Dive into Deep Learning

Chapter 10. Attention Mechanisms

Wayne L, Dec 16, 2020

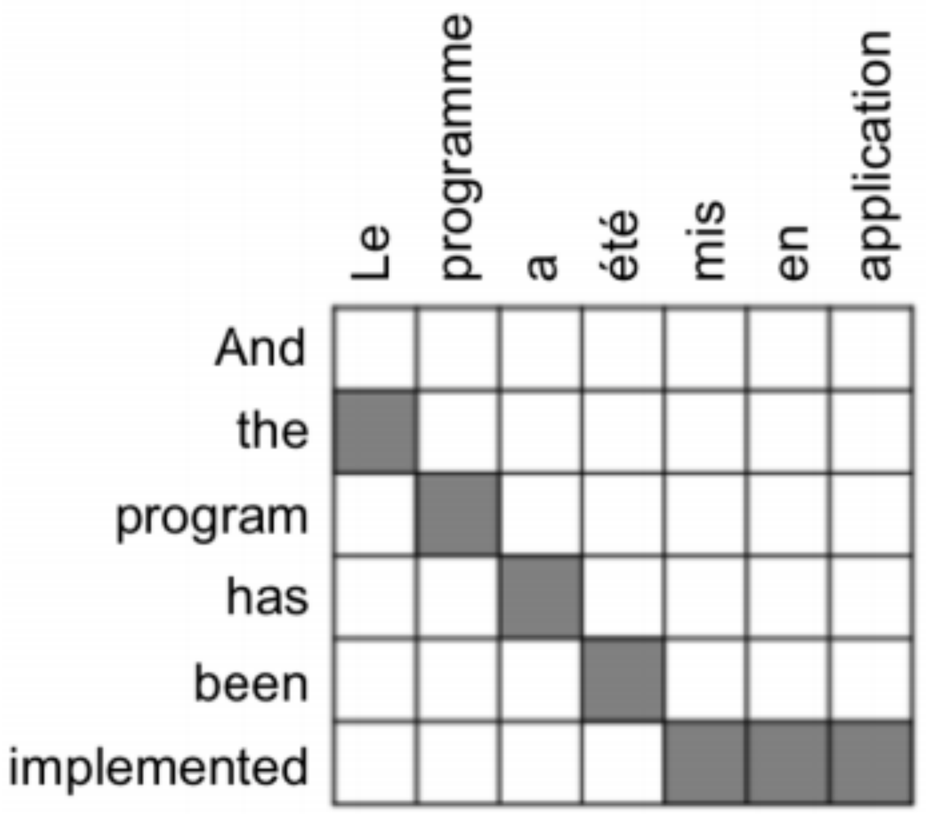
Overview

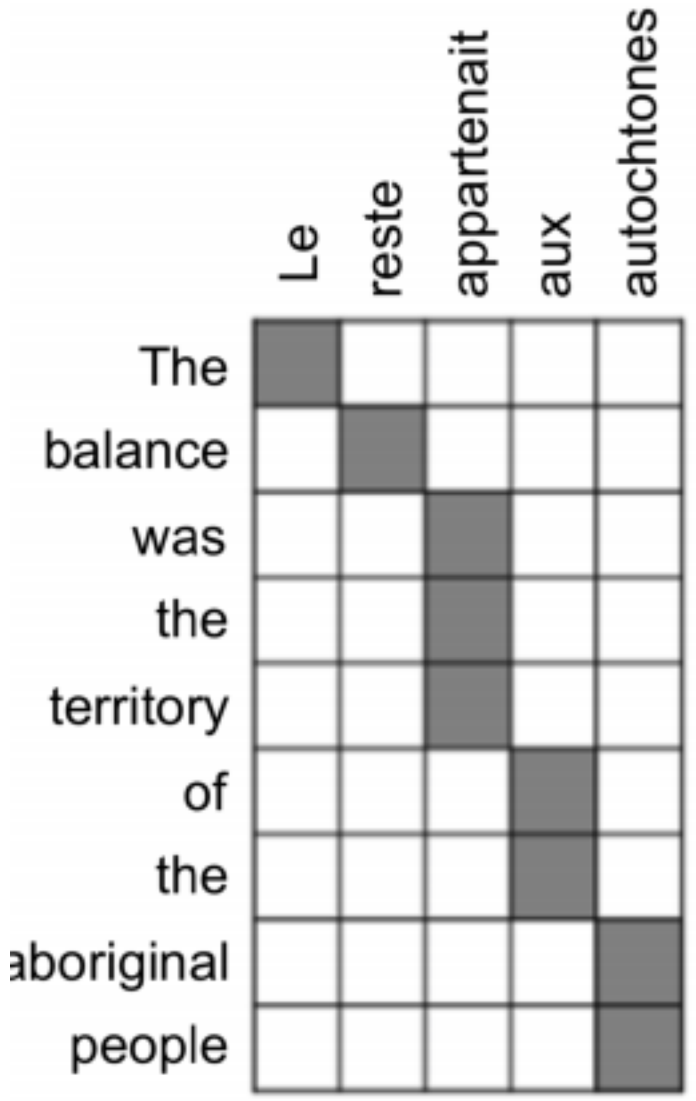
Machine Translation (From Source language to target language)

- Early 1950s, Russian —> English (motivated by the Cold War!)
    - **Rule-based**, using bilingual dictionary to map
    - <u>Alignment</u> is complex
- 1990s-2010s, **Statistical MT**
    - Learn a probabilistic model from data
    - Extremely complex and human effort to maintain

$$argmax_y P(x|y)P(y)$$

    - **Translation model**: how words and phrases should be translated (fidelity), Learnt from parallel data.
    - **Language model**: how to write good English (fluency), Learnt from monolingual data.



