Introduction to Attention

Danna Gurari
University of Colorado Boulder
Spring 2025

https://dannagurari.colorado.edu/course/neural-networks-and-deep-learning-spring-2025/

Review

- Last lecture:
  - Motivation: numeric representation of natural language
  - Tokenization: how to convert text into discrete units
  - Neural word embeddings: how to create dense representation
  - Programming tutorial

- Assignments (Canvas):
  - Problem set 3 grades are out
    - Review session will be held at 2pm today on Zoom
    - Email all regrade requests to our TA, Nick Cooper (a comment in Canvas is not sufficient)
  - Lab assignment 2 due in one week

- Questions?

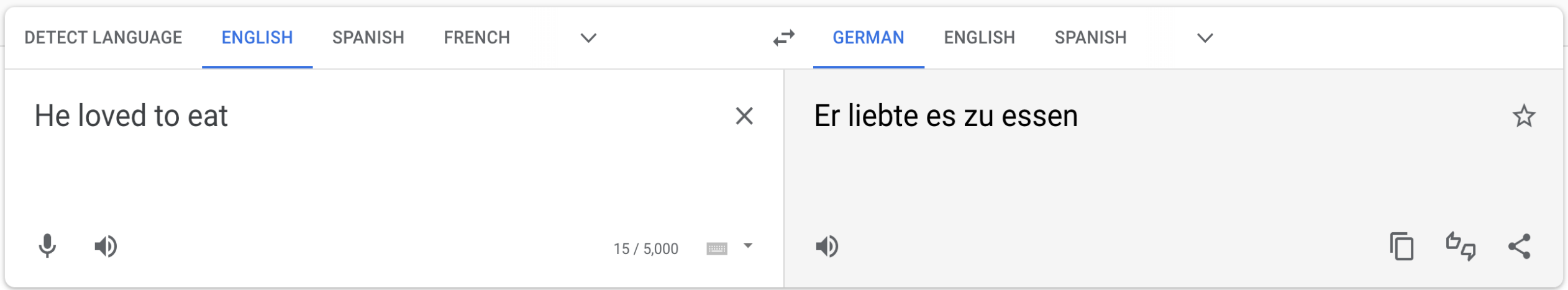
Today's Topics

- Motivation: machine neural translation for long sentences
- Decoder: attention
- Encoder
- Performance evaluation
- Final project: ways to find a partner

Today's Topics

- Motivation: machine neural translation for long sentences
- Decoder: attention
- Encoder
- Performance evaluation
- Final project: ways to find a partner

Task: Machine Translation



Which type of sequence problem is this: one-to-many, many-to-one, or many-to-many?

Pioneering Neural Network Approach

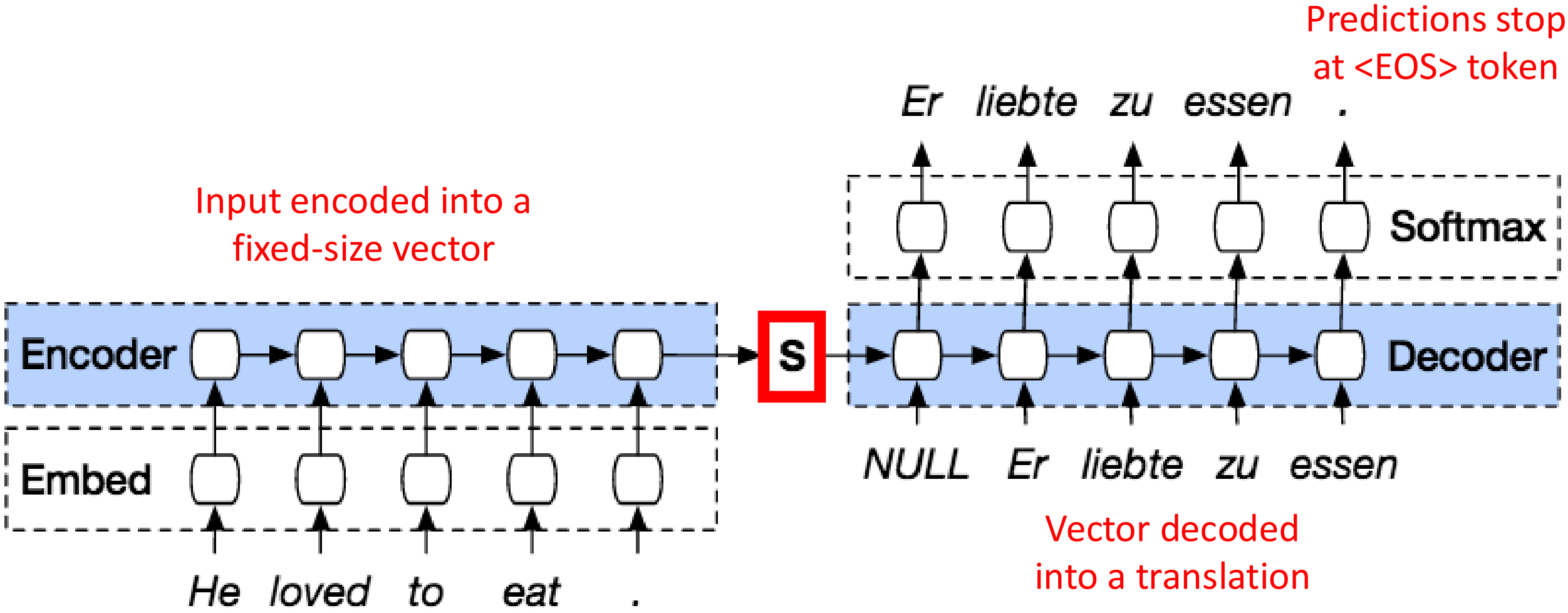


Image source: https://smerity.com/articles/2016/google_nmt_arch.html

seq2seq: Sutskever et al. Sequence to Sequence Learning with Neural Networks. Neurips 2014.

Pioneering Neural Network Approach

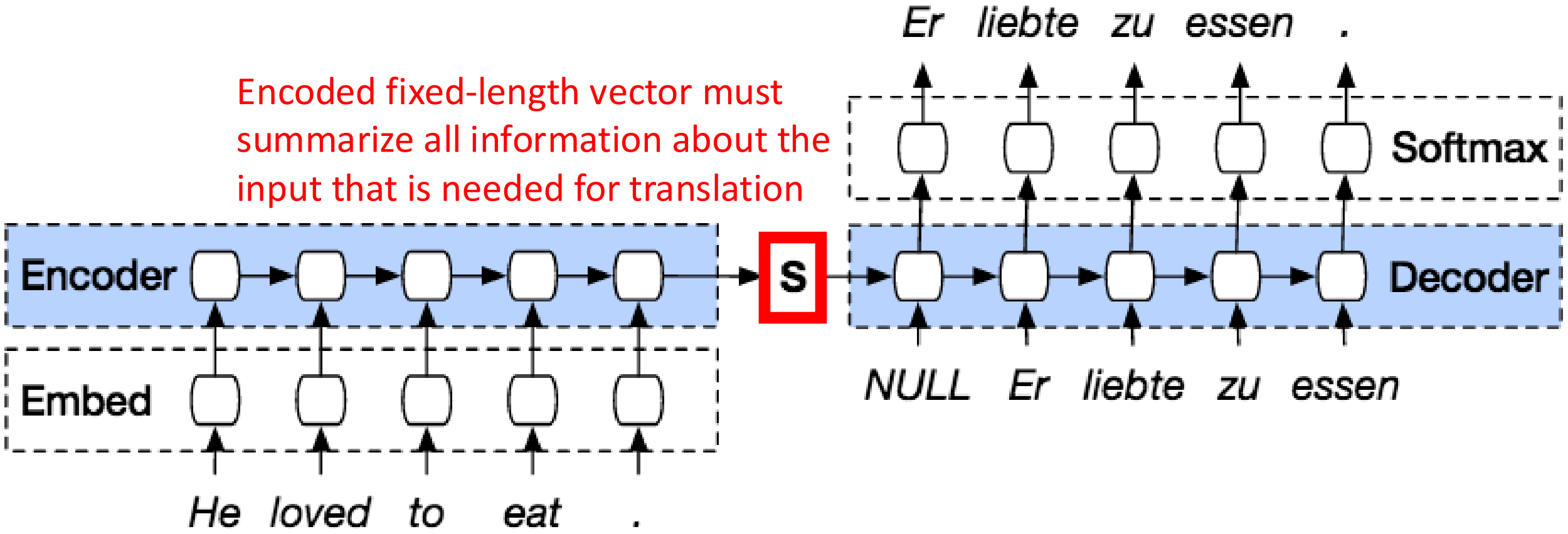
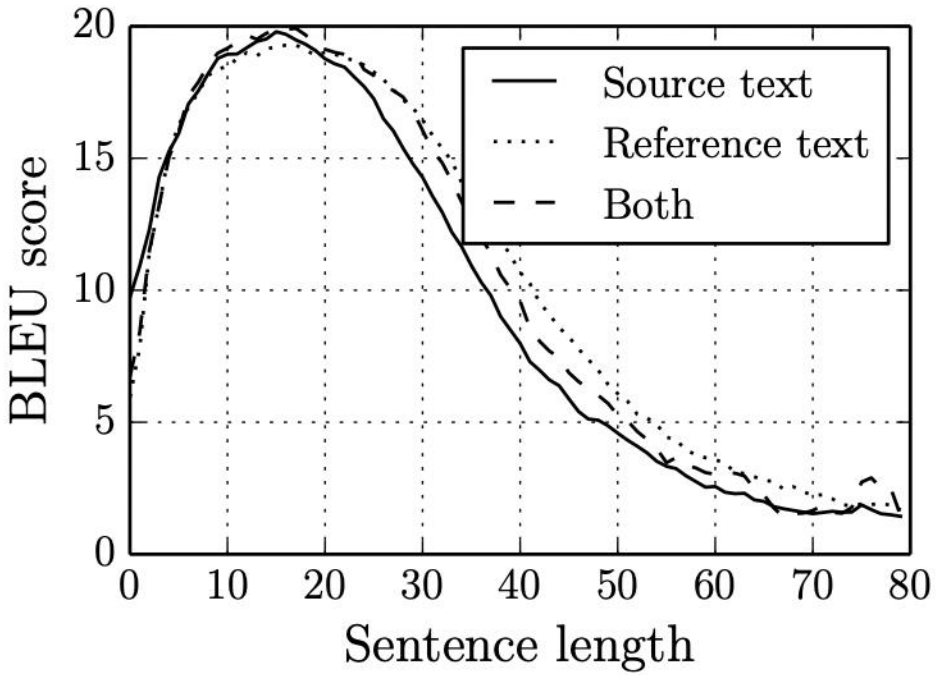


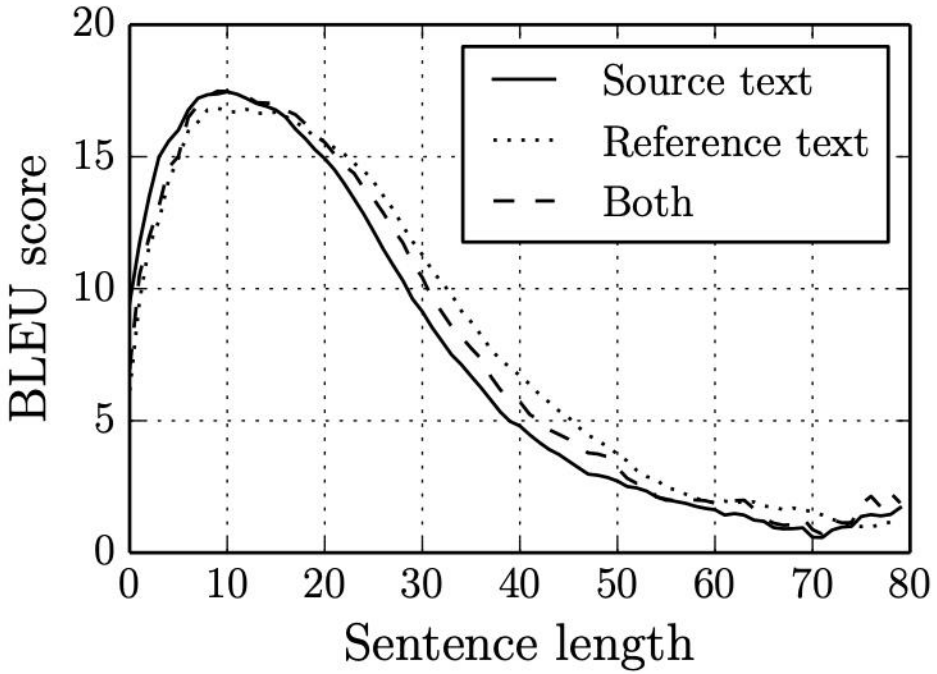
Image source: https://smerity.com/articles/2016/google_nmt_arch.html

seq2seq: Sutskever et al. Sequence to Sequence Learning with Neural Networks. Neurips 2014.

Analysis of Two Models

(larger scores are better)



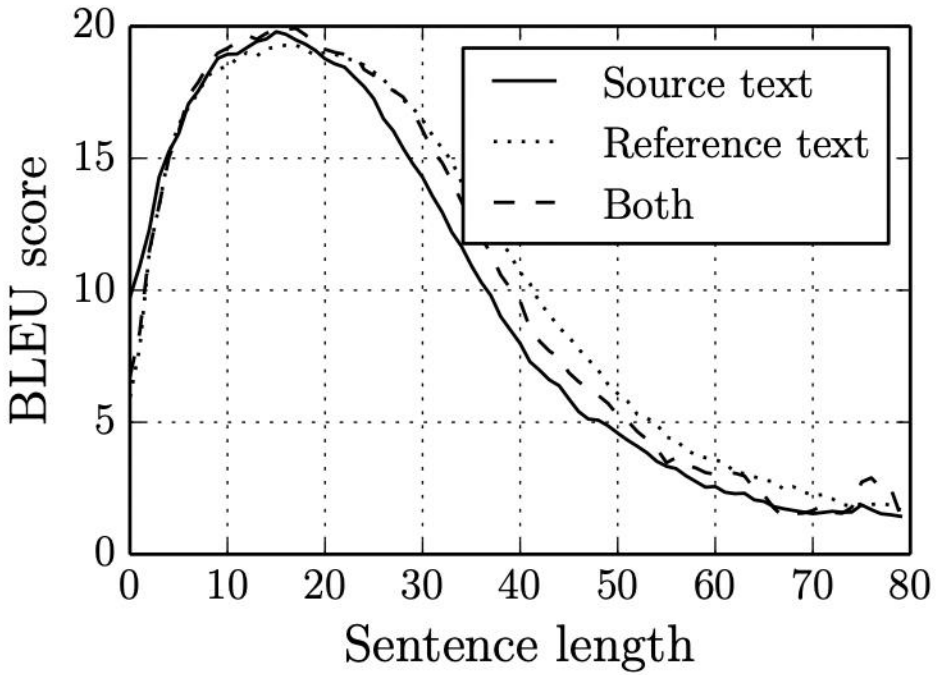


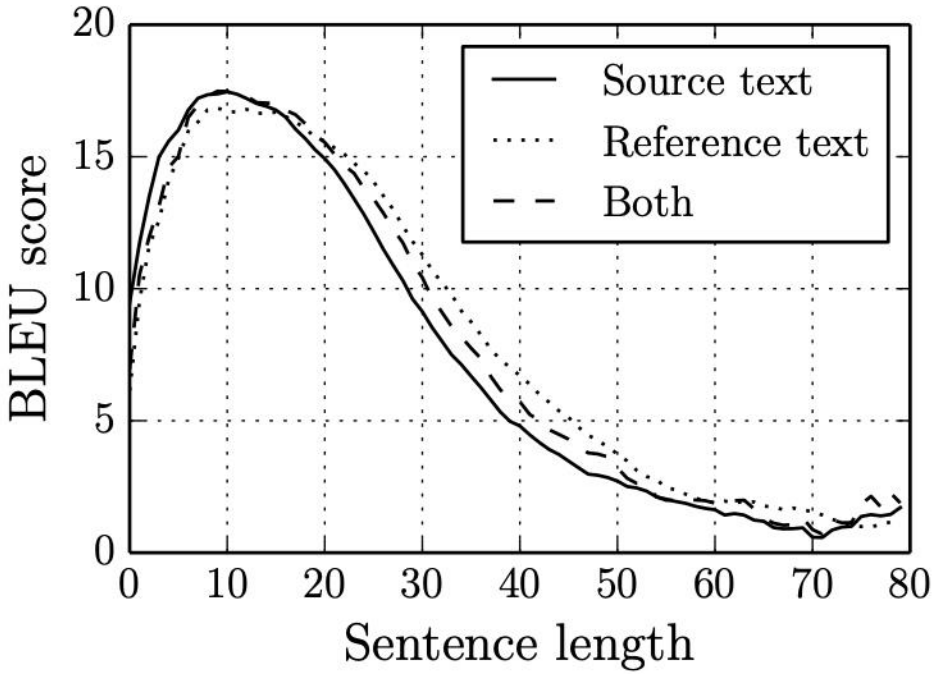
What performance trend is observed for inputs (source) and outputs (reference) as the number of words in each sentence grows?

Cho et al. On the Properties of Neural Machine Translation: Encoder–Decoder Approaches. SSST 2014

Analysis of Two Models

(larger scores are better)





Performance drops for longer sentences!

Cho et al. On the Properties of Neural Machine Translation: Encoder–Decoder Approaches. SSST 2014

Problem: Performance Drops As Sentence Length Grows

Hypothesis: insufficient capacity in fixed-length vector for all relevant information in long sentences

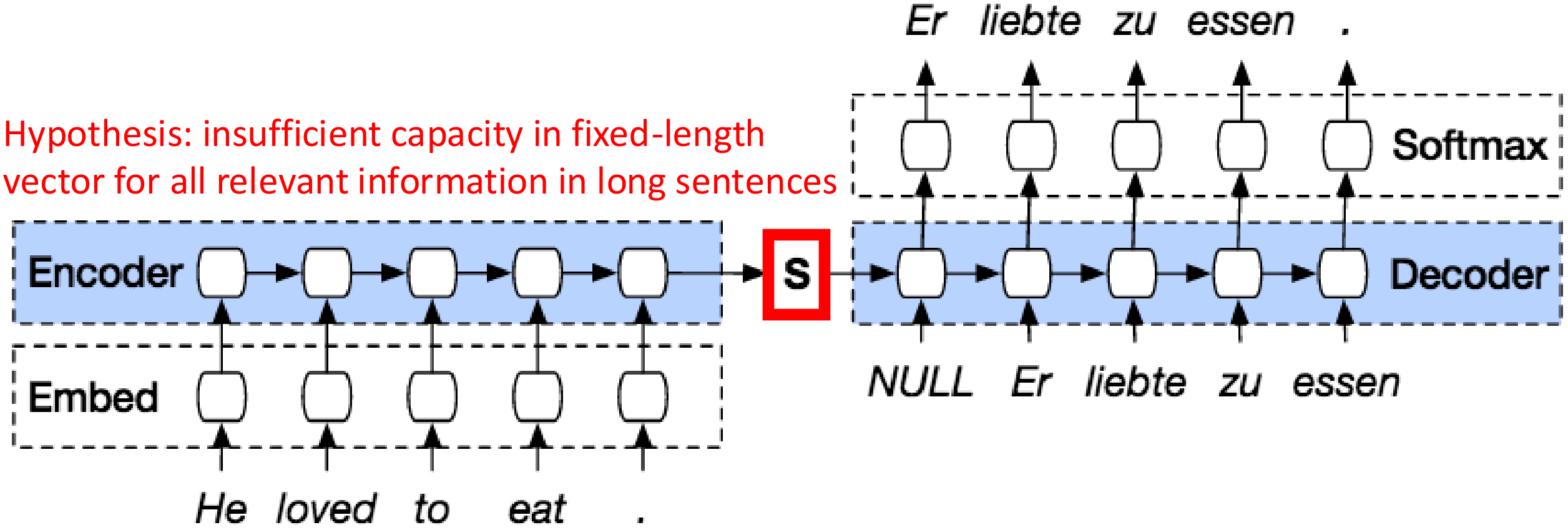


Image source: https://smerity.com/articles/2016/google_nmt_arch.html

seq2seq: Sutskever et al. Sequence to Sequence Learning with Neural Networks. Neurips 2014.

Idea to Preserve Performance for Long Sentences: **Attention**

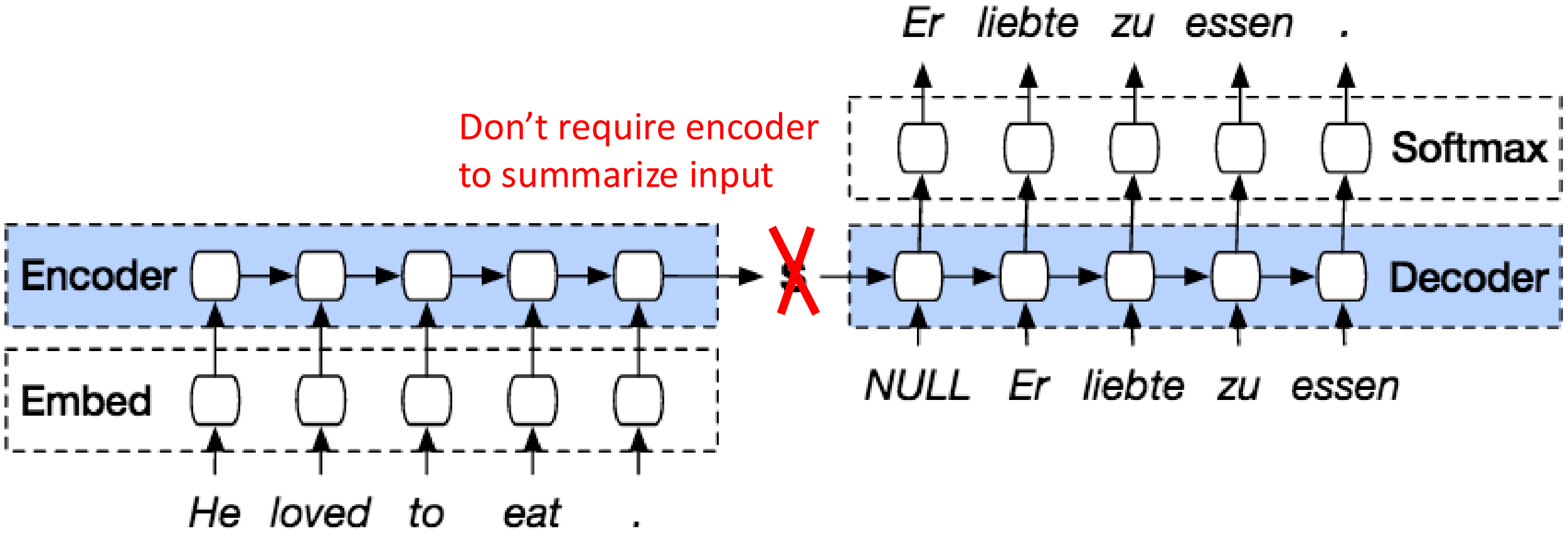


Image source: https://smerity.com/articles/2016/google_nmt_arch.html

Idea to Preserve Performance for Long Sentences: **Attention**

Instead, pass **all** hidden states of the encoder to the decoder

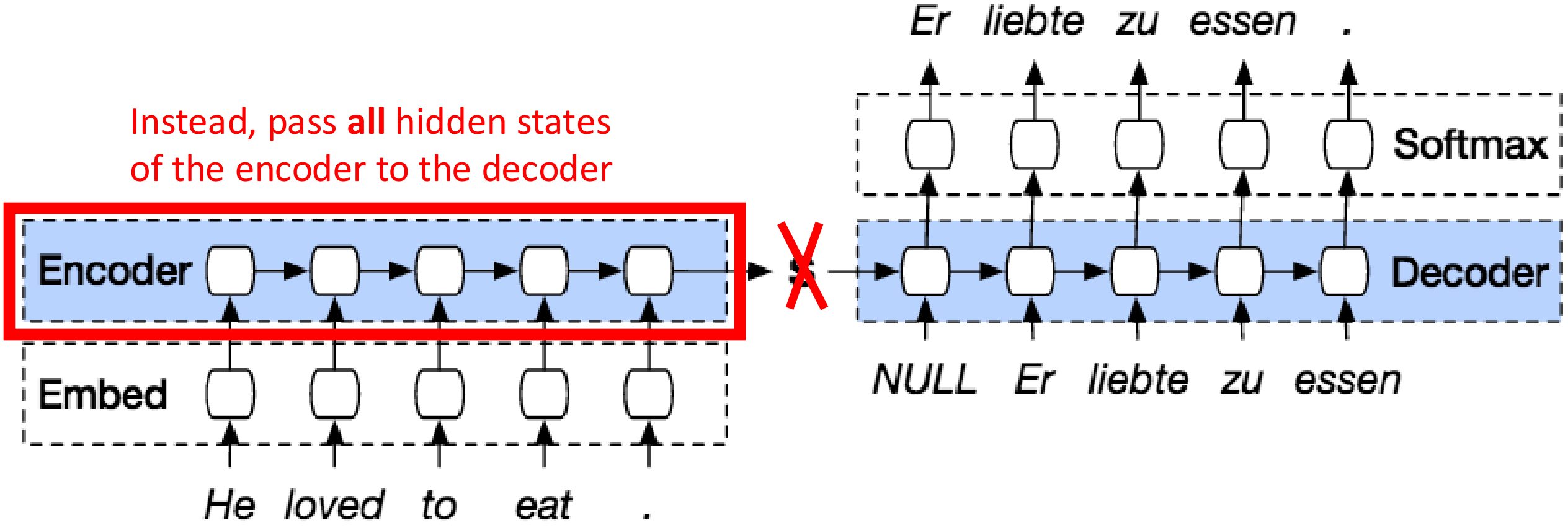


Image source: https://smerity.com/articles/2016/google_nmt_arch.html

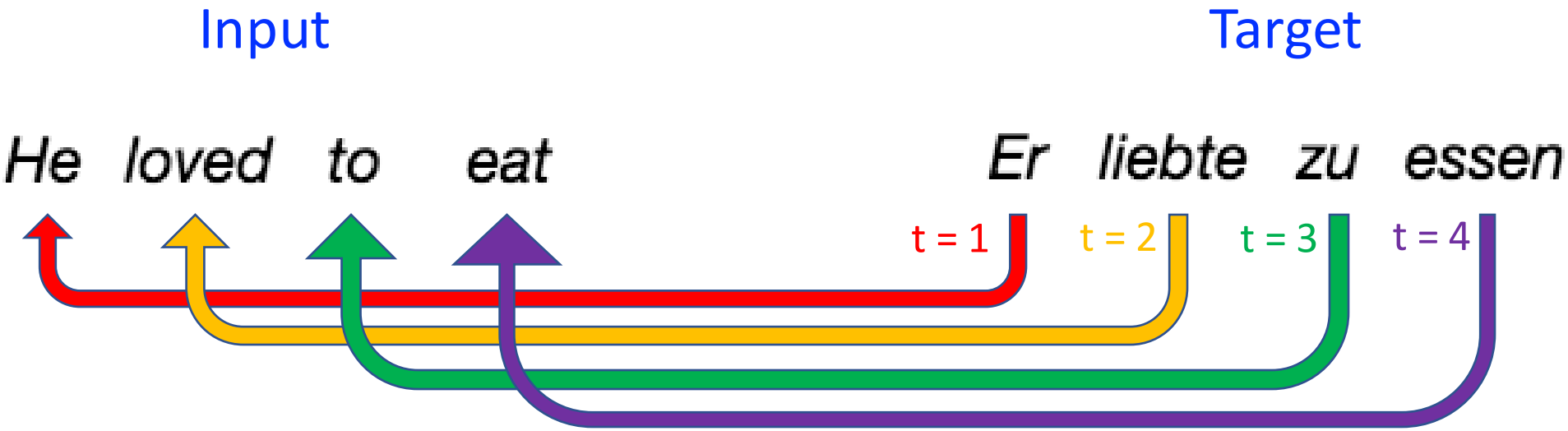
Idea to Preserve Performance for Long Sentences: **Attention**

