, h_j)$$



<source> https://medium.datadriveninvestor.com/attention-in-rnns-321fbcd64f05

Computing the attention weights and context vectors

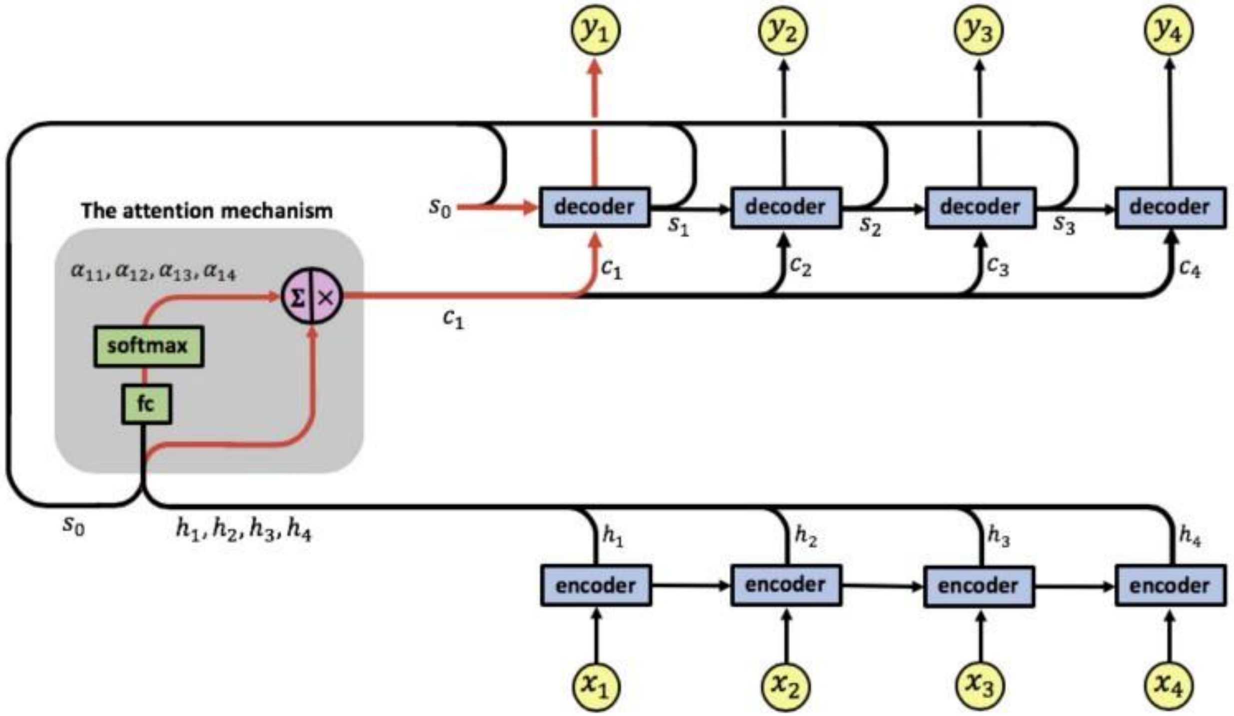
- the decoder is first involved by inputting its initial state vector s0
  - ✓ we have the first attention input sequence,
    - [s0, h1], [s0, h2], [s0, h3], [s0, h4]