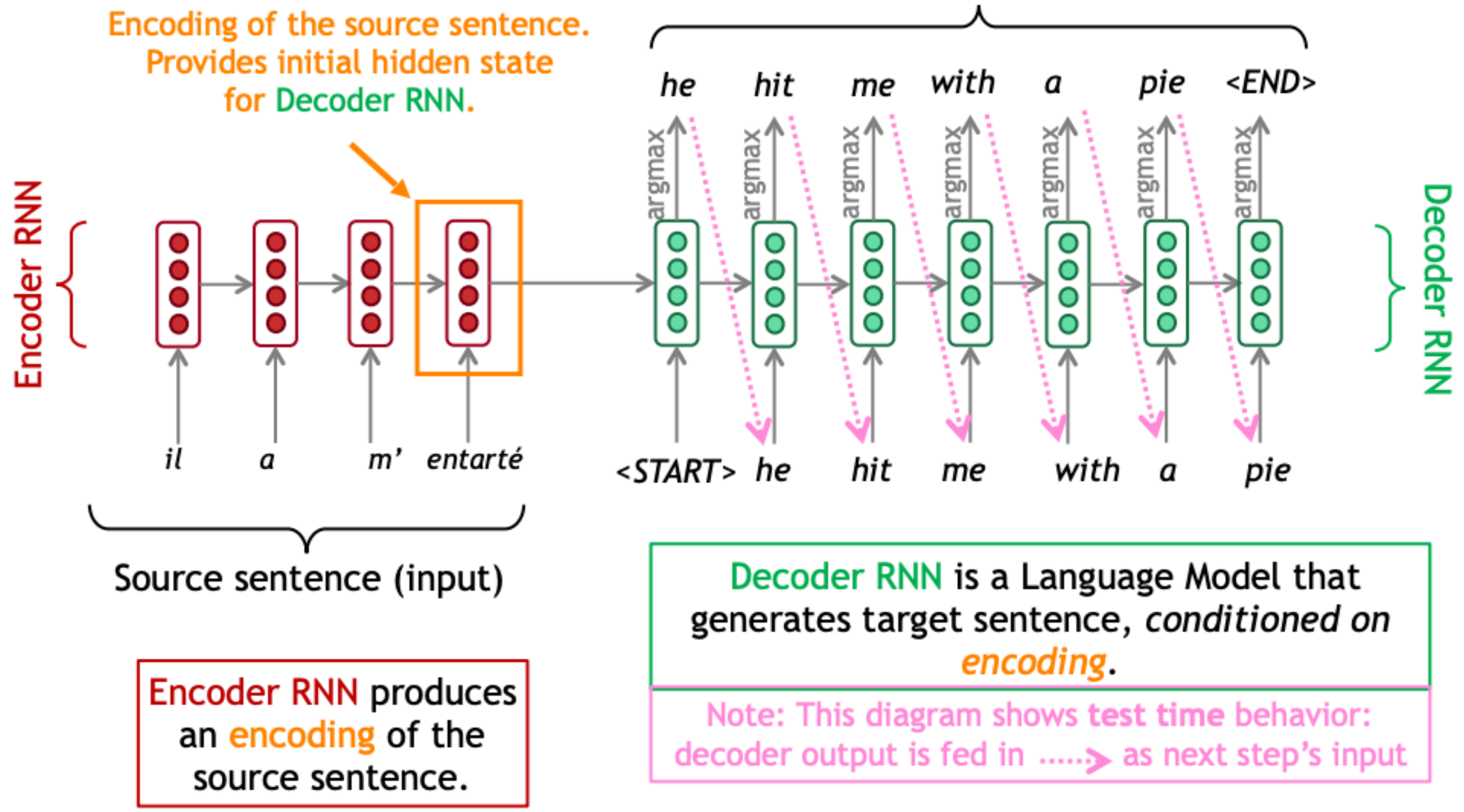


http://web.stanford.edu/class/cs224n/slides/cs224n-2020-lecture08-nmt.pdf

Overview

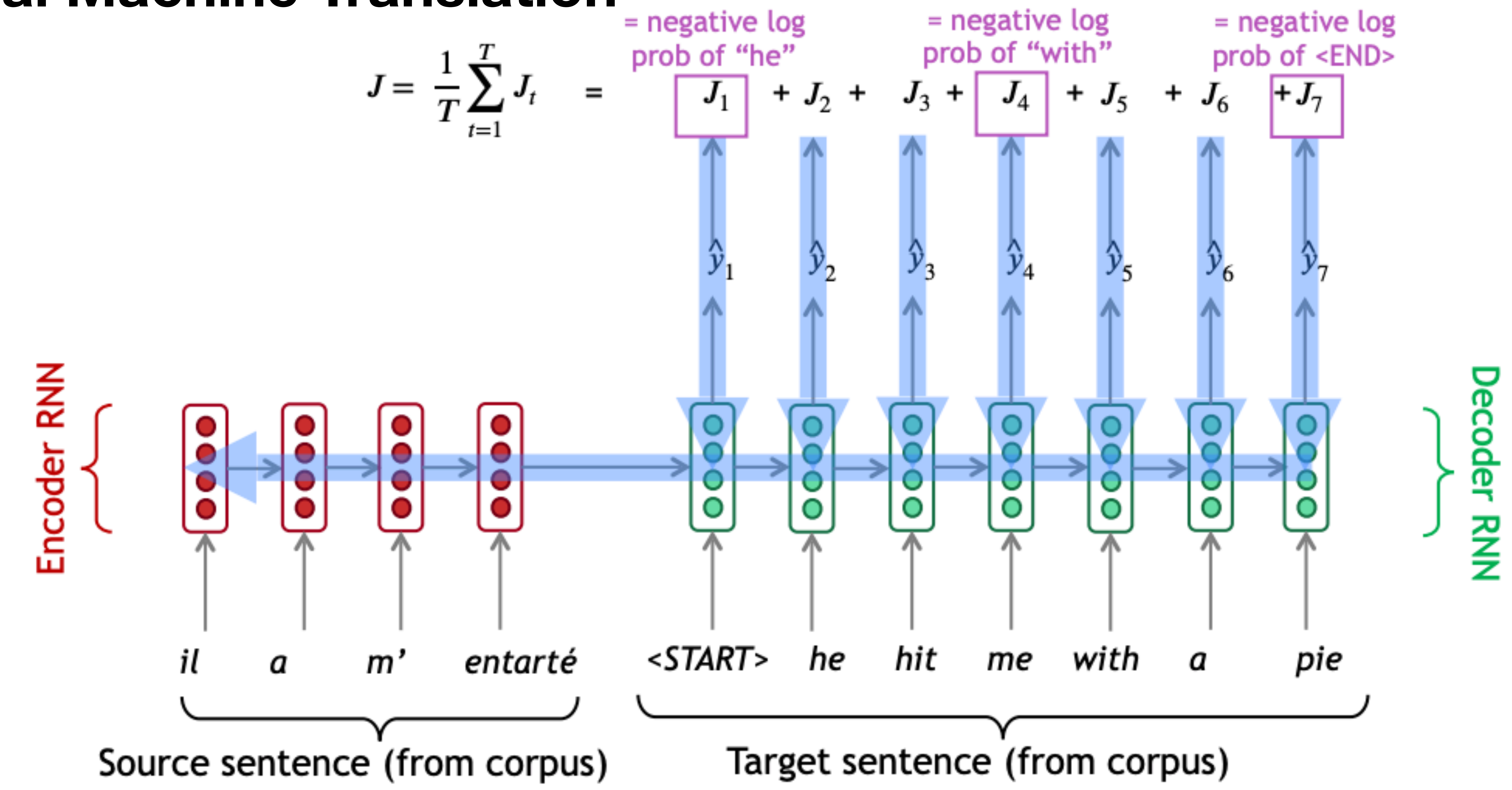
Neural Machine Translation

- Neural Machine Translation (NMT)
  - Two RNNs
  - Versatile in many NLP tasks:
    - **Summarization** (long text -> short text)
    - **Dialogue** (previous utterances -> next utterance)
    - **Parsing** (input text -> output parse as sequence
    - **Code** generation (natural language -> Python)
  - Advantages
    - Better performance (fluent, context, phrase similarities)
    - No subcomponents
    - less human engineering effort
  - Disadvantages
    - less interpretable (hard to debug)
    - difficult to control (can't easily specify rules)



Overview

Neural Machine Translation



Seq2seq is optimized as a single system.
Backpropagation operates "end-to-end".

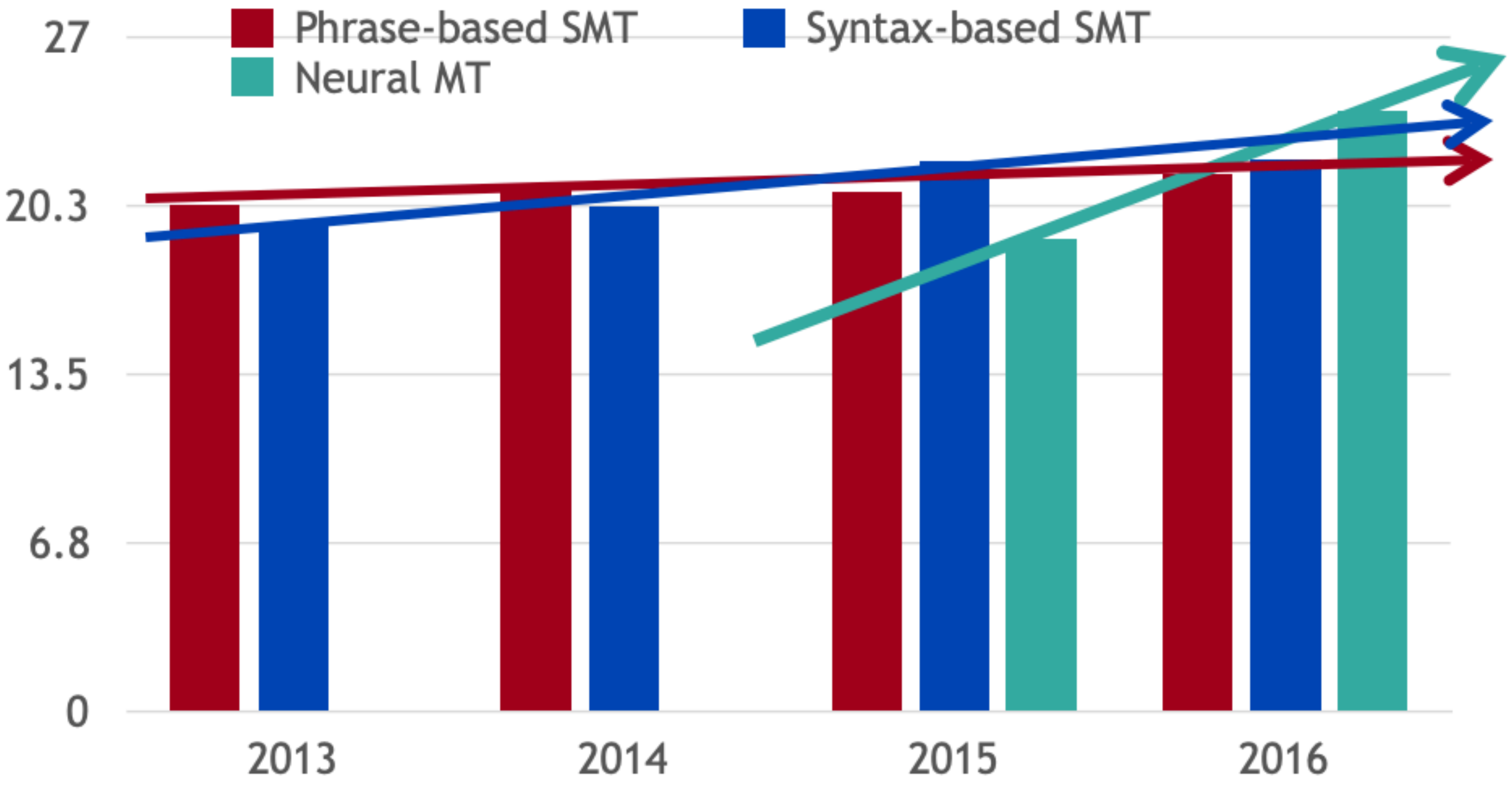
http://web.stanford.edu/class/cs224n/slides/cs224n-2020-lecture08-nmt.pdf

Overview

How do we evaluate MT?

BLEU (**B**ilingual **E**valuation **U**nderstudy)

- **BLEU** compares the machine-written translation to one or several human-written translation(s), and computes a similarity score based on:
    - *n*-gram precision (usually for 1, 2, 3 and 4-grams)
    - Plus a penalty for too-short system translations
- BLEU is **useful** but **imperfect**
    - There are many valid way to translate a sentence
    - So a **good** translation can get a **poor** BLEU score because it has low n-gram overlap with the human translation

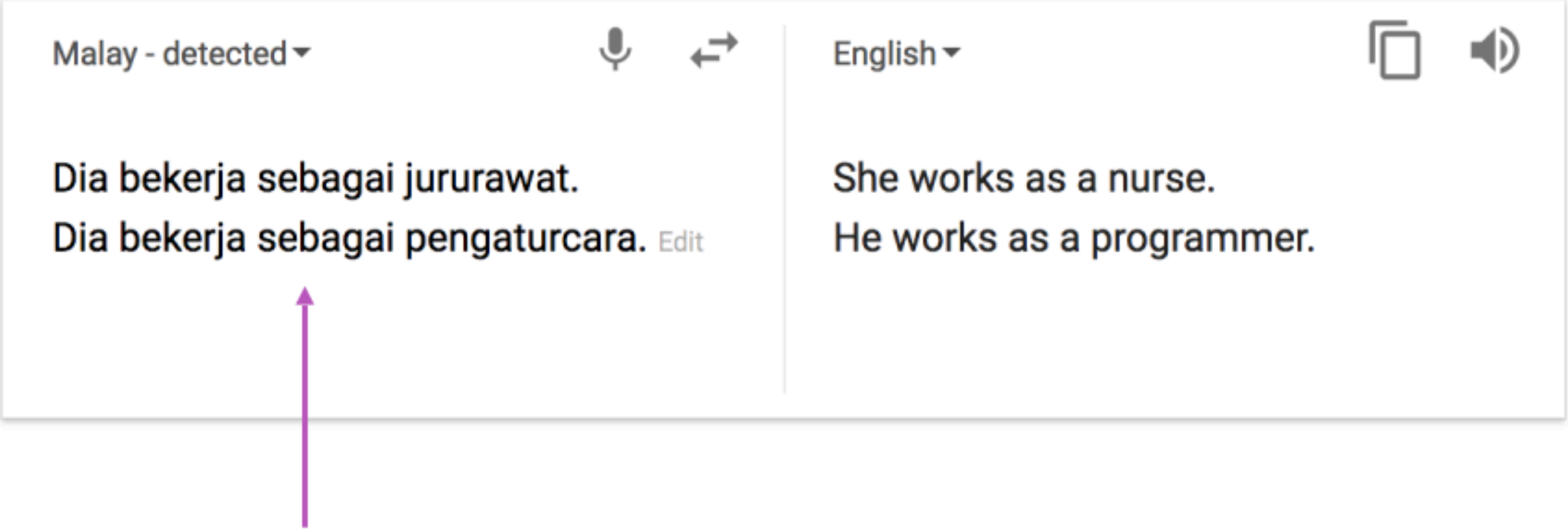


http://www.meta-net.eu/events/meta-forum-2016/slides/09_sennrich.pdf

Overview

The biggest success story of NLP Deep learning

- **2014**: First seq2seq paper published
- **2016**: Google Translate switches from SMT to NMT
- **This is amazing!**
    - SMT systems, built by **hundreds** of engineers over **many years**, outperformed by NMT systems trained by a **handful** of engineers in **a few months**
- Many difficulties remain:
    - Out-of-vocabulary
    - Domain mismatch between train and test data
    - Maintaining context over longer text
    - Low-resource language pairs
    - Common sense and Idioms
    - Biases



