Deep Learning Lab 13-1: Seq2Seq Learning & Neural Machine Translation

Chia-Hung Yuan & DataLab

Department of Computer Science,
National Tsing Hua University, Taiwan

Outline

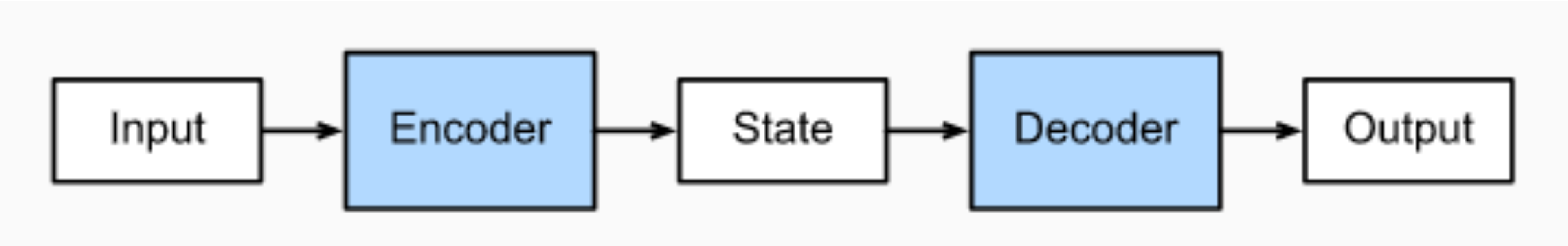
- Encoder-Decoder Model
- Sequence-to-Sequence (Seq2Seq)
- Attention Mechanism
- Teacher Forcing
- Assignment
- Extra Material: Transformer
- Reference

Outline

- Encoder-Decoder Model
- Sequence-to-Sequence (Seq2Seq)
- Attention Mechanism
- Teacher Forcing
- Assignment
- Extra Material: Transformer
- Reference

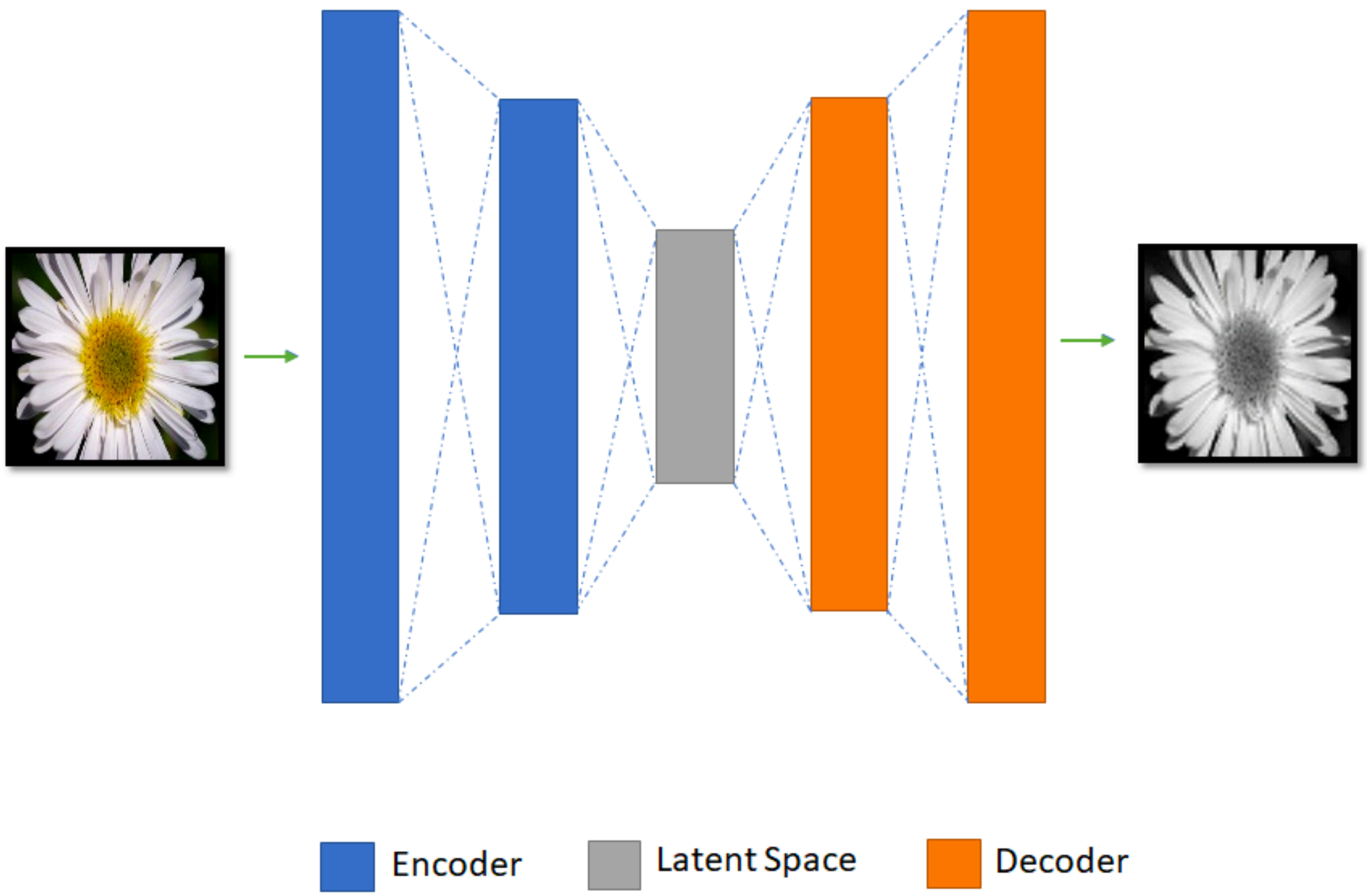
Encoder-Decoder Model

- As we have seen in lab12-2 (AdaIN),
  - Encoder: encode the inputs into **hidden state/context/code**
  - Decoder: the hidden state is passed into the decoder to generate the desired output



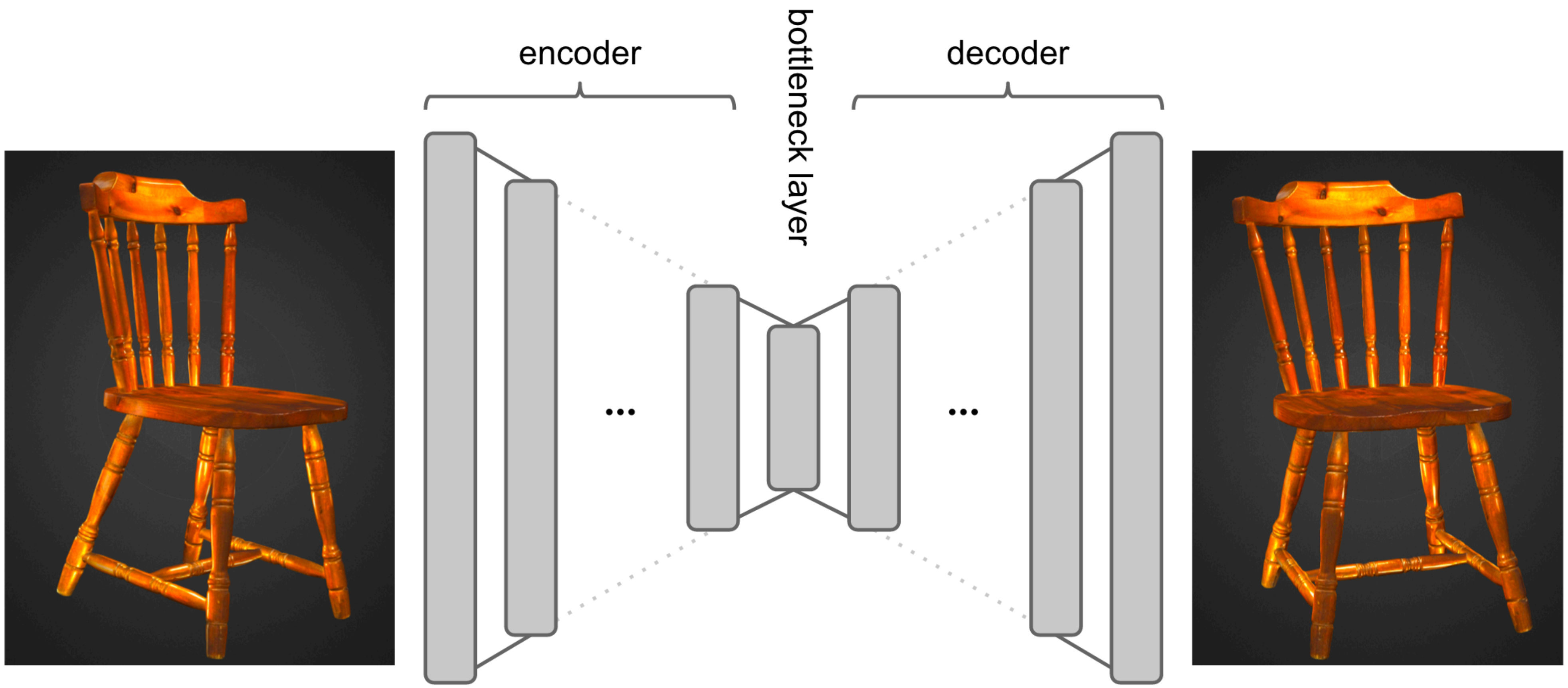
Encoder-Decoder Model

- As we have seen in lab12-2 (AdaIN),
  - Encoder: encode the inputs into **hidden state/context/code**
  - Decoder: the hidden state is passed into the decoder to generate the desired output



Encoder-Decoder Model

- As we have seen in lab12-2 (AdaIN),
  - Encoder: encode the inputs into **hidden state/context/code**
  - Decoder: the hidden state is passed into the decoder to generate the desired output



Outline

- Encoder-Decoder Model
- Sequence-to-Sequence (Seq2Seq)
- Attention Mechanism
- Teacher Forcing
- Assignment
- Extra Material: Transformer

