Neural Machine Translation

CSE 447 / 517

March 10th, 2022 (Week 10)

Logistics

- A9 is due **tomorrow 11:59 PM** (March 11th, 2022)

Agenda

- Neural Machine Translation
- Quiz 9
- Q & A

Neural Machine Translation (NMT)

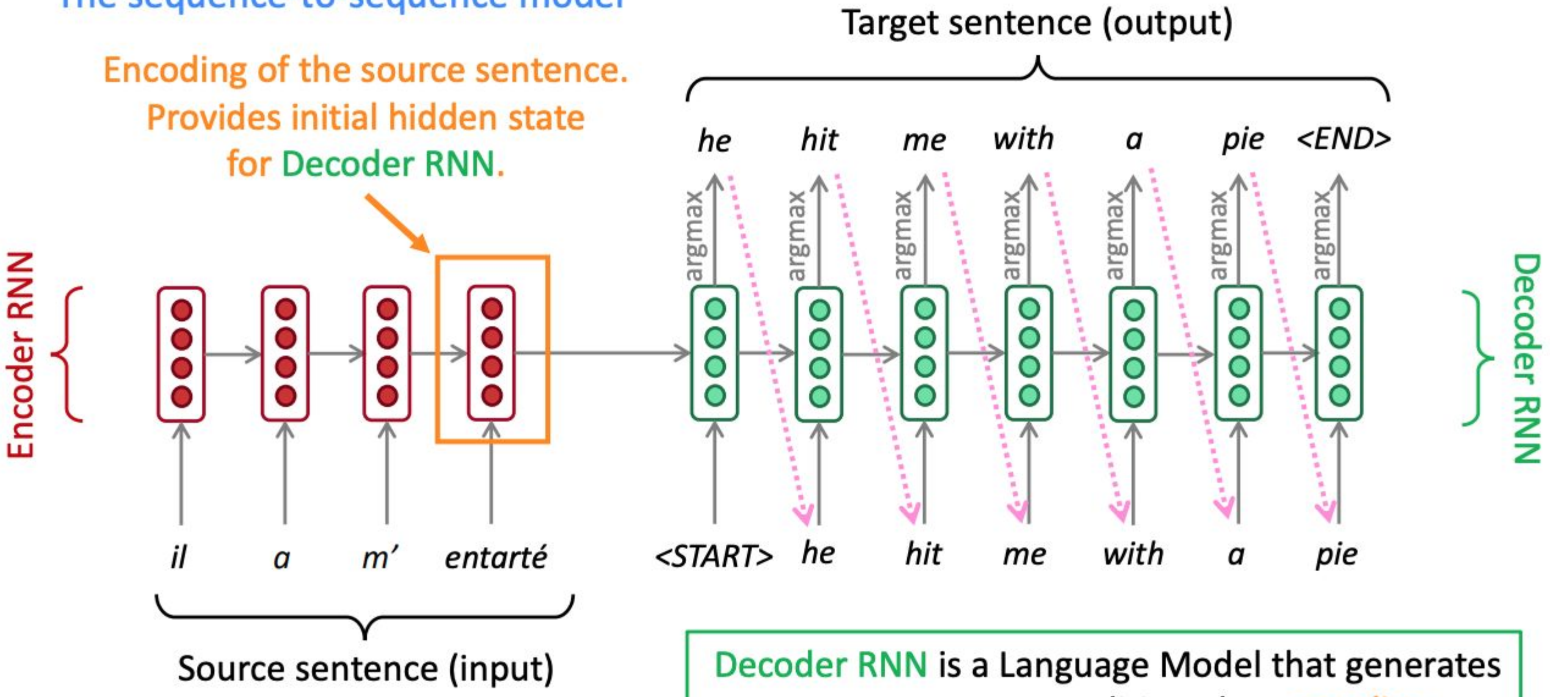
- Based on new model archetype: **seq-to-seq** or **encoder-decoder**
- High-level model:

$$p(\boldsymbol{E} = \mathbf{e} \mid \boldsymbol{f}) = p(\boldsymbol{E} = \mathbf{e} \mid \text{encode}(\boldsymbol{f}))$$
$$= \prod_{j=1}^{\ell} p(e_j \mid e_0, \ldots, e_{j-1}, \text{encode}(\boldsymbol{f}))$$

- The model has two parts:
    - **Encoder** that takes in the source language sentence **f** and outputs an encoding of the sentence **encode(f)**
    - **Decoder** that at step **j** predicts the target language word $\mathbf{e_j}$ from the previously output target language words $\mathbf{e_{<j}}$ and **encode(f)**

Neural Machine Translation (NMT)