Decoder decides which inputs are needed for prediction at each time step; e.g., “hard attention” focuses on one input

Note: while word order between the input and target align in this example, it can differ

Idea to Preserve Performance for Long Sentences: **Attention**

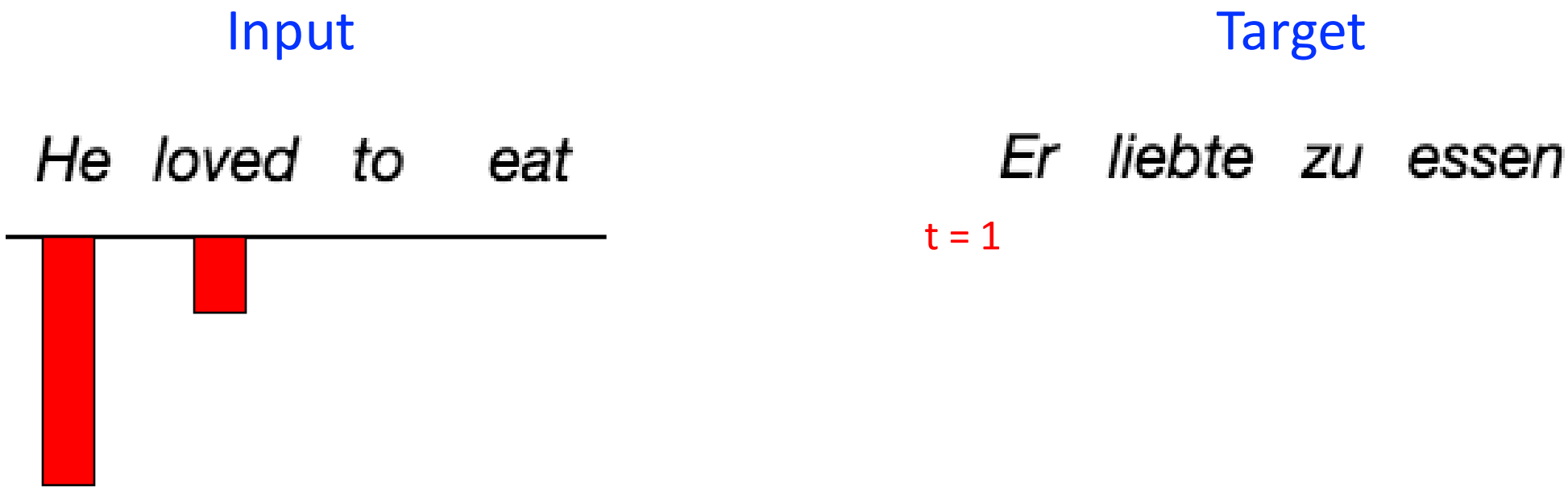
Decoder decides which inputs are needed for prediction at each time step; e.g., “hard attention” focuses on one input



Limitations: a target word relies on information about one input word and “hard attention” is not differentiable

Idea to Preserve Performance for Long Sentences: **Attention**

Decoder decides which inputs are needed for prediction at each time step; e.g., “soft attention” uses a weighted combination of the input



Idea to Preserve Performance for Long Sentences: **Attention**

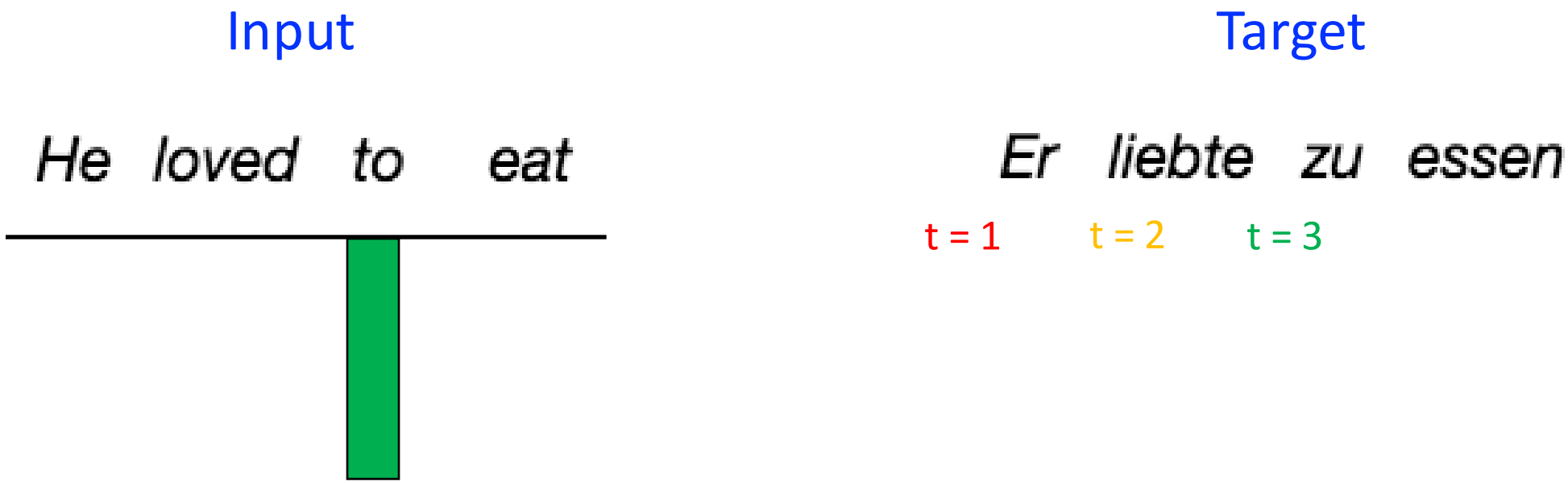
Decoder decides which inputs are needed for prediction at each time step; e.g., “soft attention” uses a weighted combination of the input

Input

Target

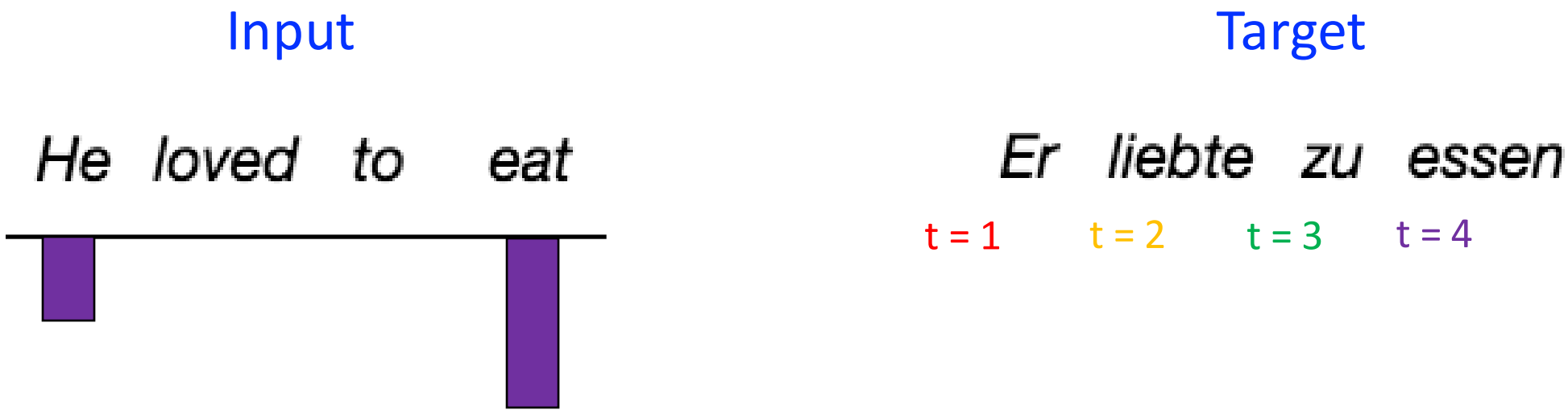
Idea to Preserve Performance for Long Sentences: **Attention**

Decoder decides which inputs are needed for prediction at each time step; e.g., “soft attention” uses a weighted combination of the input



Idea to Preserve Performance for Long Sentences: **Attention**

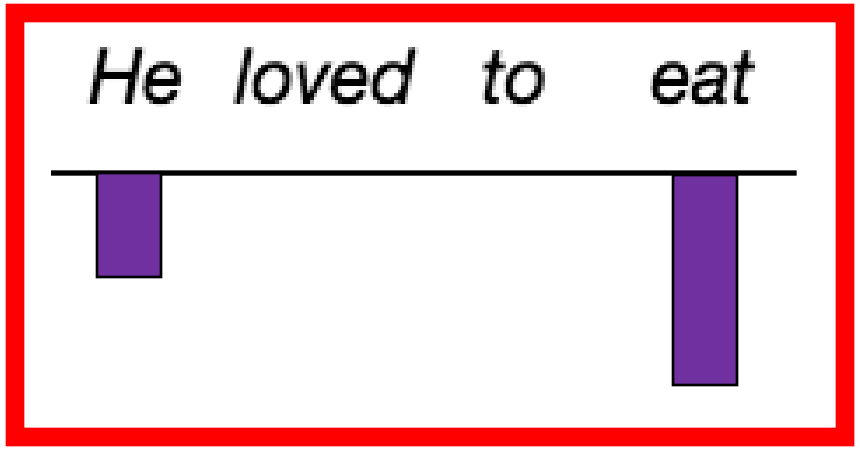
Decoder decides which inputs are needed for prediction at each time step; e.g., “soft attention” uses a weighted combination of the input



“Soft” Attention: Challenge

Decoder decides which inputs are needed for prediction at each time step; e.g., “soft attention” uses a weighted combination of the input

Input



Target

Er liebte zu essen

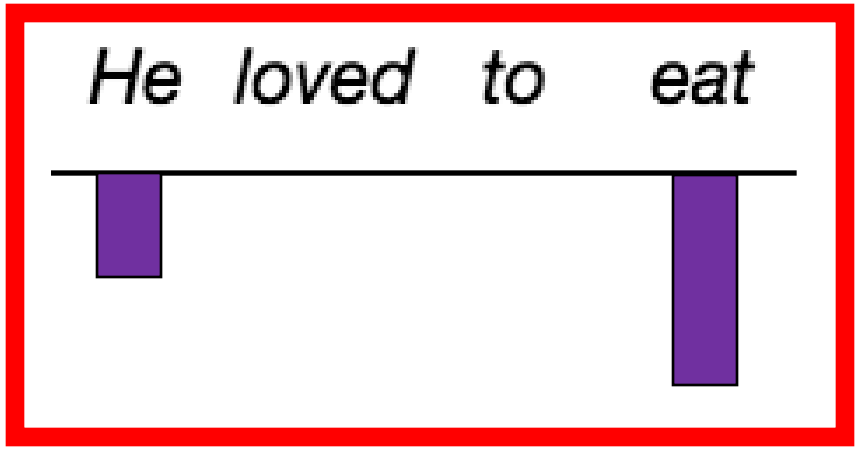
t = 1 t = 2 t = 3 t = 4

How should weights be chosen for each input?

“Soft” Attention: Challenge

Decoder decides which inputs are needed for prediction at each time step; e.g., “soft attention” uses a weighted combination of the input

Input



Target

Er liebte zu essen

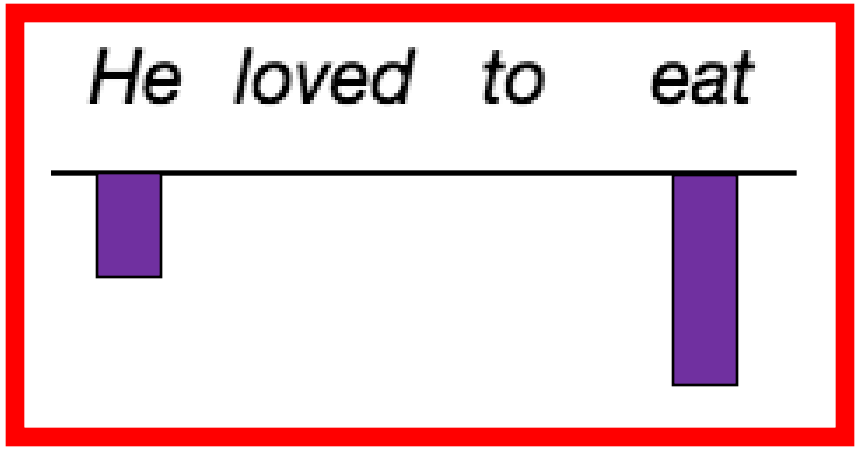
t = 1 t = 2 t = 3 t = 4

Collect manual annotations and then have loss function push predicted parameters to match ground truth parameters… but this is impractical

“Soft” Attention: Challenge

Decoder decides which inputs are needed for prediction at each time step; e.g., “soft attention” uses a weighted combination of the input

Input



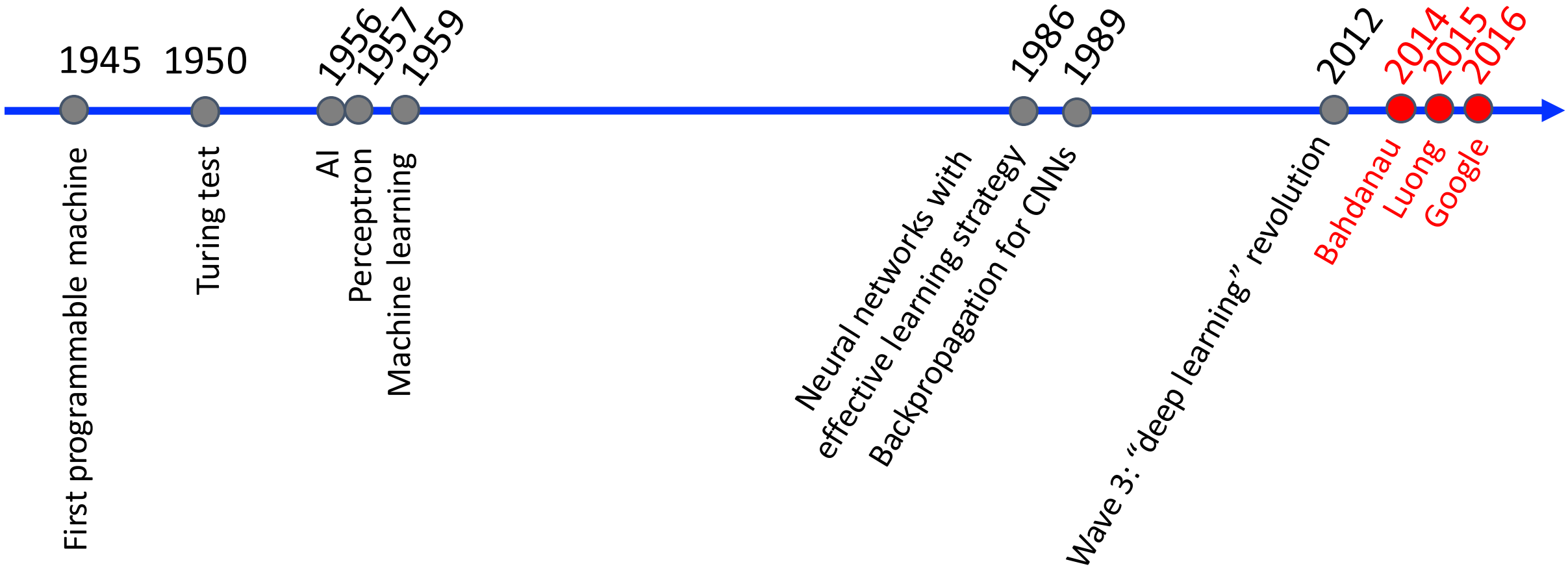
Instead, have the model learn how to weight each input!

Target

Er liebte zu essen

t = 1 t = 2 t = 3 t = 4

Popular Solutions

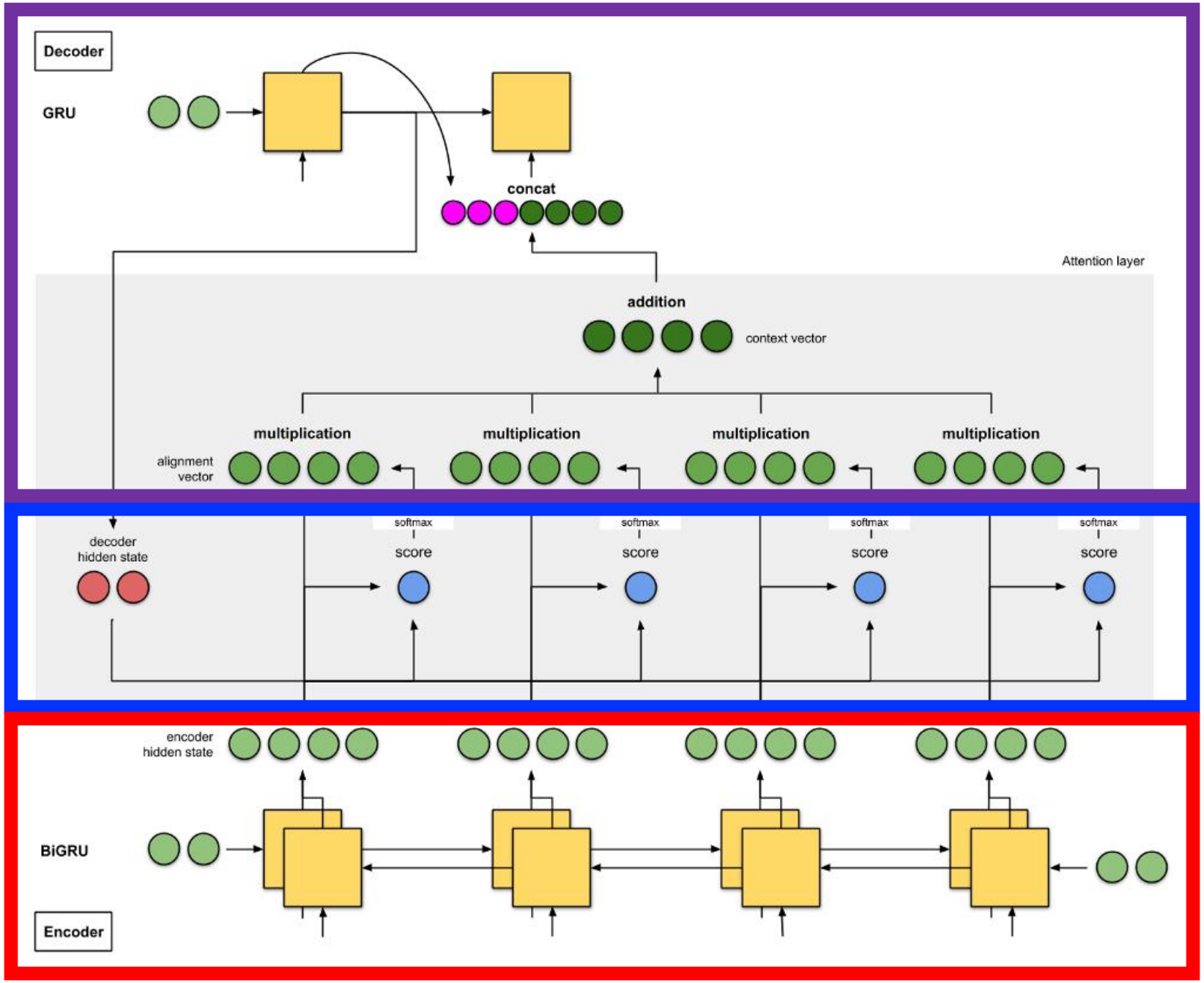


A Solution

3. At each decoder time step, a prediction is made based on the weighted sum of the inputs

2. At each decoder time step, attention weights are computed that determine each input's relevance for the prediction

1. Encoder produces hidden state for every input



https://towardsdatascience.com/attn-illustrated-attention-5ec4ad276ee3

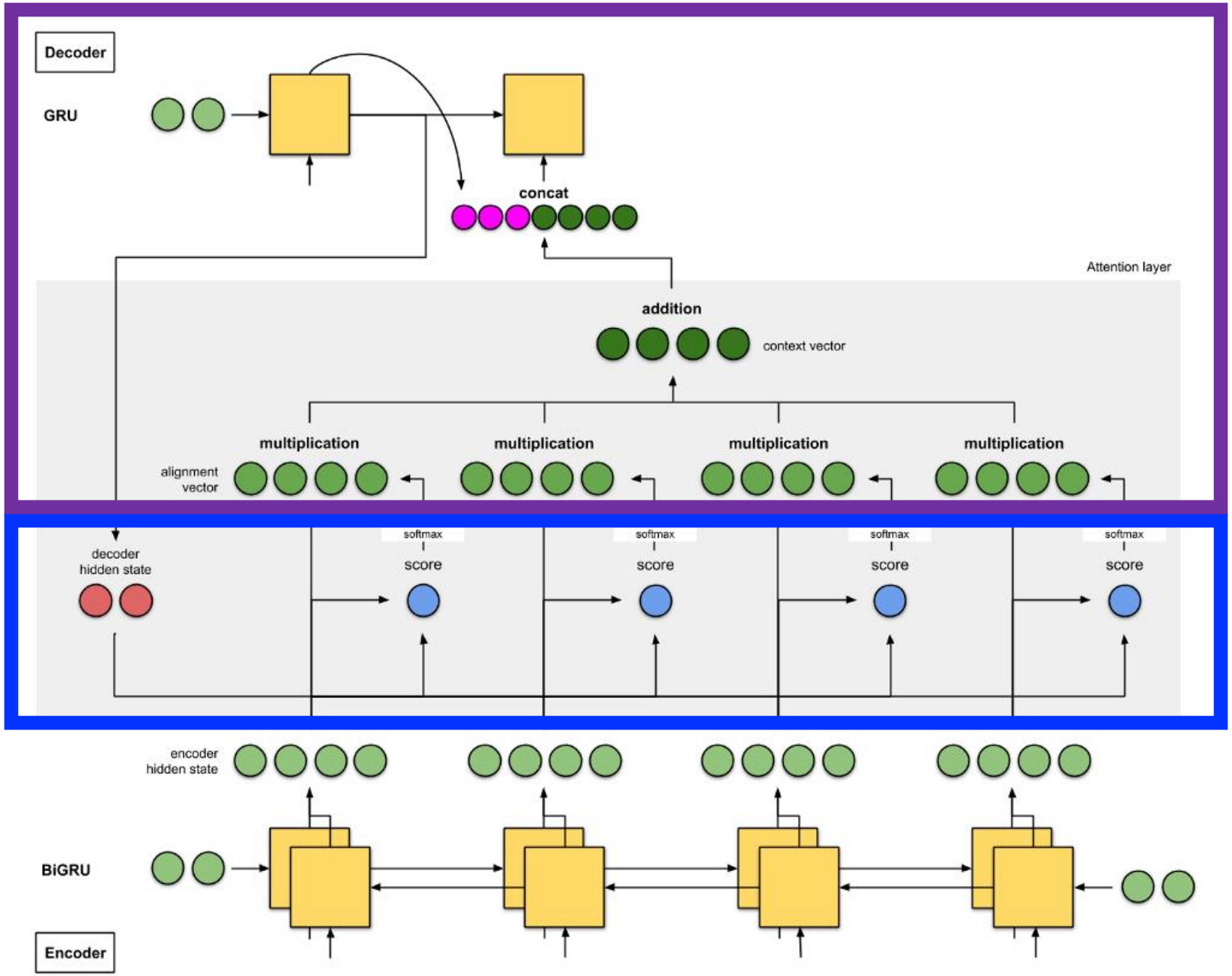
Today's Topics

- Motivation: machine neural translation for long sentences
- Decoder: attention
- Encoder
- Performance evaluation
- Final project: ways to find a partner