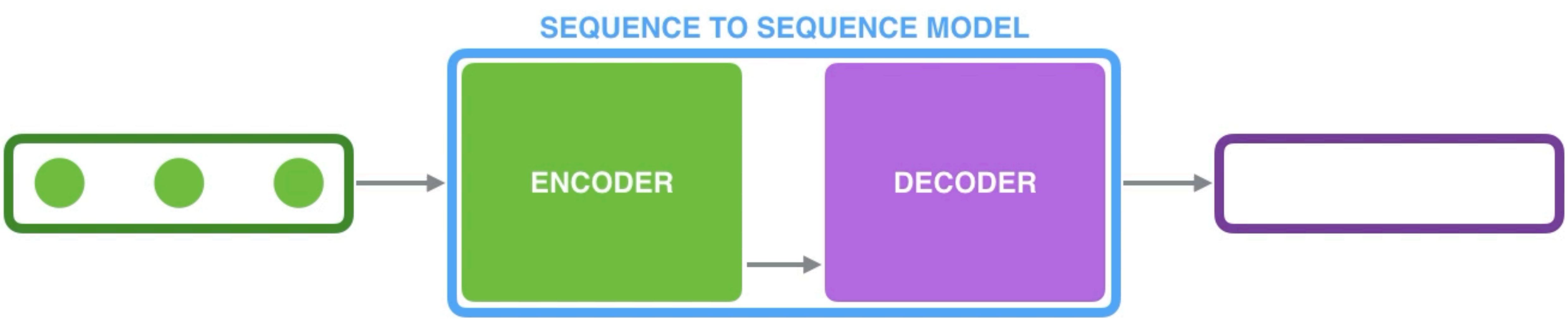
Sequence-to-Sequence (Seq2Seq)

- Sequence-to-Sequence (Seq2Seq) is an architecture based on the encoder-decoder, which transforms an input sequence to the target sequence
  - Both sequences can have arbitrary lengths
  - Have achieved a lot of success in tasks like machine translation, text summarization, and image captioning
  - Google Translate started using such a model in production in late 2016

Google Translate

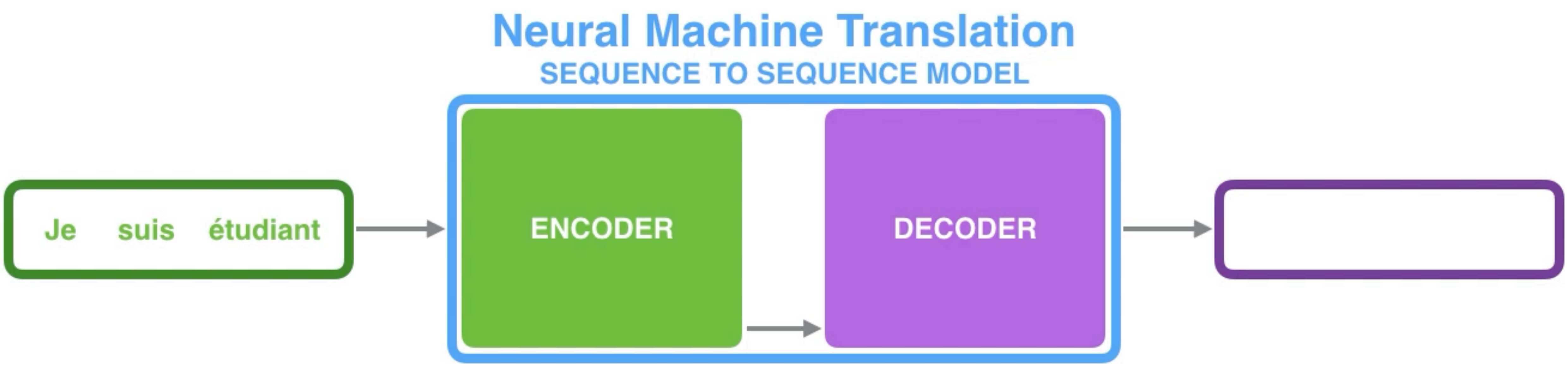
Seq2Seq

- Sequence-to-Sequence (Seq2Seq) is an architecture based on the encoder-decoder, which transforms an input sequence to the target sequence
    - The encoder processes each item in the input sequence, it compiles the information it captures into a vector, called the context. After processing the entire input sequence, the encoder sends the context over to the decoder, which begins producing the output sequence item by item



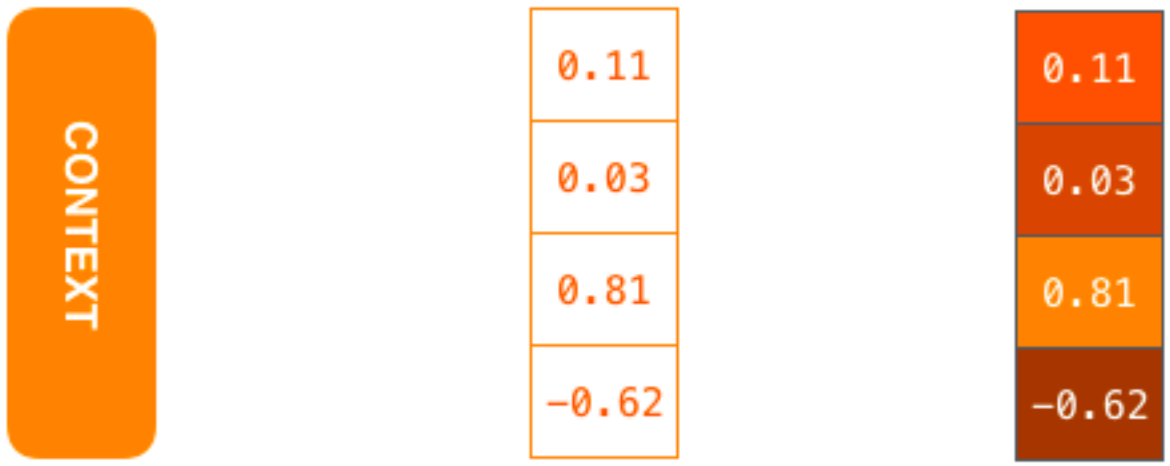
Seq2Seq

- In neural machine translation, a sequence is a series of words, processed one after another. The output is, likewise, a series of words:



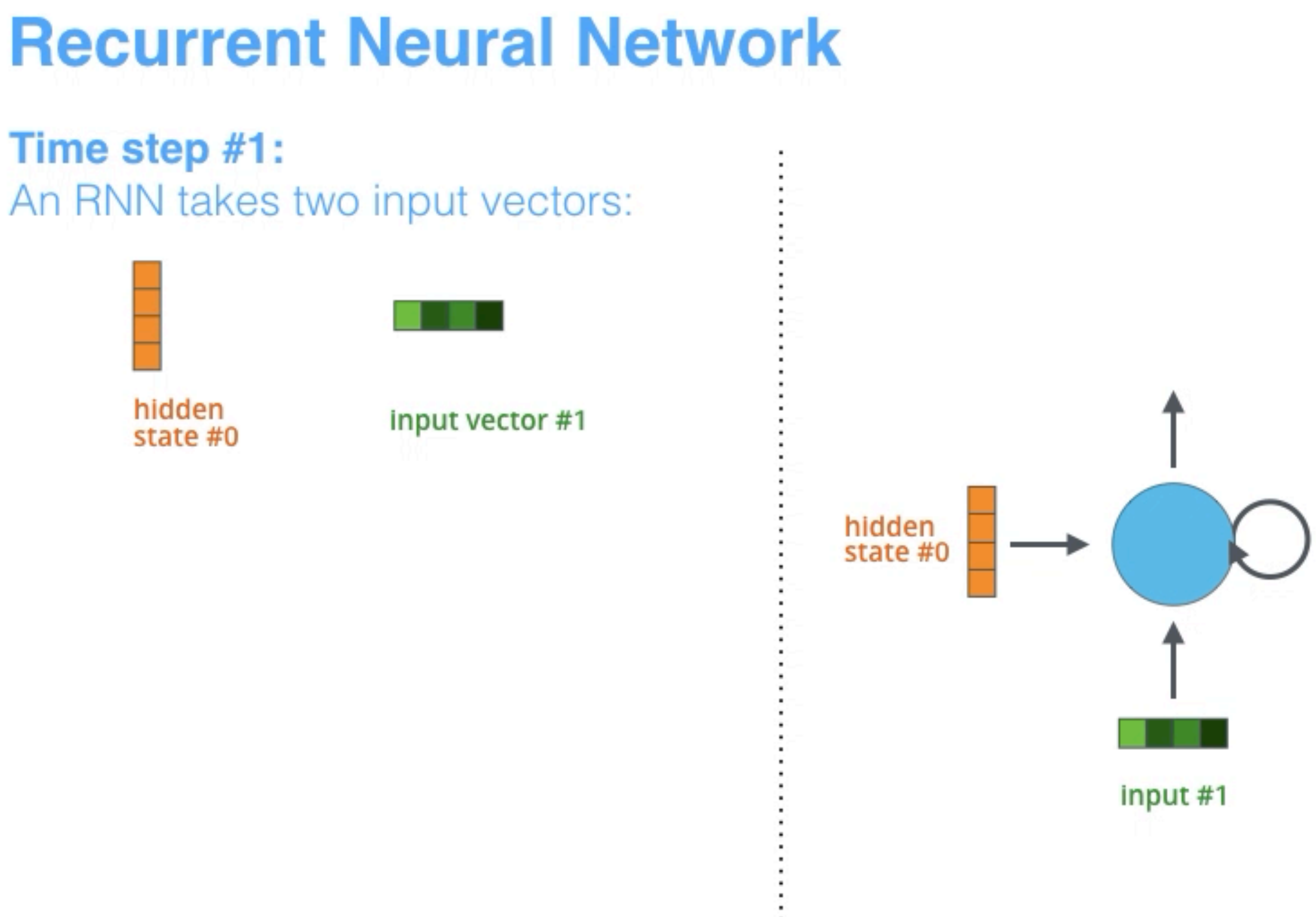
Seq2Seq

- The context is a vector (an array of numbers, basically) in the case of machine translation. The encoder and decoder tend to both be recurrent neural networks (RNNs)
  - You can set the size of the context vector when you set up your model. It is basically the number of hidden units in the encoder RNN