The sequence-to-sequence model

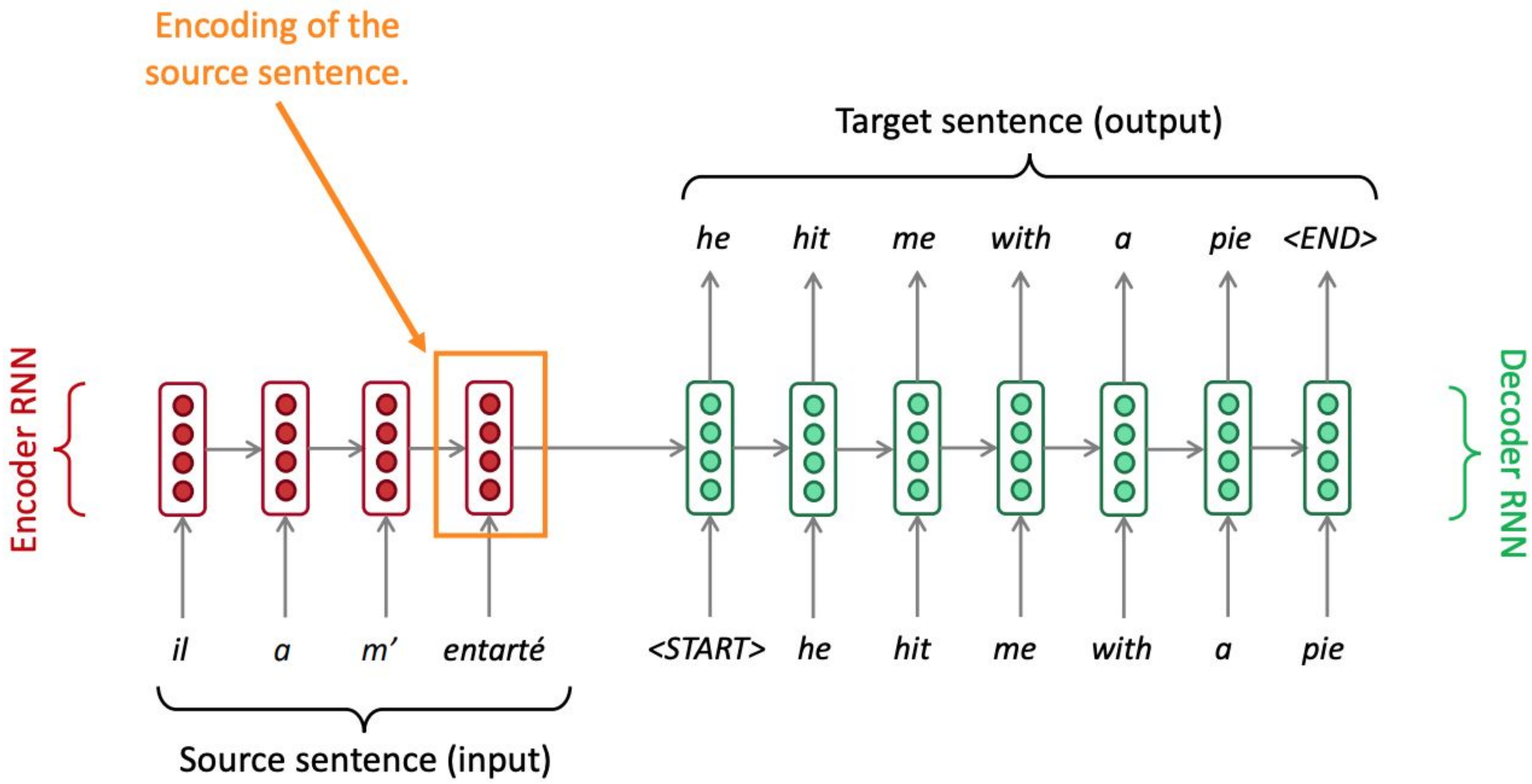


Decoder RNN is a Language Model that generates target sentence, *conditioned on encoding*.

Encoder RNN produces an encoding of the source sentence.

Note: This diagram shows **test time** behavior: decoder output is fed in ······> as next step's input

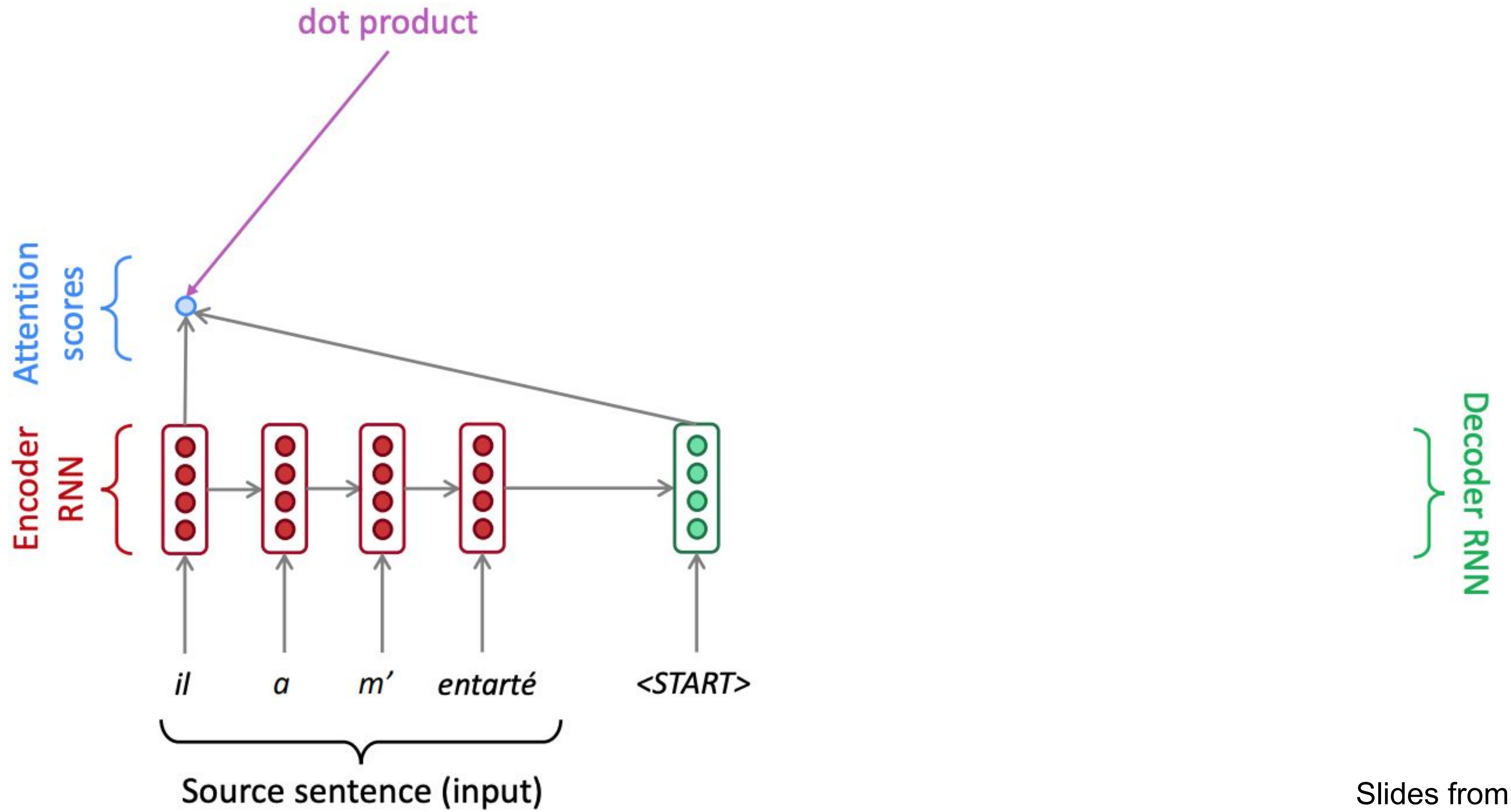
Sequence-to-sequence: the bottleneck problem



Problems with this architecture?

