Attention Network

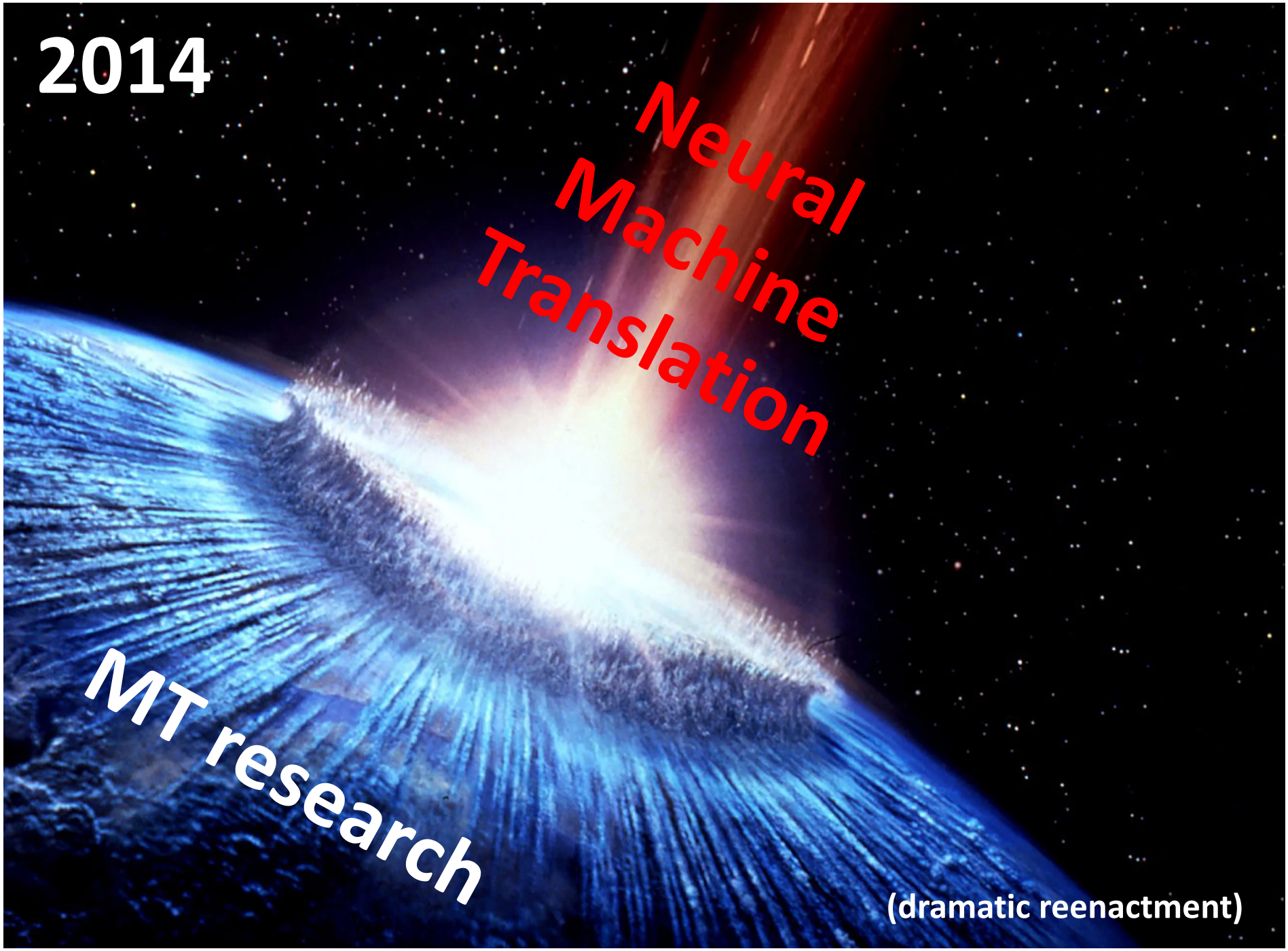
- Stanford CS 224n Lecture 10, 11, 12 slides
- Stanford CS 231n Lecture 10 slides

Machine Translation

Machine Translation (MT) is the task of translating a sentence *x* from one language (the source language) to a sentence *y* in another language (the target language).