Didn’t specify gender

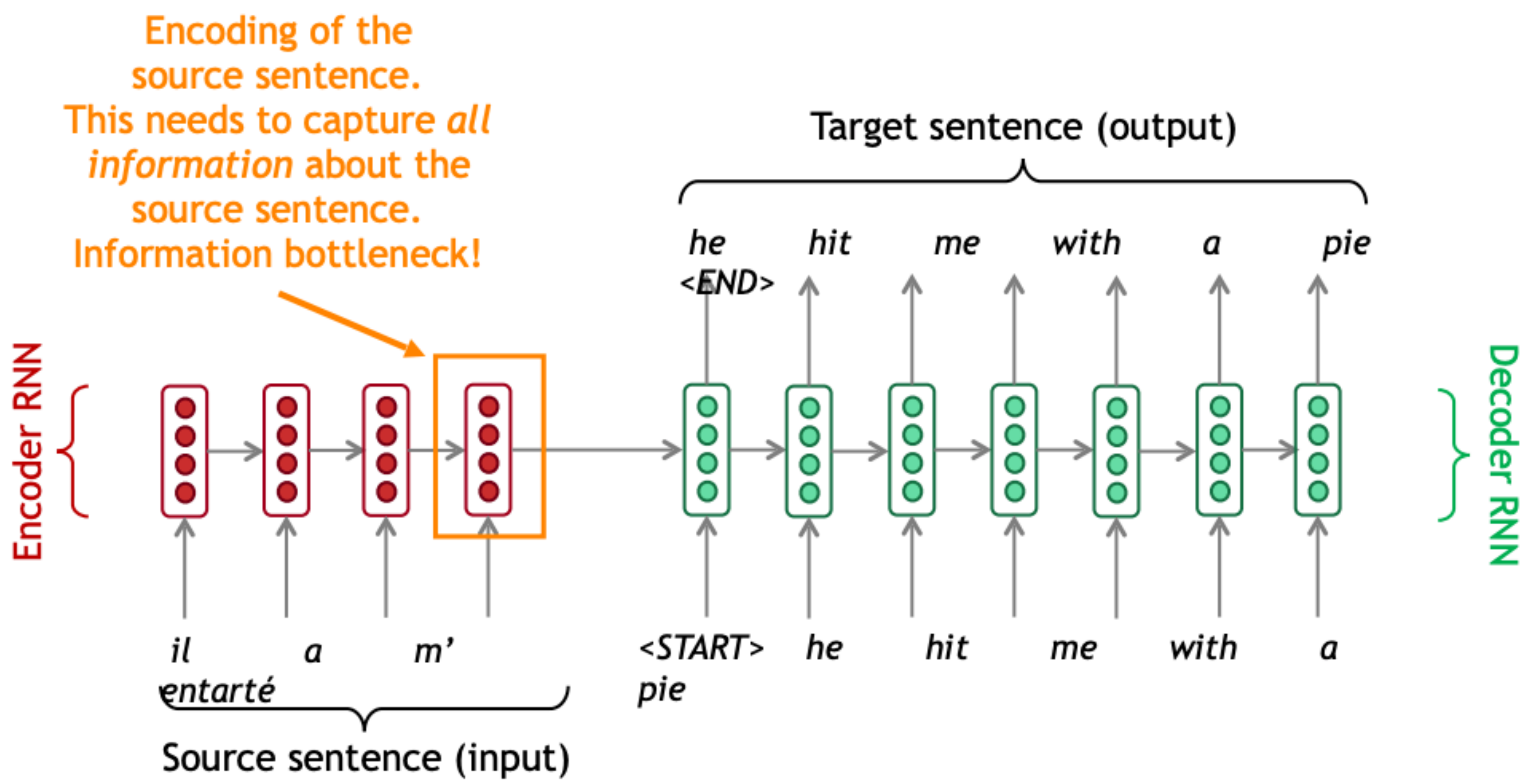
Attention Mechanisms
NMT research continues

NMT is the **flagship task** for NLP Deep Learning

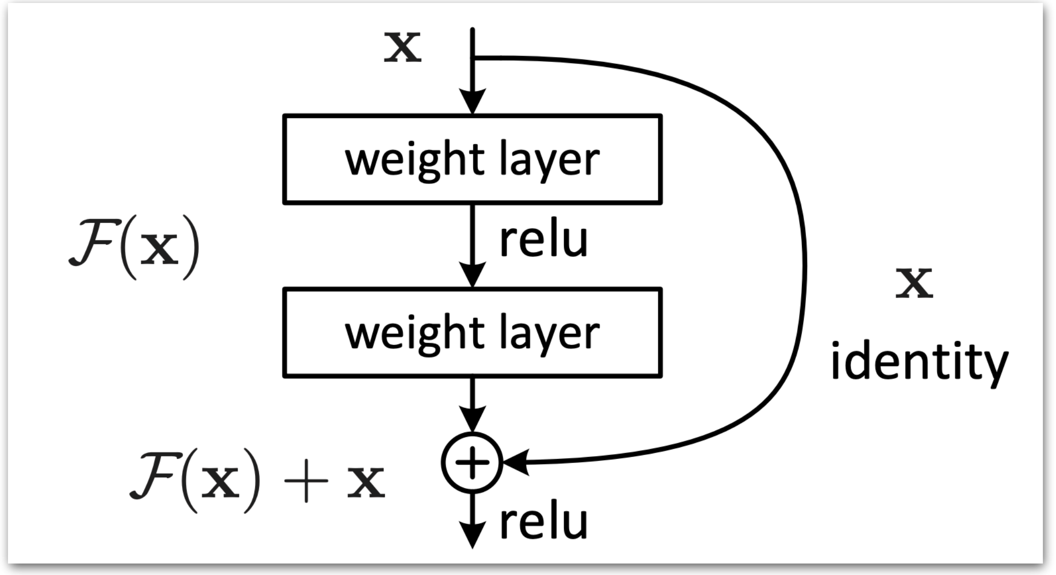
- NMT research has **pioneered** many of the recent innovations of NLP Deep Learning
- In 2019: NMT research continues to thrive
  - Researchers have found **many, many improvements** to the “vanilla” seq2seq NMT system we’ve presented today
  - But one improvement is so **integral** that it is the new vanilla …

Attention Mechanisms

Seq2Seq

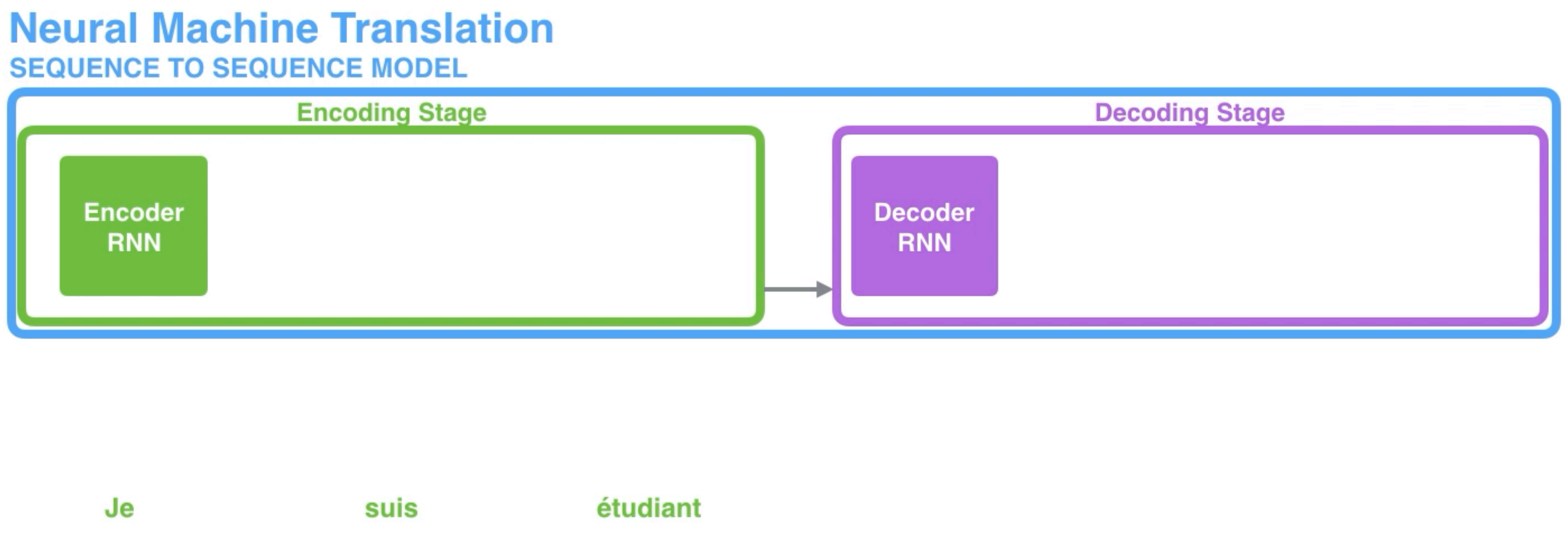


- **Attention** provides a solution to <u>the bottleneck problem</u>.
- <u>Core idea</u>: on each step of the decoder, use direct connection to the encoder to focus on a particular part of the source sequence