Solution

3. At each decoder time step, a prediction is made based on the weighted sum of the inputs

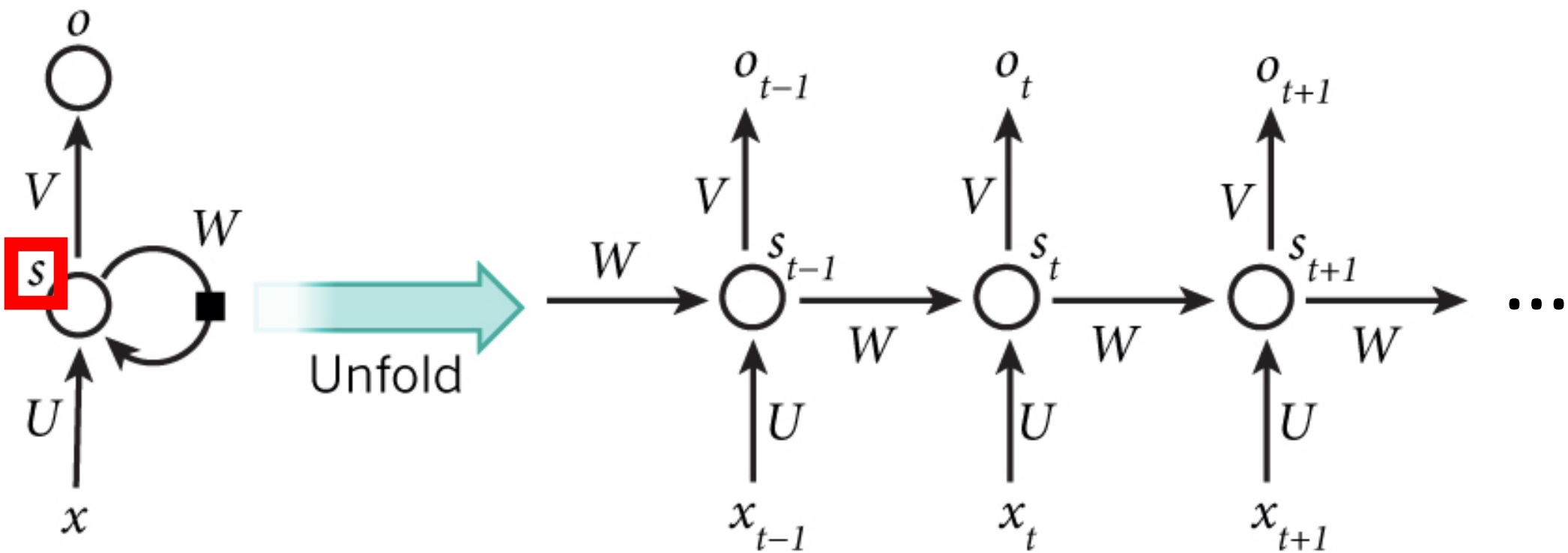
2. At each decoder time step, attention weights are computed that determine each input's relevance for the prediction



https://towardsdatascience.com/attn-illustrated-attention-5ec4ad276ee3

Recall: GRU is a Type of RNN

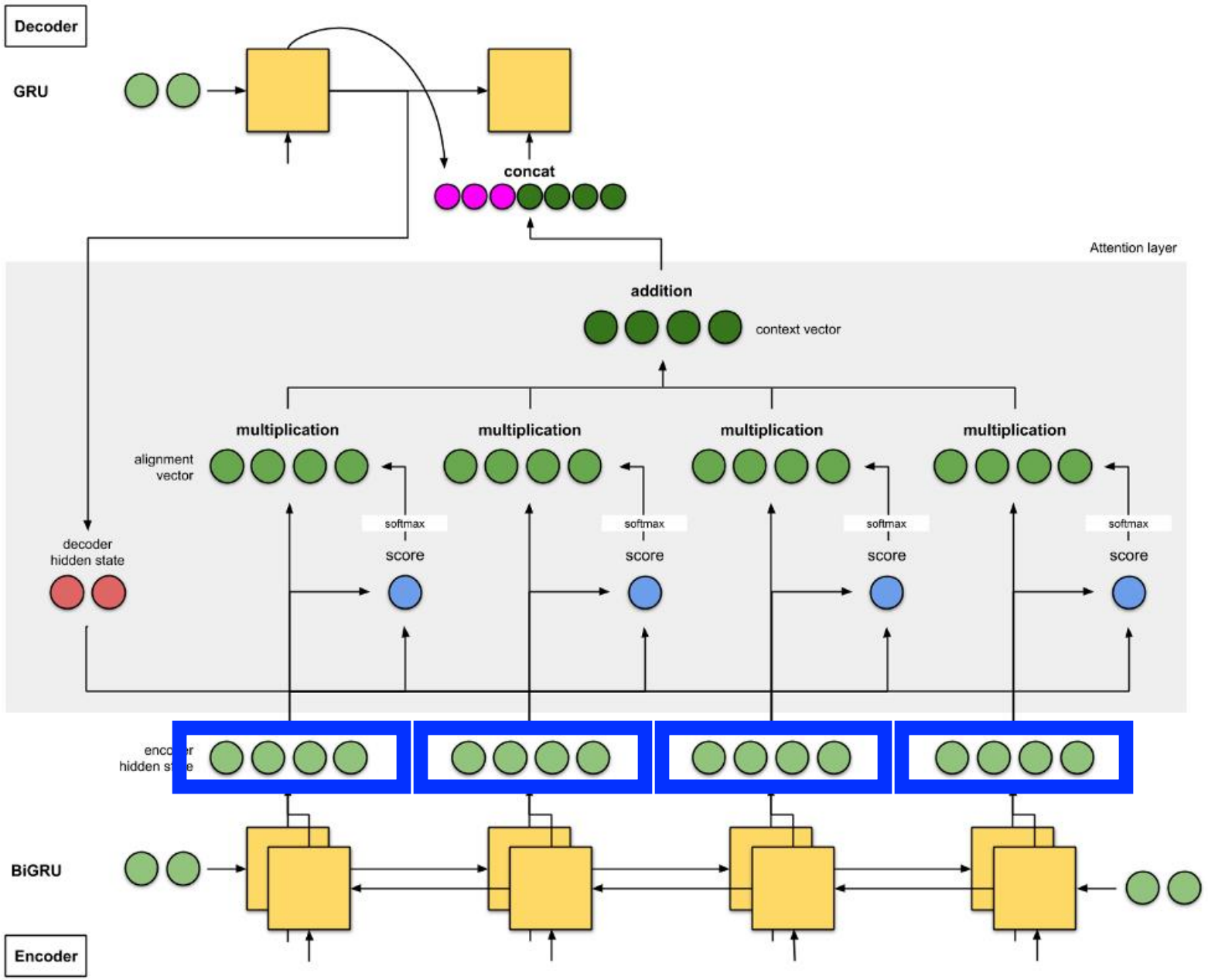
- Its hidden state captures information about the past



http://www.wildml.com/2015/09/recurrent-neural-networks-tutorial-part-1-introduction-to-rnns/

Measuring Each Input's Relevance on the Prediction

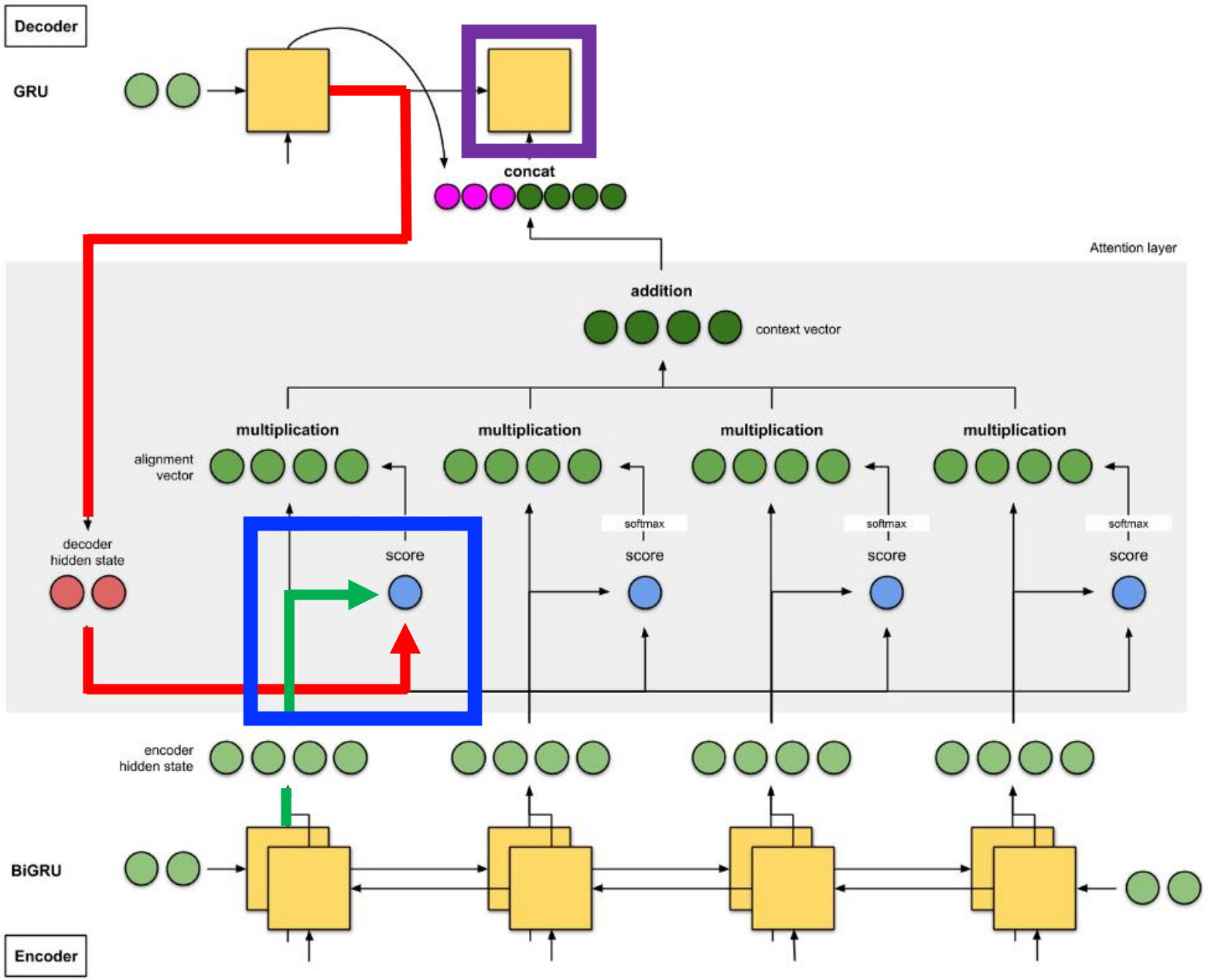
How many input are considered by the decoder in this example?



https://towardsdatascience.com/attn-illustrated-attention-5ec4ad276ee3

Measuring Each Input's Relevance on the Prediction

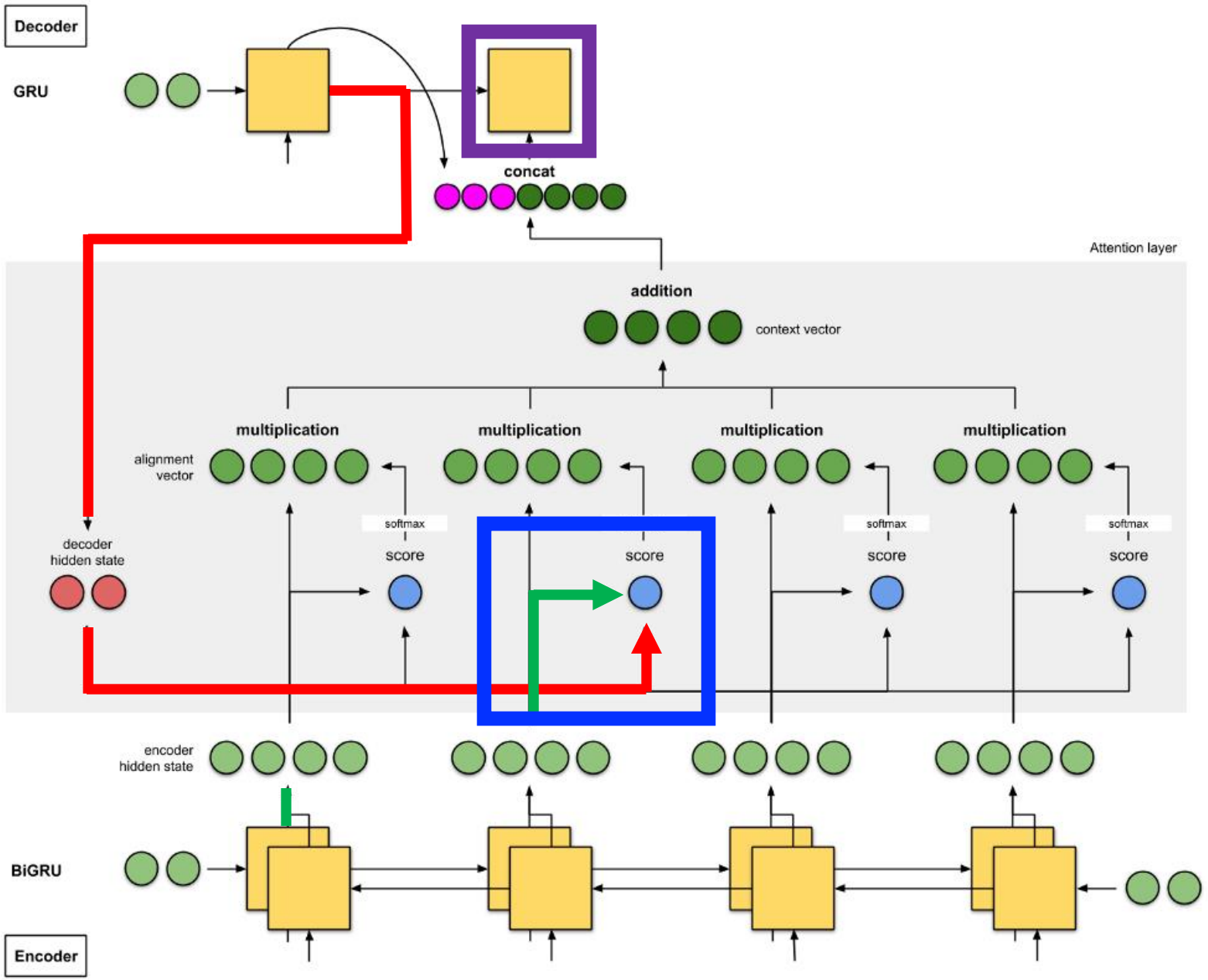
At each decoder time step, the compatibility between the decoder's hidden state and each input's hidden state is computed to decide each input's score at the time step



https://towardsdatascience.com/attn-illustrated-attention-5ec4ad276ee3

Measuring Each Input's Relevance on the Prediction

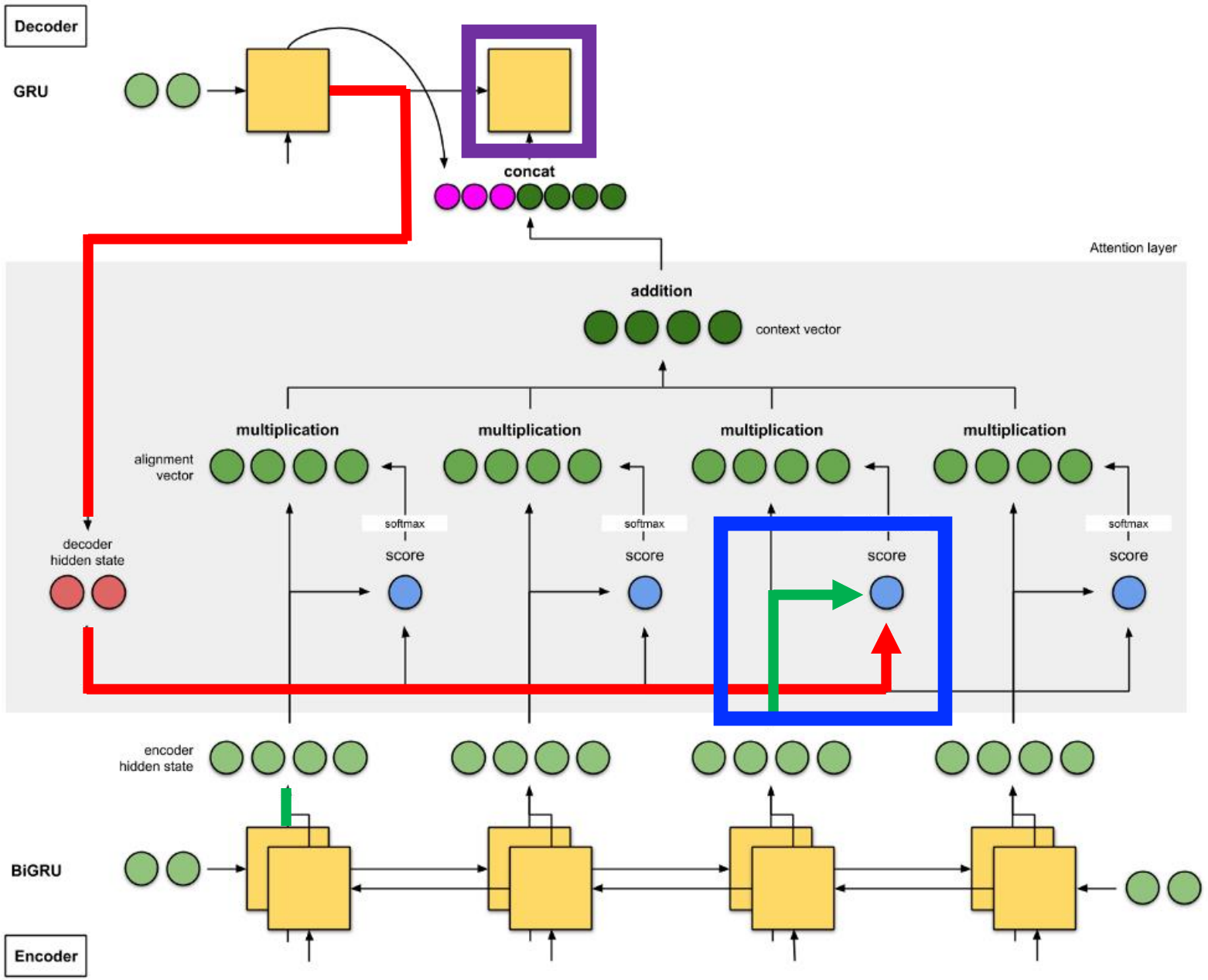
At each decoder time step, the compatibility between the decoder's hidden state and each input's hidden state is computed to decide each input's score at the time step



https://towardsdatascience.com/attn-illustrated-attention-5ec4ad276ee3

Measuring Each Input's Relevance on the Prediction

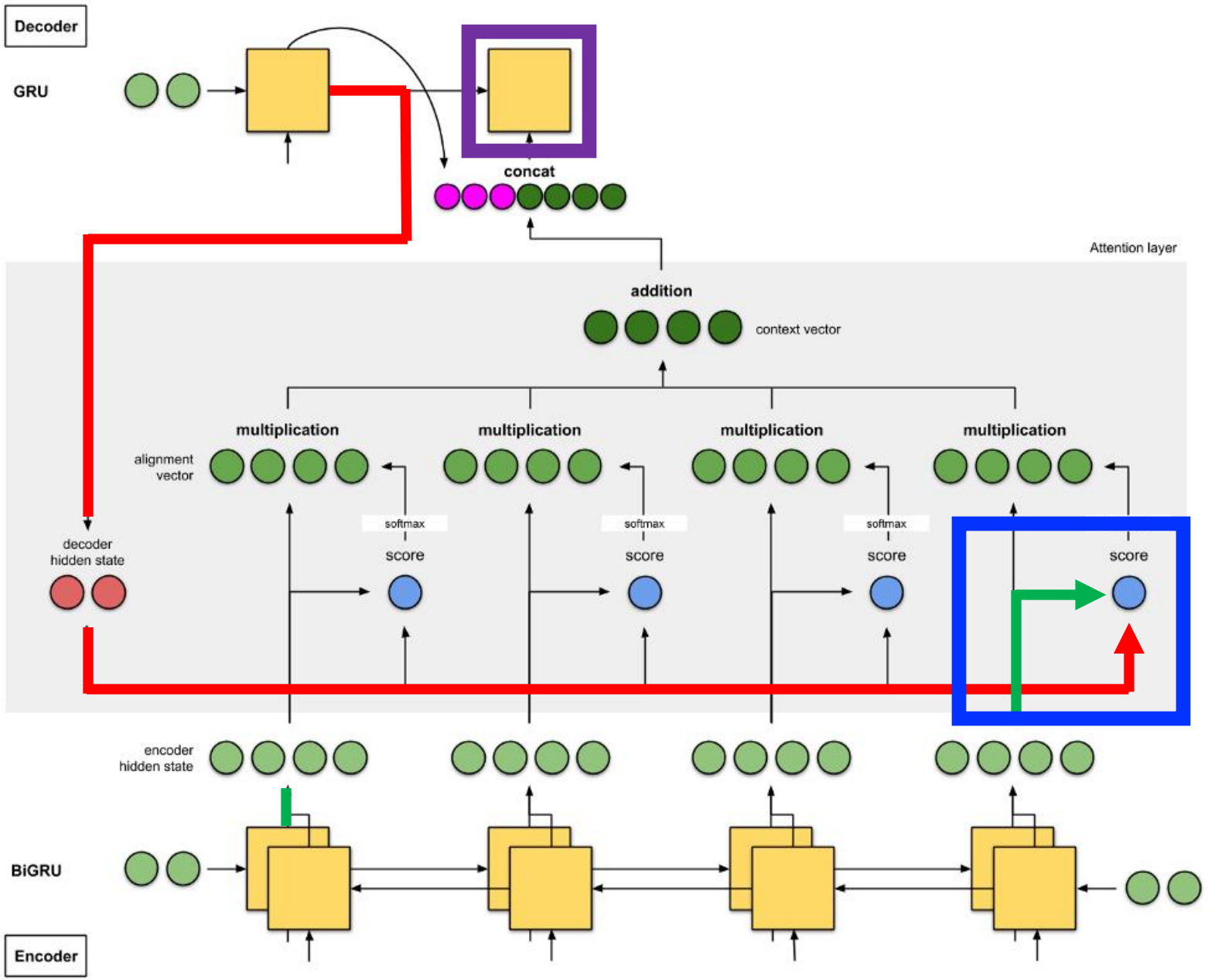
At each decoder time step, the compatibility between the decoder's hidden state and each input's hidden state is computed to decide each input's score at the time step