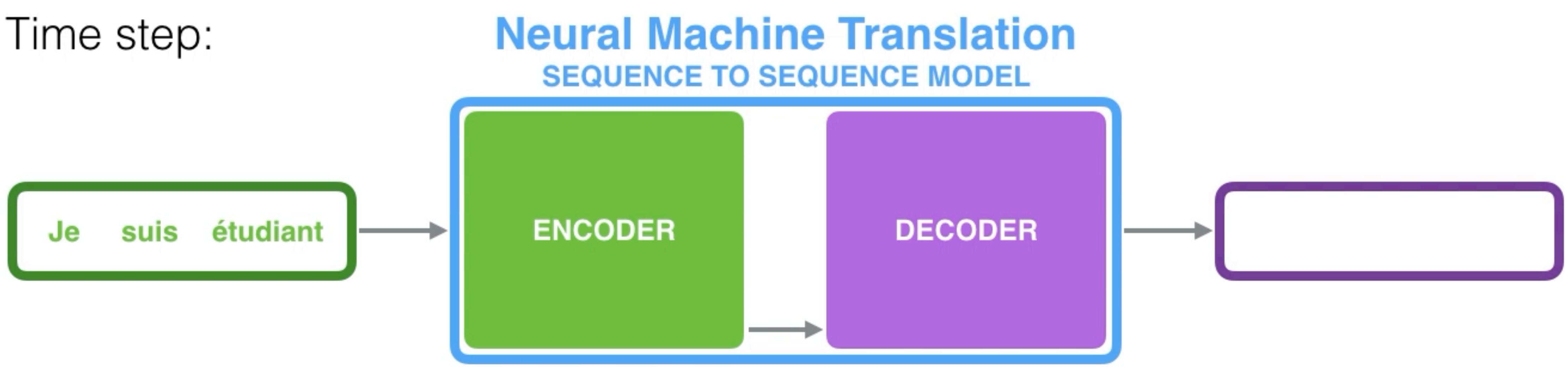
Seq2Seq

- By design, a RNN takes two inputs at each time step: an input (in the case of the encoder, one word from the input sentence), and a hidden state



Seq2Seq

- Since the encoder and decoder are both RNNs, each time step one of the RNNs does some processing, it updates its hidden state based on its inputs and previous inputs it has seen
    - Notice the last hidden state is actually the context we pass along to the decoder



Outline

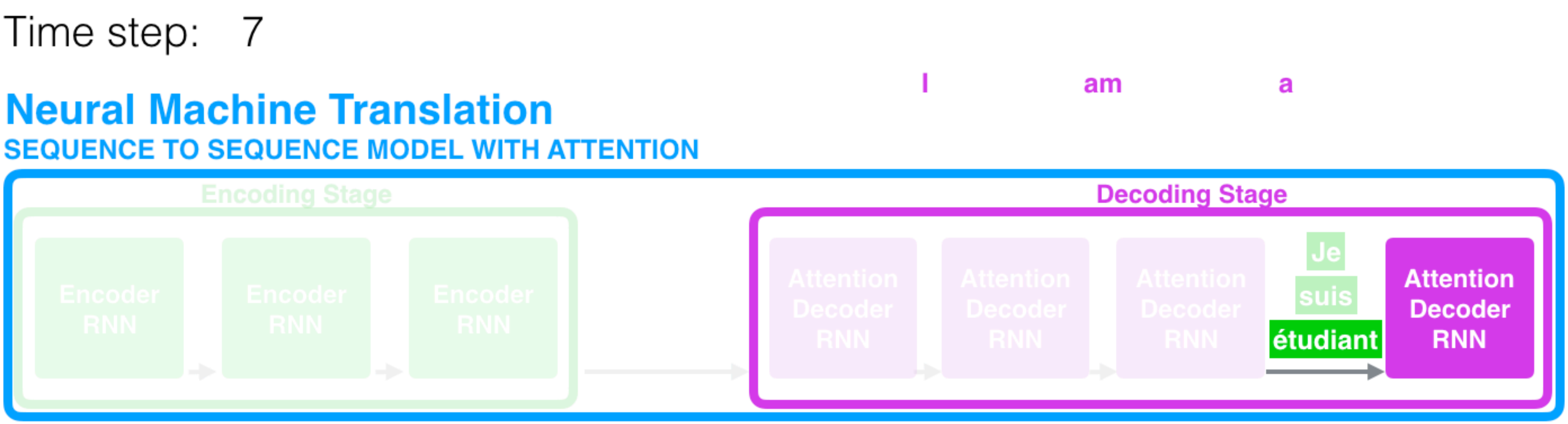
- Encoder-Decoder Model
- Sequence-to-Sequence (Seq2Seq)
- Attention Mechanism
- Teacher Forcing
- Assignment
- Extra Material: Transformer
- Reference

Why Attention?

- The **fixed-length** context vector turned out to be a bottleneck for these types of models. It made it challenging for the models to deal with long sentences
    - It is hard for the fixed-length context vector to store all information as the input sequence getting longer and longer
    - It has often forgotten the earlier part of input once it processed the whole input sequence

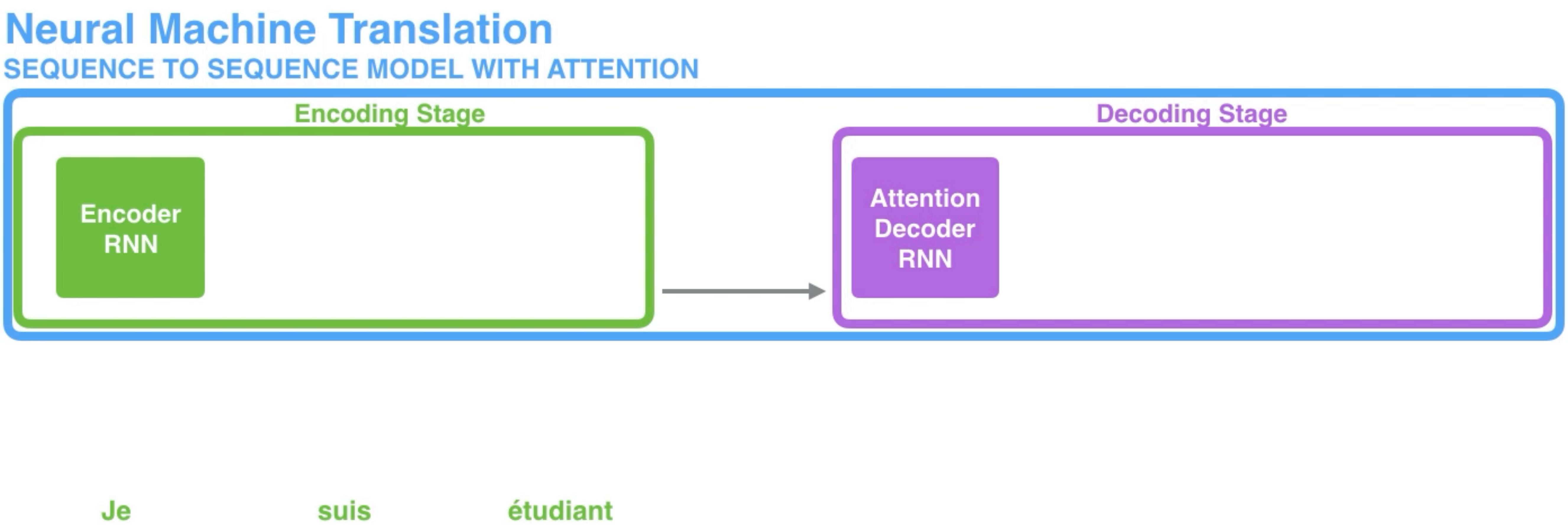
Attention Mechanism

- Attention allows the model to focus on the relevant parts of the input sequence as needed
    - The decoder can focus on different part of the input sequence at each time step, in order to make a better prediction



Attention Mechanism

1. First, the encoder passes a lot more data to the decoder. Instead of passing the last hidden state of the encoding stage, the encoder passes **all** the hidden states to the decoder:



Attention Mechanism

2. Second, an attention decoder does an extra step before producing its output. In order to focus on the parts of the input that are relevant to this decoding time step, the decoder does the following:

    1. Look at the set of encoder hidden states it received – each encoder hidden states is most associated with a certain word in the input sentence

    2. Give each hidden states a score (let's ignore how the scoring is done for now)

    3. Multiply each hidden states by its softmax score, thus amplifying hidden states with high scores, and drowning out hidden states with low scores

Attention Mechanism

Attention at time step 4

Attention Mechanism

- Let us now bring the whole thing together in the following visualization and look at how the attention process works:

    1. The attention decoder RNN takes in the embedding of the <END> token, and an initial decoder hidden state

    2. The RNN processes its inputs, producing an output and a new hidden state vector (h4). The output is discarded

    3. Attention Step: We use the encoder hidden states and the h4 vector to calculate a context vector (C4) for this time step