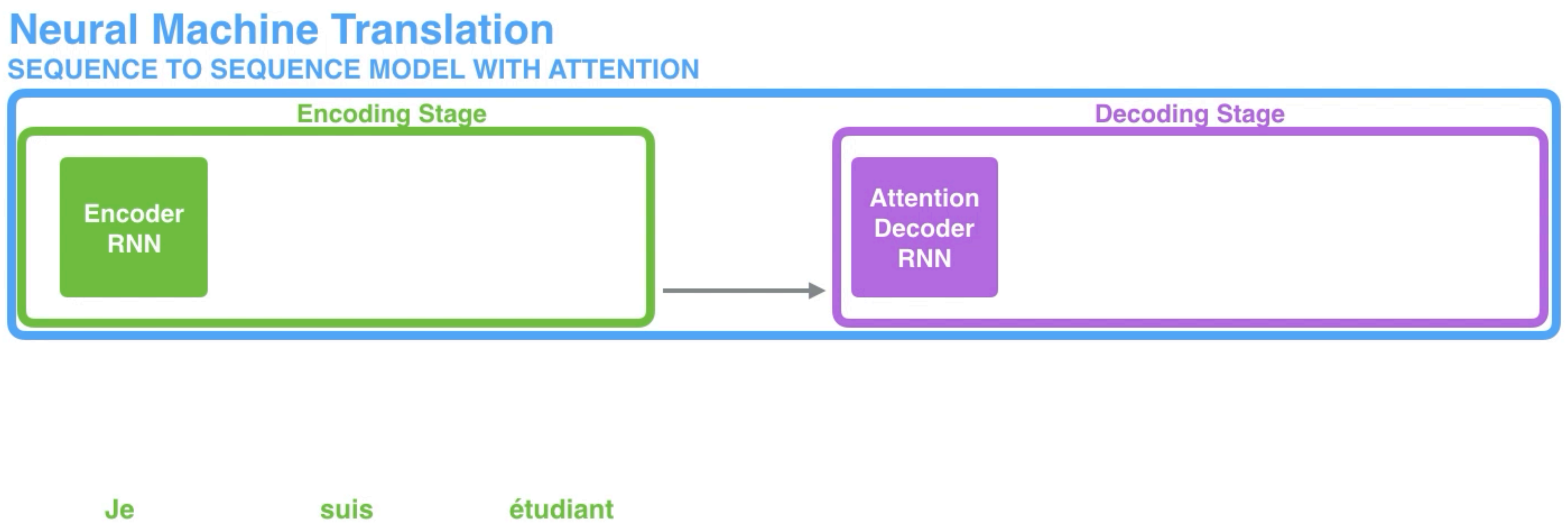


Attention Mechanisms

Visualizing a NMT



w/o attention

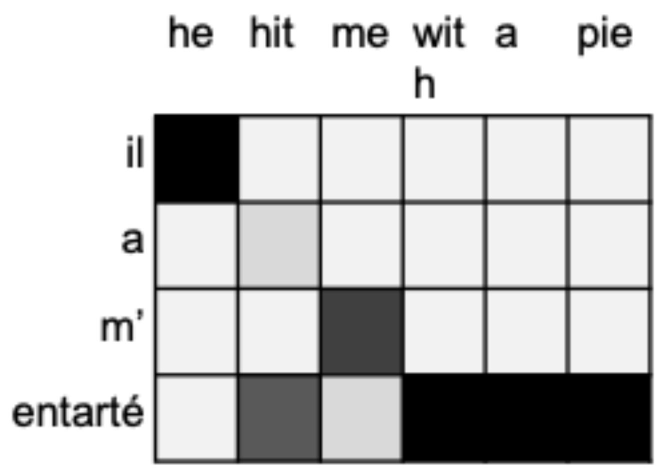


with attention

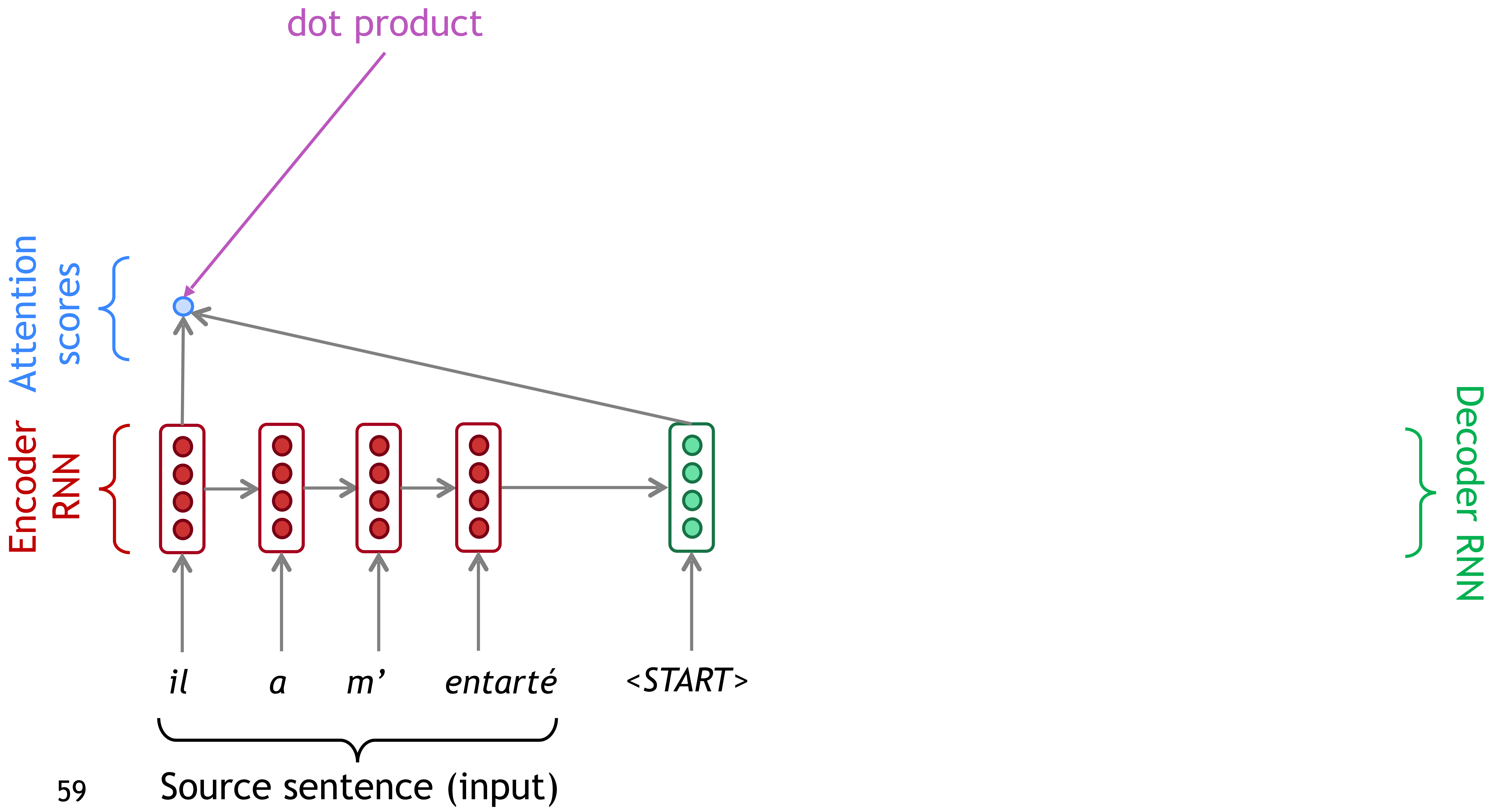
Attention Mechanisms

Attention is great

- significantly **improves NMT performance**
  - It’s very useful to allow decoder to focus on certain parts of the source
- **solves the bottleneck problem**
  - it allows decoder to look directly at source; bypass bottleneck
- **helps with vanishing gradient problem**
  - provides shortcut to faraway sates
- **provides some interpretability**
  - By inspecting attention distribution, we can see what the decoder was focusing on
  - We get (soft) **alignment for free**
  - This is cool because we never explicitly trained an alignment system
  - The network just learned alignment by itself