https://towardsdatascience.com/attn-illustrated-attention-5ec4ad276ee3

Measuring Each Input's Relevance on the Prediction

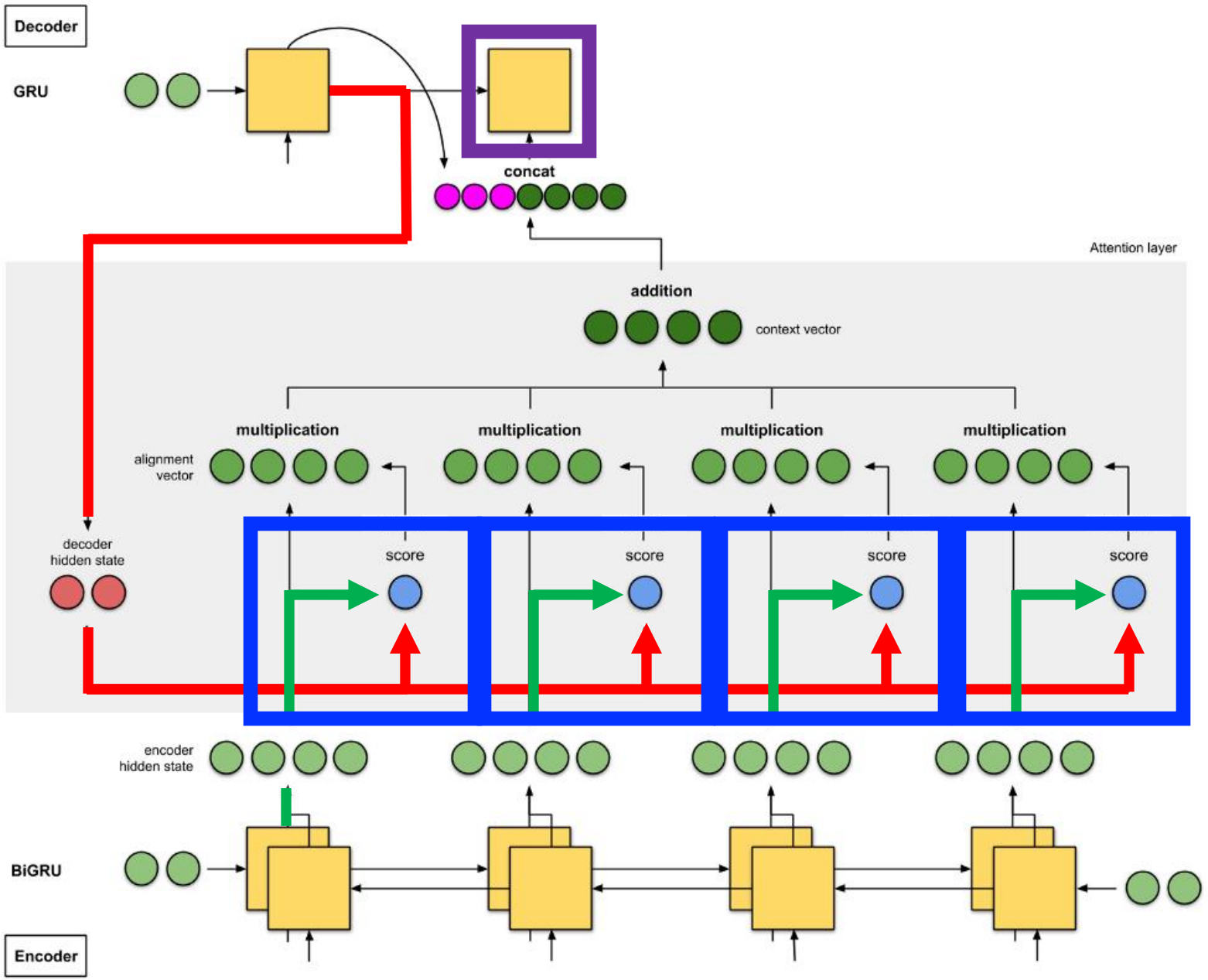
At each decoder time step, the compatibility between the decoder's hidden state and each input's hidden state is computed to decide each input's score at the time step



https://towardsdatascience.com/attn-illustrated-attention-5ec4ad276ee3

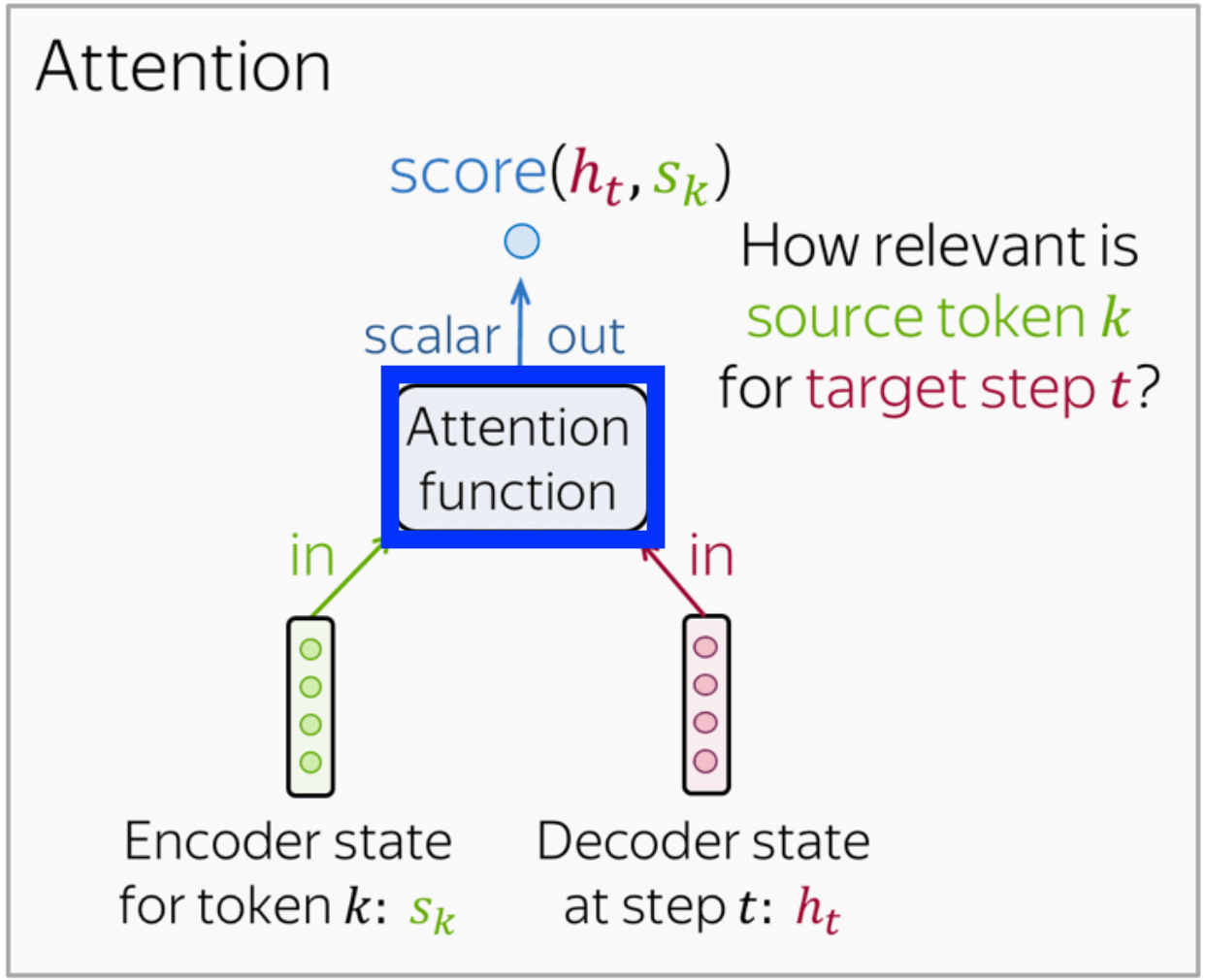
Measuring Each Input's Relevance on the Prediction

How to measure compatibility between hidden states of the decoder and input?



https://towardsdatascience.com/attn-illustrated-attention-5ec4ad276ee3

Compatibility Measures

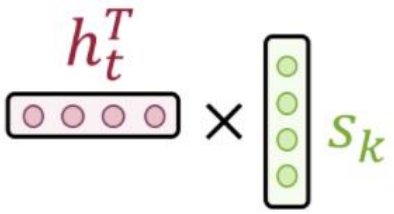


https://lena-voita.github.io/nlp_course/seq2seq_and_attention.html

Compatibility Measures

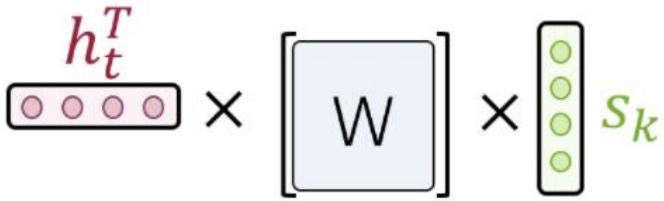
- Many options (function should be differentiable)

Dot-product



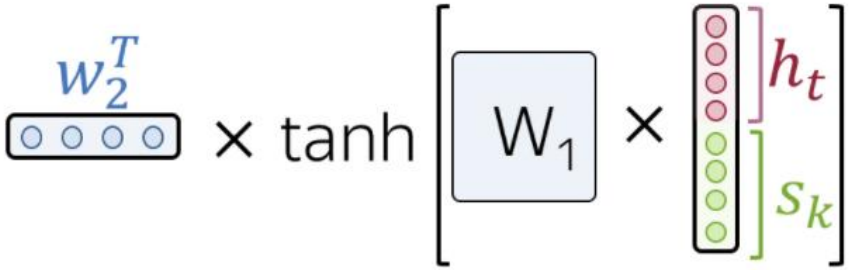
$$\text{score}(h_t, s_k) = h_t^T \, s_k$$

Bilinear



$$\text{score}(h_t, s_k) = h_t^T \, W \, s_k$$

Multi-Layer Perceptron



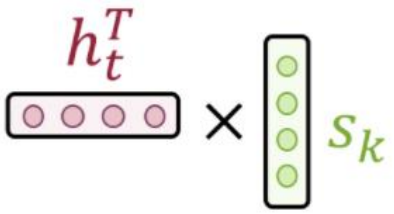
$$\text{score}(h_t, s_k) = w_2^T \cdot \tanh(W_1[h_t, s_k])$$

https://lena-voita.github.io/nlp_course/seq2seq_and_attention.html

Compatibility Measures

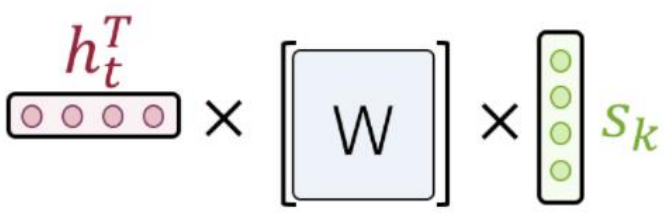
- Many options (function should be differentiable)

Dot-product



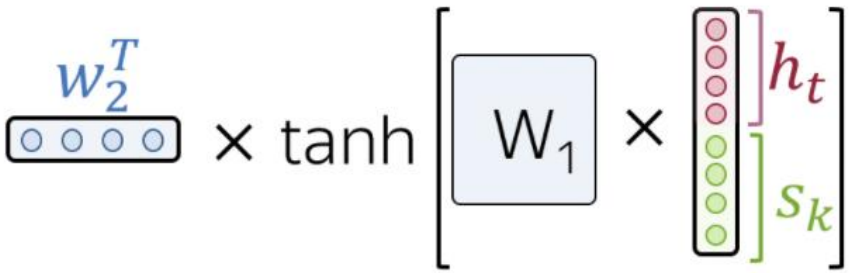
$\text{score}(h_t, s_k) = h_t^T \; s_k$

Bilinear



$\text{score}(h_t, s_k) = h_t^T \; W s_k$

Multi-Layer Perceptron



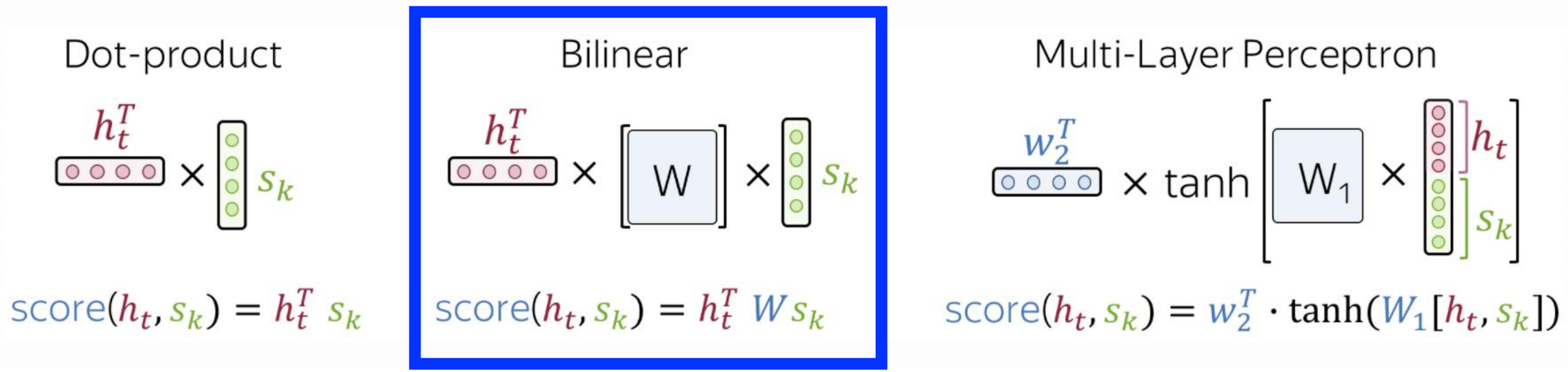
$\text{score}(h_t, s_k) = w_2^T \cdot \tanh(W_1[h_t, s_k])$

What model parameters must be learned when using dot-product?

https://lena-voita.github.io/nlp_course/seq2seq_and_attention.html

Compatibility Measures

- Many options (function should be differentiable)



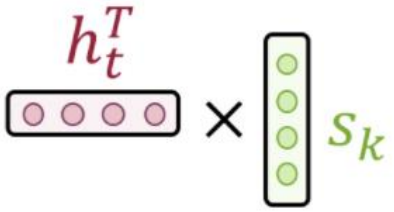
What model parameters must be learned when using bilinear?

https://lena-voita.github.io/nlp_course/seq2seq_and_attention.html

Compatibility Measures

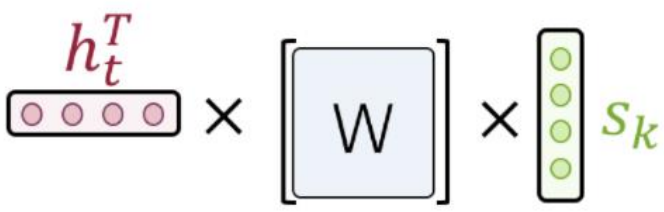
- Many options (function should be differentiable)

Dot-product



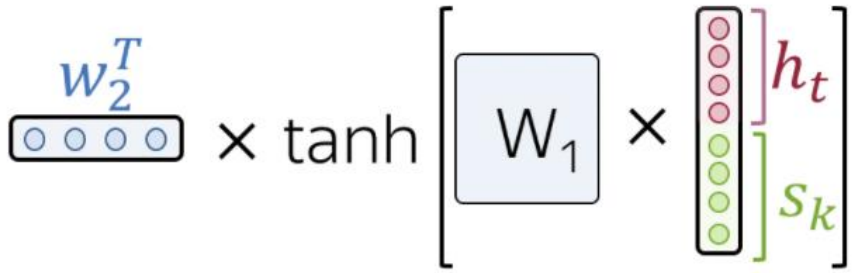
$$\text{score}(h_t, s_k) = h_t^T \, s_k$$

Bilinear



$$\text{score}(h_t, s_k) = h_t^T \, W \, s_k$$

Multi-Layer Perceptron



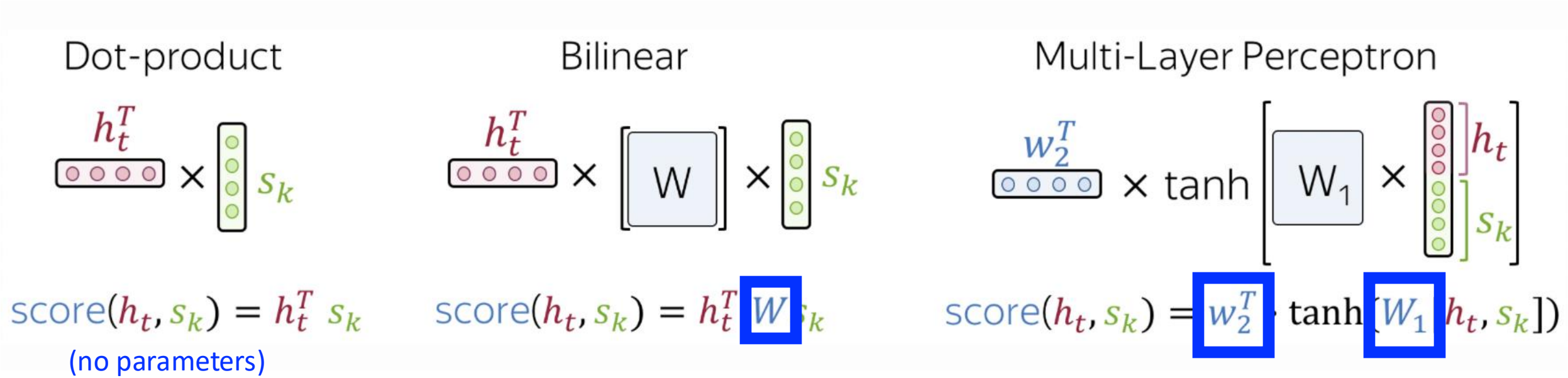
$$\text{score}(h_t, s_k) = w_2^T \cdot \tanh(W_1[h_t, s_k])$$

What model parameters must be learned when using multi-layer perceptron?

https://lena-voita.github.io/nlp_course/seq2seq_and_attention.html

Compatibility Measures

- Many options (function should be differentiable)

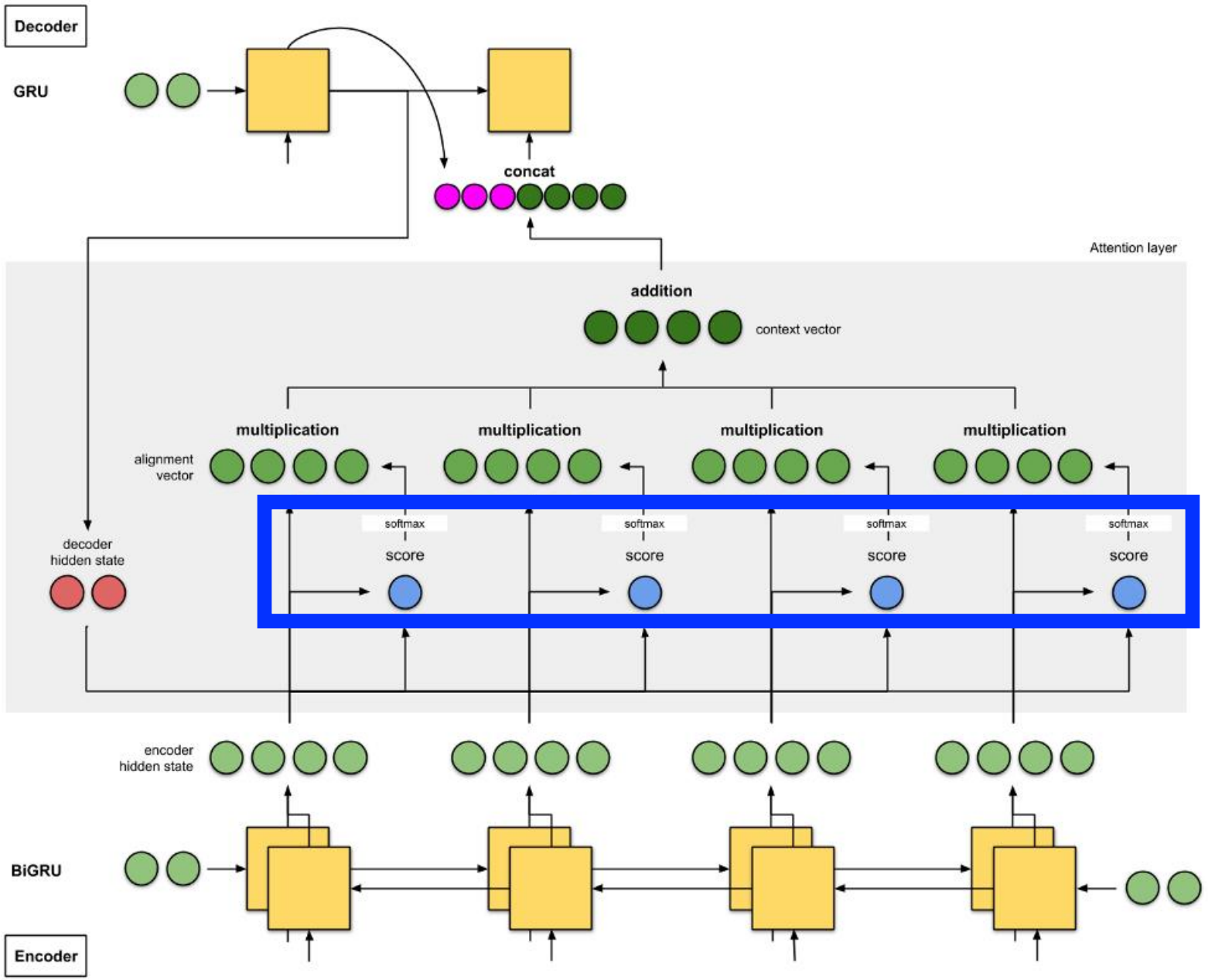


Model parameters that must be learned

https://lena-voita.github.io/nlp_course/seq2seq_and_attention.html

Measuring Each Input's Influence on the Prediction

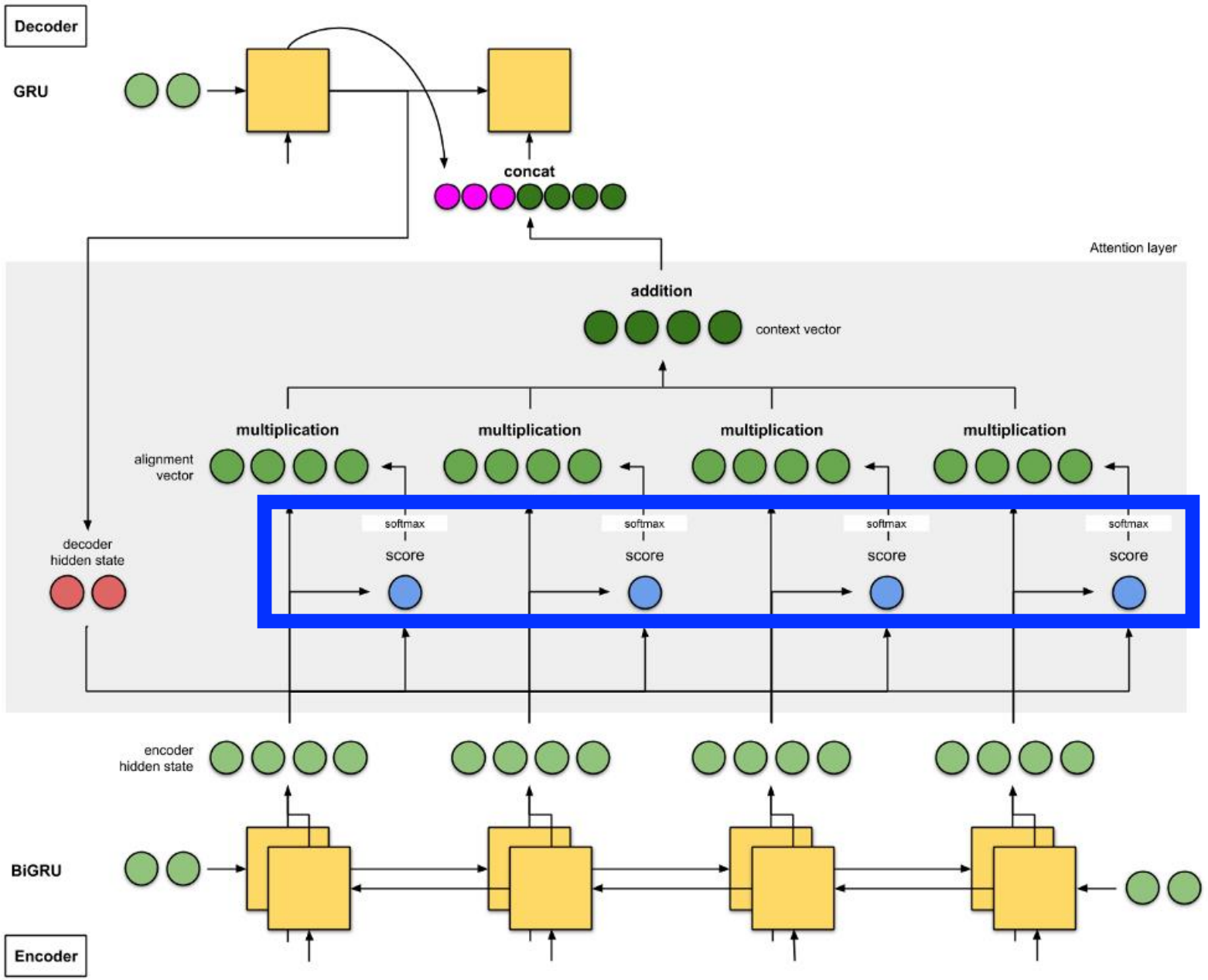
Next, apply softmax so all inputs' weights sum to 1



https://towardsdatascience.com/attn-illustrated-attention-5ec4ad276ee3

Measuring Each Input's Influence on the Prediction

We now have our attention weights!