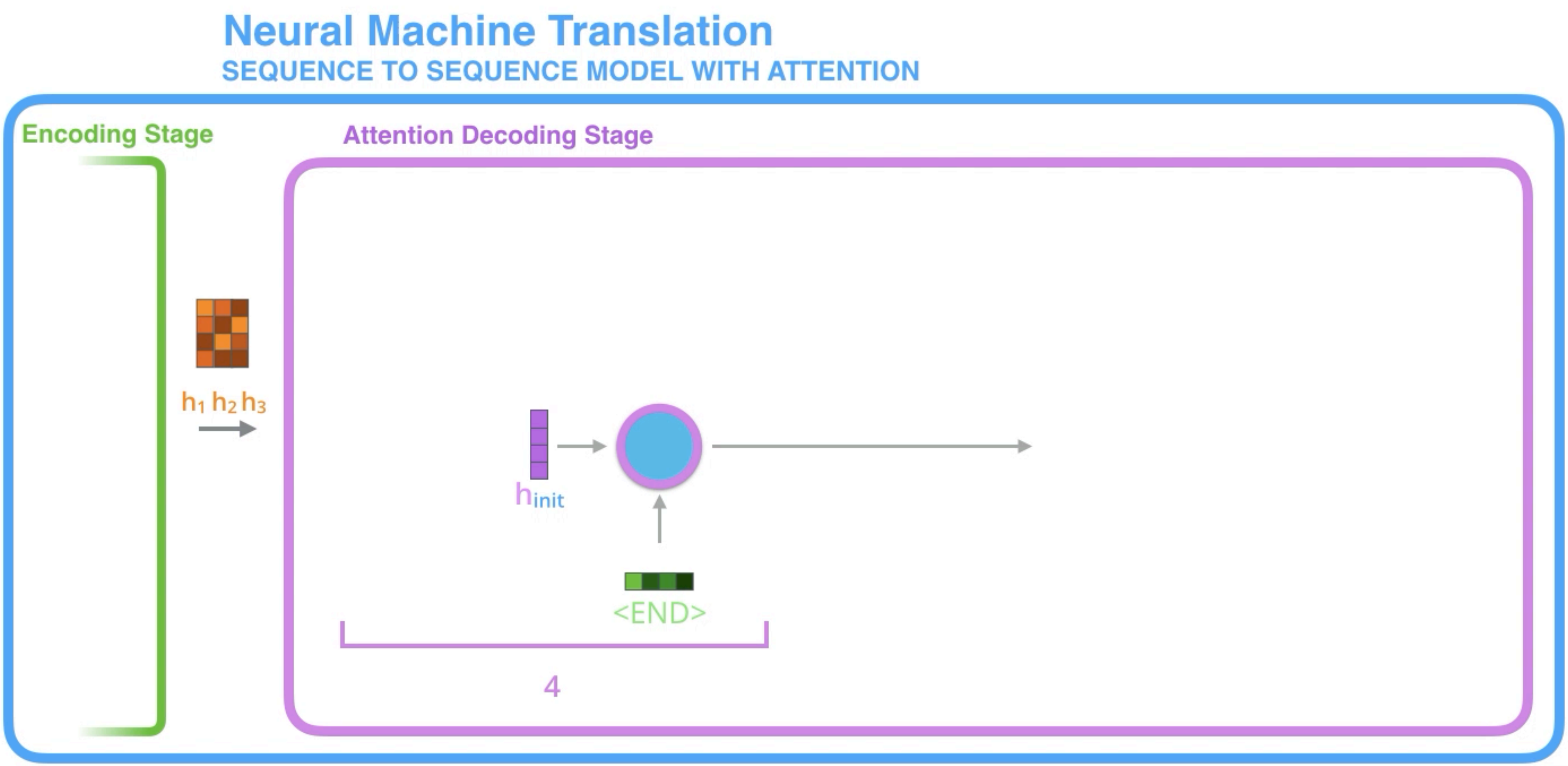
    4. We concatenate h4 and C4 into one vector

    5. We pass this vector through a feedforward neural network (one trained jointly with the model)

    6. The output of the feedforward neural networks indicates the output word of this time step

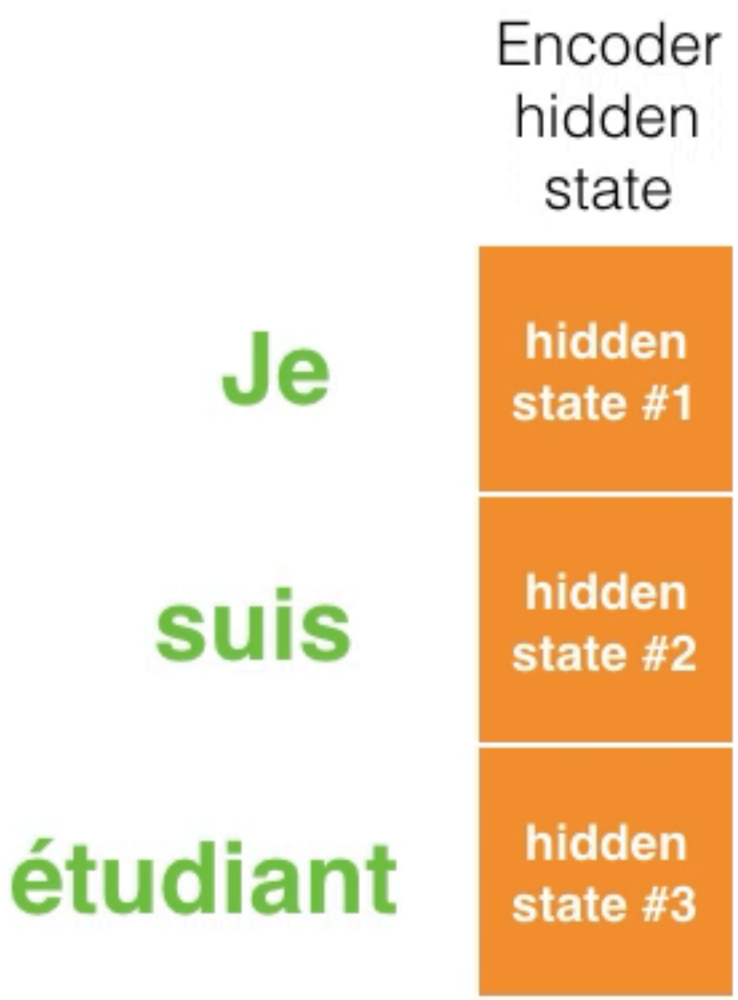
    7. Repeat for the next time steps

Attention Mechanism



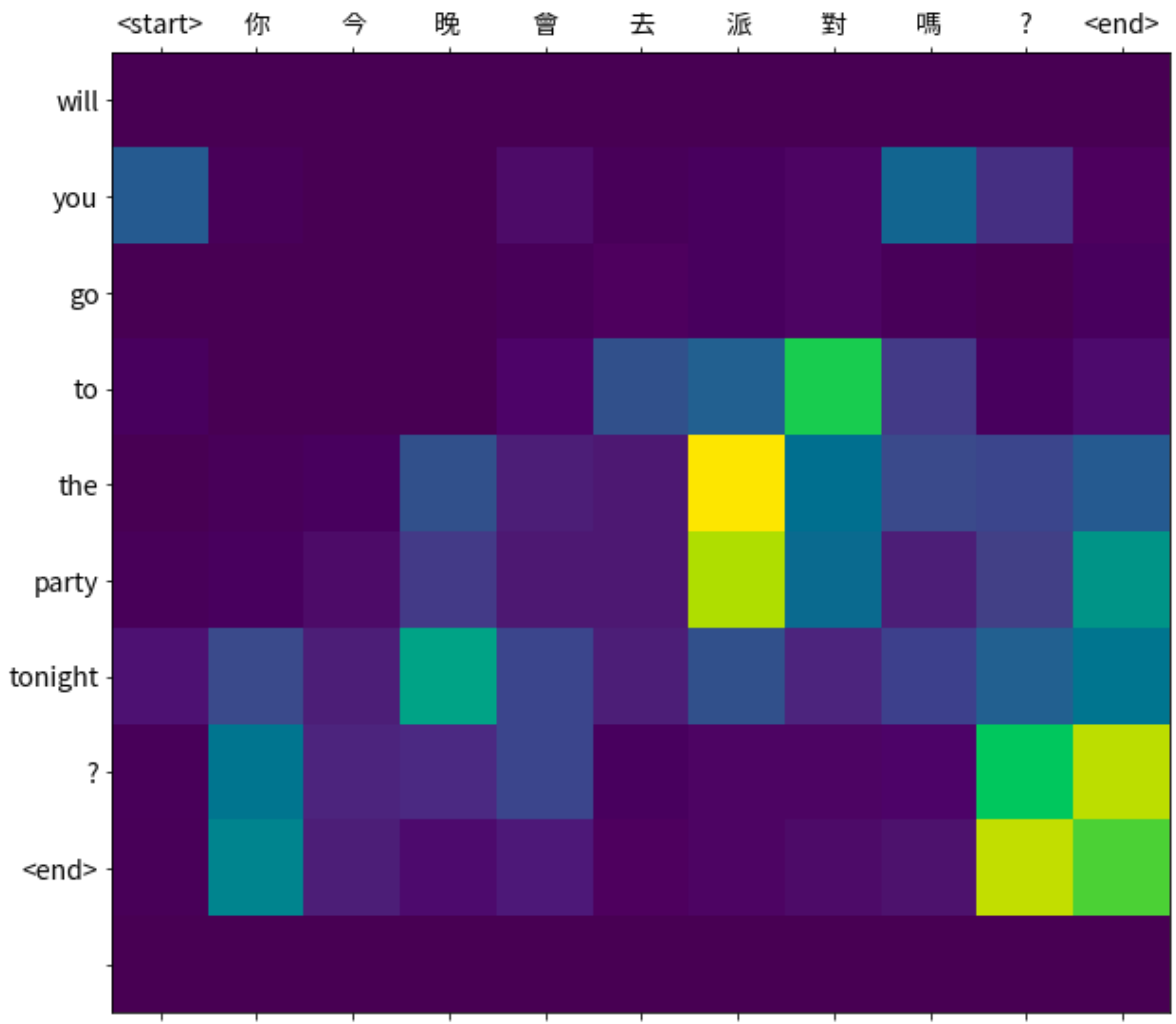
Attention Mechanism

- This is another way to look at which part of the input sentence we’re paying attention to at each decoding step:



Attention Mechanism

- It actually learned from the training phase how to align words in that language pair



Attention Mechanism

- Let's dive into math!
  - Score: $e_{ij} = a(s_{i-1}, h_j)$
  - Score after softmax: $\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})}$
  - Context vector: $c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j$

Attention Mechanism

- How to calculate the score $e_{ij} = a(s_{i-1}, h_j)$?
  - , where $a(\cdot)$ is the dot product in the following example

```
decoder_hidden = [10, 5, 10]

encoder_hidden  score
---------------------
     [0, 1, 1]     15 (= 10×0 + 5×1 + 10×1, the dot product)
     [5, 0, 1]     60
     [1, 1, 0]     15
     [0, 5, 1]     35
```

Attention Mechanism

- Score after softmax $\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})}$ is straightforward

```
encoder_hidden  score  score^
-----------------------------
     [0, 1, 1]     15       0
     [5, 0, 1]     60       1
     [1, 1, 0]     15       0
     [0, 5, 1]     35       0
```

Attention Mechanism

- Finally, we can multiply each hidden state of encoder by its score and sum up the alignment vectors to get the context vector: $c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j$

```
encoder   score   score^   alignment
-----------------------------------
[0, 1, 1]   15      0   [0, 0, 0]
[5, 0, 1]   60      1   [5, 0, 1]
[1, 1, 0]   15      0   [0, 0, 0]
[0, 5, 1]   35      0   [0, 0, 0]


context = [0+5+0+0, 0+0+0+0, 0+1+0+0] = [5, 0, 1]
```