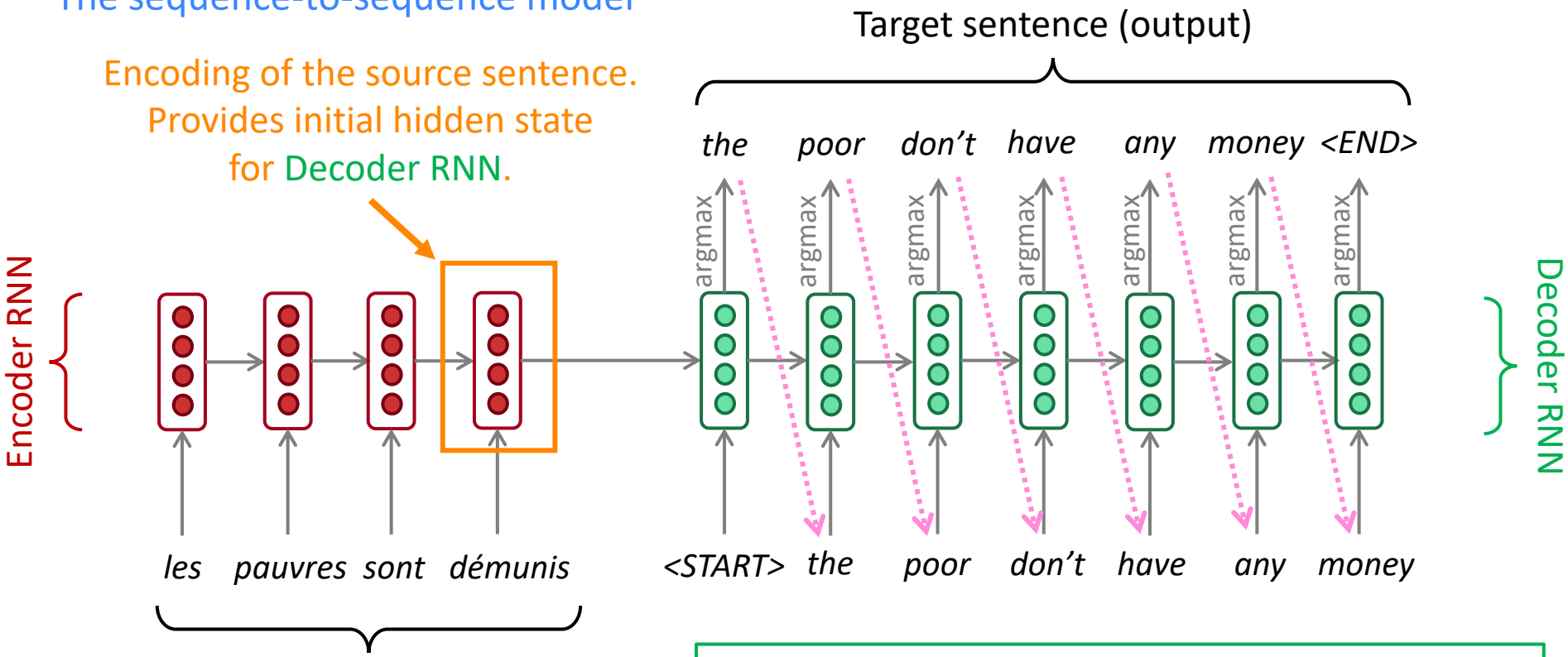
x: *L'homme est né libre, et partout il est dans les fers*

y: *Man is born free, but everywhere he is in chains*


2014
Neural
Machine
Translation
MT research
(dramatic reenactment)

Neural Machine Translation (NMT)

The sequence-to-sequence model



Encoder RNN produces an encoding of the source sentence.