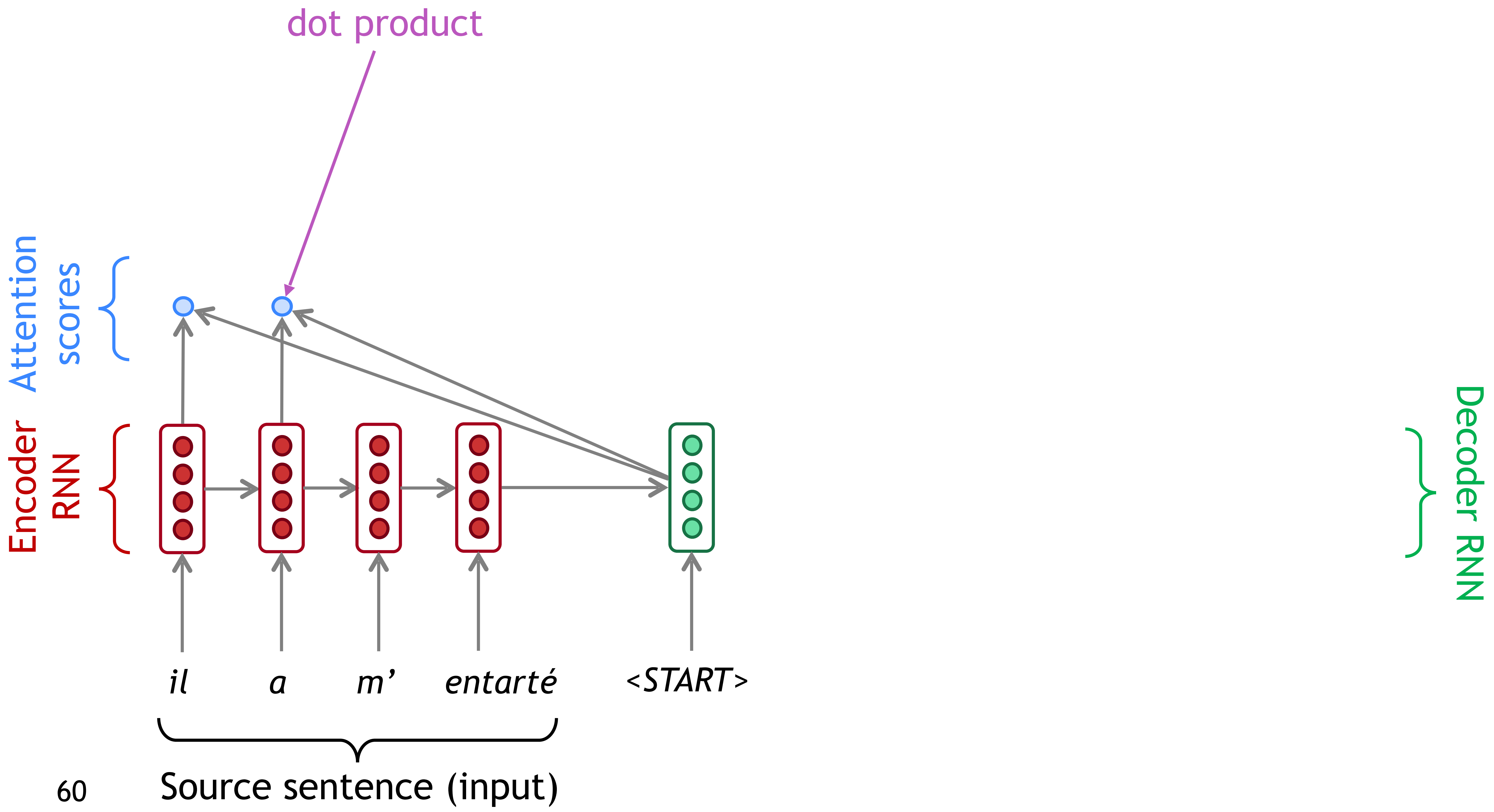


Sequence-to-sequence with attention



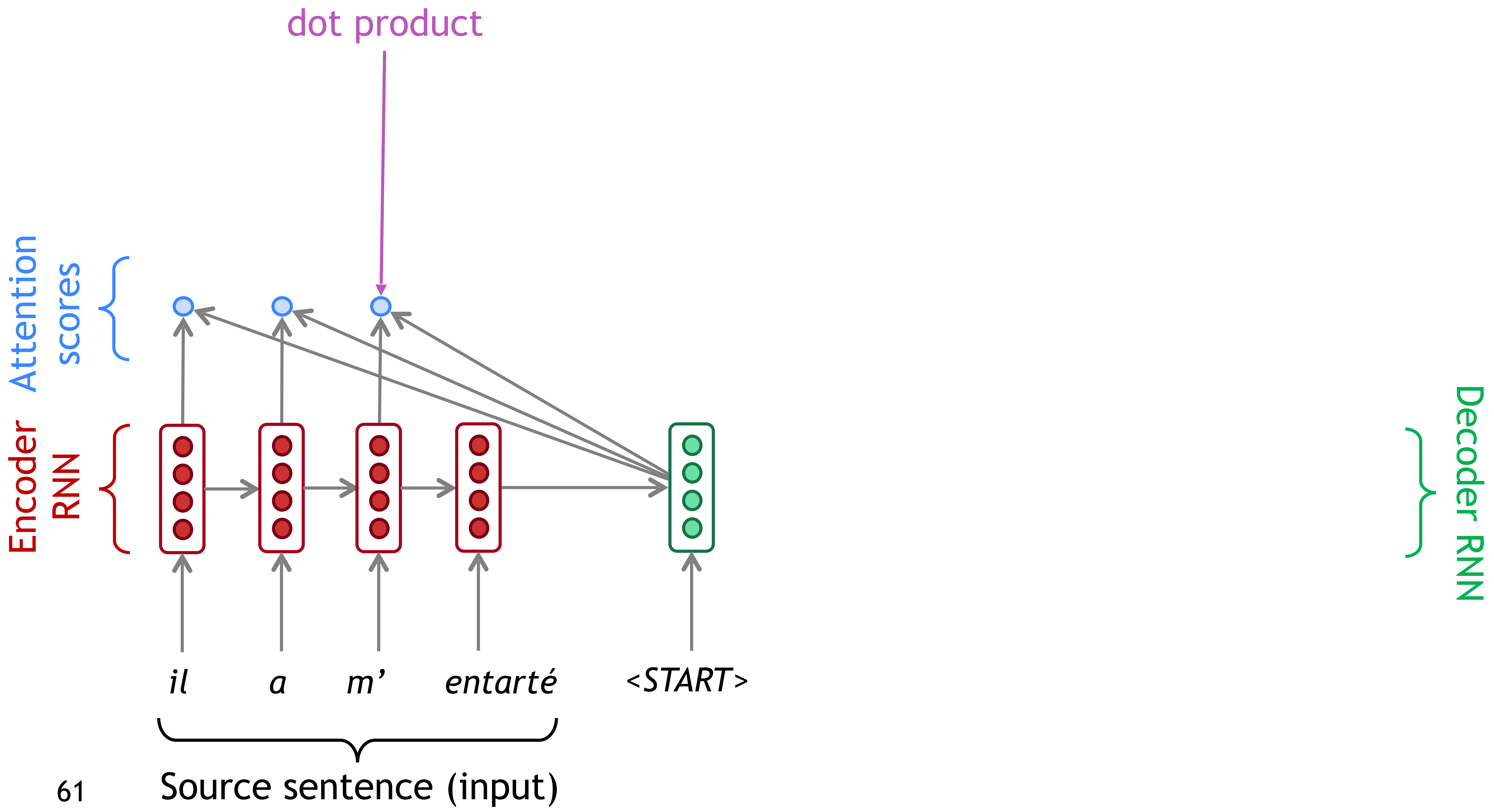
http://web.stanford.edu/class/cs224n/slides/cs224n-2020-lecture08-nmt.pdf

Sequence-to-sequence with attention



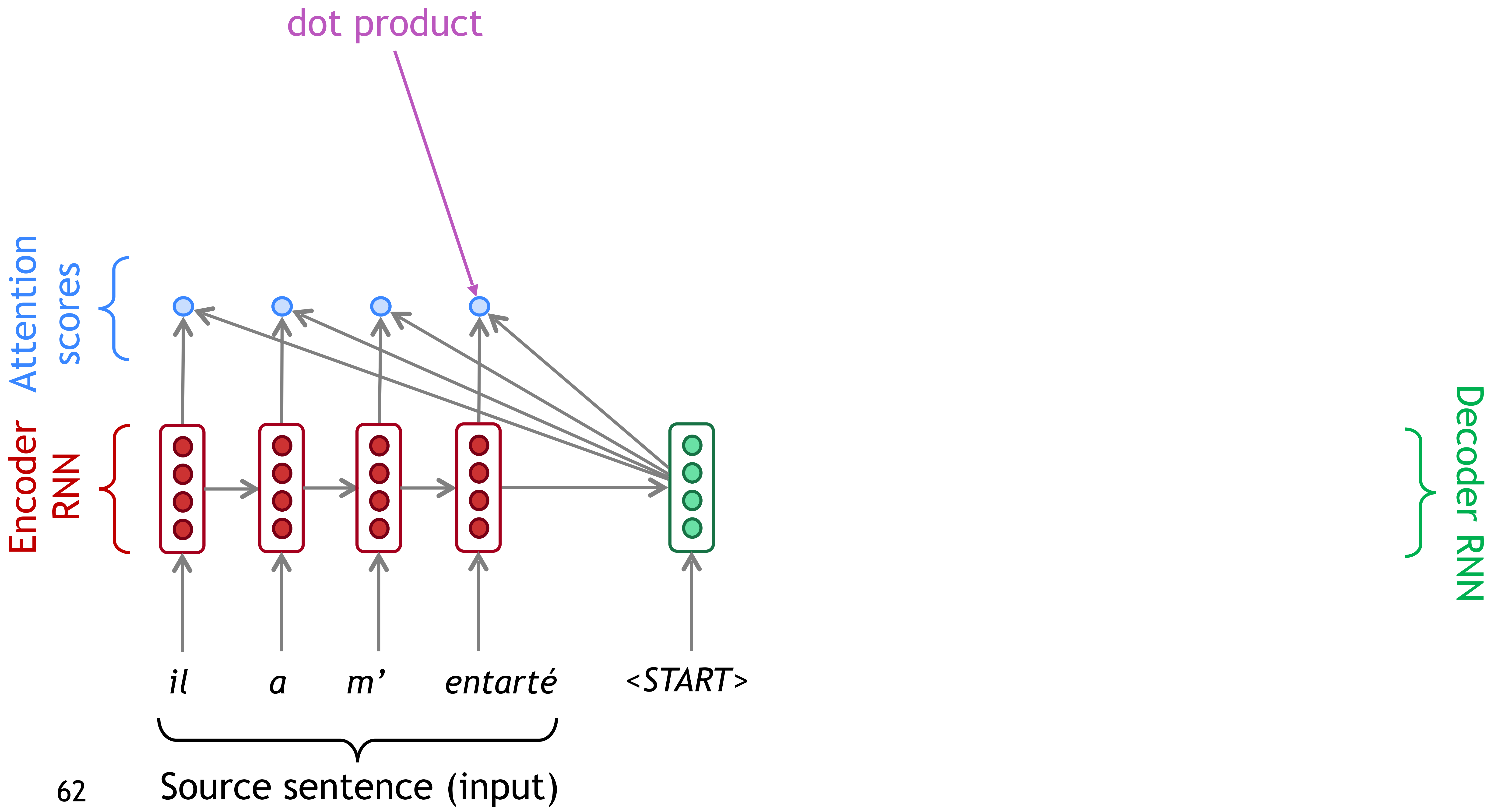
http://web.stanford.edu/class/cs224n/slides/cs224n-2020-lecture08-nmt.pdf

Sequence-to-sequence with attention



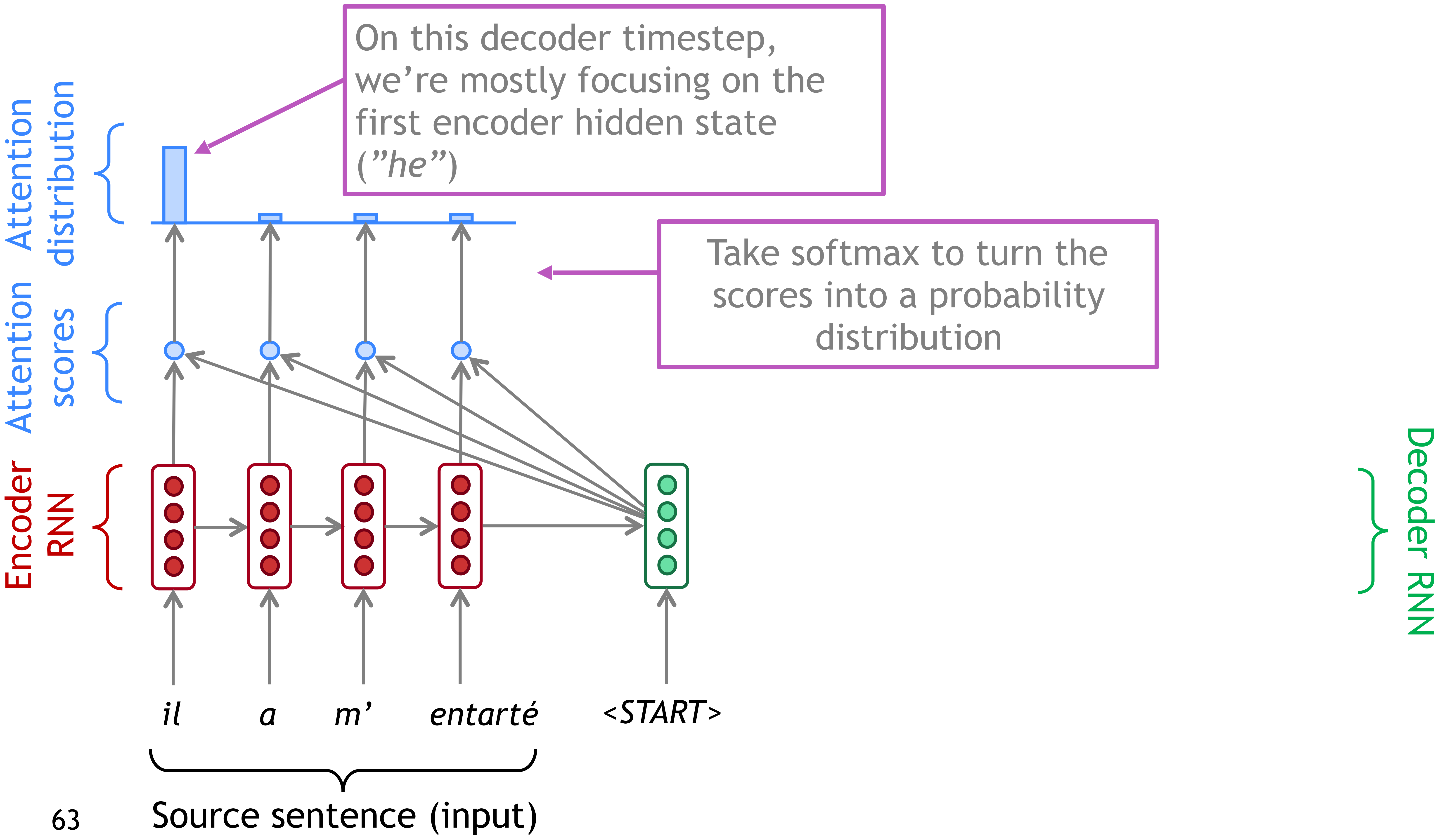
http://web.stanford.edu/class/cs224n/slides/cs224n-2020-lecture08-nmt.pdf