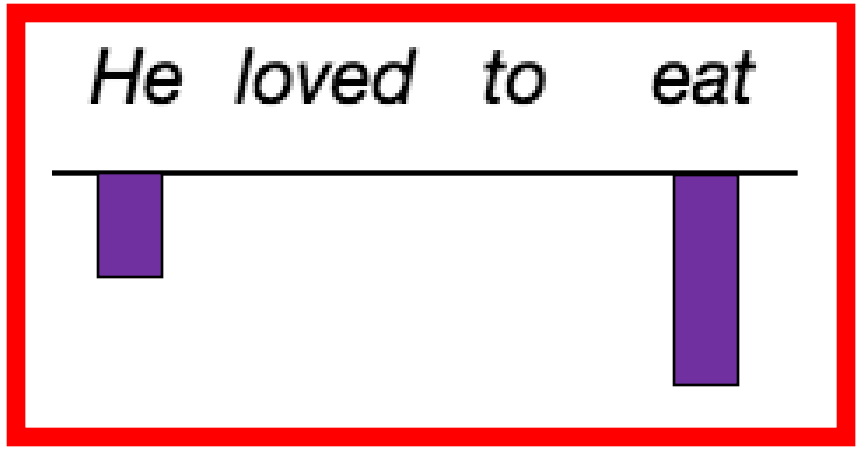


https://towardsdatascience.com/attn-illustrated-attention-5ec4ad276ee3

Measuring Each Input's Influence on the Prediction

Intuitively:

Input



Target

Er liebte zu essen

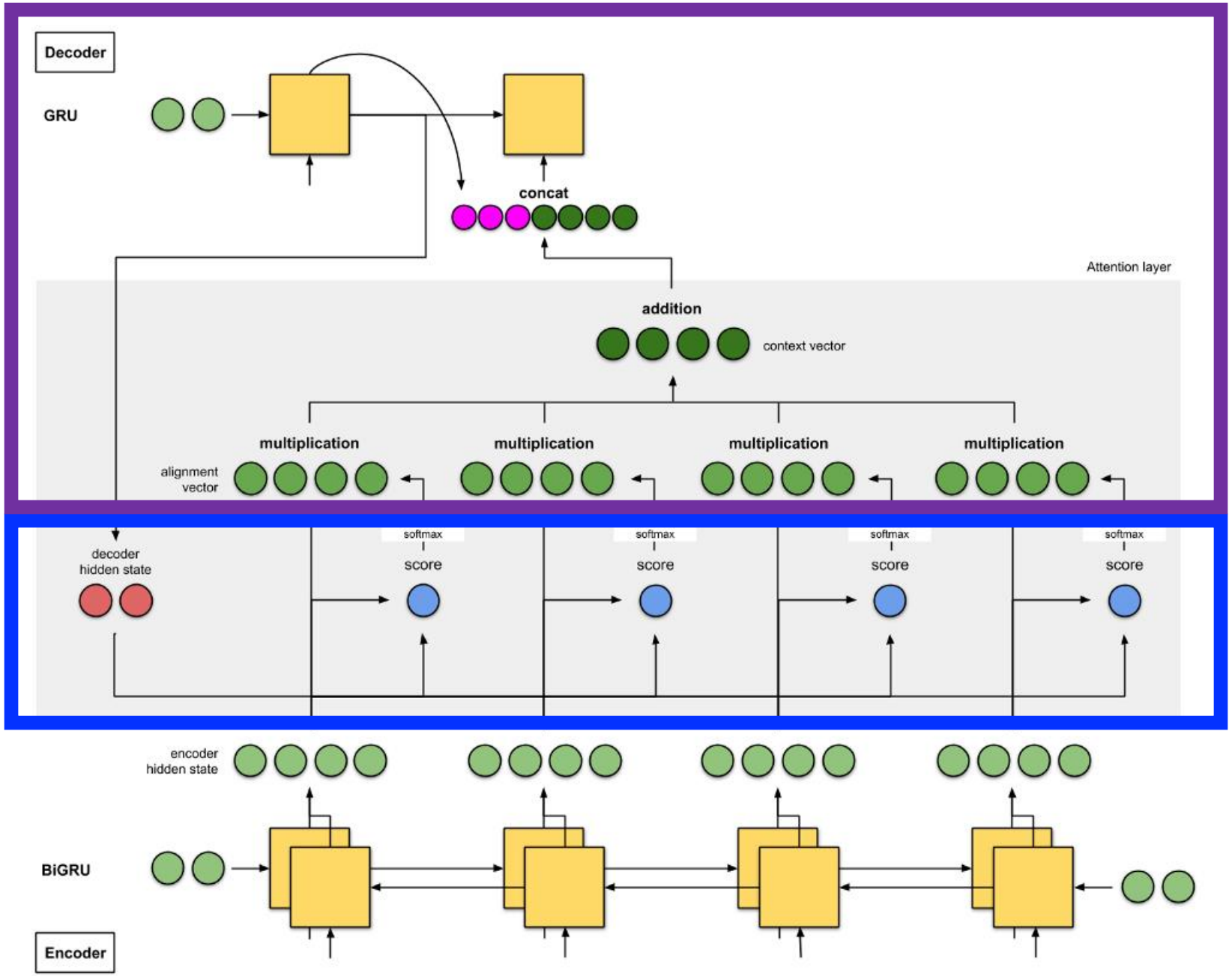
t = 4

The model can weight each input at each time step!

Solution

3. At each decoder time step, a prediction is made based on the weighted sum of the inputs

2. At each decoder time step, attention weights are computed that determine each input's relevance for the prediction



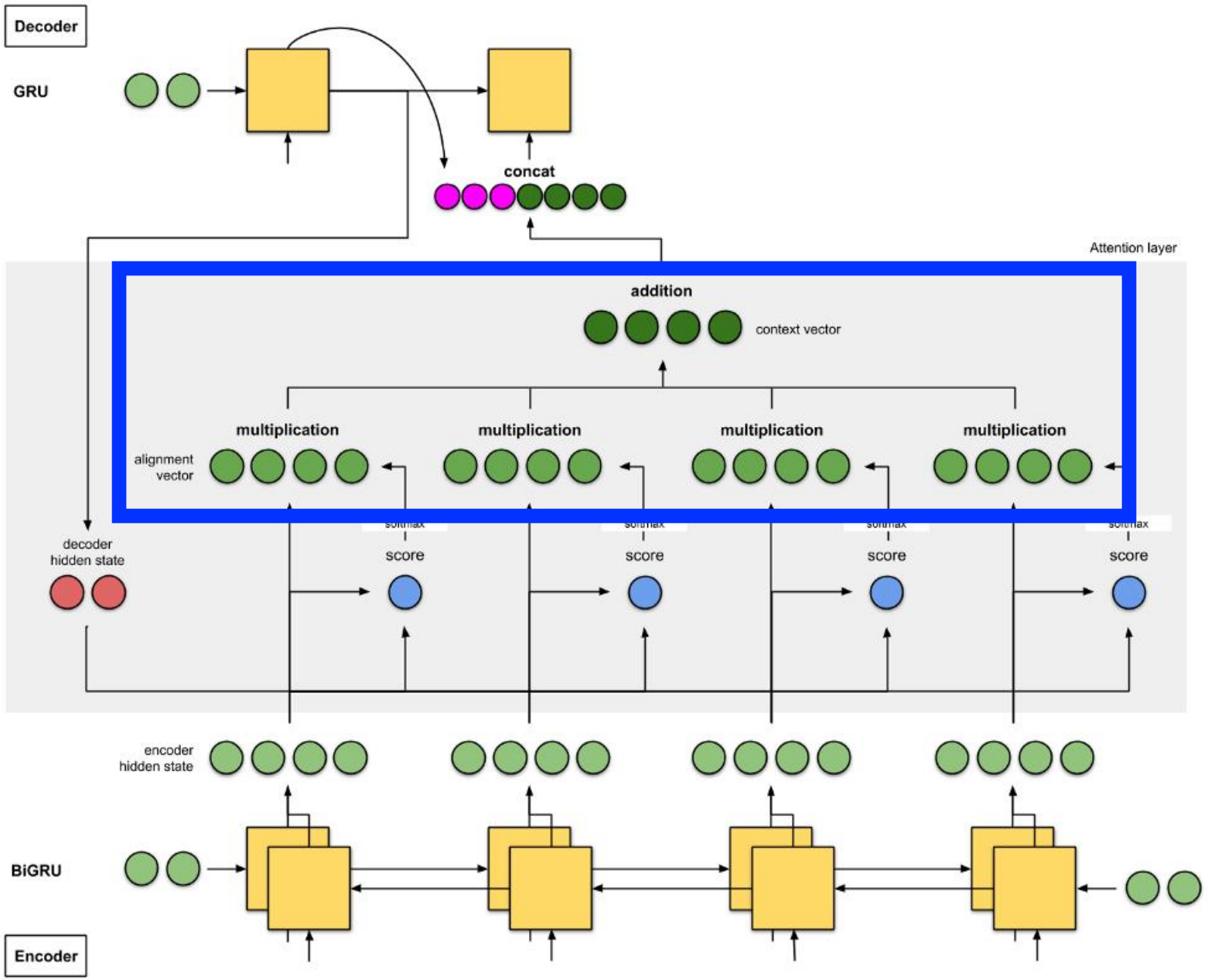
https://towardsdatascience.com/attn-illustrated-attention-5ec4ad276ee3

Word Prediction

Compute at time step *t* for all *n* inputs weighted sum*:*

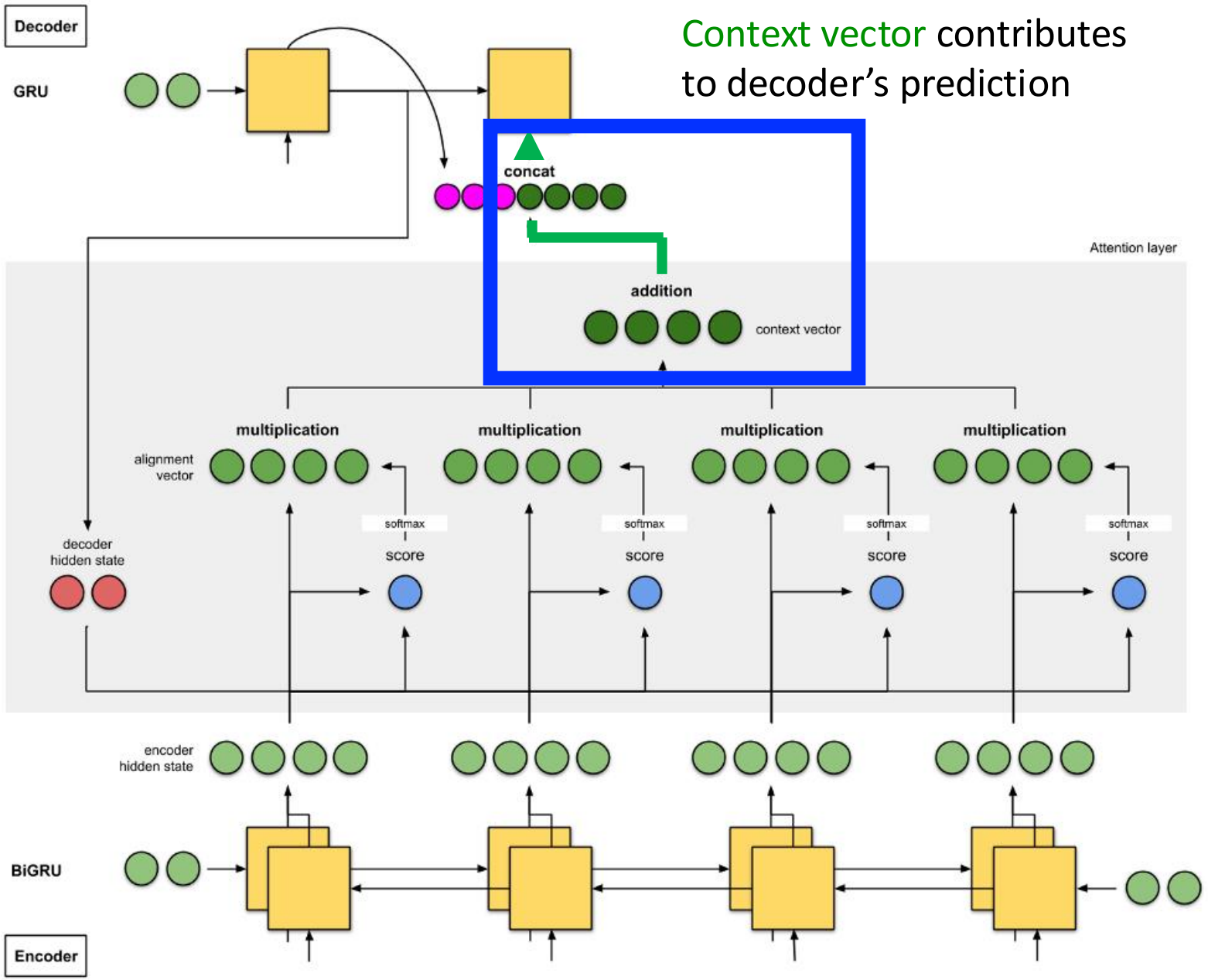
$$\mathbf{c}_t = \sum_{i=1}^{n} \alpha_{t,i} \boldsymbol{h}_i$$

Influence of inputs are **amplified** for large attention weights and repressed otherwise



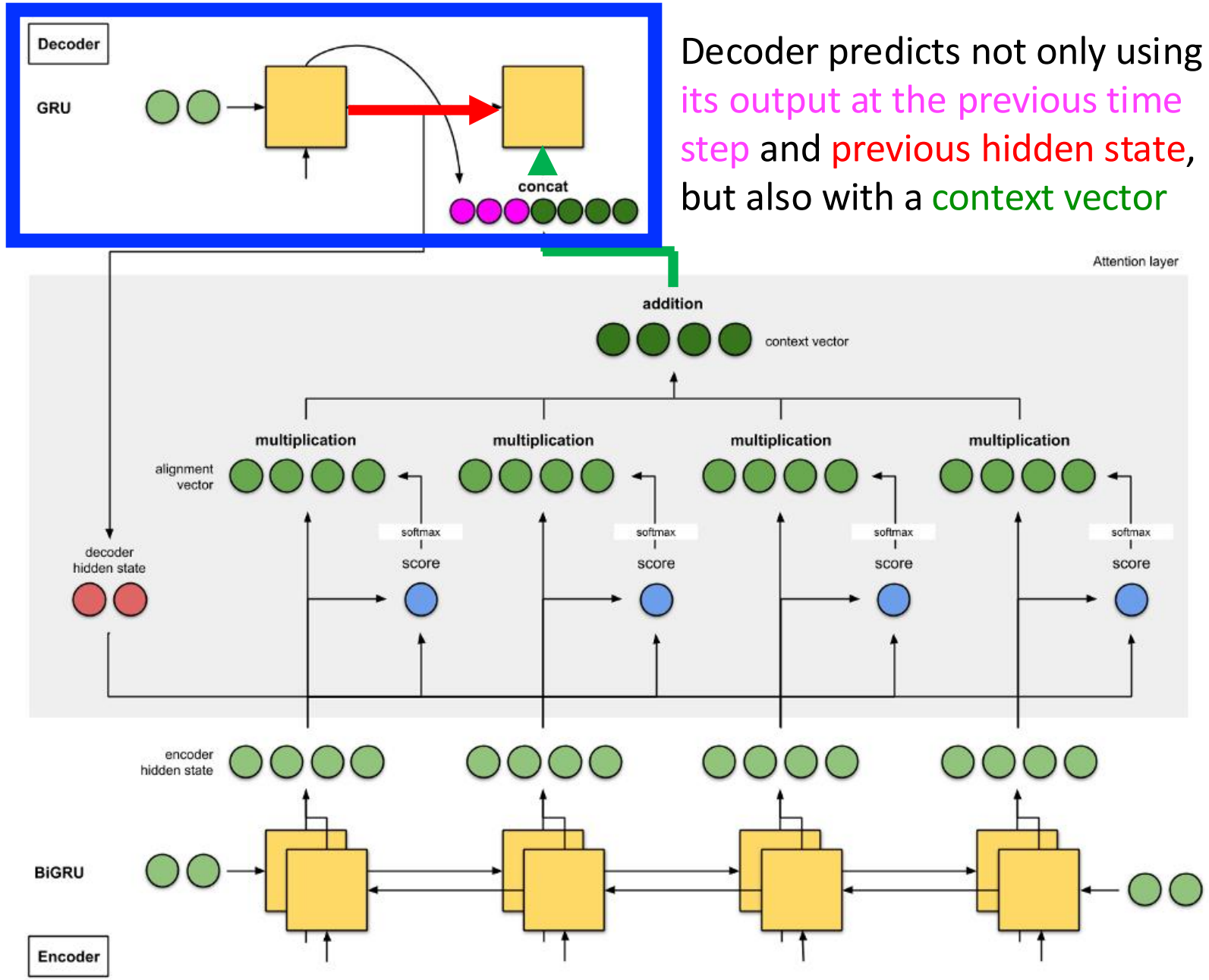
https://towardsdatascience.com/attn-illustrated-attention-5ec4ad276ee3

Word Prediction



https://towardsdatascience.com/attn-illustrated-attention-5ec4ad276ee3

Word Prediction

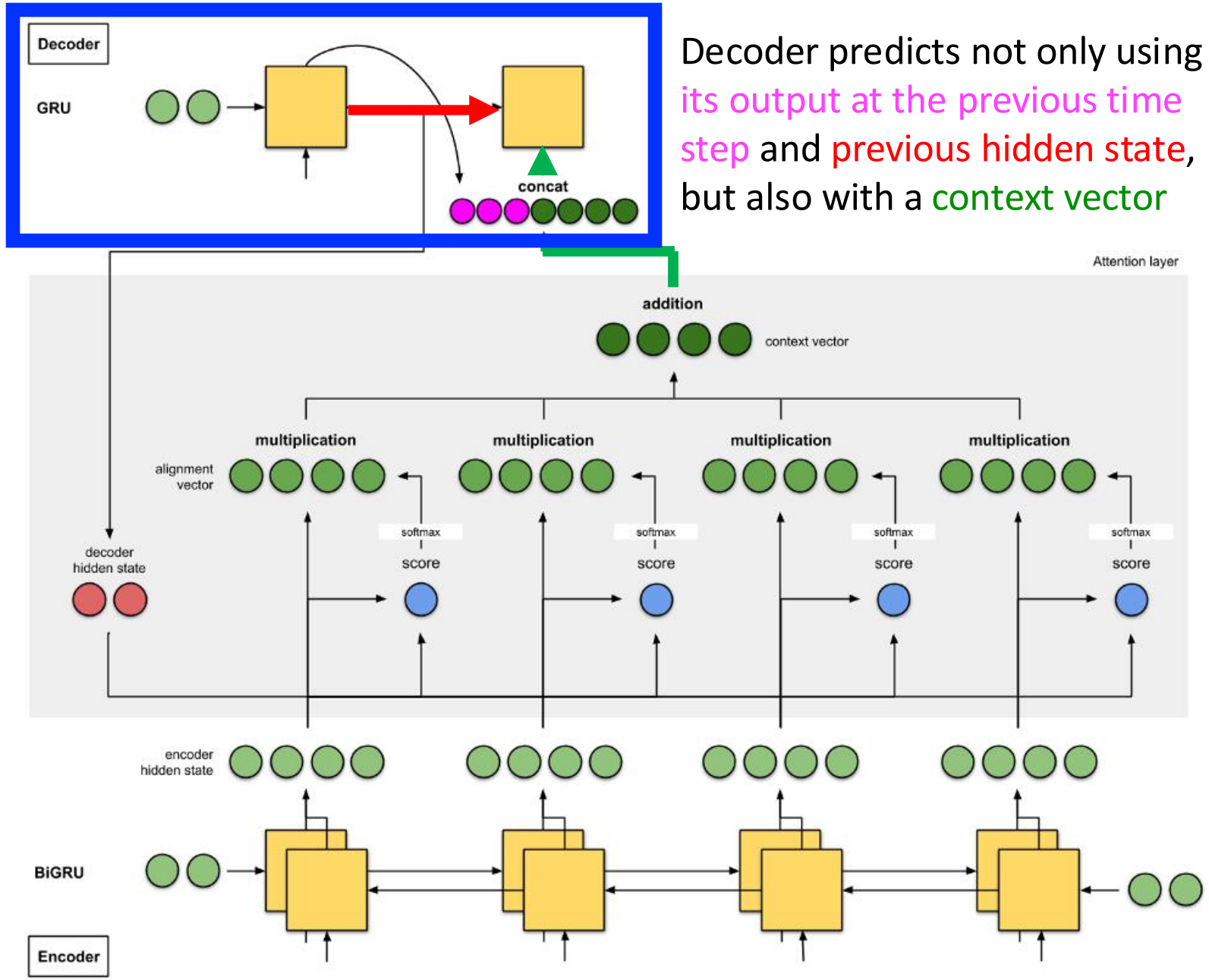


Decoder predicts not only using its output at the previous time step and previous hidden state, but also with a context vector

https://towardsdatascience.com/attn-illustrated-attention-5ec4ad276ee3

Bahdanau method

Many options exist for how to use the context vector with the decoder's output at the previous time step to produce an output at each decoder time step



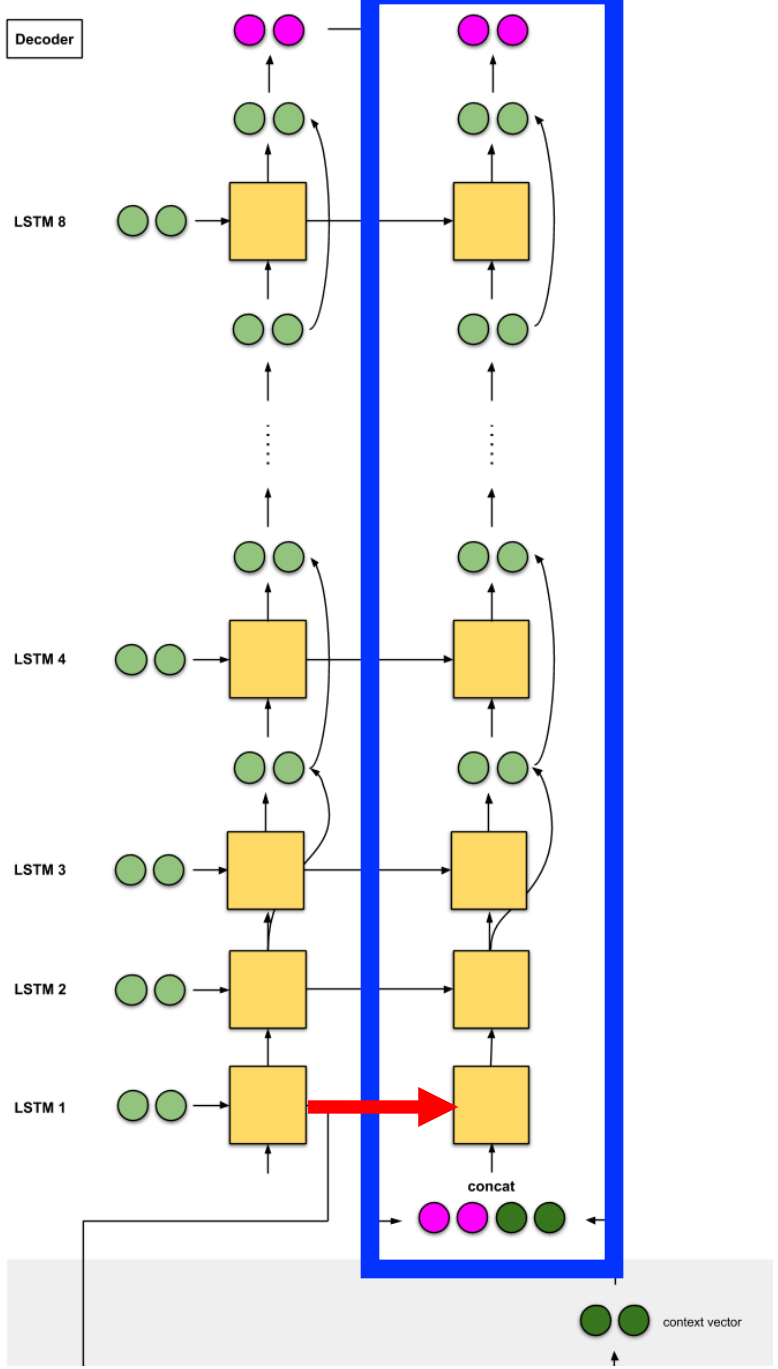
Decoder predicts not only using its output at the previous time step and previous hidden state, but also with a context vector

https://towardsdatascience.com/attn-illustrated-attention-5ec4ad276ee3

Google method

Many options exist for how to use the context vector with the decoder's output at the previous time step to produce an output at each decoder time step