<source> https://medium.datadriveninvestor.com/attention-in-rnns-321fbcd64f05

❖ The first set of attention weights α11, α12, α13, α14

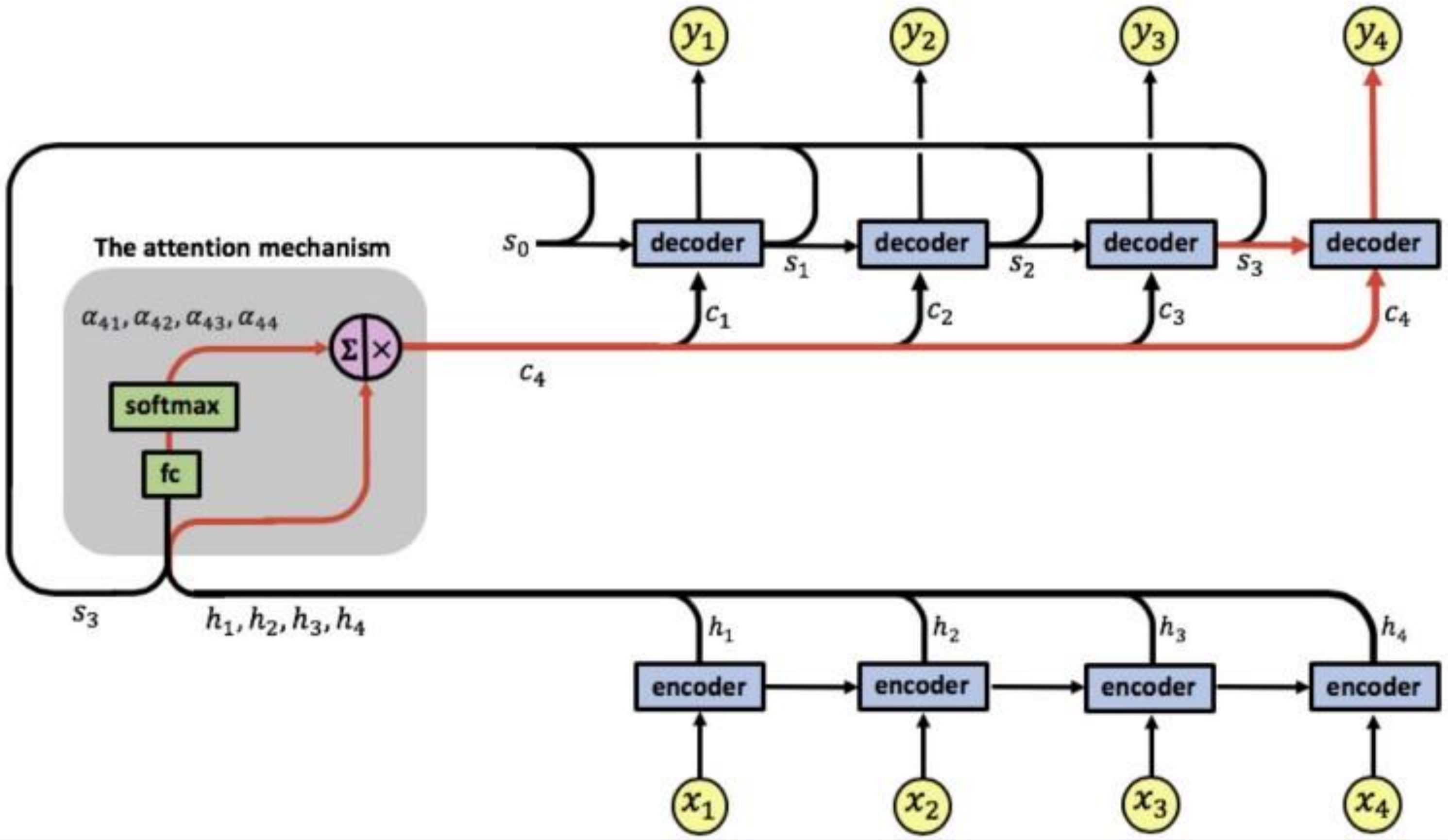
- ✓enabling the computation of the first context vector c1
- ✓The decoder now uses [s0,c1] and computes the first RNN output y1



<source> https://medium.datadriveninvestor.com/attention-in-rnns-321fbcd64f05

Computing the attention weights and context vectors

- It computes a fourth set of attention weights α41, α42, α43, α44 enabling the computation of the fourth context vector c4