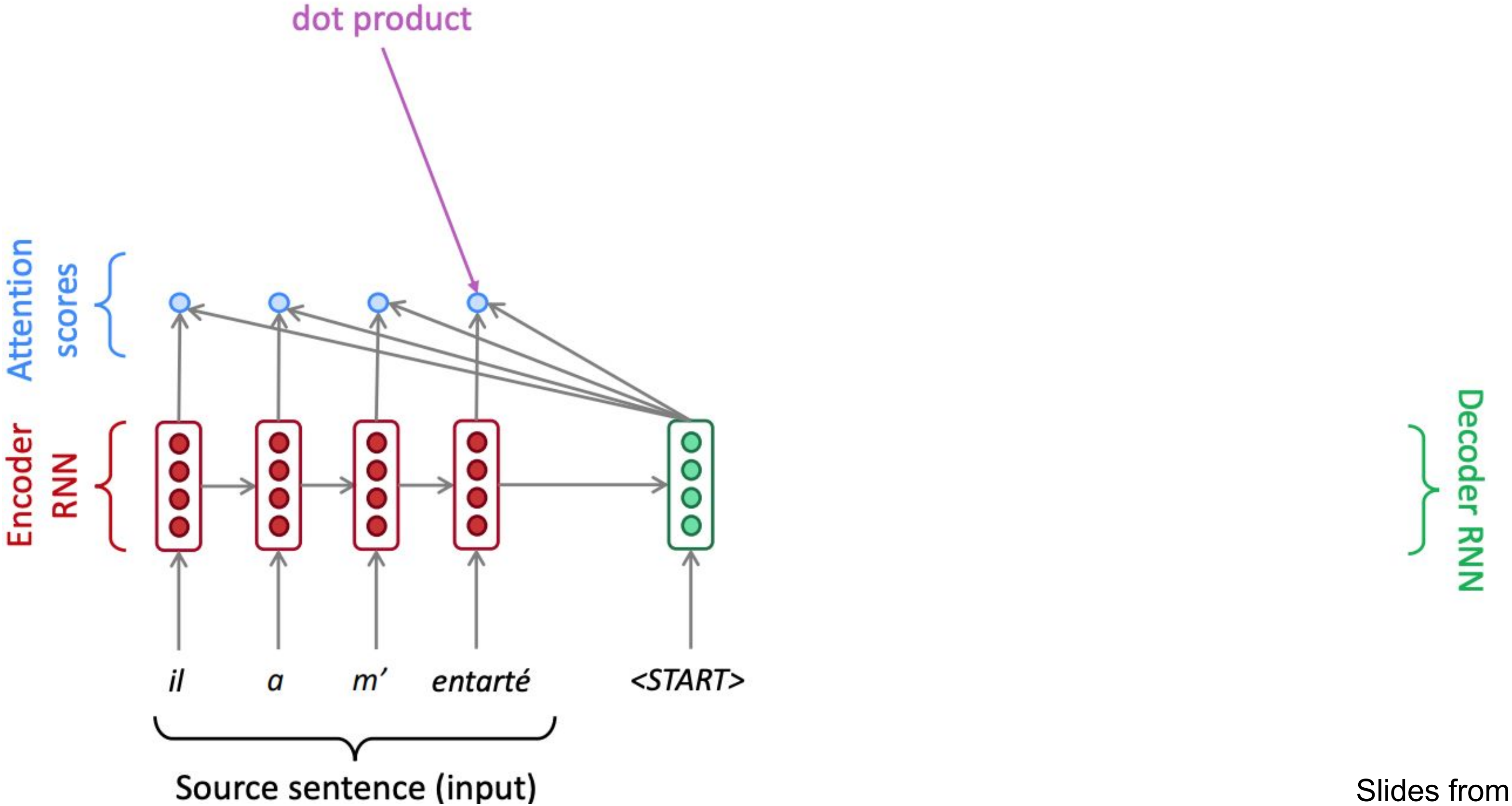
Slides from Abigail See

Sequence-to-sequence with attention



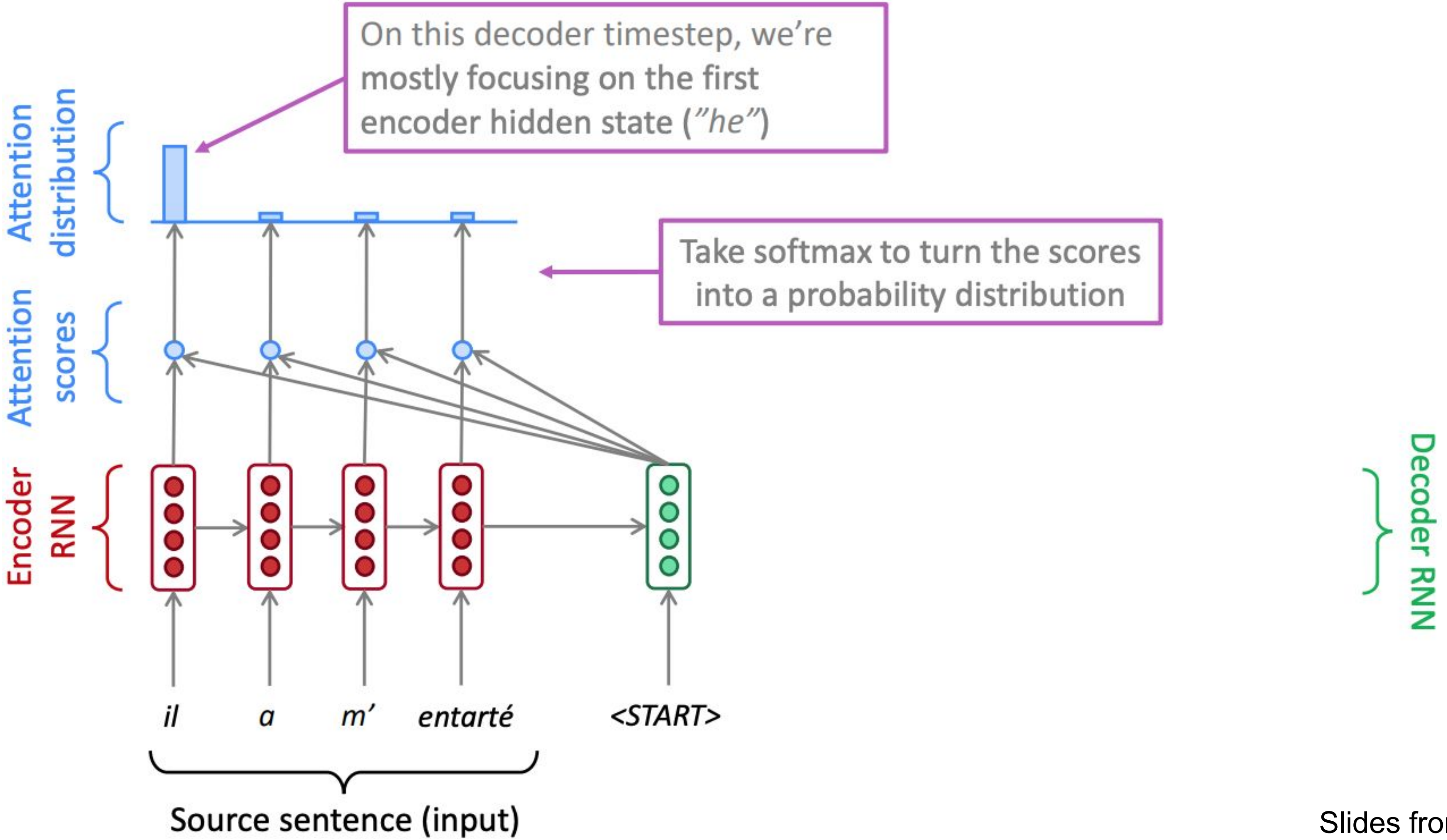
Slides from Abigail See

Sequence-to-sequence with attention