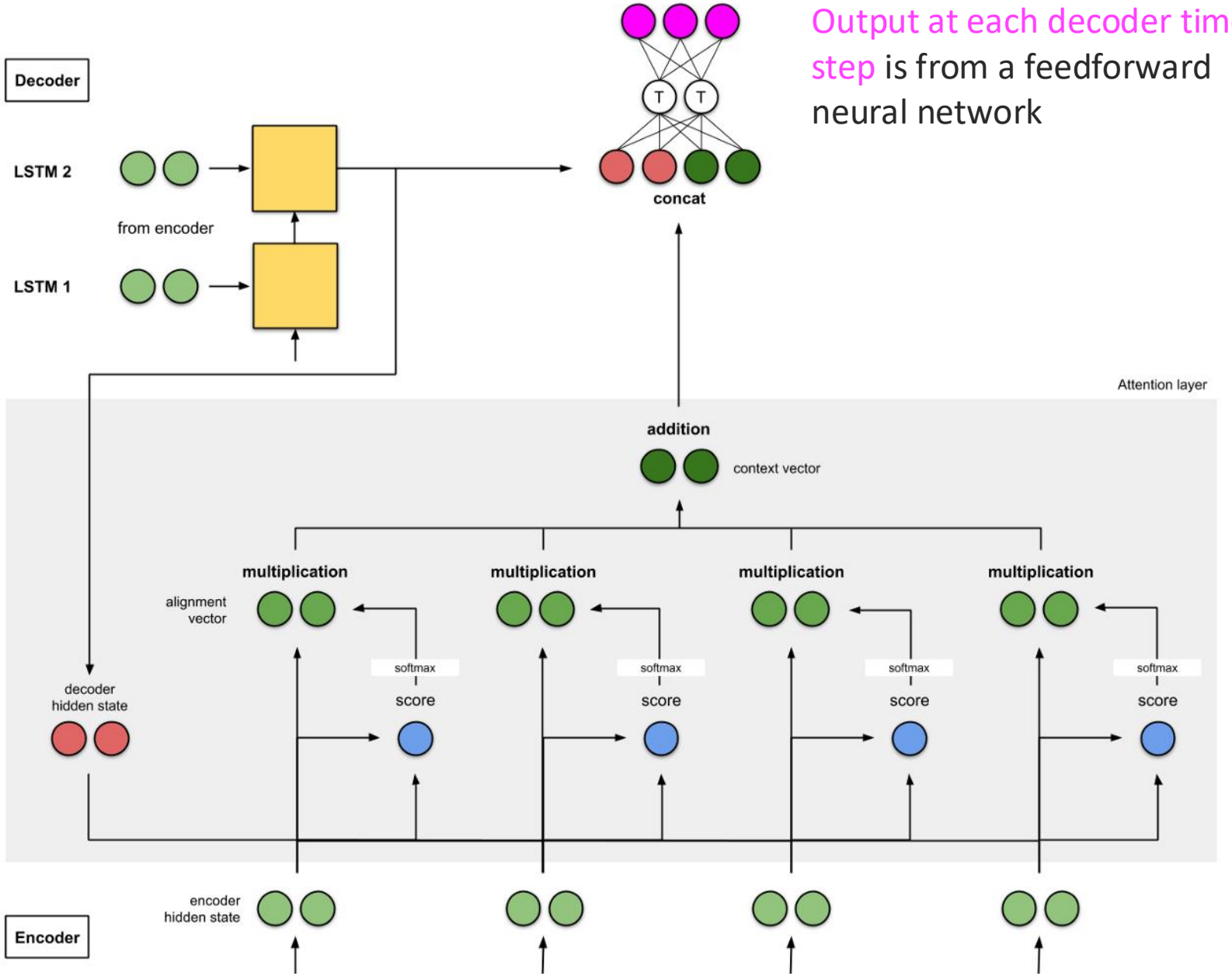
Decoder also uses previous hidden state



https://towardsdatascience.com/attn-illustrated-attention-5ec4ad276ee3

Luong method

Many options exist for how to use the context vector with the decoder's output at the previous time step to produce an output at each decoder time step

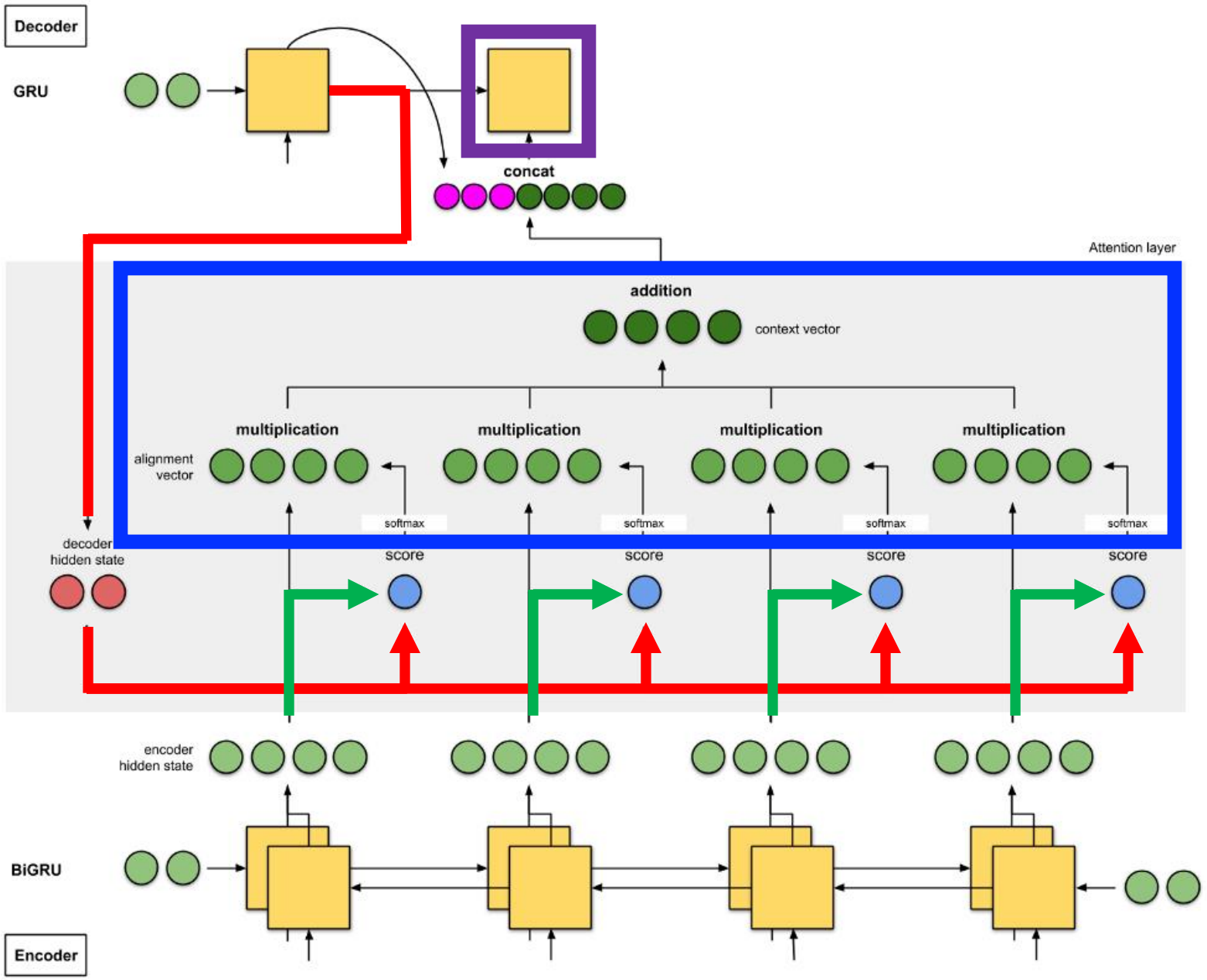


https://towardsdatascience.com/attn-illustrated-attention-5ec4ad276ee3

Decoder

What stays the same at each decoder time step?
- input's hidden state

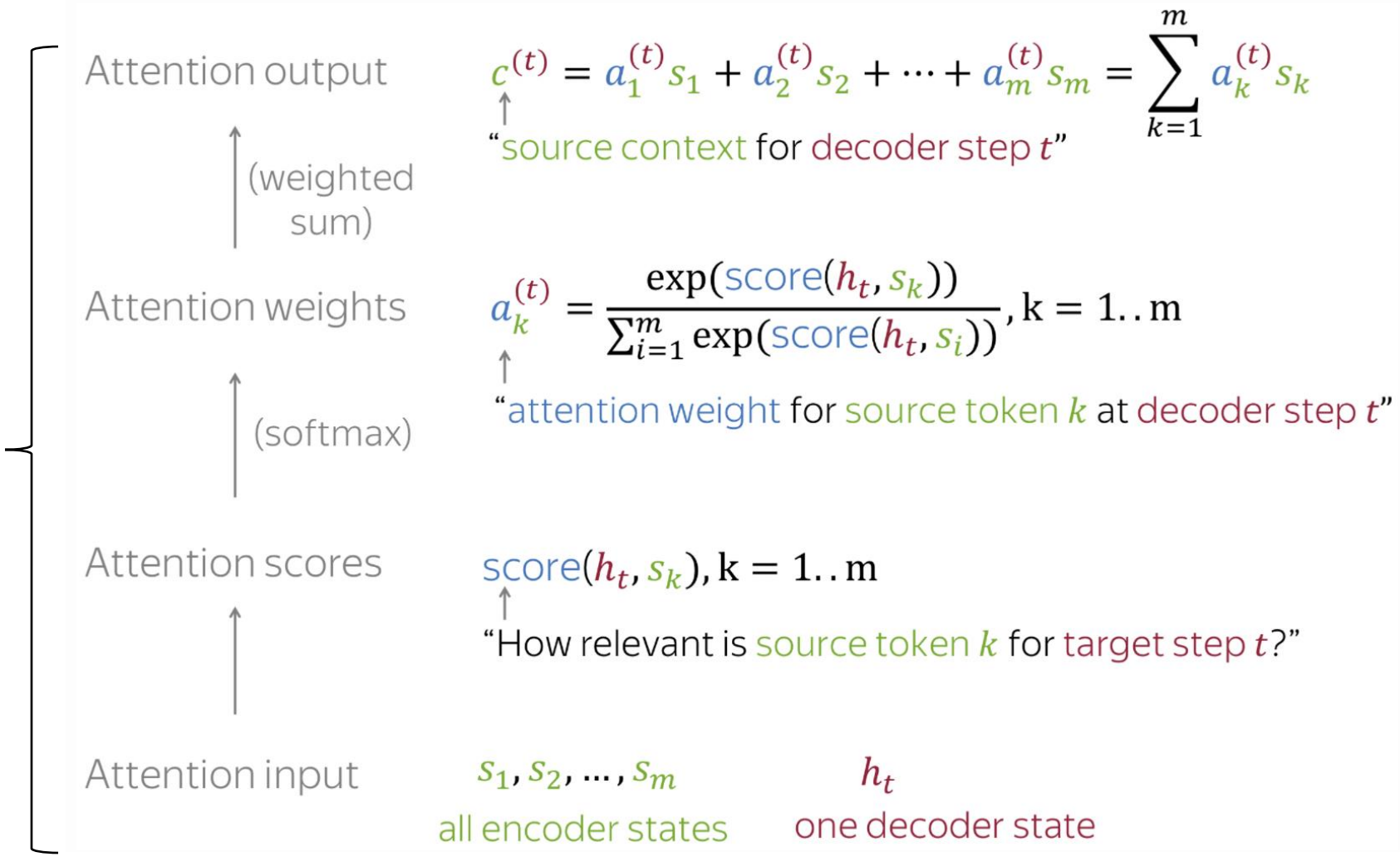
What changes at each decoder time step?
- decoder's hidden state
- (and so) attention weights and context vector
- decoder's prediction at the previous time step



https://towardsdatascience.com/attn-illustrated-attention-5ec4ad276ee3

Summary: Attention (Computations at Each Decoder Step)

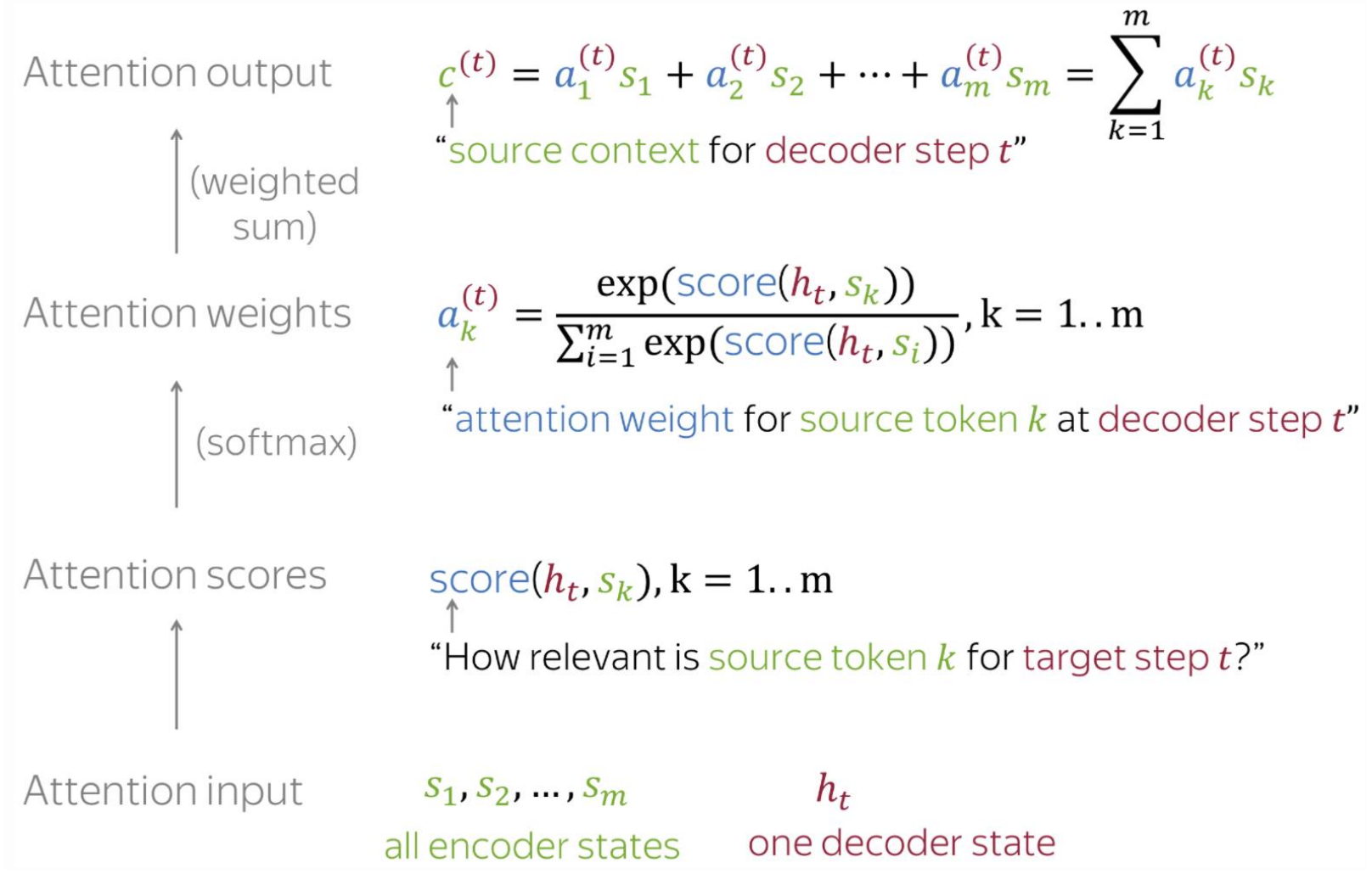
Decoder decides which inputs are needed for prediction at each time step with “soft attention”



https://lena-voita.github.io/nlp_course/seq2seq_and_attention.html

Summary: Attention (Computations at Each Decoder Step)

All parts are differentiable which means end-to-end training is possible

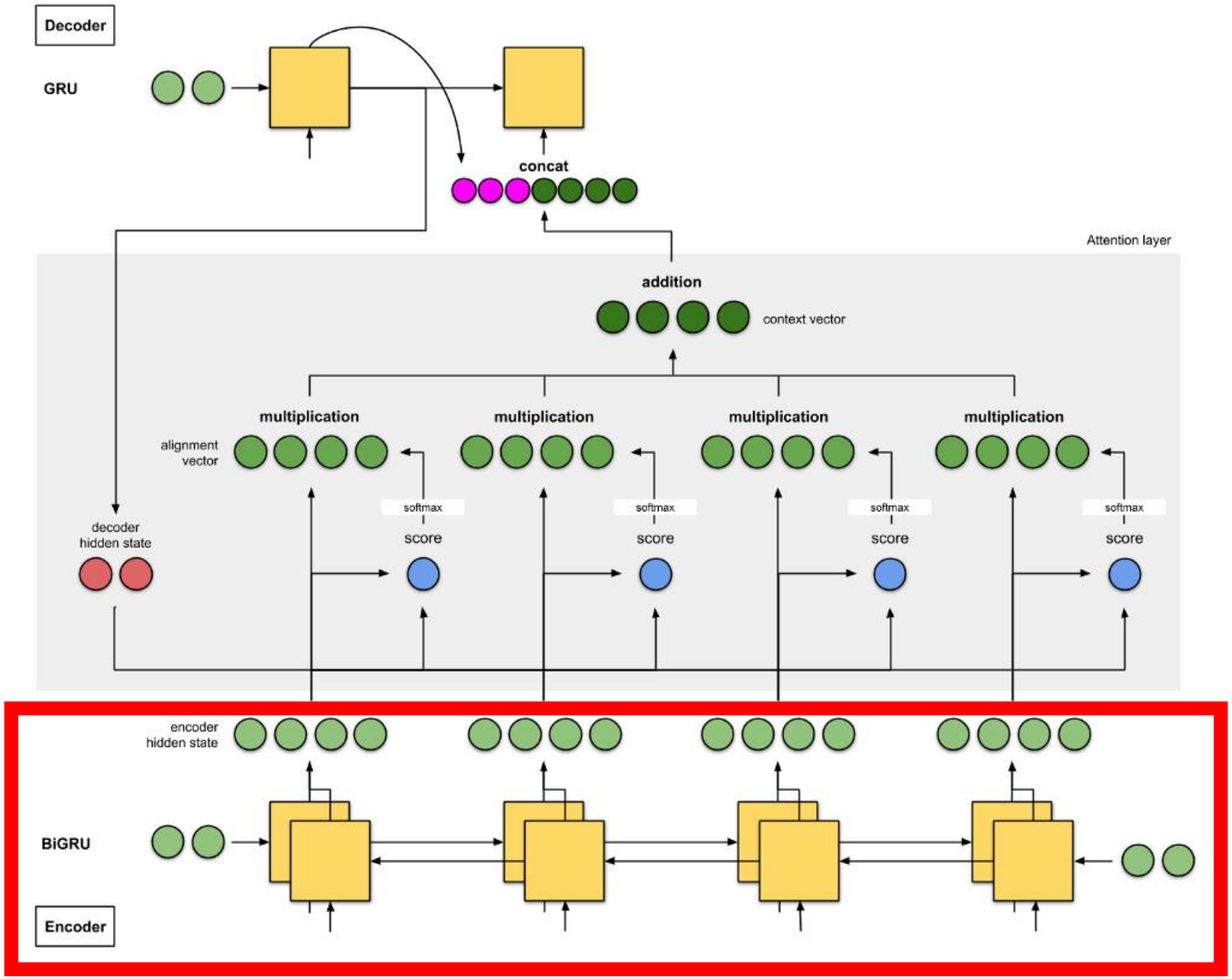


https://lena-voita.github.io/nlp_course/seq2seq_and_attention.html

Today's Topics

- Motivation: machine neural translation for long sentences
- Decoder: attention
- Encoder
- Performance evaluation
- Final project: ways to find a partner

Solution

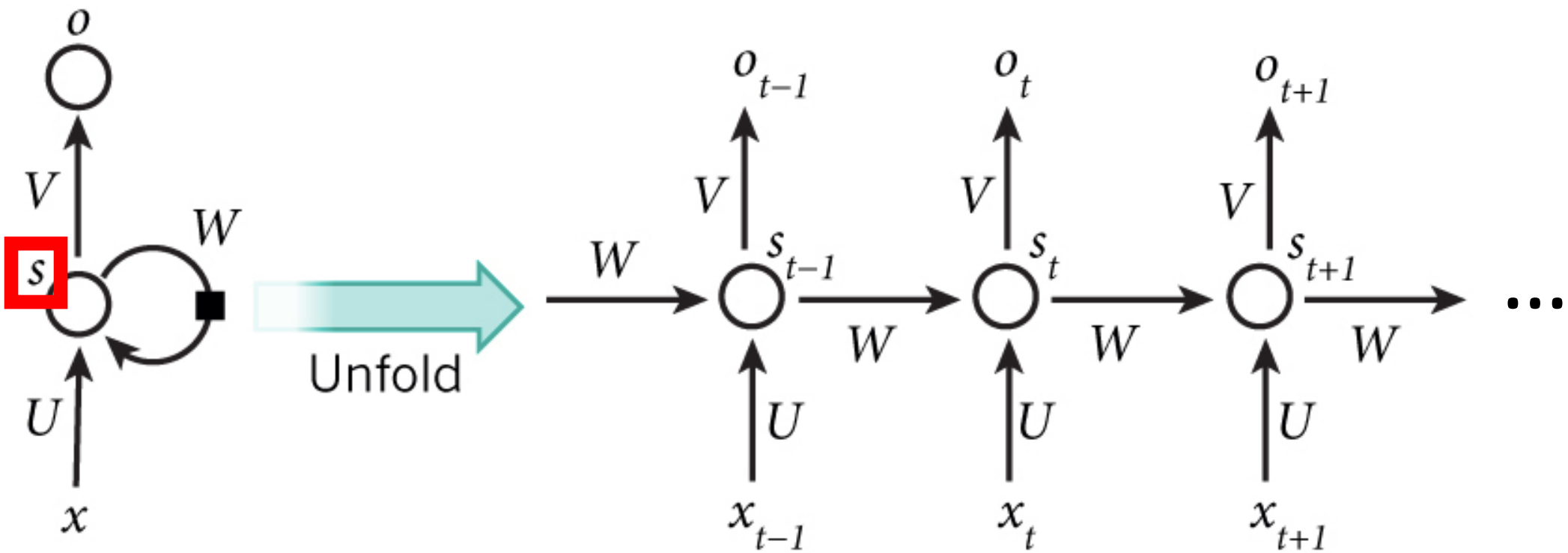


1. Encoder produces hidden state for every input

https://towardsdatascience.com/attn-illustrated-attention-5ec4ad276ee3

Recall: Hidden States from RNNs

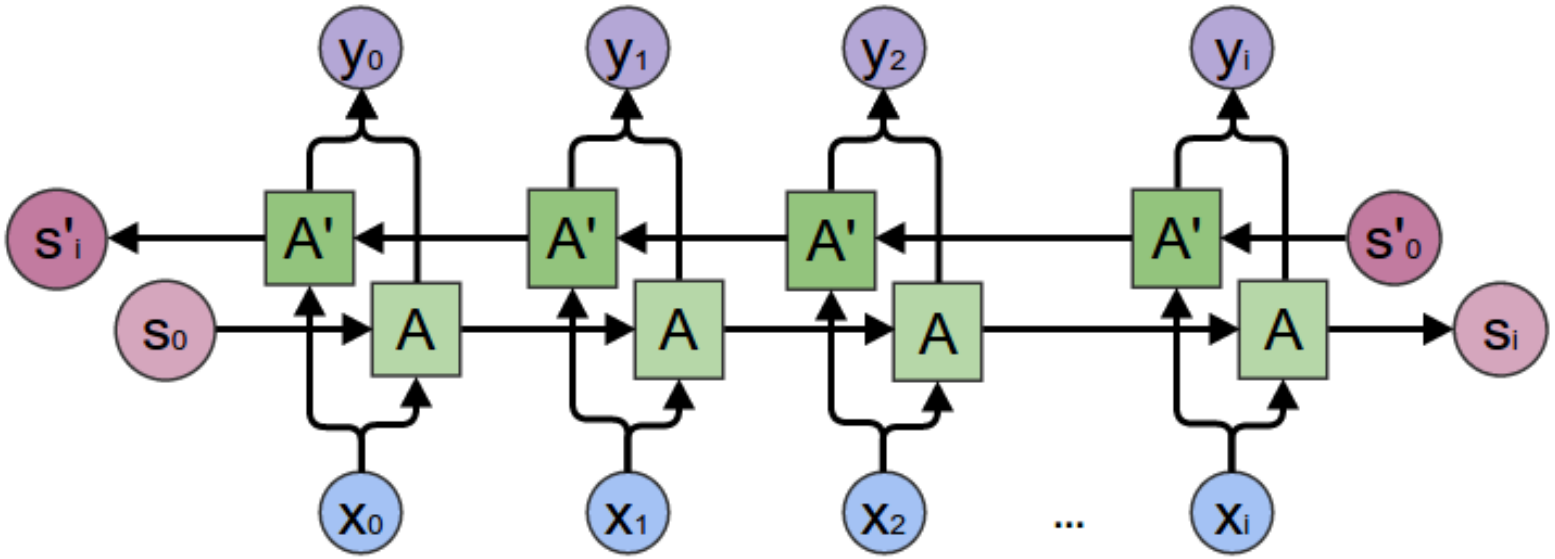
- Hidden state captures information about the past



http://www.wildml.com/2015/09/recurrent-neural-networks-tutorial-part-1-introduction-to-rnns/