Attention Mechanism

- There are many well-known score functions as follows
  - h represents hidden state of encoder
  - s represents decoder hidden states

Name	Alignment score function	Citation
Content-base attention	$\text{score}(\boldsymbol{s}_t, \boldsymbol{h}_i) = \text{cosine}[\boldsymbol{s}_t, \boldsymbol{h}_i]$	Graves2014
Additive(*)	$\text{score}(\boldsymbol{s}_t, \boldsymbol{h}_i) = \mathbf{v}_a^\top \tanh(\mathbf{W}_a[\boldsymbol{s}_t; \boldsymbol{h}_i])$	Bahdanau2015
Location-Base	$\alpha_{t,i} = \text{softmax}(\mathbf{W}_a \boldsymbol{s}_t)$ Note: This simplifies the softmax alignment to only depend on the target position.	Luong2015
General	$\text{score}(\boldsymbol{s}_t, \boldsymbol{h}_i) = \boldsymbol{s}_t^\top \mathbf{W}_a \boldsymbol{h}_i$ where $\mathbf{W}_a$ is a trainable weight matrix in the attention layer.	Luong2015
Dot-Product	$\text{score}(\boldsymbol{s}_t, \boldsymbol{h}_i) = \boldsymbol{s}_t^\top \boldsymbol{h}_i$	Luong2015
Scaled Dot-Product(^)	$\text{score}(\boldsymbol{s}_t, \boldsymbol{h}_i) = \frac{\boldsymbol{s}_t^\top \boldsymbol{h}_i}{\sqrt{n}}$ Note: very similar to the dot-product attention except for a scaling factor; where n is the dimension of the source hidden state.	Vaswani2017

Attention Mechanism

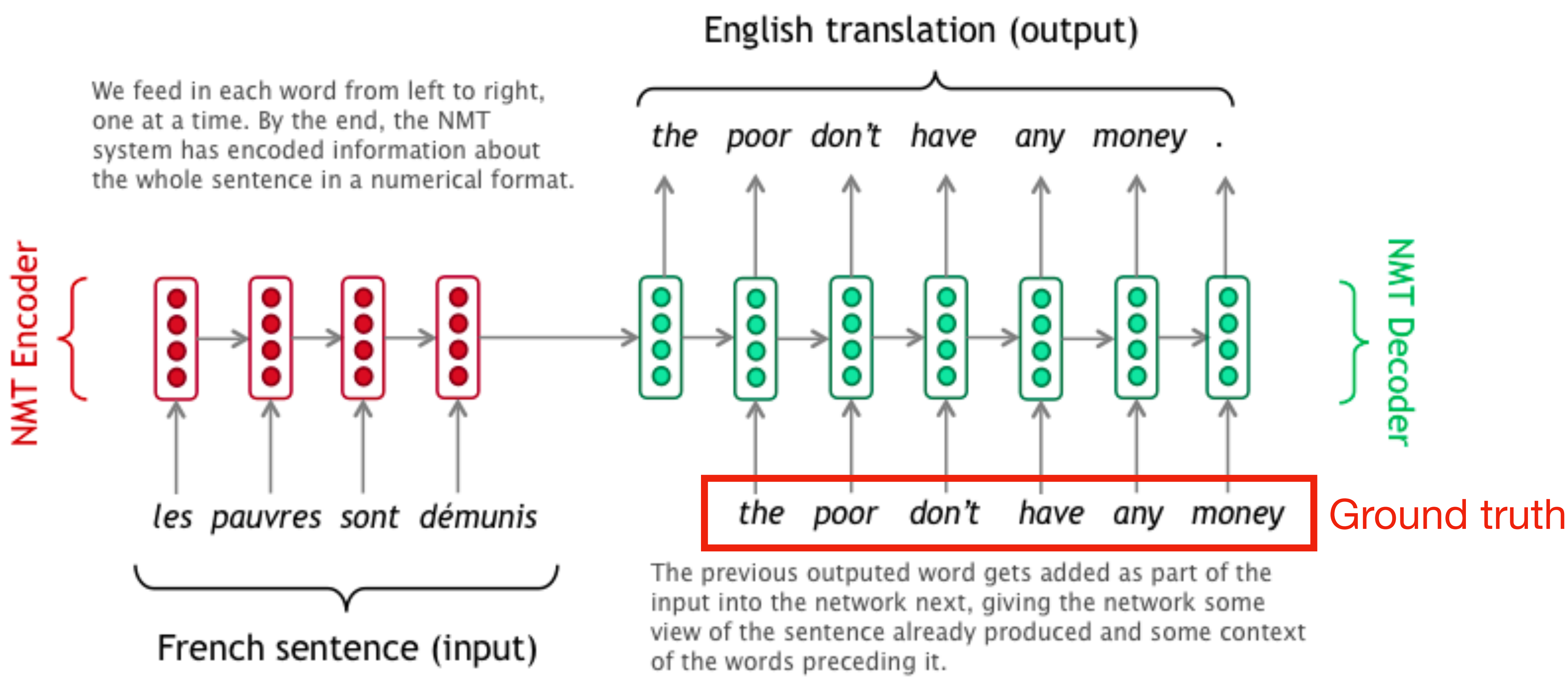
- Attention mechanism allows the decoder to focus on various part of input sequence, instead of forcing it to encode all information into one fixed-length vector
  - Pros: model interpretability, better performance
  - Cons: computationally expensive

Outline

- Encoder-Decoder Model
- Sequence-to-Sequence (Seq2Seq)
- Attention Mechanism
- Teacher Forcing
- Assignment
- Extra Material: Transformer
- Reference

Teacher Forcing

- Teacher forcing is a method for quickly and efficiently training recurrent neural network models that use the ground truth from a prior time step as input
  - Accelerate the convergence speed
  - Stabilize the training process



Outline

- Encoder-Decoder Model
- Sequence-to-Sequence (Seq2Seq)
- Attention Mechanism
- Teacher Forcing
- Assignment
- Extra Material: Transformer
- Reference

Assignment

- There are two parts in the notebook
  - Part I: neural machine translation
  - Part II: sentiment analysis (assignment)

Neural Machine Translation

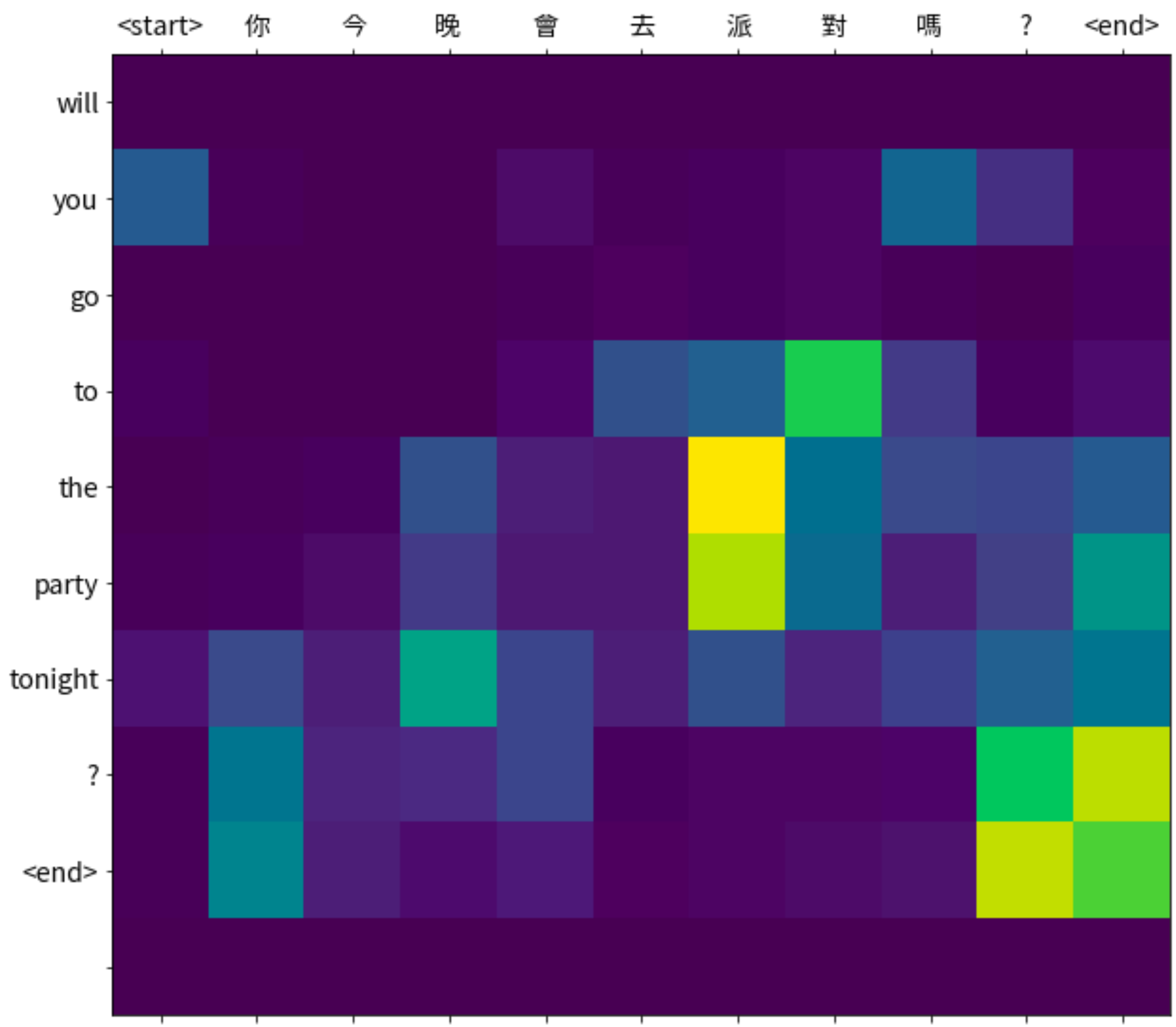
- Overview
  - Task: translate the sentence from Chinese to English
  - Dataset size: 20289
  - Encoder: RNN with GRU cell
  - Decoder: RNN with GRU cell
  - Attention machanism: Bahdanau Attention