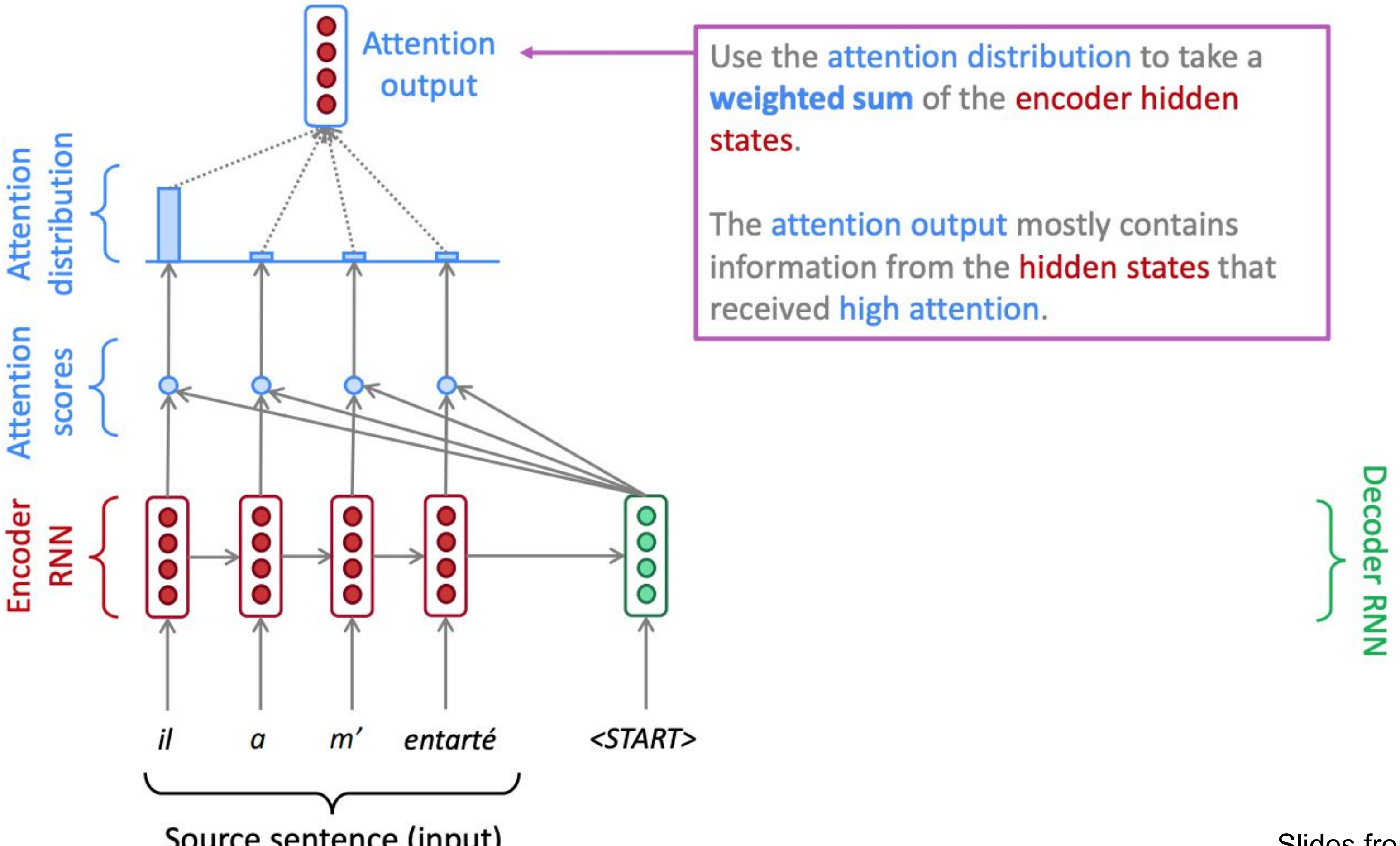
Slides from Abigail See

Sequence-to-sequence with attention



Slides from Abigail See

Sequence-to-sequence with attention

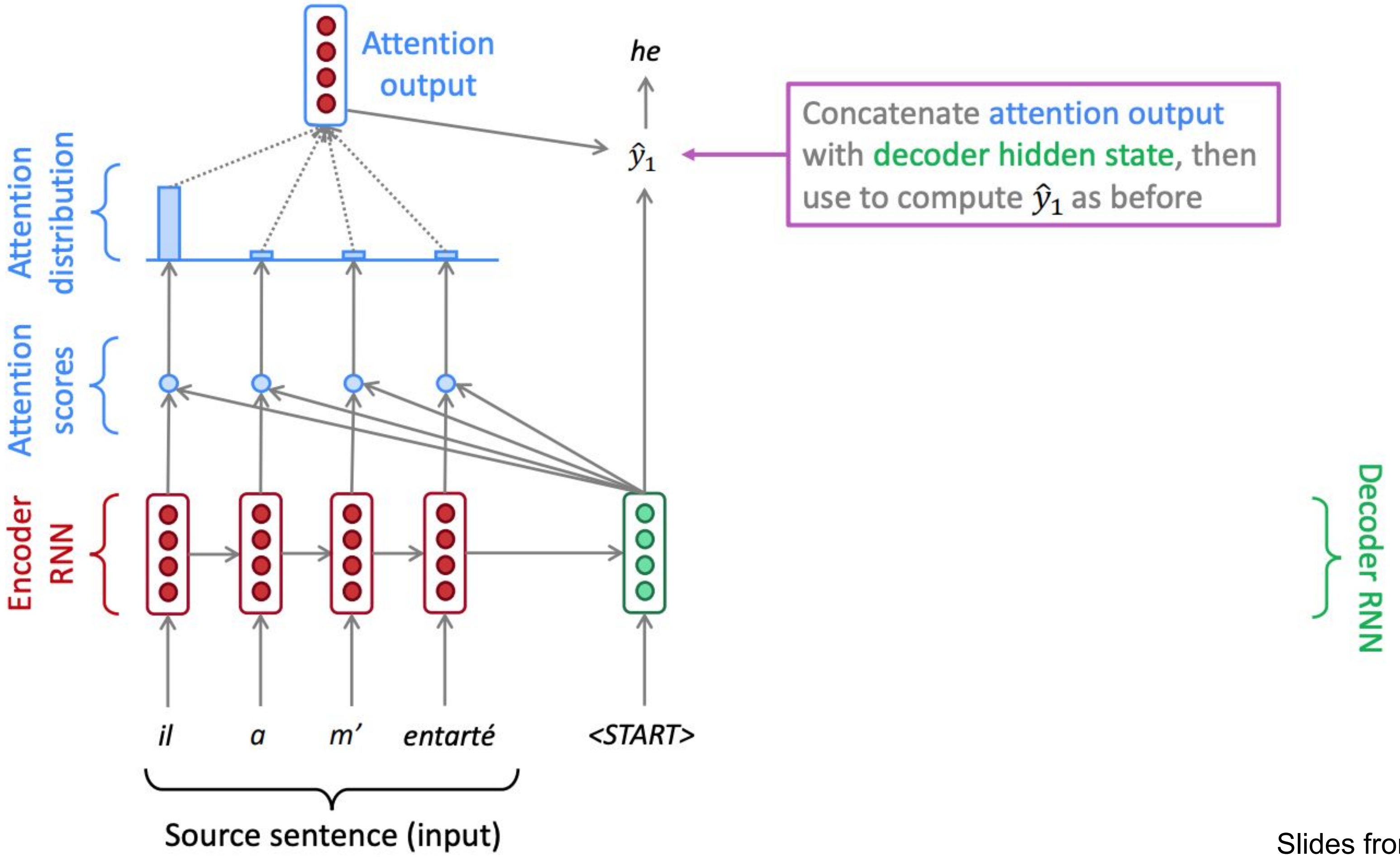


Use the attention distribution to take a **weighted sum** of the encoder hidden states.

The attention output mostly contains information from the hidden states that received high attention.