Bahdanau's Encoder

- Two RNNs where input is fed forward and backward respectively and then the hidden states (typically) are concatenated into a hidden state

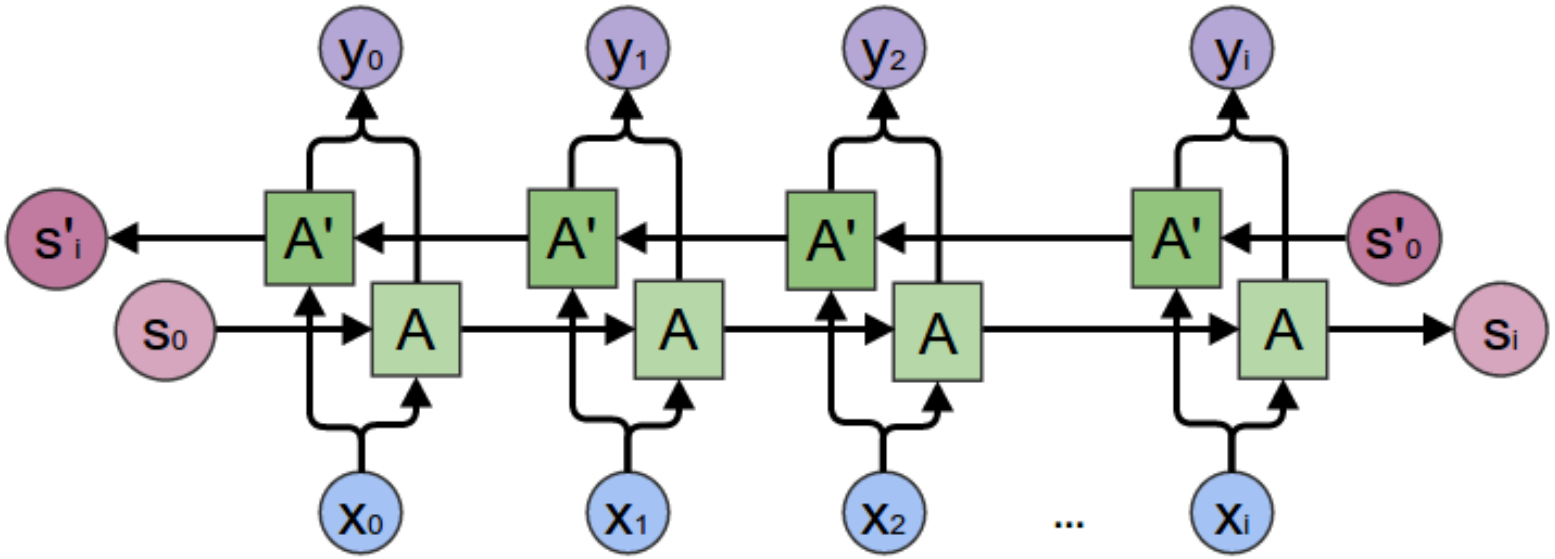


Input encoded with a bidirectional GRU

https://towardsdatascience.com/understanding-bidirectional-rnn-in-pytorch-5bd25a5dd66

Bahdanau's Encoder

- Two RNNs where input is fed forward and backward respectively and then the hidden states (typically) are concatenated into a hidden state

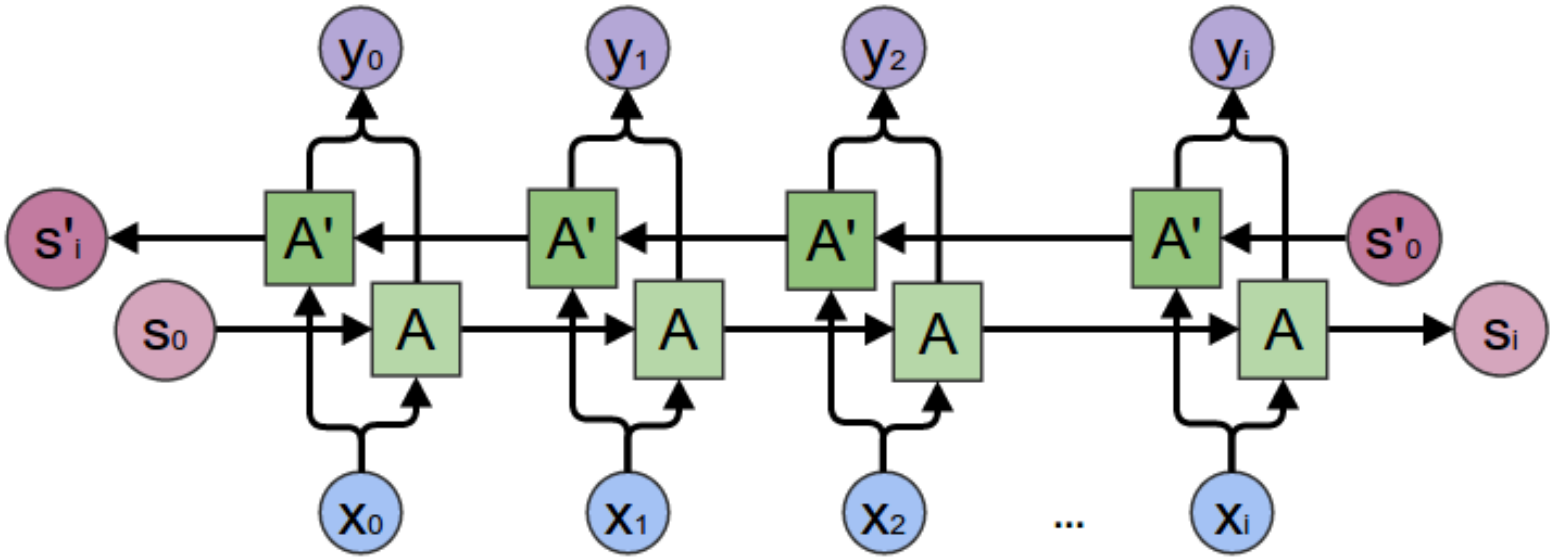


What are advantages of a bi-directional RNN compared to a single RNN?

https://towardsdatascience.com/understanding-bidirectional-rnn-in-pytorch-5bd25a5dd66

Bahdanau's Encoder

- Two RNNs where input is fed forward and backward respectively and then the hidden states (typically) are concatenated into a hidden state

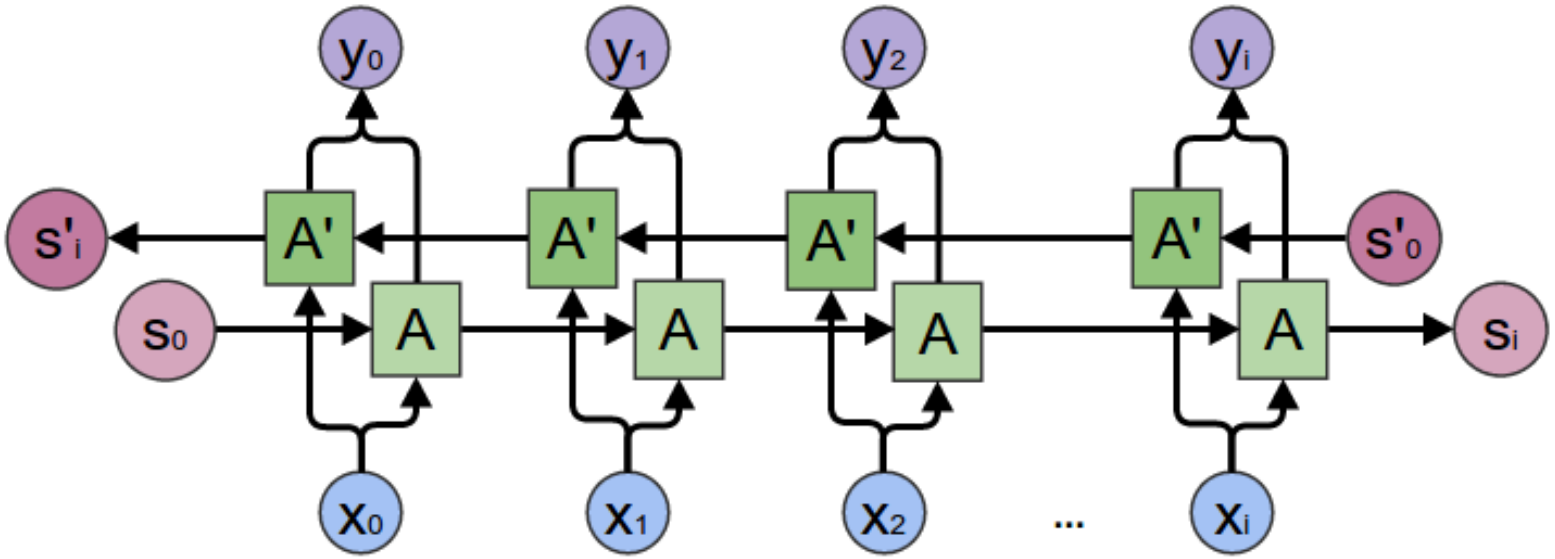


Can use information from the past and **future** to make predictions: e.g., can resolve for "Teddy is a …?" if Teddy refers to a "bear" or former US President Roosevelt

https://towardsdatascience.com/understanding-bidirectional-rnn-in-pytorch-5bd25a5dd66

Bahdanau's Encoder

- Two RNNs where input is fed forward and backward respectively and then the hidden states (typically) are concatenated into a hidden state

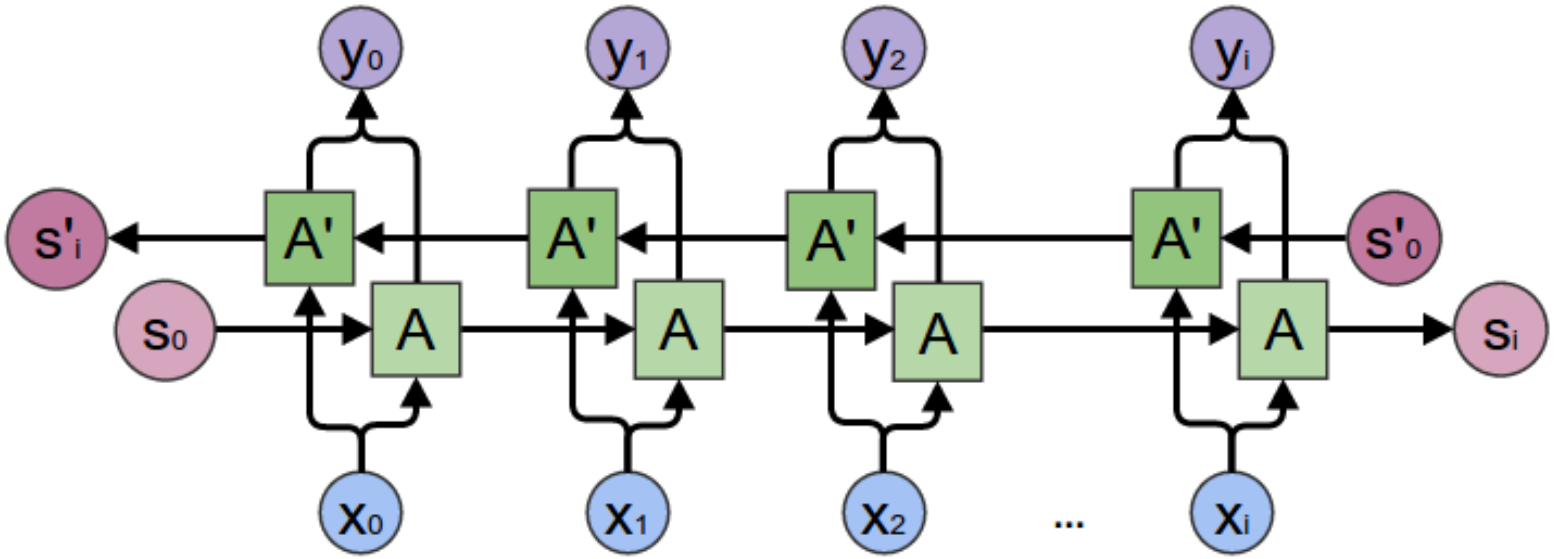


What are disadvantages of a bi-directional RNN compared to a single RNN?

https://towardsdatascience.com/understanding-bidirectional-rnn-in-pytorch-5bd25a5dd66

Bahdanau's Encoder

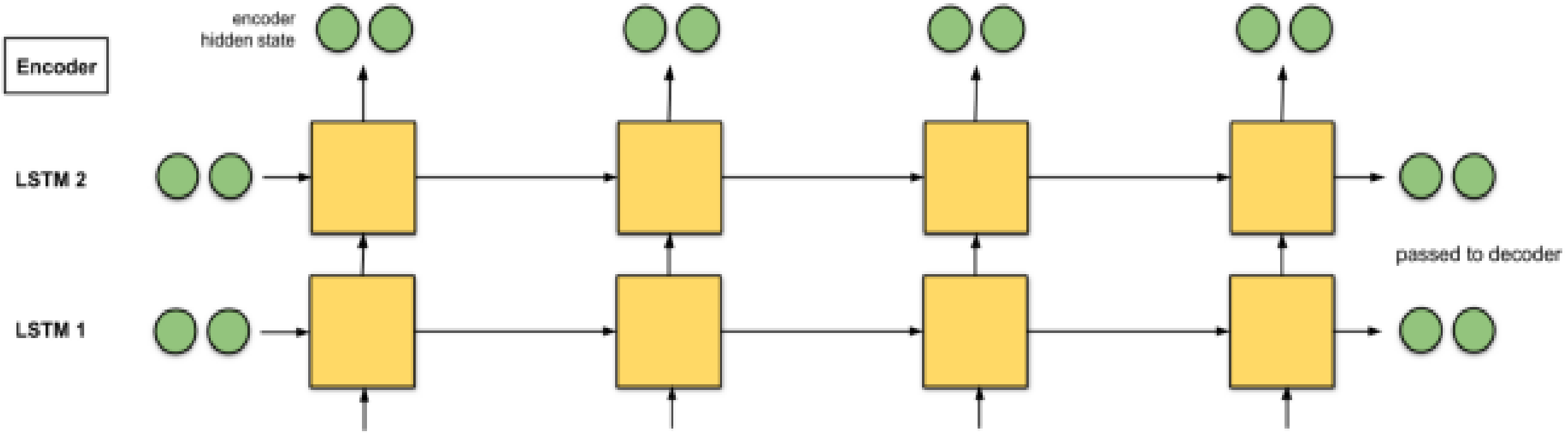
- Two RNNs where input is fed forward and backward respectively and then the hidden states (typically) are concatenated into a hidden state



Entire sequence must be observed to make a prediction (e.g., unsuitable for text prediction)

https://towardsdatascience.com/understanding-bidirectional-rnn-in-pytorch-5bd25a5dd66

Luong's Encoder



Input encoded with a 2-layer stacked LSTM

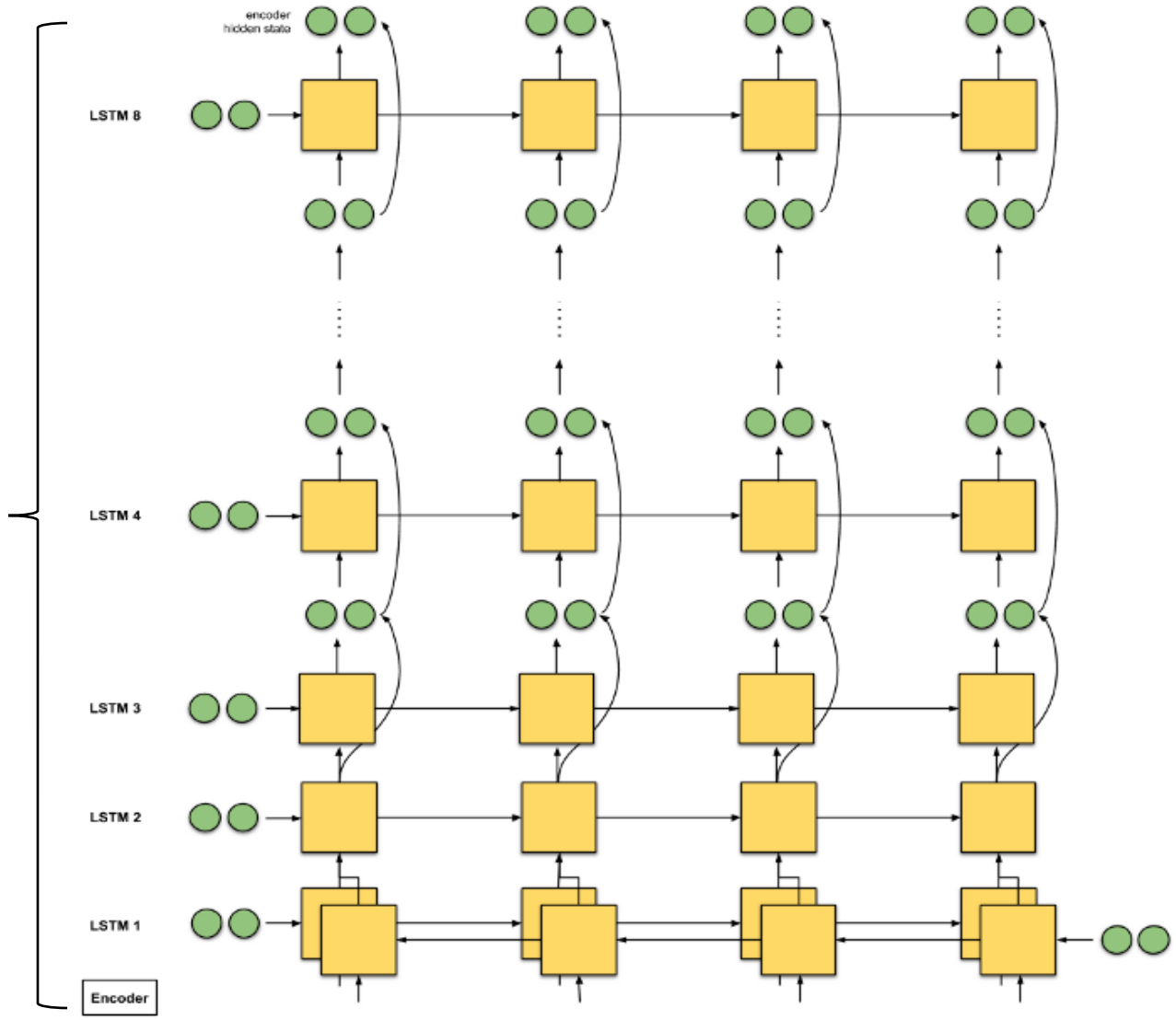
https://towardsdatascience.com/attn-illustrated-attention-5ec4ad276ee3#df28

Luong et al. Effective Approaches to Attention-based Neural Machine Translation. EMNLP 2015

Google's Encoder

8 layers with a bi-directional first layer and skip connections between layers

(greater level of abstraction for input)



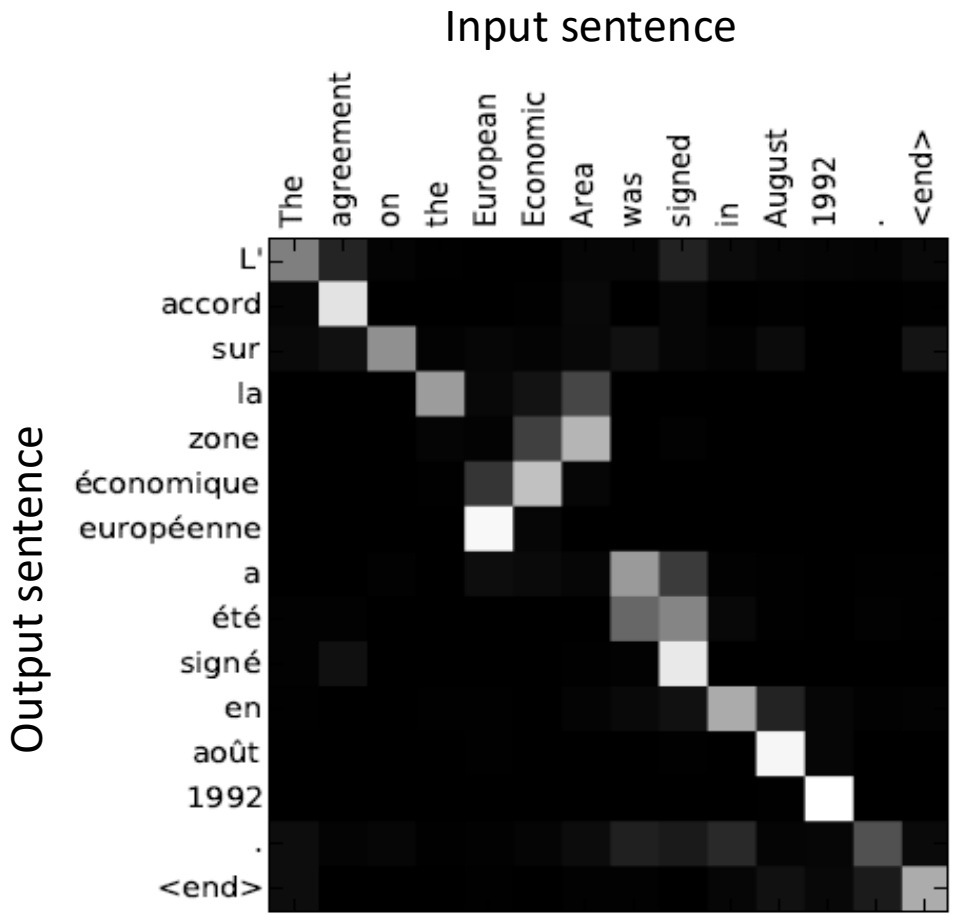
https://towardsdatascience.com/attn-illustrated-attention-5ec4ad276ee3#df28

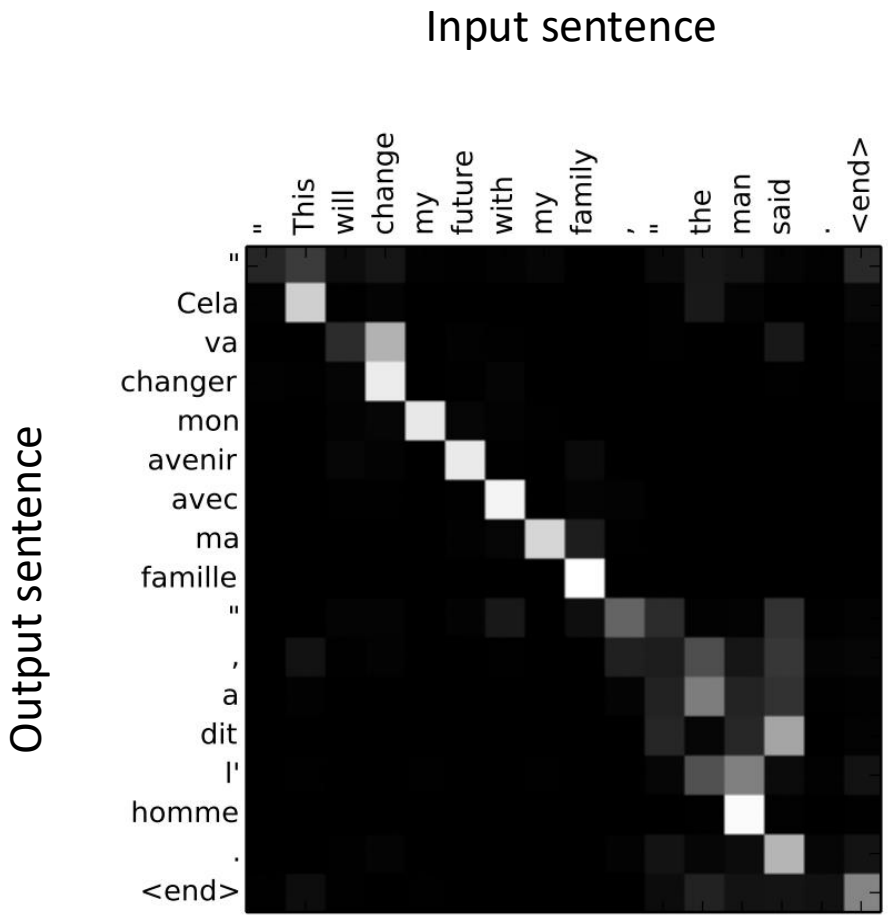
Wu et al. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. arXiv 2016.