$$\text{score}(s_t, h_i) = v_a^T \tanh(W_a[s_t; h_i])$$

  - It is worth noticing that in GRU, **the hidden state and the output are same**

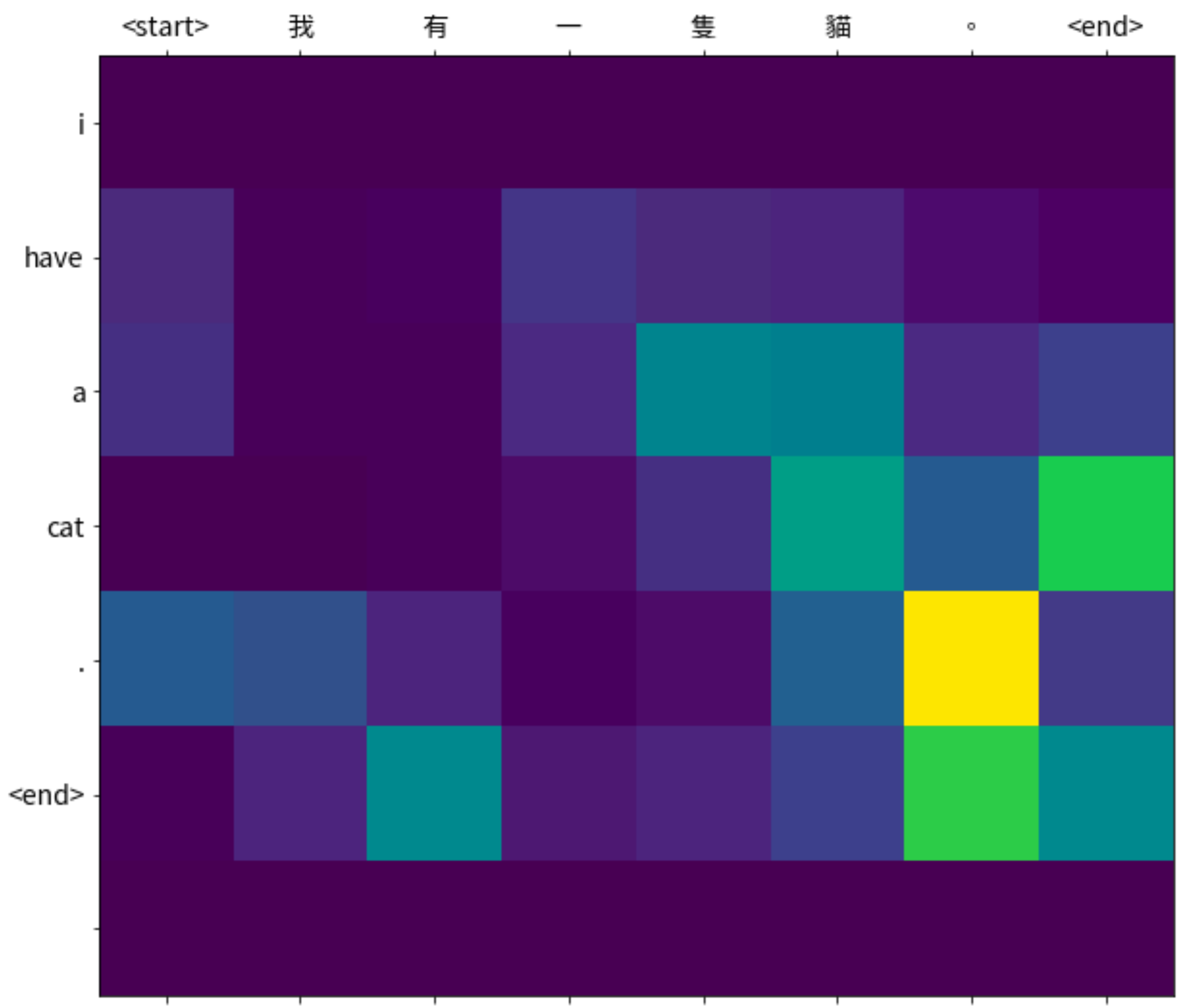
Neural Machine Translation

- It's a toy model with toy dataset. Here we focus on the main idea behind the model



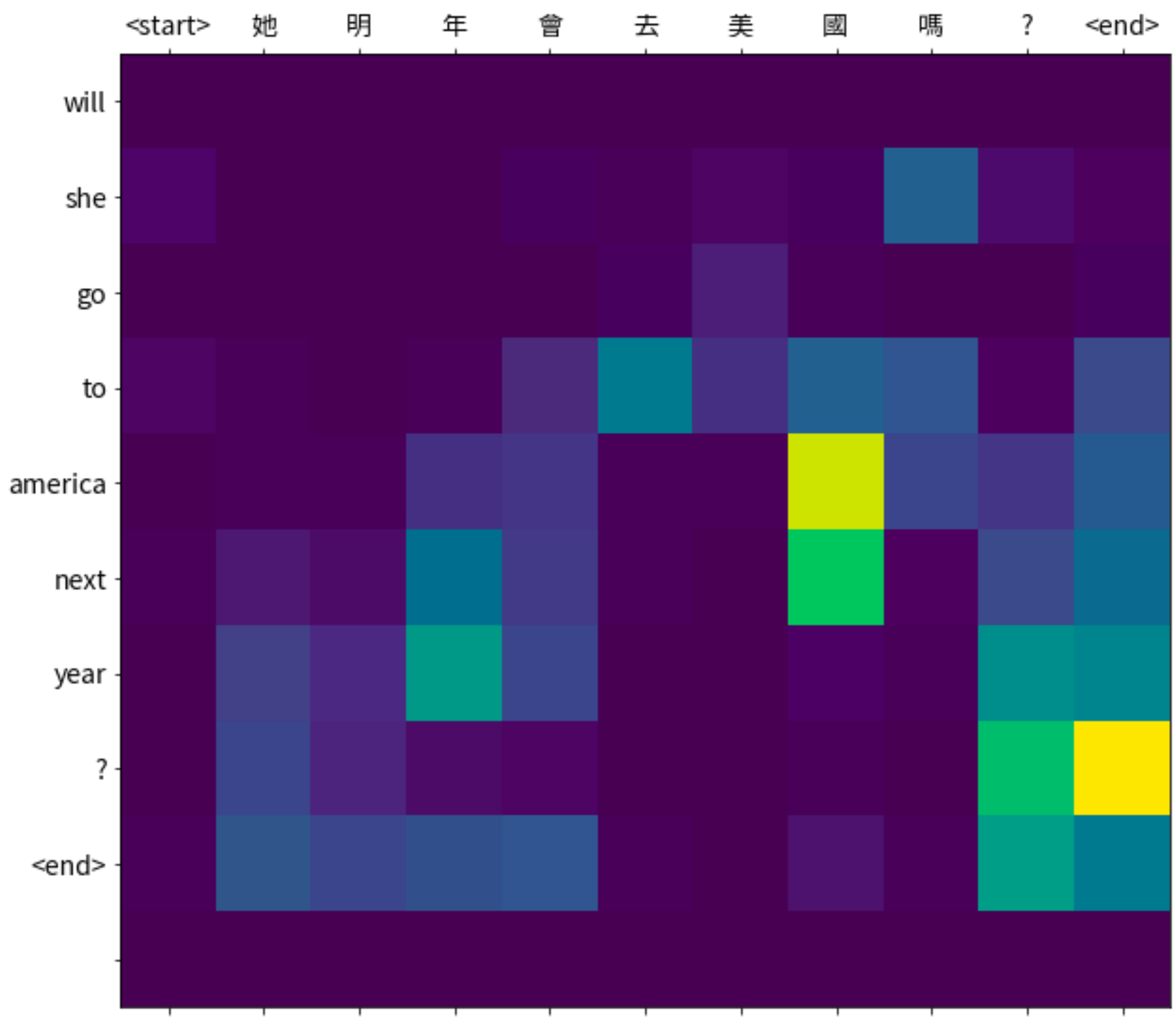
Neural Machine Translation

- It's a toy model with toy dataset. Here we focus on the main idea behind the model



Neural Machine Translation

- It's a toy model with toy dataset. Here we focus on the main idea behind the model



Neural Machine Translation

- In the plotting function, you need to change the path of the Chinese font. Otherwise, the Chinese character will not be displayed in the plot

```
def plot_attention(attention, sentence, predicted_sentence):
    # you need to change the fname based on your system, and the Chinese can be displayed in the plot
    font = FontProperties(fname=r"./data/TaipeiSansTCBeta-Regular.ttf", size=14)
```

Sentiment Analysis

- Overview
  - Task: predict whether the comment is positive or negative
  - Dataset: IMDB
  - Dataset size: 50000
  - Encoder: RNN with GRU cell
  - Decoder: 4 fully-connected layers
  - Attention machanism: Luong Attention

	review	sentiment
0	One of the other reviewers has mentioned that ...	positive
1	A wonderful little production. The...	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

TODO

- Implement the **Luong Attention**, where the formula of the score function is:

$$\text{score}(s_t, h_i) = s_t^T W_a h_i$$

  - h_i: hidden state of the encoder
  - s_t: hidden state of the decoder
  - W_a: the trainable weights

Demo

- This simple model achieves ~84.5% accuracy with only 10 epochs! Not bad at all!

- Besides the nice accuracy, let's try to do some more fascinating things. How about **visualize** our results?