Sequence-to-sequence with attention

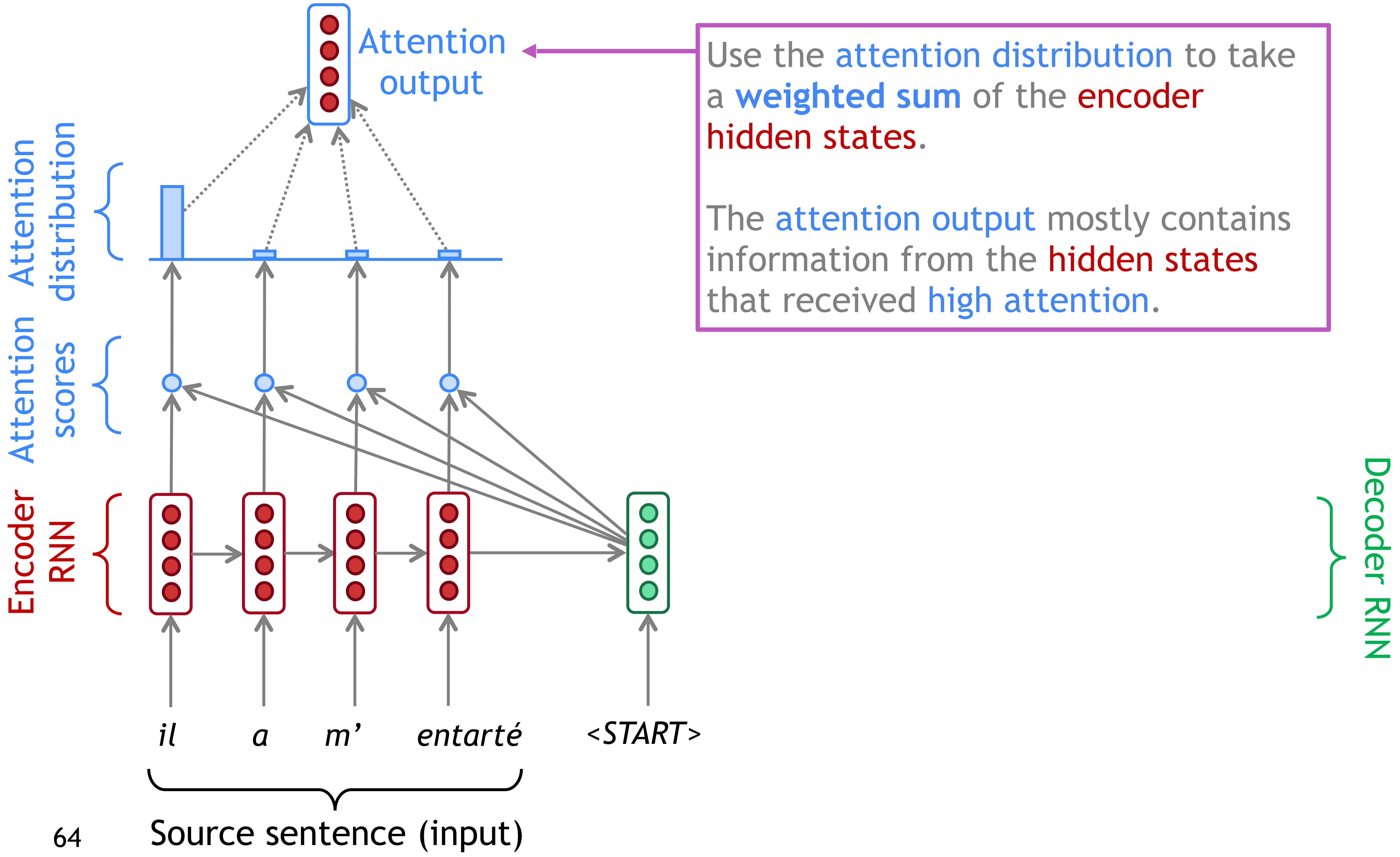


http://web.stanford.edu/class/cs224n/slides/cs224n-2020-lecture08-nmt.pdf

Sequence-to-sequence with attention



Sequence-to-sequence with attention

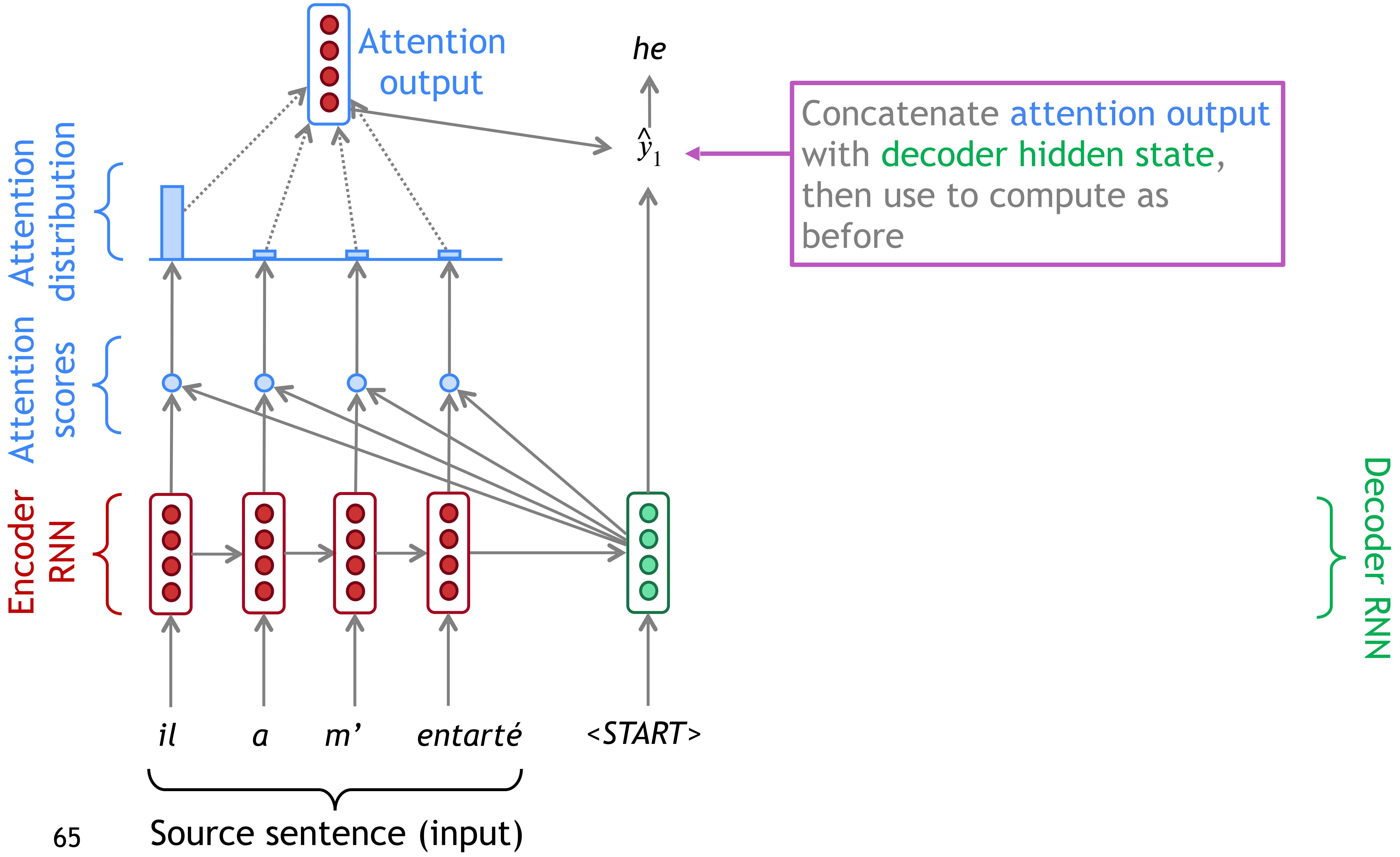


Use the attention distribution to take a **weighted sum** of the encoder hidden states.

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

http://web.stanford.edu/class/cs224n/slides/cs224n-2020-lecture08-nmt.pdf

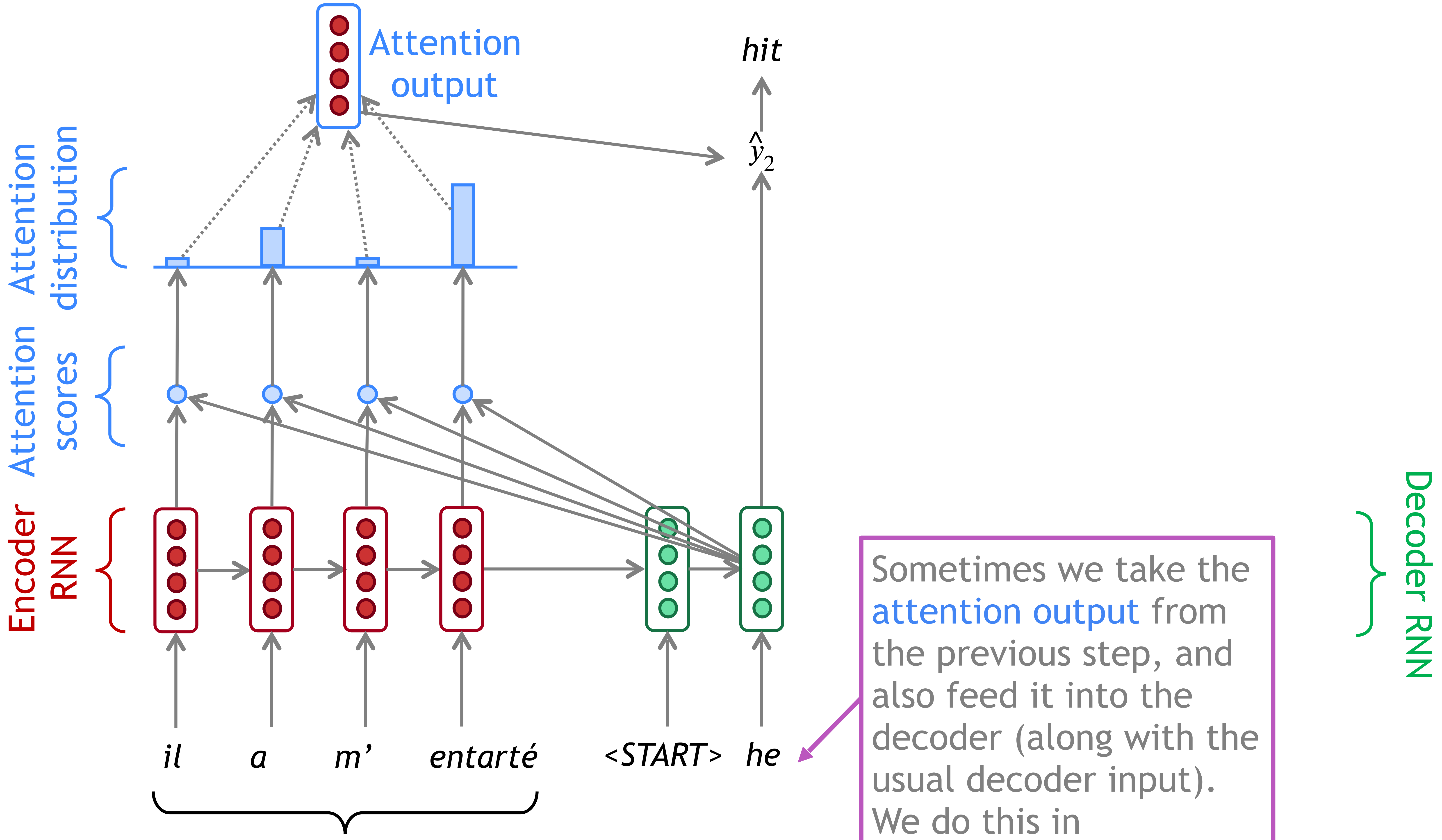
Sequence-to-sequence with attention



Concatenate attention output with decoder hidden state, then use to compute as before