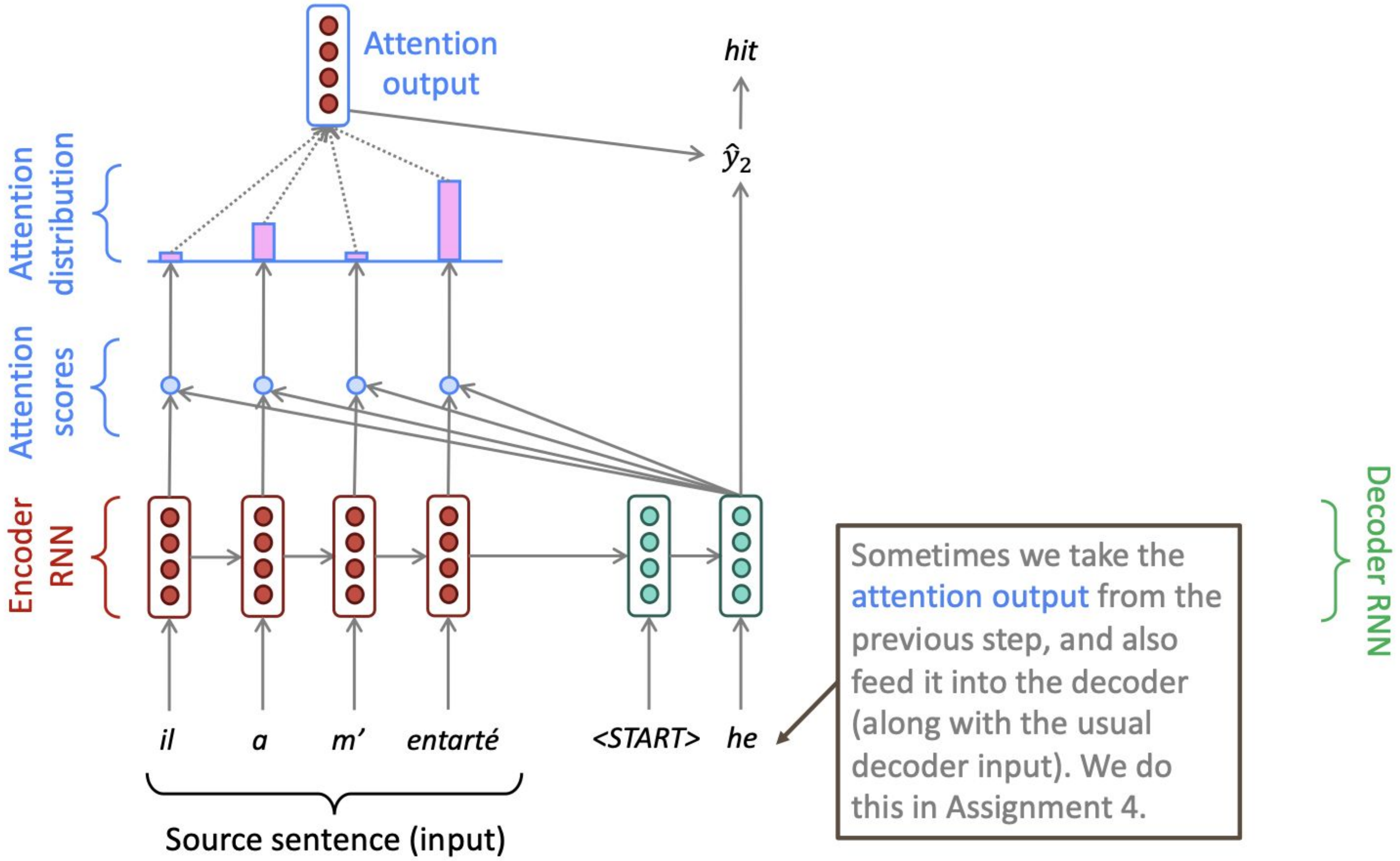
Slides from Abigail See

Sequence-to-sequence with attention



Slides from Abigail See

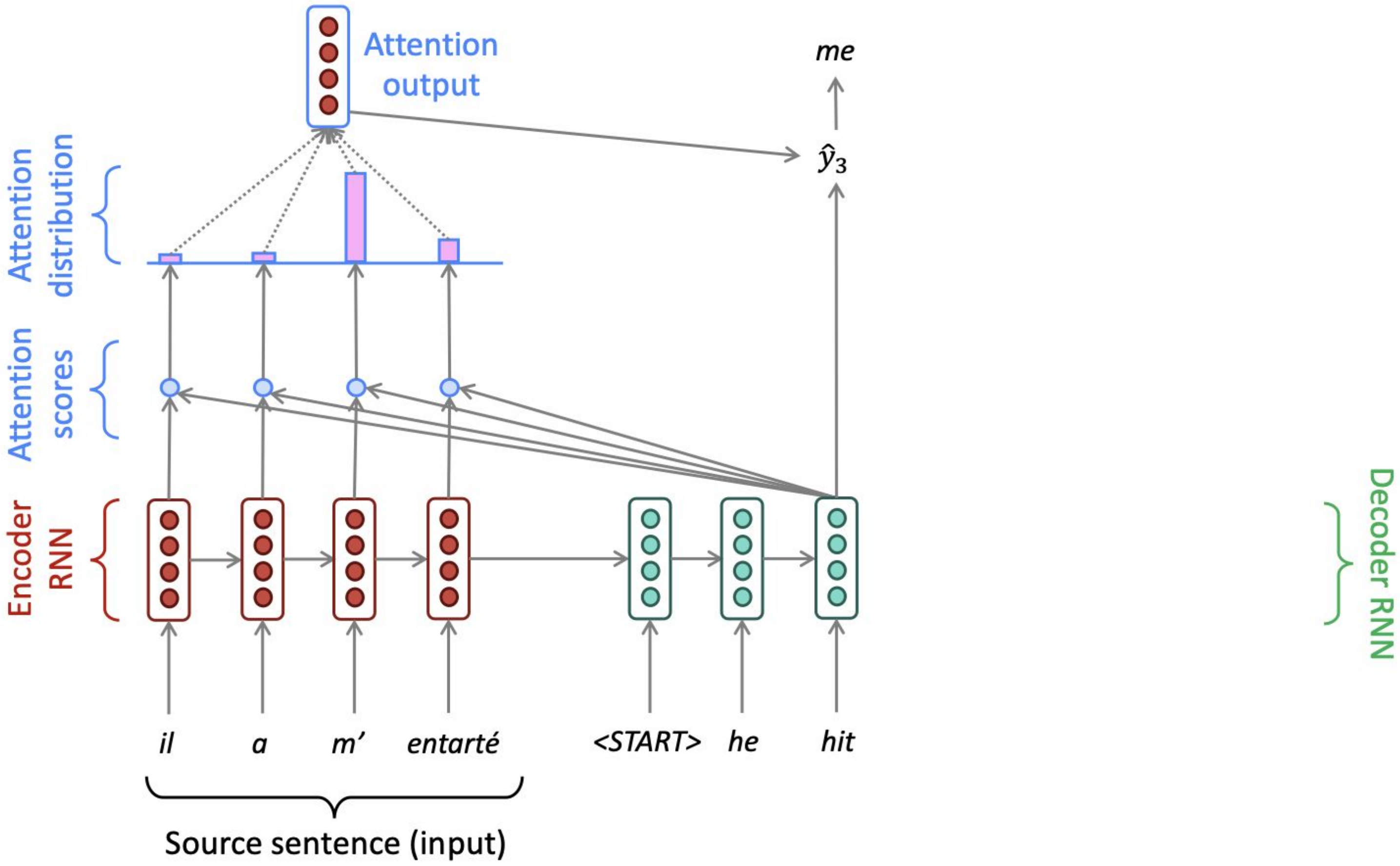
Sequence-to-sequence with attention



Sometimes we take the **attention output** from the previous step, and also feed it into the decoder (along with the usual decoder input). We do this in Assignment 4.