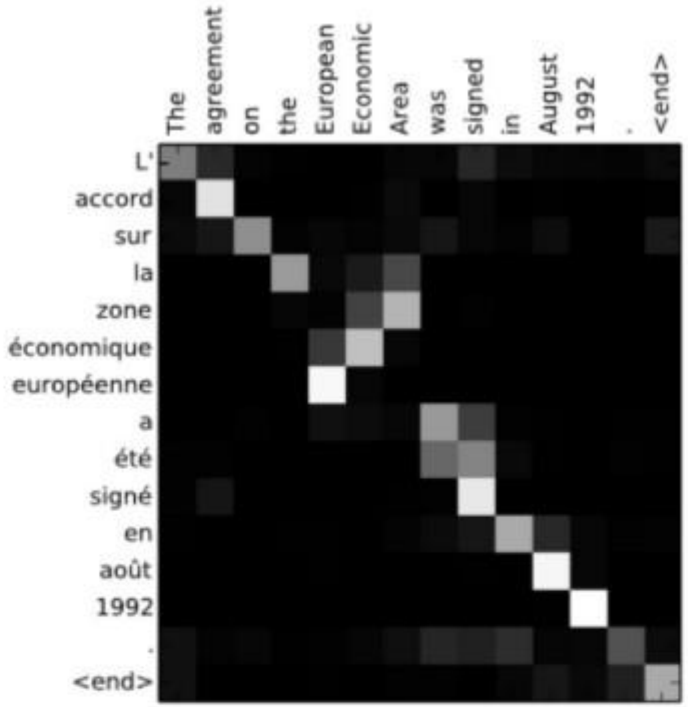


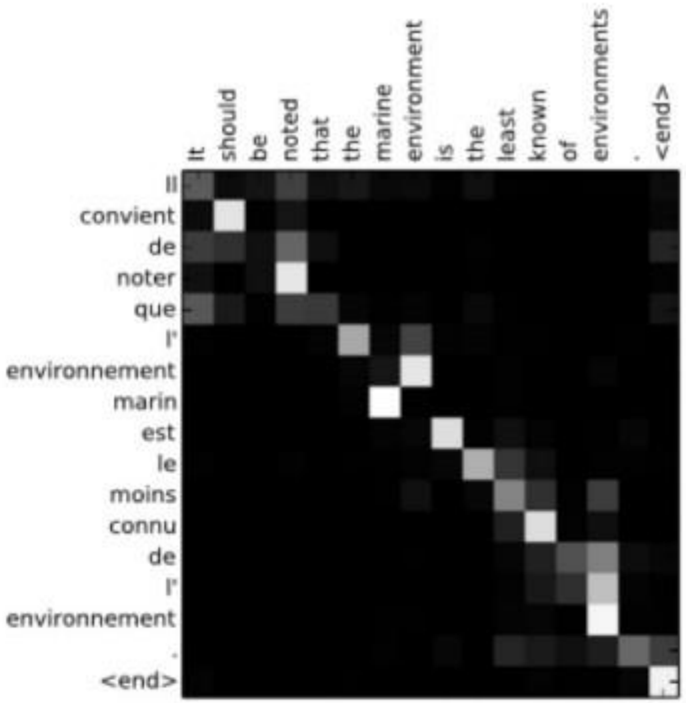
<source> https://medium.datadriveninvestor.com/attention-in-rnns-321fbcd64f05

An example task is English-French machine translation

✓ Each pixel shows the weight αij of the j-th source word and the i-th target word, in grayscale

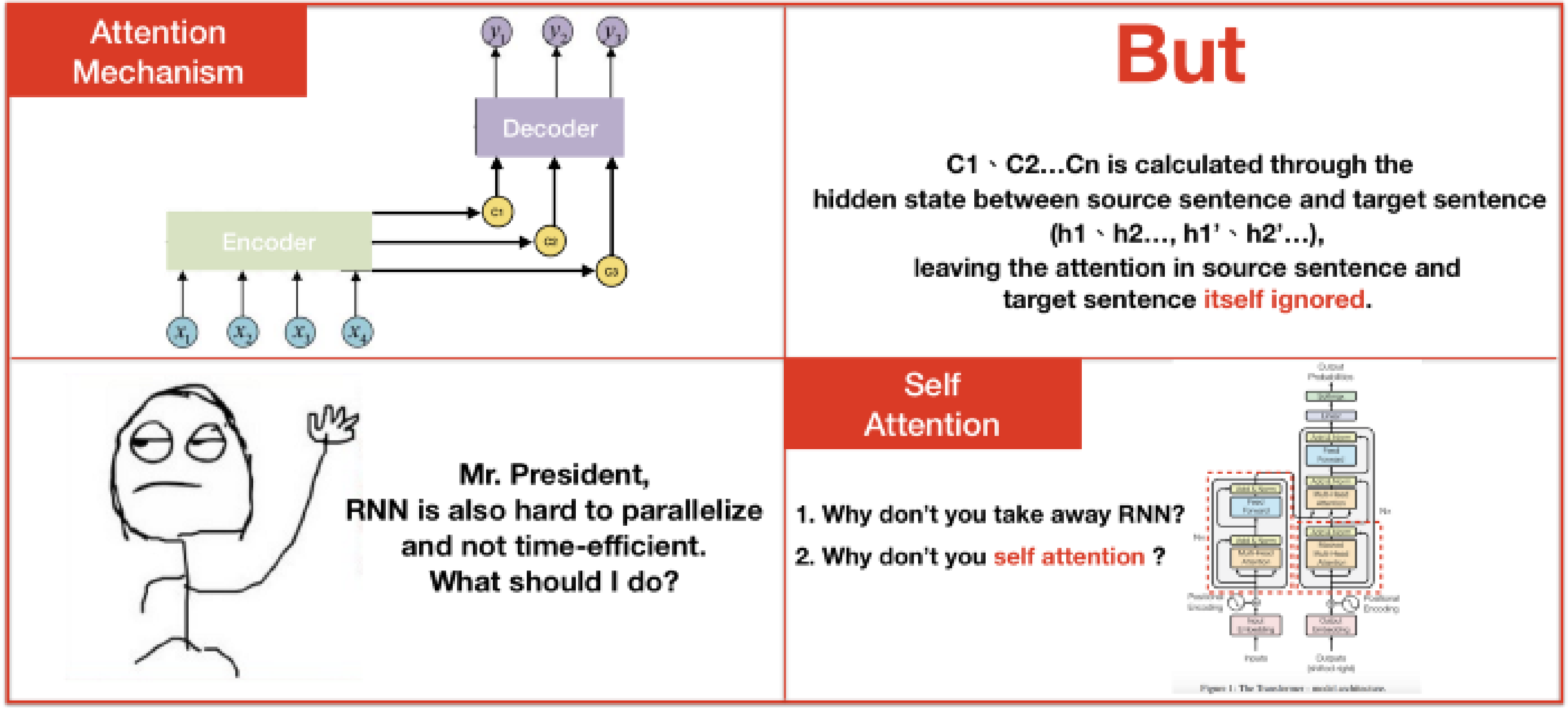
✓ The larger attention parameters (given by the white pixels) connect corresponding parts of the English and French sentences