http://web.stanford.edu/class/cs224n/slides/cs224n-2020-lecture08-nmt.pdf

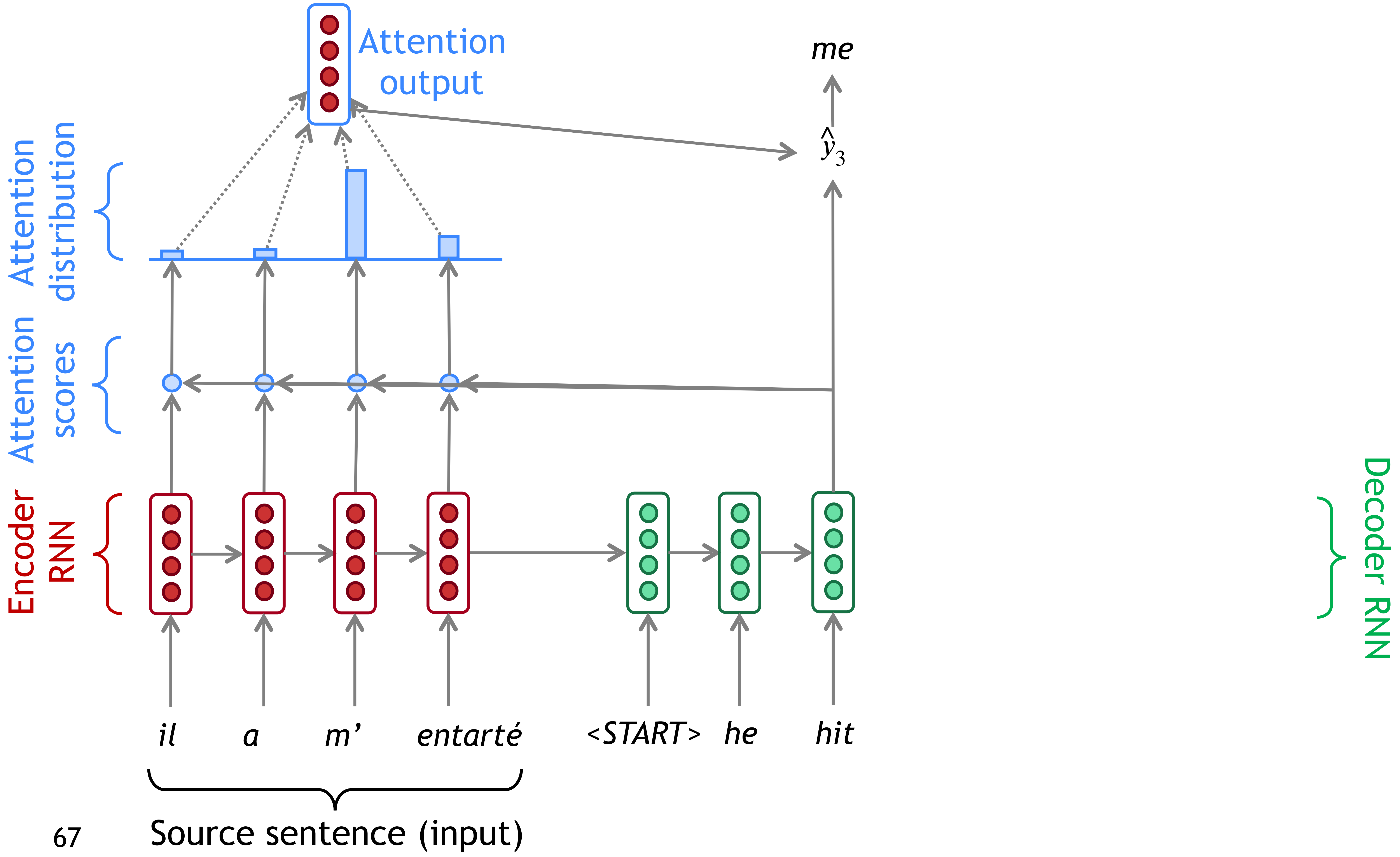
Sequence-to-sequence with attention



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.

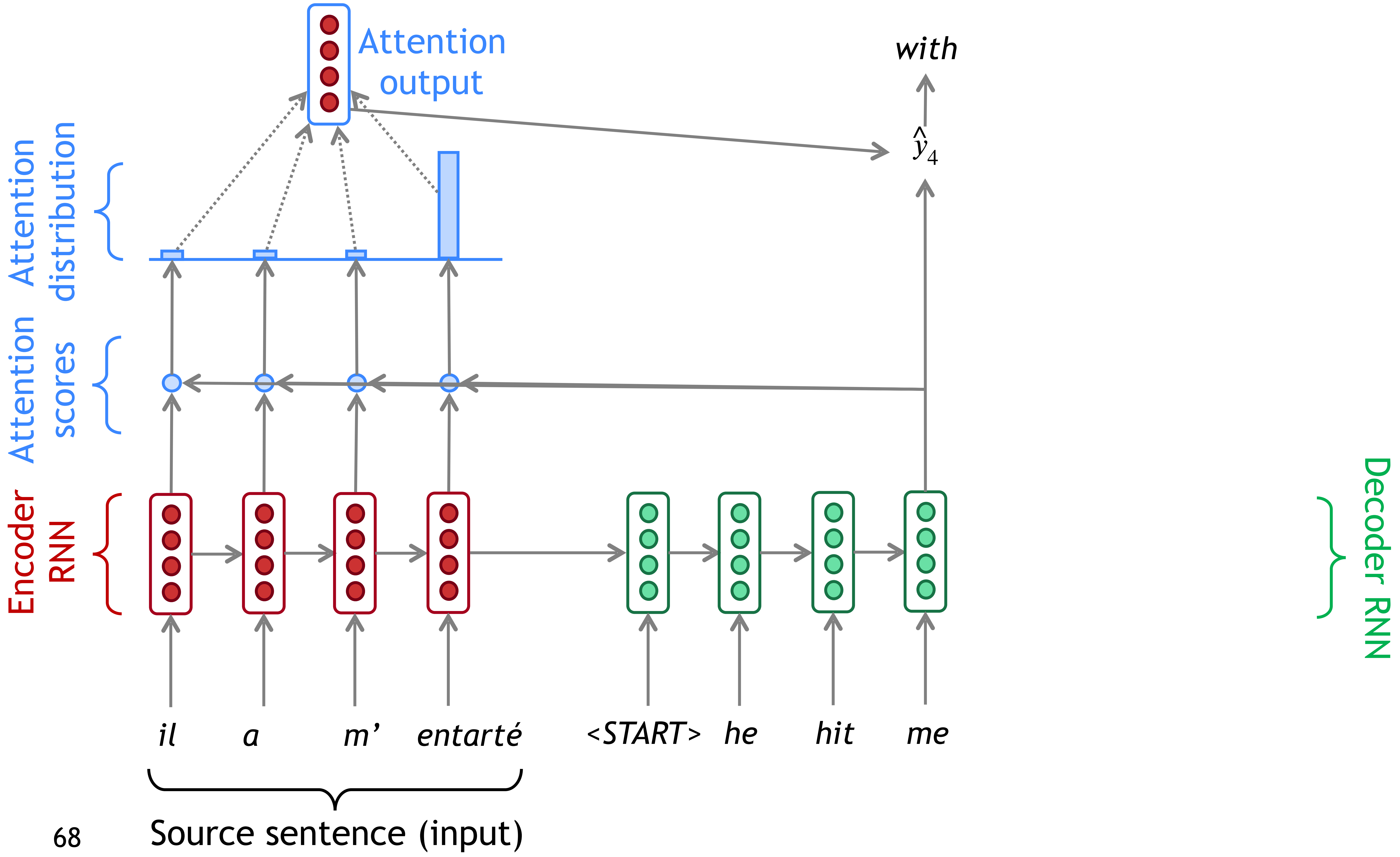
http://web.stanford.edu/class/cs224n/slides/cs224n-2020-lecture08-nmt.pdf

Sequence-to-sequence with attention



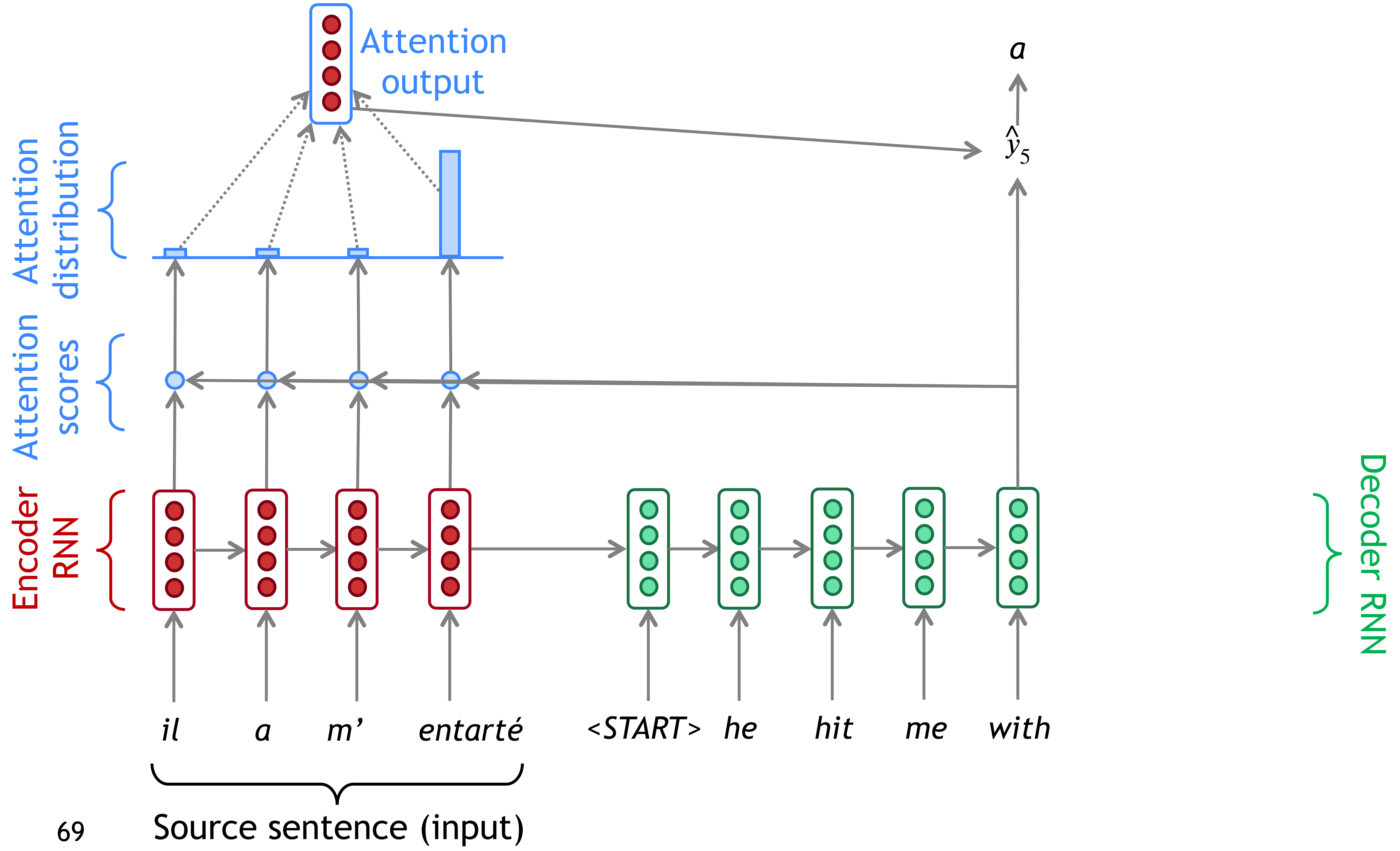
http://web.stanford.edu/class/cs224n/slides/cs224n-2020-lecture08-nmt.pdf