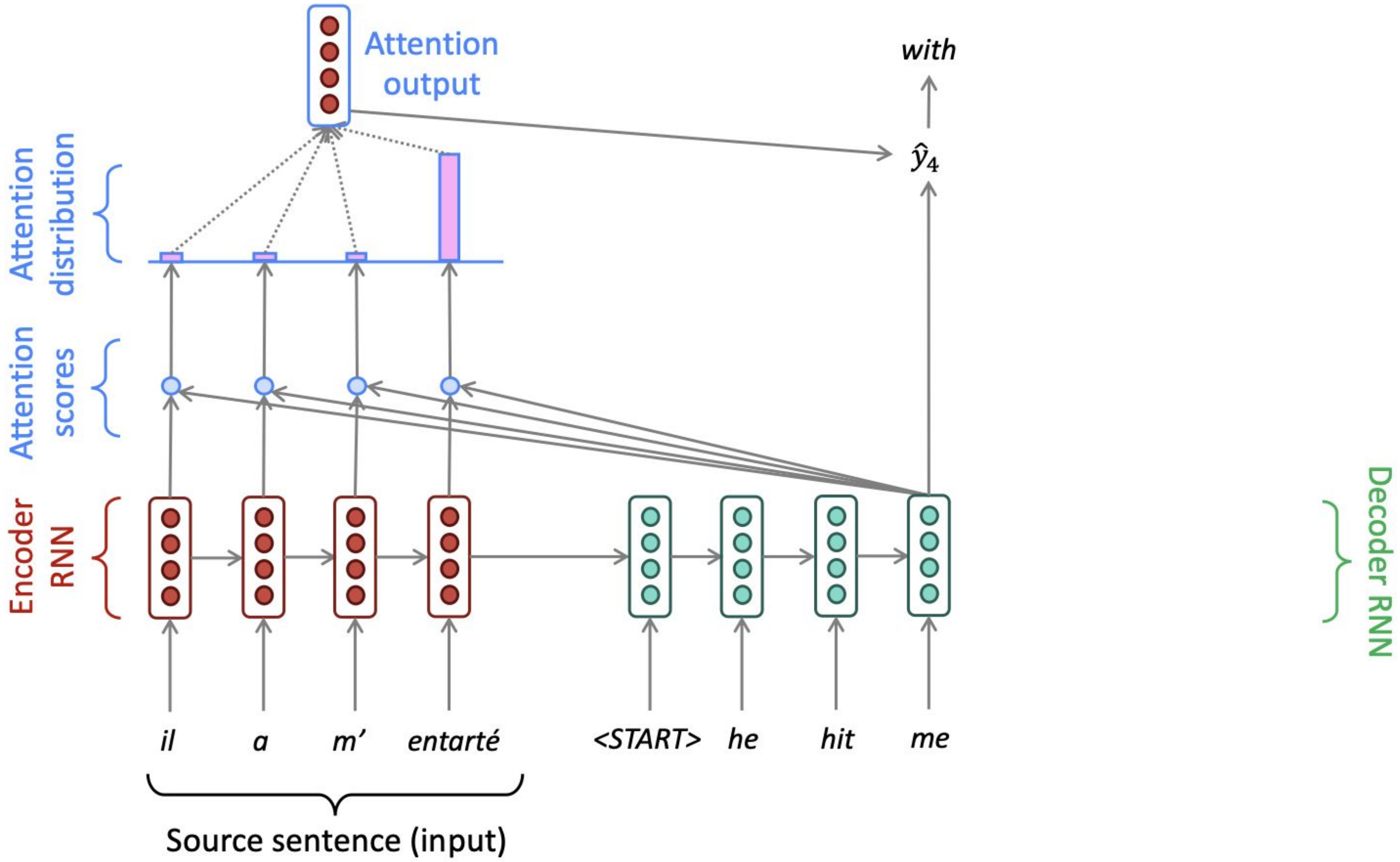
Slides from Abigail See

Sequence-to-sequence with attention



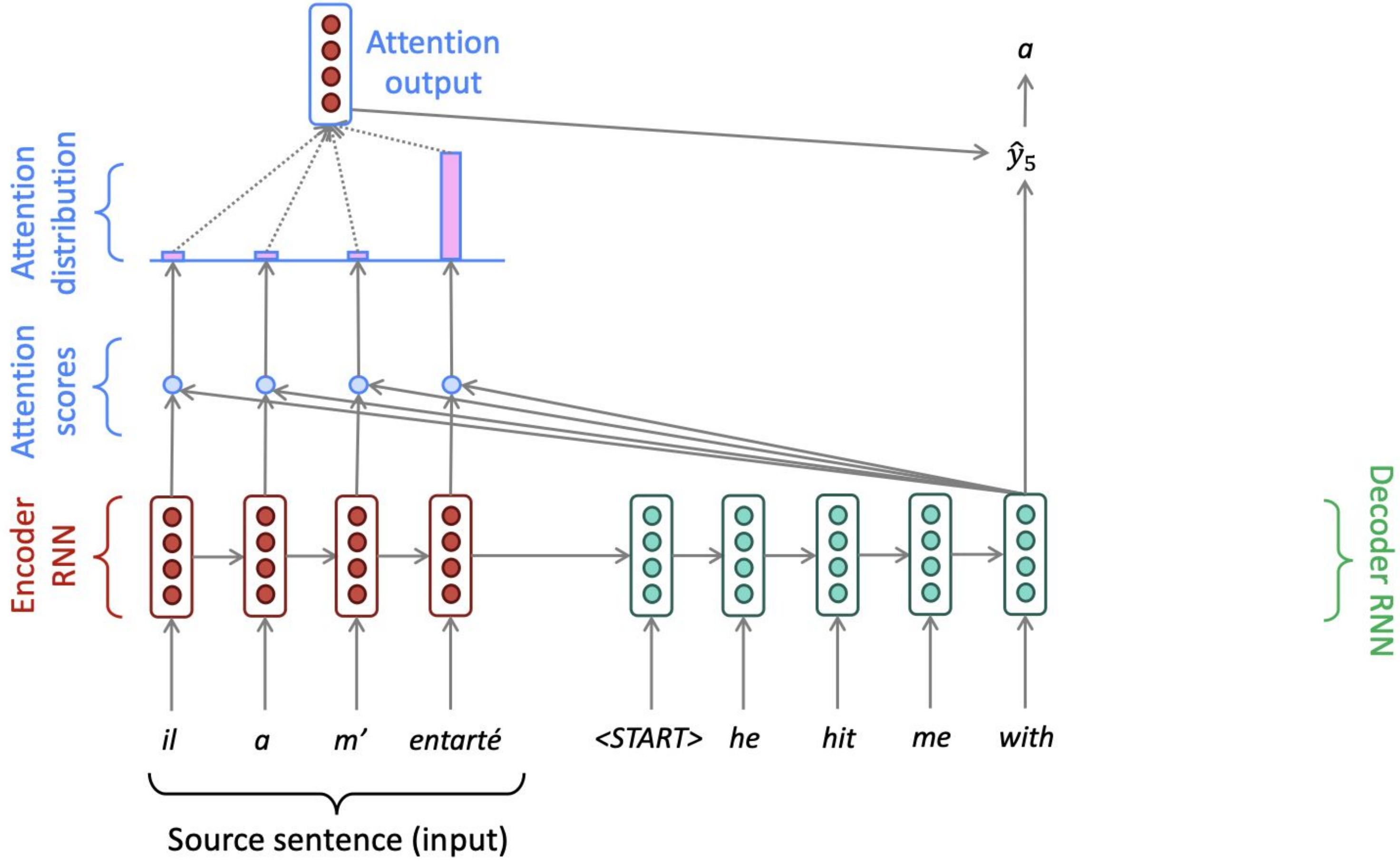
Slides from Abigail See

Sequence-to-sequence with attention



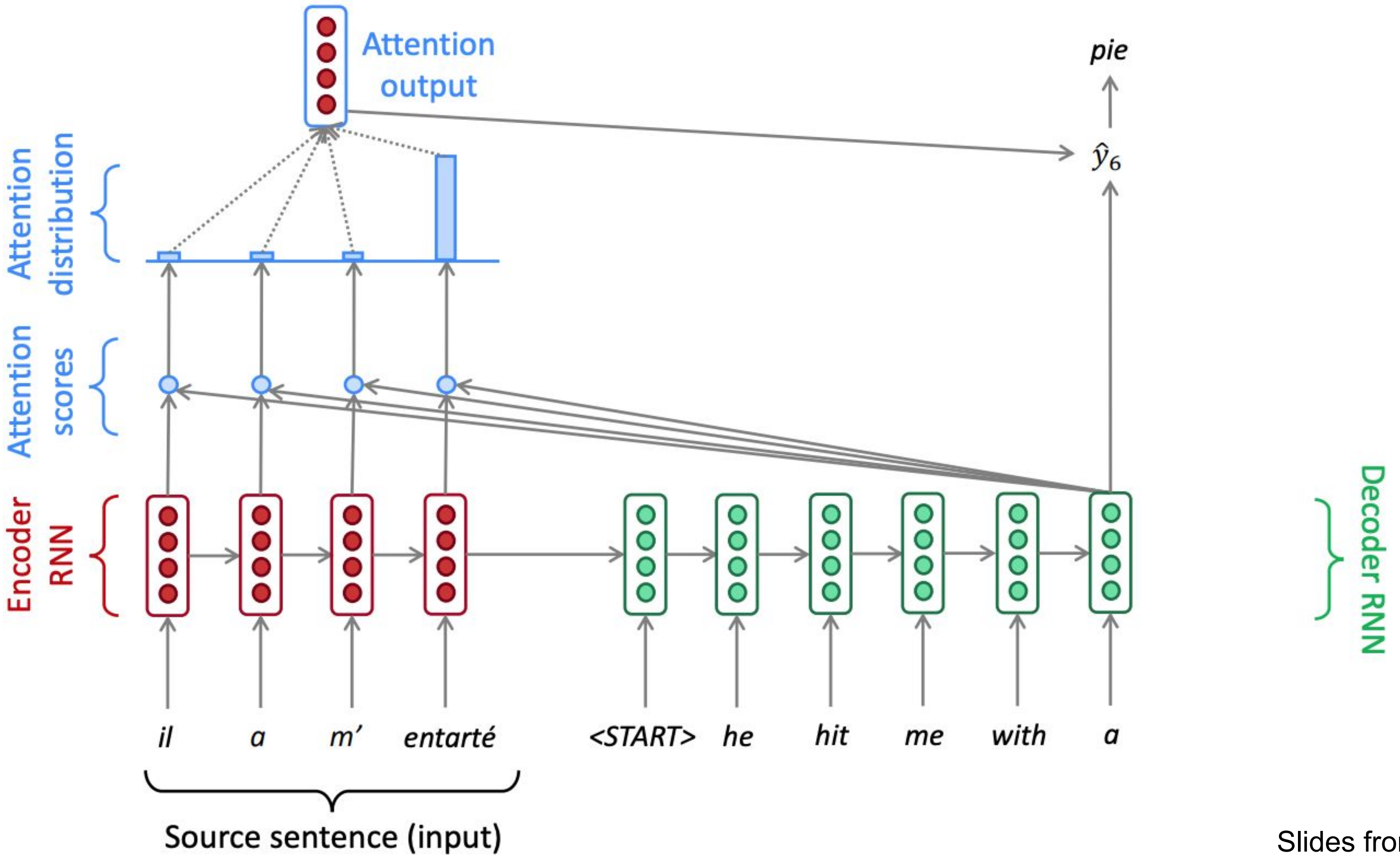
Slides from Abigail See

Sequence-to-sequence with attention



Slides from Abigail See

Sequence-to-sequence with attention



Slides from Abigail See

Attention: in equations

- We have encoder hidden states $h_1, \ldots, h_N \in \mathbb{R}^h$
- On timestep *t*, we have decoder hidden state $s_t \in \mathbb{R}^h$
- We get the attention scores $\boldsymbol{e}^t$ for this step:

$$\boldsymbol{e}^t = [\boldsymbol{s}_t^T \boldsymbol{h}_1, \ldots, \boldsymbol{s}_t^T \boldsymbol{h}_N] \in \mathbb{R}^N$$

- We take softmax to get the attention distribution α^t for this step (this is a probability distribution and sums to 1)

$$\alpha^t = \mathrm{softmax}(\boldsymbol{e}^t) \in \mathbb{R}^N$$

- We use α^t to take a weighted sum of the encoder hidden states to get the attention output $\boldsymbol{a}_t$

$$\boldsymbol{a}_t = \sum_{i=1}^{N} \alpha_i^t \boldsymbol{h}_i \in \mathbb{R}^h$$

- Finally we concatenate the attention output $\boldsymbol{a}_t$ with the decoder hidden state s_t and proceed as in the non-attention seq2seq model

$$[\boldsymbol{a}_t; \boldsymbol{s}_t] \in \mathbb{R}^{2h}$$

Slides from Abigail See

Final notes on NMT

- To **decode** (get a translated sentence from the MT model), we can use methods discussed for previous sequence labeling tasks: greedy decoding, beam search, etc.