Today's Topics

- Motivation: machine neural translation for long sentences
- Encoder
- Decoder: attention
- Performance evaluation
- Final project: ways to find a partner

Visualizing Attention

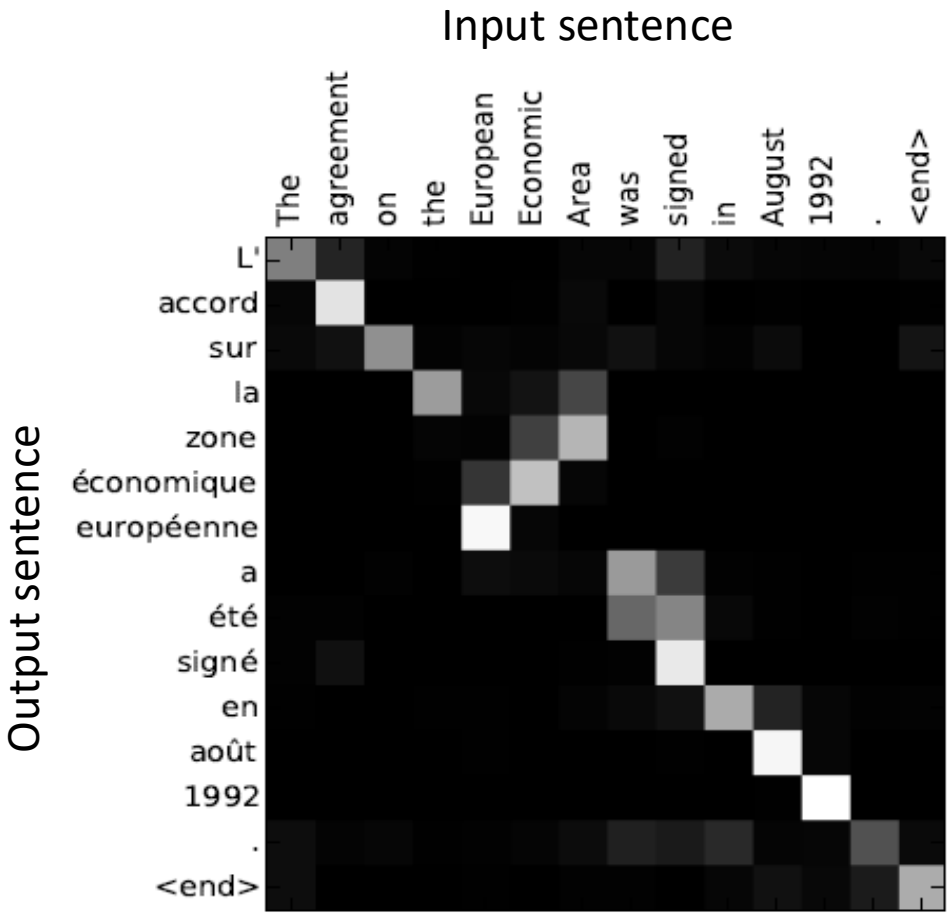


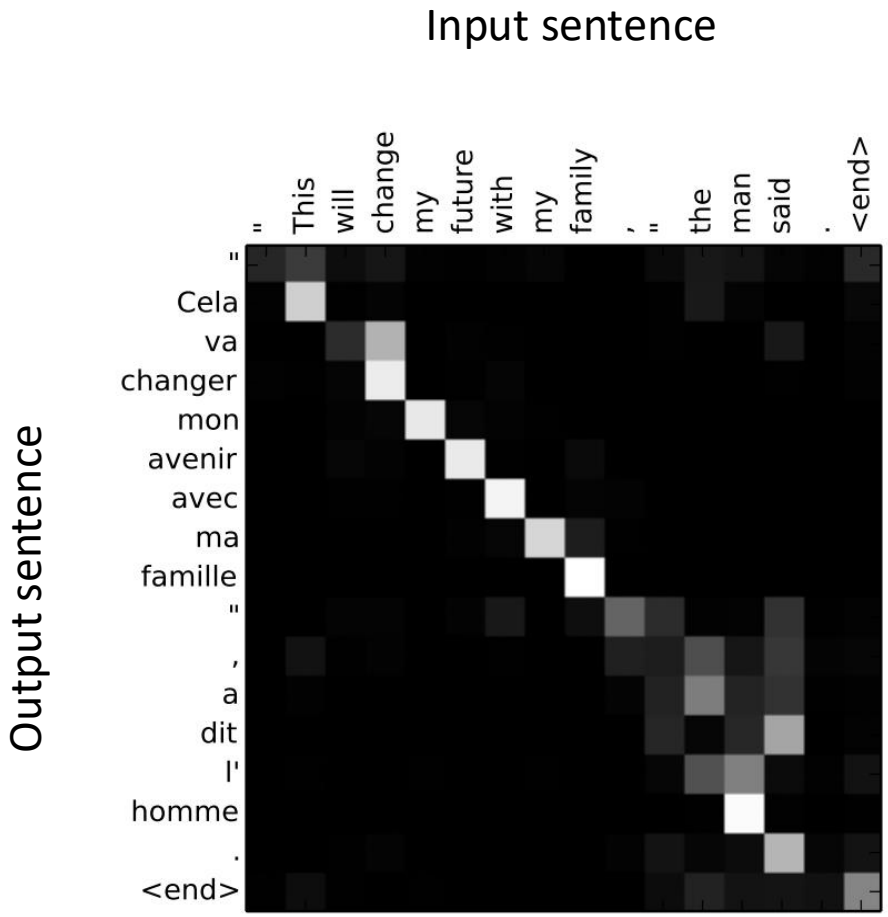


Values are 0 to 1, with whiter pixels indicating larger attention weights

Bahdanau et al. Neural Machine Translation by Jointly Learning to Align and Translate. ICLR 2015

Visualizing Attention: Group Discussion

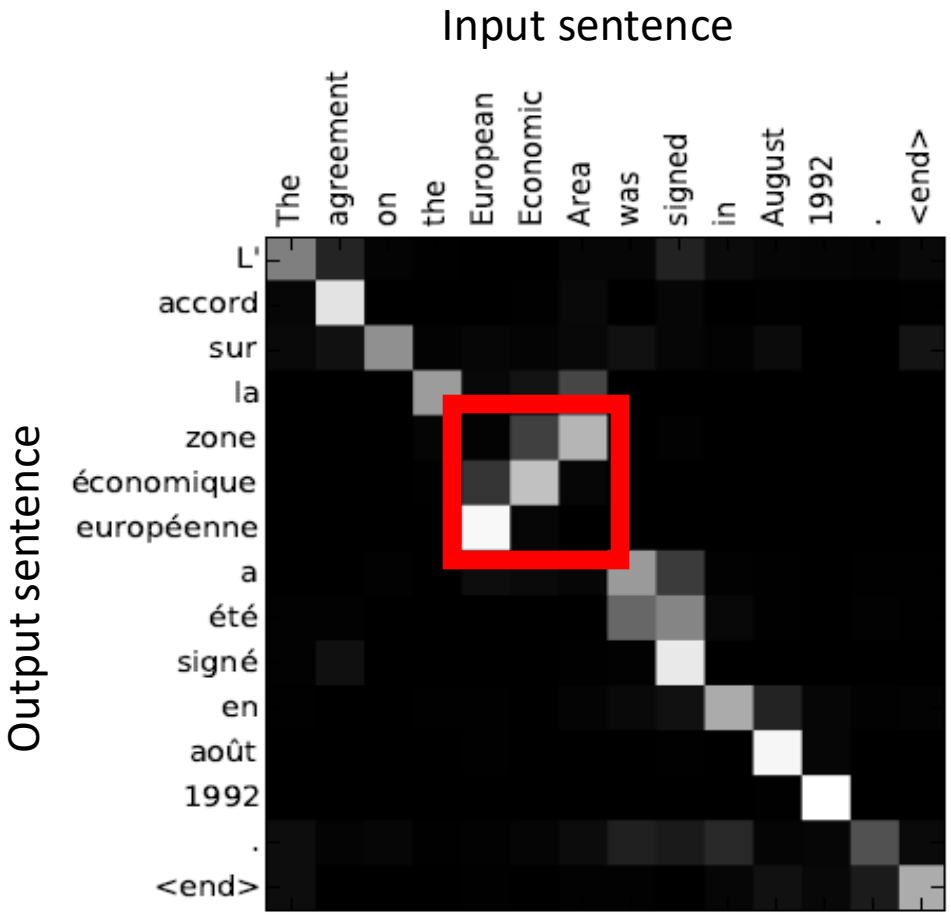


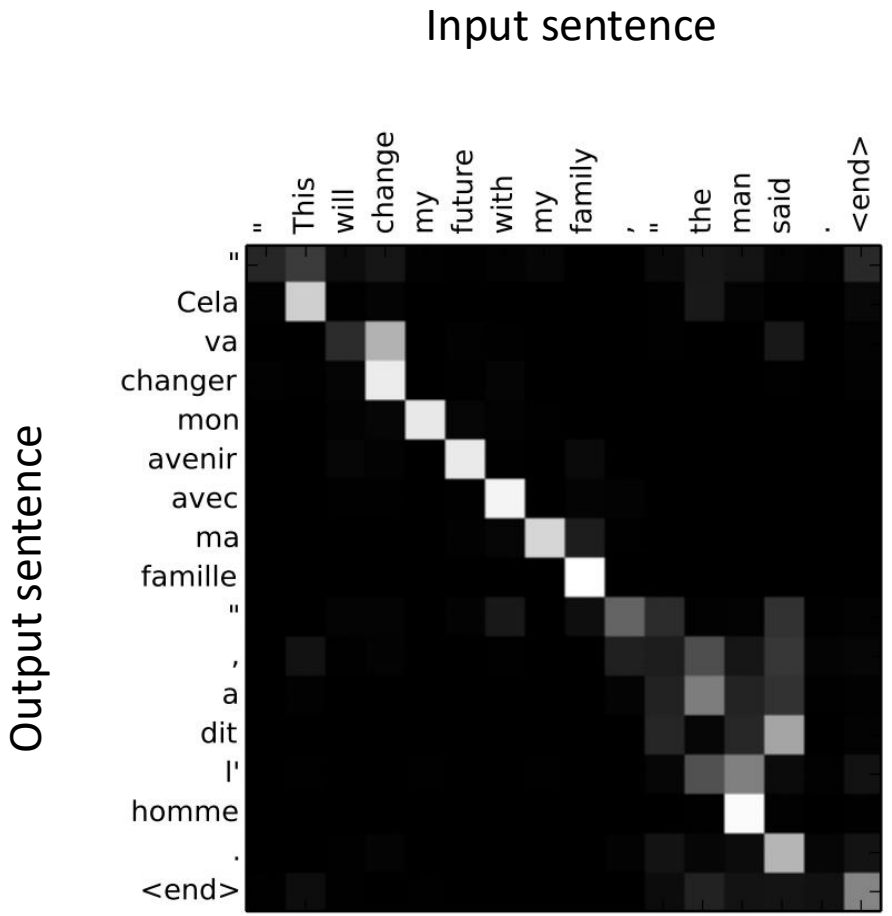


What insights can we glean from these examples?

Bahdanau et al. Neural Machine Translation by Jointly Learning to Align and Translate. ICLR 2015

Visualizing Attention

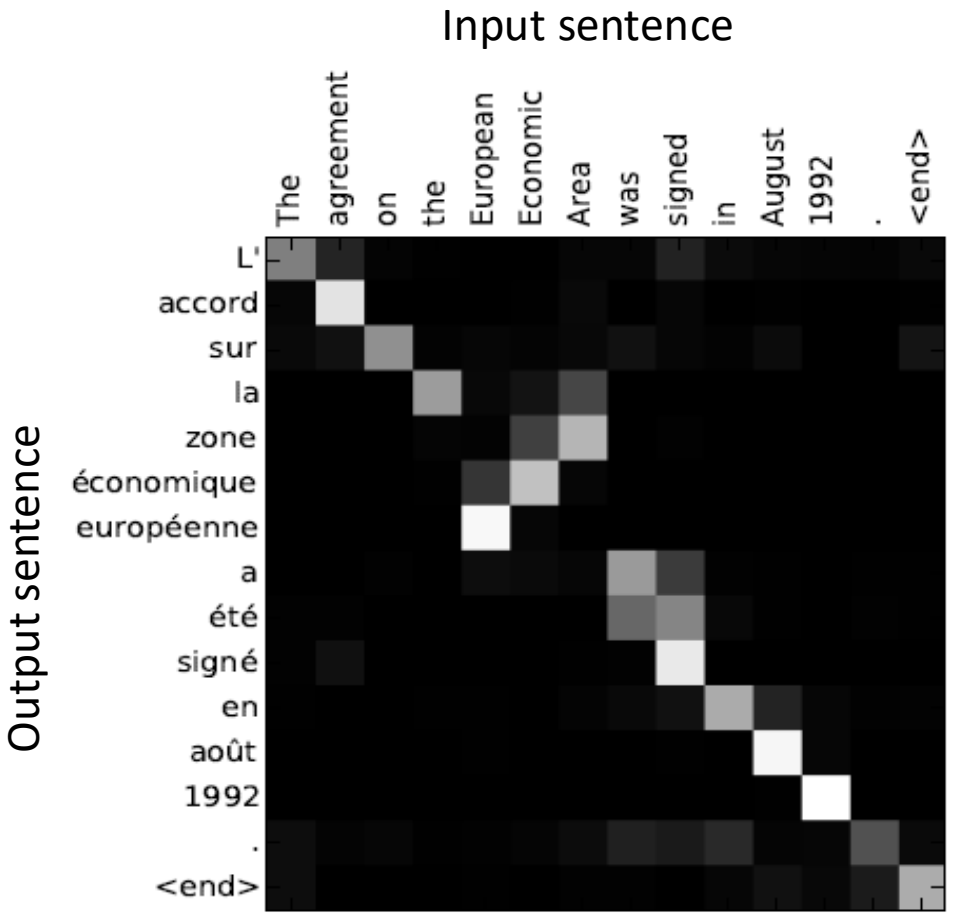


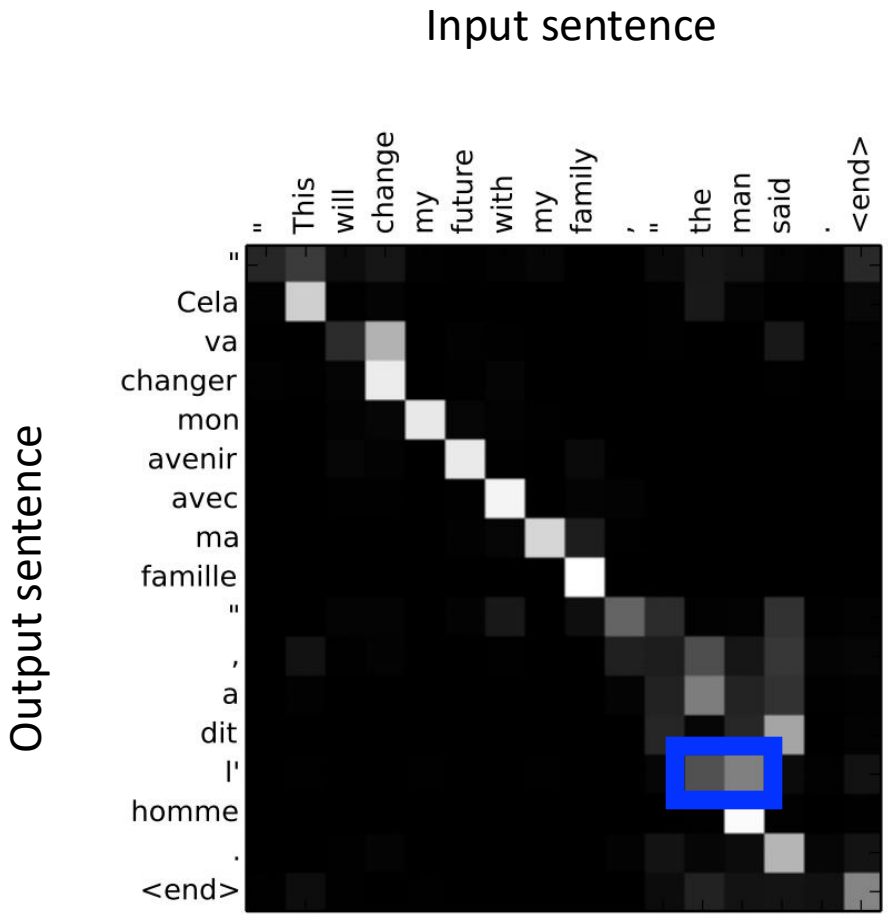


While a linear alignment between input and output sentences is common, there are exceptions (e.g., order of adjectives and nouns can differ)

Bahdanau et al. Neural Machine Translation by Jointly Learning to Align and Translate. ICLR 2015

Visualizing Attention

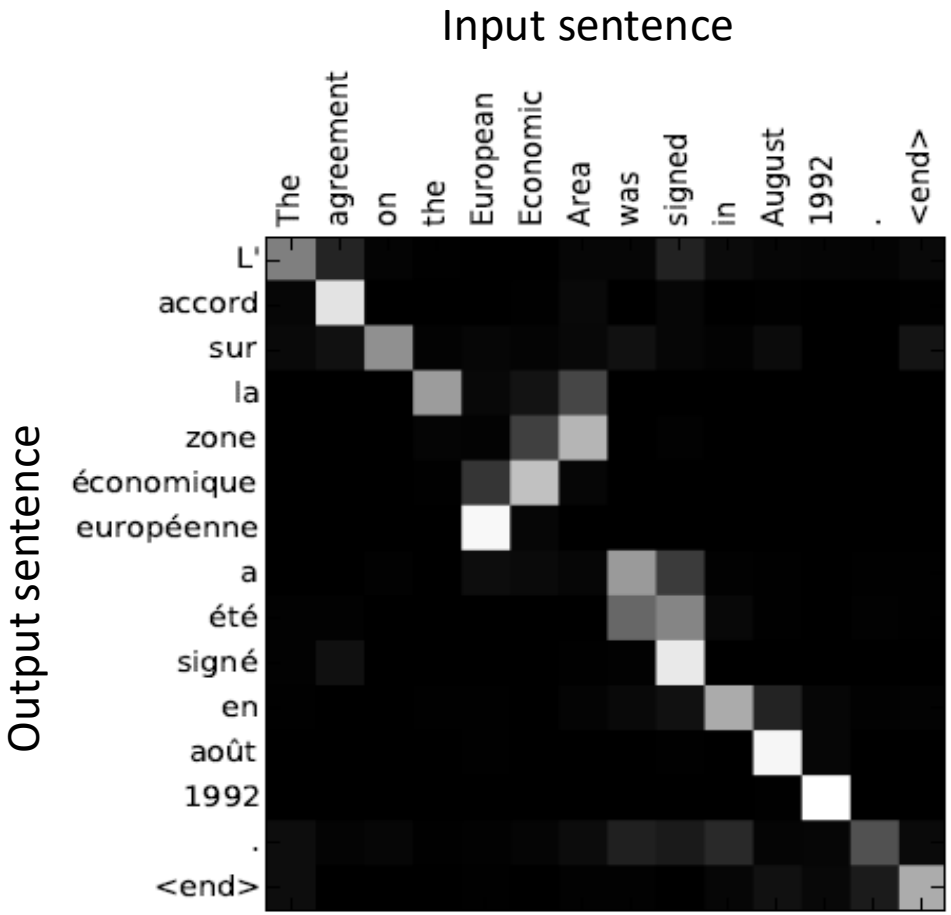


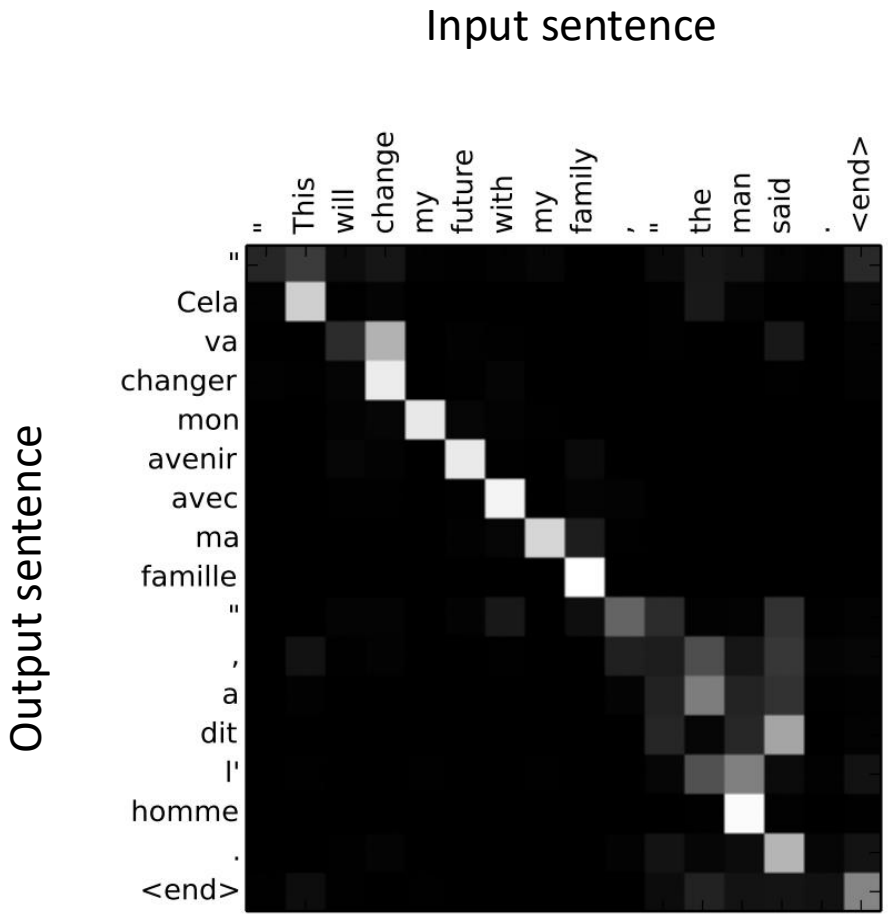


Output words are often informed by more than one input word;
e.g., “man” indicates translation of “the” to l’ instead of le, la, or les

Bahdanau et al. Neural Machine Translation by Jointly Learning to Align and Translate. ICLR 2015

Visualizing Attention





It naturally handles different input and output lengths
(e.g., 1 extra output word for both examples)

Bahdanau et al. Neural Machine Translation by Jointly Learning to Align and Translate. ICLR 2015

Today's Topics

- Motivation: machine neural translation for long sentences
- Encoder
- Decoder: attention
- Performance evaluation
- Final project: ways to find a partner

Final Project

- Requirements on course website: https://dannagurari.colorado.edu/course/neural-networks-and-deep-learning-spring-2025/final-project/

- We will facilitate a 3-stage process to help you find a partner
  - Described in doc linked to from Canvas

Today's Topics

- Motivation: machine neural translation for long sentences
- Encoder
- Decoder: attention
- Performance evaluation
- Final project: ways to find a partner


The End