<source> https://medium.datadriveninvestor.com/attention-in-rnns-321fbcd64f05

Attention Based Model



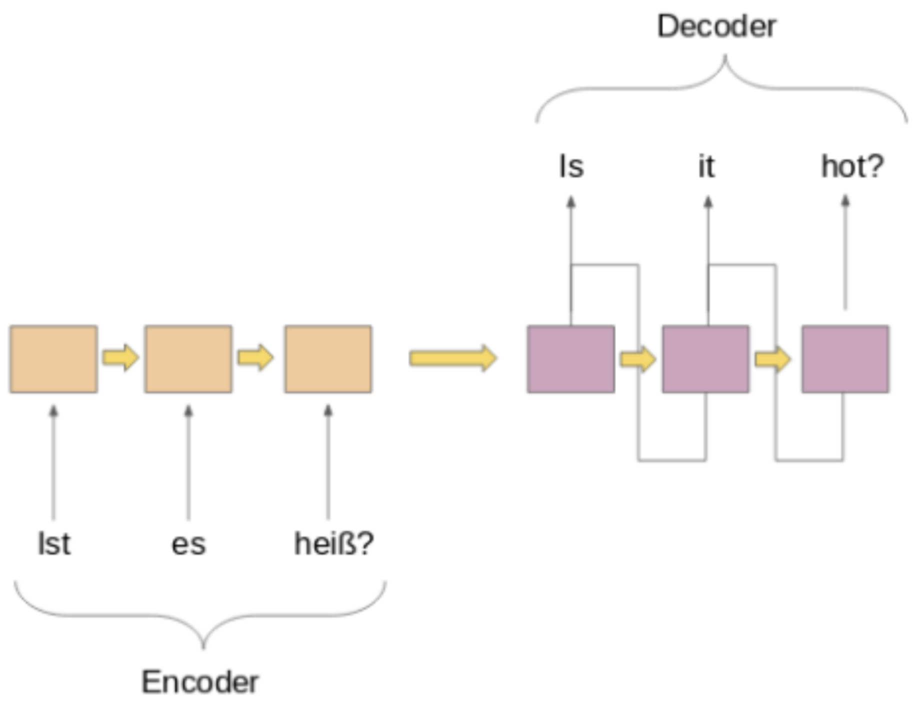
<source> https://bgg.medium.com/seq2seq-pay-attention-to-self-attention-part-1-d332e85e9aad

Comparing RNNs to Transformers

RNNs
(+) LSTMs work reasonably well for long sequences.
(-) Expects an ordered sequences of inputs
(-) Sequential computation: subsequent hidden states can only be computed after the previous ones are done.

Transformers:
(+) Good at long sequences. Each attention calculation looks at all inputs.
(+) Can operate over unordered sets or ordered sequences with positional encodings.
(+) Parallel computation: All alignment and attention scores for all inputs can be done in parallel.
(-) Requires a lot of memory: N x M alignment and attention scalers need to be calculated and stored for a single self-attention head. (but GPUs are getting bigger and better)

- Seq2Seq
  - ✓ https://blog.keras.io/a-ten-minute-introduction-to-sequence-to-sequence-learning-in-keras.html
- Seq2seq may basically have different lengths of the input sequence and the output sequence
- Corpus with two or more languages in parallel
  - ✓ http://www.manythings.org/anki
    - fra-eng.zip
    - kor-eng.zip



2022

Korea Institute of Science
and Technology Information

TRUST KISTI