Sequence-to-sequence with attention



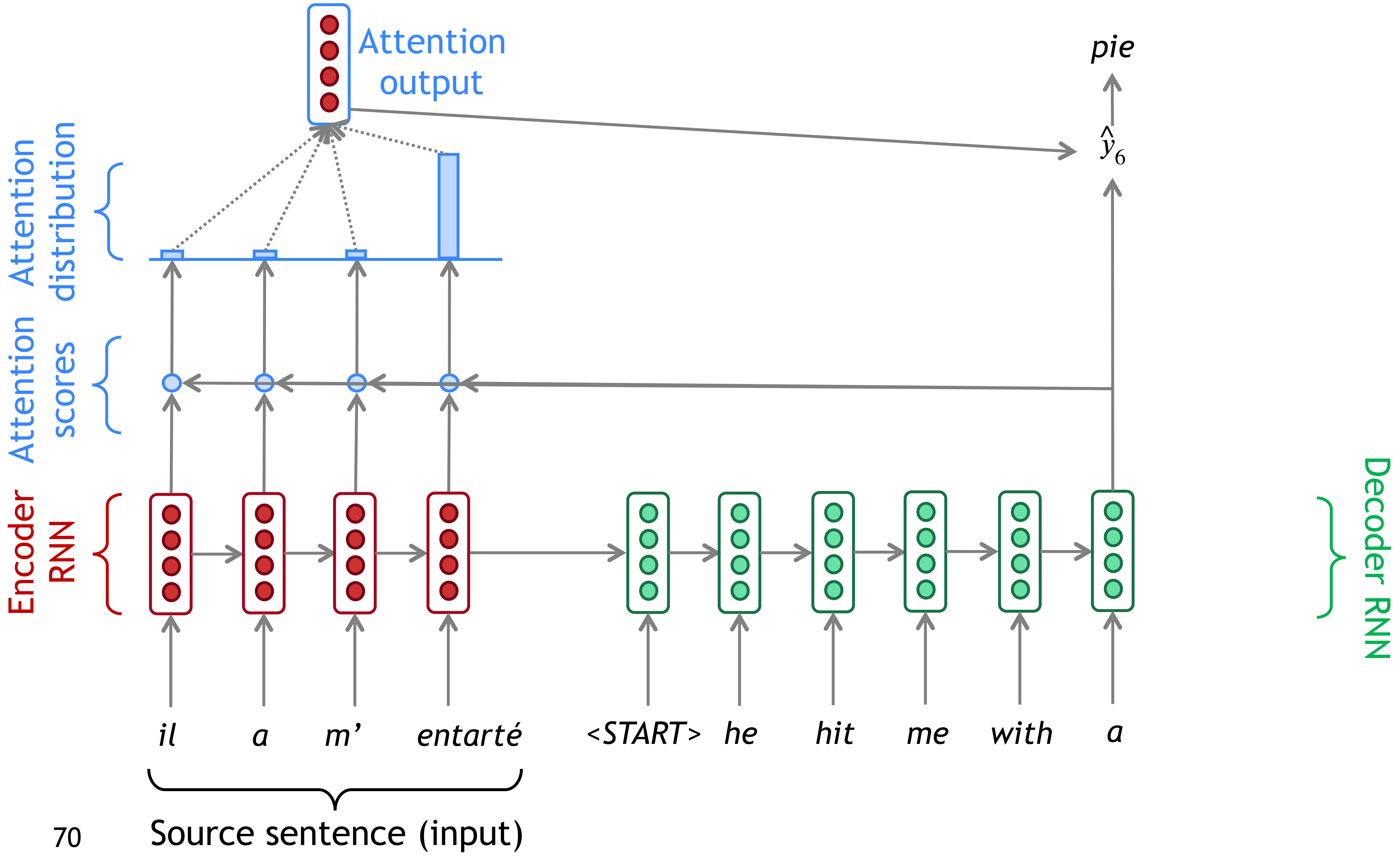
http://web.stanford.edu/class/cs224n/slides/cs224n-2020-lecture08-nmt.pdf

Sequence-to-sequence with attention



http://web.stanford.edu/class/cs224n/slides/cs224n-2020-lecture08-nmt.pdf

Sequence-to-sequence with attention



http://web.stanford.edu/class/cs224n/slides/cs224n-2020-lecture08-nmt.pdf

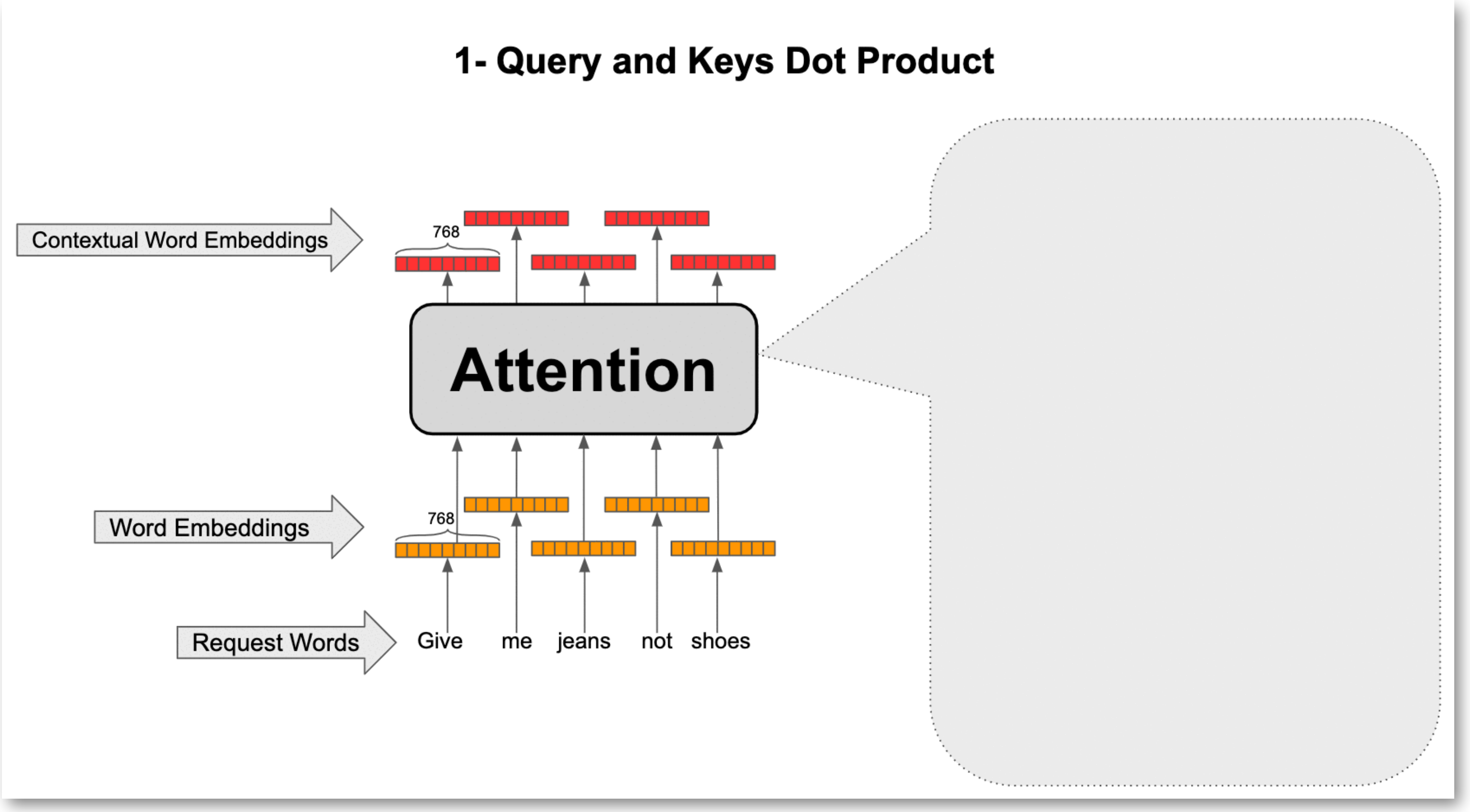
Attention Mechanisms

Dot Product

- $\mathbf{q}, \mathbf{k}_i \in \mathbb{R}^d$ for all i
- dot product between the query and a key, which is then divided by $\sqrt{d}$ to minimize the unrelated influence of the dimension d on the scores. $\alpha(\mathbf{q}, \mathbf{k}) = \langle \mathbf{q}, \mathbf{k} \rangle / \sqrt{d}$.
- $\mathbf{Q} \in \mathbb{R}^{m \times d}$ contains m queries and $\mathbf{K} \in \mathbb{R}^{n \times d}$ has all the n keys. We can compute all mn scores by $\alpha(\mathbf{Q}, \mathbf{K}) = \mathbf{Q}\mathbf{K}^\top / \sqrt{d}$.

MLP

- Both Query and keys into $\mathbb{R}^h$ by learnable weights parameters.
- Learnable weights are $\mathbf{W}_k \in \mathbb{R}^{h \times d_k}$, $\mathbf{W}_q \in \mathbb{R}^{h \times d_q}$, and $\mathbf{v} \in \mathbb{R}^h$.
- $\alpha(\mathbf{k}, \mathbf{q}) = \mathbf{v}^\top \tanh(\mathbf{W}_k \mathbf{k} + \mathbf{W}_q \mathbf{q})$.



Attention Mechanisms