Demo

Positive

y_true: 1
y_predict: 1
jane austen would definitely of this one paltrow does an awesome job capturing the attitude of emma she is funny without being silly yet elegant she puts on very convincing british accent n ot being british myself maybe m not the best judge but she fooled me she was also excellent in doors sometimes forget she american also brilliant are jeremy northam and sophie thompson and law emma thompson sister and mother as the bates women they nearly steal the show and ms law d oesn even have any lines highly recommended

Negative

y_true: 0
y_predict: 0
reaches the point where they become obnoxious and simply frustrating touch football puzzle fami ly and talent shows are not how actual people behave it almost sickening another big flaw is th e woman carell is supposed to be falling for her in her first scene with steve carell is like w atching stroke victim trying to be what imagine is supposed to be unique and original in this w oman comes off as mildly retarded it makes me think that this movie is taking place on another planet left the theater wondering what just saw after thinking further don think it was much

Requirement

- The accuracy should be at least **0.80**
- Show the **10-most-focused words** in the sentence
- Only need to show the **first 10 results** in the test data
- Submit on iLMS your code file `Lab13-1_{student id}.ipynb`
- No need to submit the checkpoints file, but you should show the results in the notebook
- Deadline: **2020-12-03(Thur) 23:59**

Outline

- Encoder-Decoder Model
- Sequence-to-Sequence (Seq2Seq)
- Attention Mechanism
- Teacher Forcing
- Assignment
- Extra Material: Transformer
- Reference

Attention Is All You Need

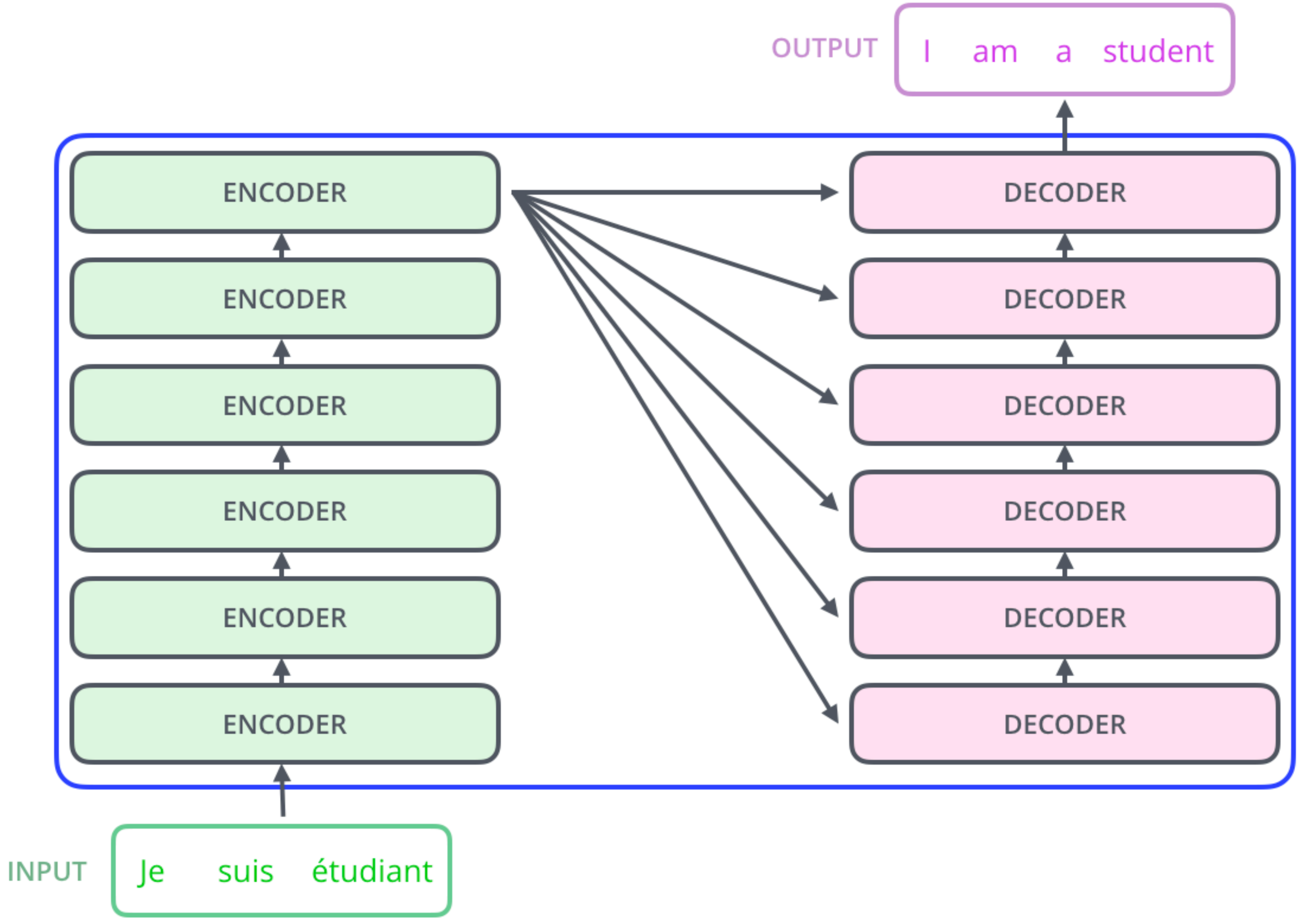
- The authors proposed the Transformer – a model that uses attention to **boost the performance** and **shorten the required training time**
- The Transformer is a **stand-alone attention-based model**, which entirely replace recurrence with self-attention

A High-Level Look

- Let’s begin by looking at the model as a single black box. In a machine translation application, it would take a sentence in one language, and output its translation in another

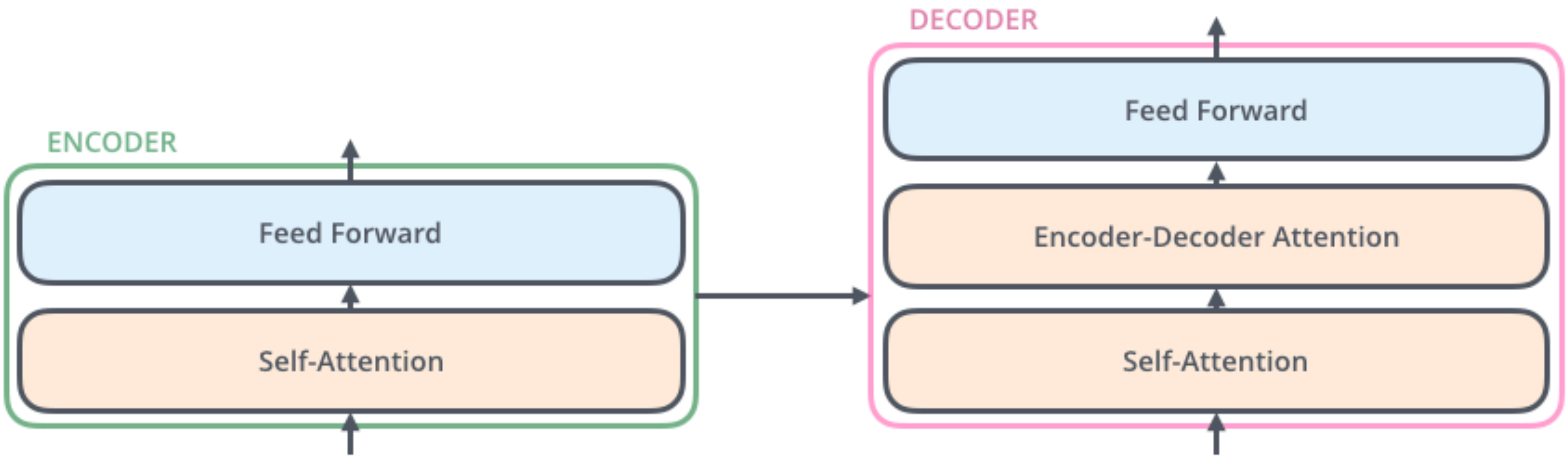
A High-Level Look

- The encoding component is a stack of encoders, while the decoding component is a stack of decoders of the same number



A High-Level Look

- The encoders are all identical in structure. Each one is broken down into two sub-layers, self-attention layer and feed-forward network



Self-Attention at a High Level

- A self-attention layer helps the encoder look at other words in the input sentence as it encodes a specific word, capturing long-distance interactions

- Different from tradition attention mechanism, self-attention is defined as attention applied to single context instead of across multiple contexts

Self-Attention at a High Level

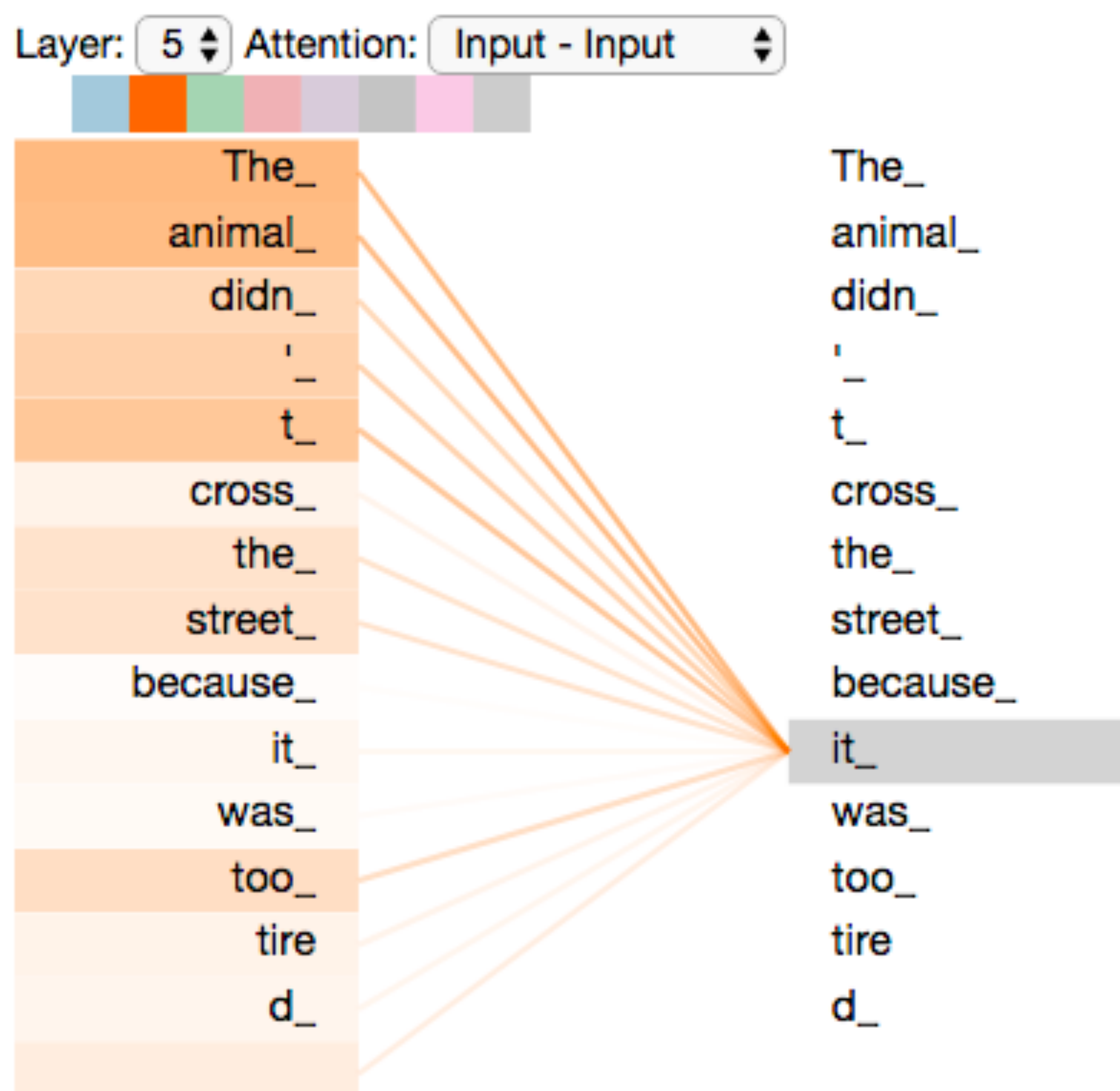
- For example, let's say the following sentence is an input sentence we want to translate:

The animal didn't cross the street because it was too tired.