- We show how to use the encoder-decoder model for MT, but this is a general setup that works:
  - For many different NLP tasks
  - With different NN architectures (RNNs, Transformers)

Quiz 9 - Problem 1

Instantiation of IBM model 1 trained to give the probability of English given Latin:

p(* | dubito) = [I : 0.5, doubt : 0.5, ...]
p(* | cogito) = [I : 0.49, think : 0.51, ...]
p(* | sum) = [I : 0.51, am : 0.49, ...]
p(* | ergo) = [therefore : 0.99, I : 0, ...]
p(* | ,) = [, : 1, ...]
p(* | .) = [. : 1, ...]

Consider the parallel sentences:

dubito , ergo cogito , ergo sum .

I doubt , therefore I think , therefore I am .

Infer the **single most probable alignment** of each English word to a Latin word in this sentence pair. Which Latin word will the first instance of "I" align to?

Quiz 9 - Problem 1

Suppose you are going to infer the **single most probable alignment** of each English word to a Latin word in this sentence pair. Which Latin word will the first instance of "I" align to?

The posterior probability of each Latin word *w* given the English word I is proportional to p(I | *w*), so we have:

0.5 dubito 0.49 cogito 0.51 sum

Renormalizing this in the same way as the E step gives:

p(dubito | I) = 0.333 p(cogito | I) = 0.327 p(sum | I) = 0.340

The most probable alignment for the first "I" in sentence 2 will be to "sum" in sentence 1 ... and indeed all instances of "I" in sentence 2 will align to "sum".

Q & A