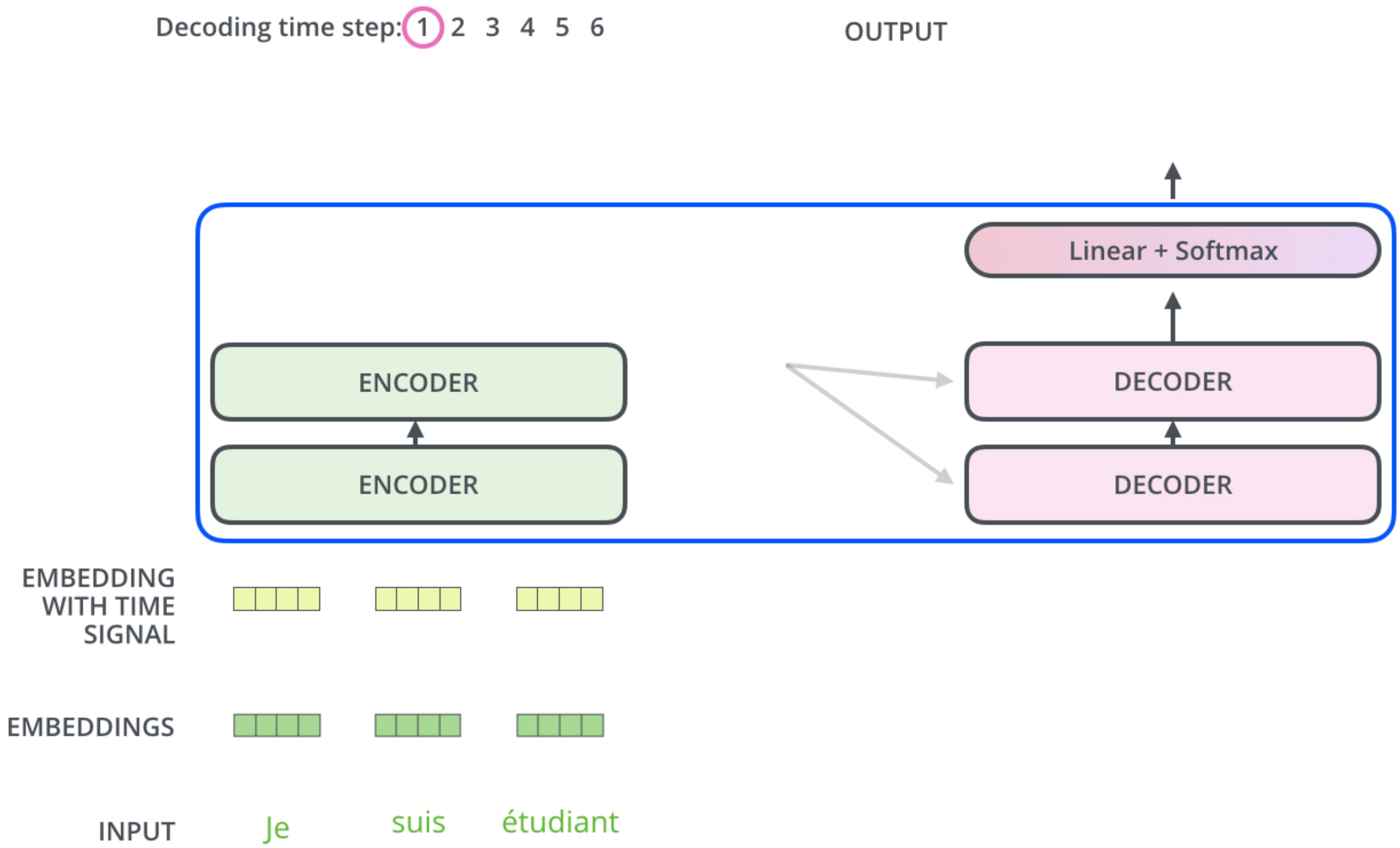
- What does "it" in this sentence refer to? Is it referring to the street or to the animal? It's a simple question to a human, but not as simple to an algorithm

Self-Attention at a High Level

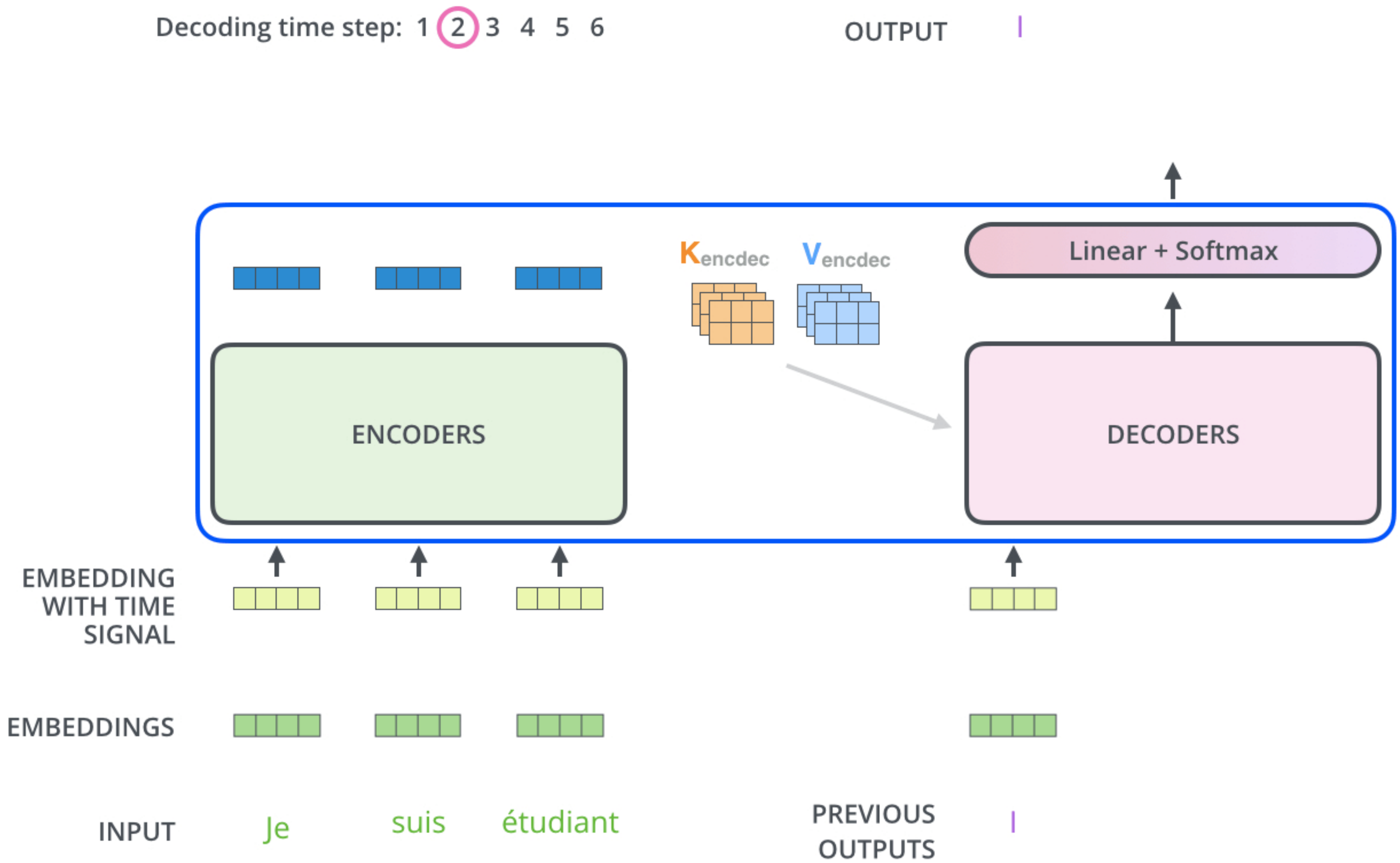
- As the model processes each word, self-attention allows it to look at other positions in the input sequence for clues that can help lead to a better encoding for this word



Self-Attention at a High Level



Self-Attention at a High Level



Transformer

- Self-attention is a powerful mechanism and has been already widely used in many applications including natural language processing (i.e. BERT, GPT-3, XLNet), signal processing

- For more details, please refer to the main paper

Outline

- Encoder-Decoder Model
- Sequence-to-Sequence (Seq2Seq)
- Attention Mechanism
- Teacher Forcing
- Assignment
- Extra Material: Transformer
- Reference

Reference

- Sequence to Sequence Learning with Neural Networks, I. Sutskever et al., NeurIPS'14
- Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation, K. Cho et al., EMNLP'14
- Neural Machine Translation by Jointly Learning to Align and Translate, D. Bahdanau et al., ICLR'15
- Effective Approaches to Attention-based Neural Machine Translation, M. T. Luong, EMNLP'15
- Attention Is All You Need, Google Brain, NeurIPS' 17
- Visualizing A Neural Machine Translation Model
- The Illustrated Transformer