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

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

Image Captioning

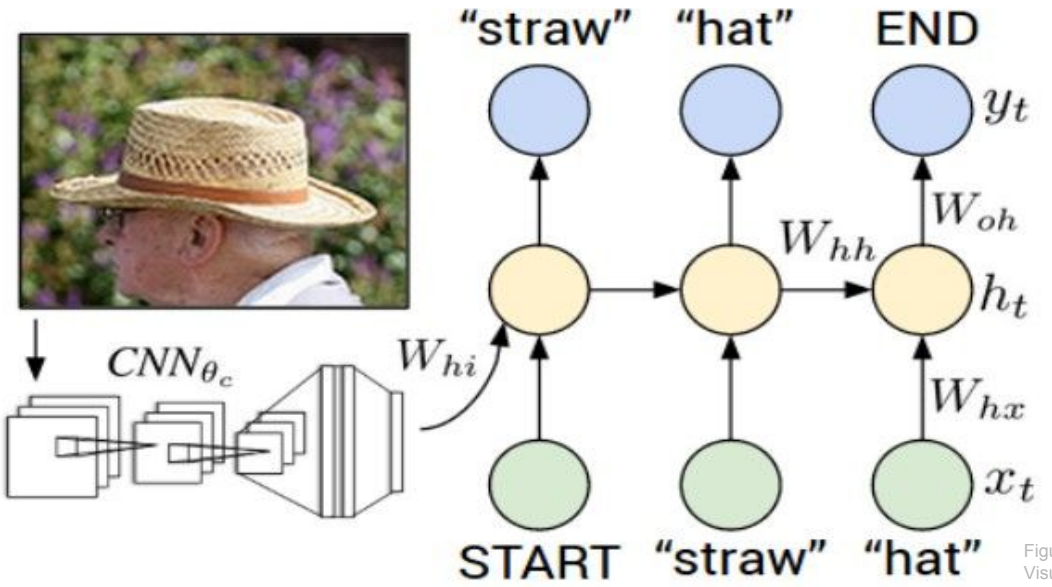
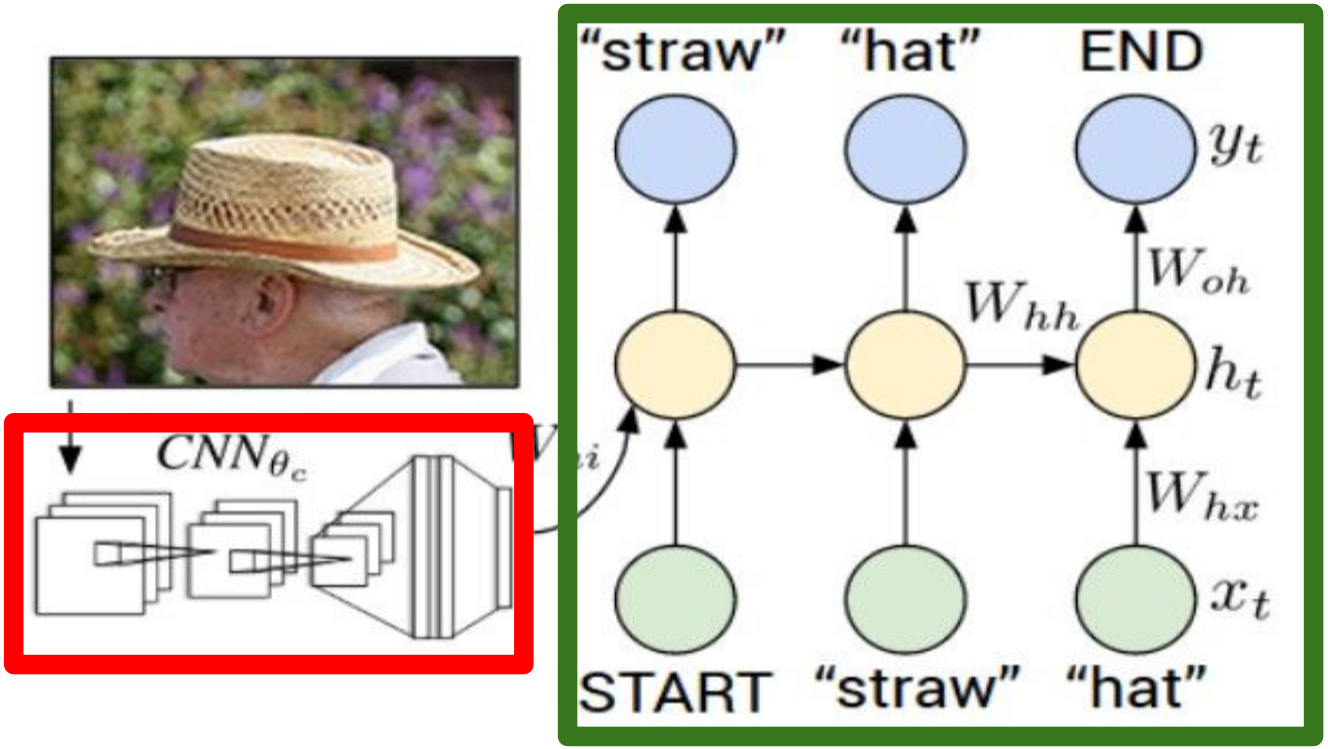


Figure from Karpathy et a, "Deep Visual-Semantic Alignments for Generating Image Descriptions", CVPR 2015; figure copyright IEEE, 2015.
Reproduced for educational purposes.

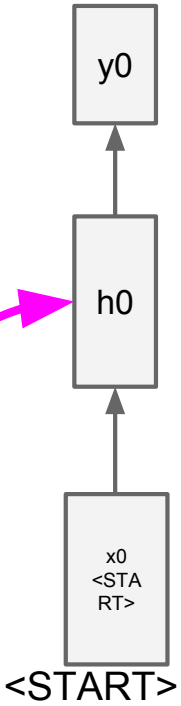
Explain Images with Multimodal Recurrent Neural Networks, Mao et al.
Deep Visual-Semantic Alignments for Generating Image Descriptions, Karpathy and Fei-Fei
Show and Tell: A Neural Image Caption Generator, Vinyals et al.
Long-term Recurrent Convolutional Networks for Visual Recognition and Description, Donahue et al.
Learning a Recurrent Visual Representation for Image Caption Generation, Chen and Zitnick

Recurrent Neural Network



Convolutional Neural Network

test image



before:

h = tanh(Wxh * x + Whh * h)

now:

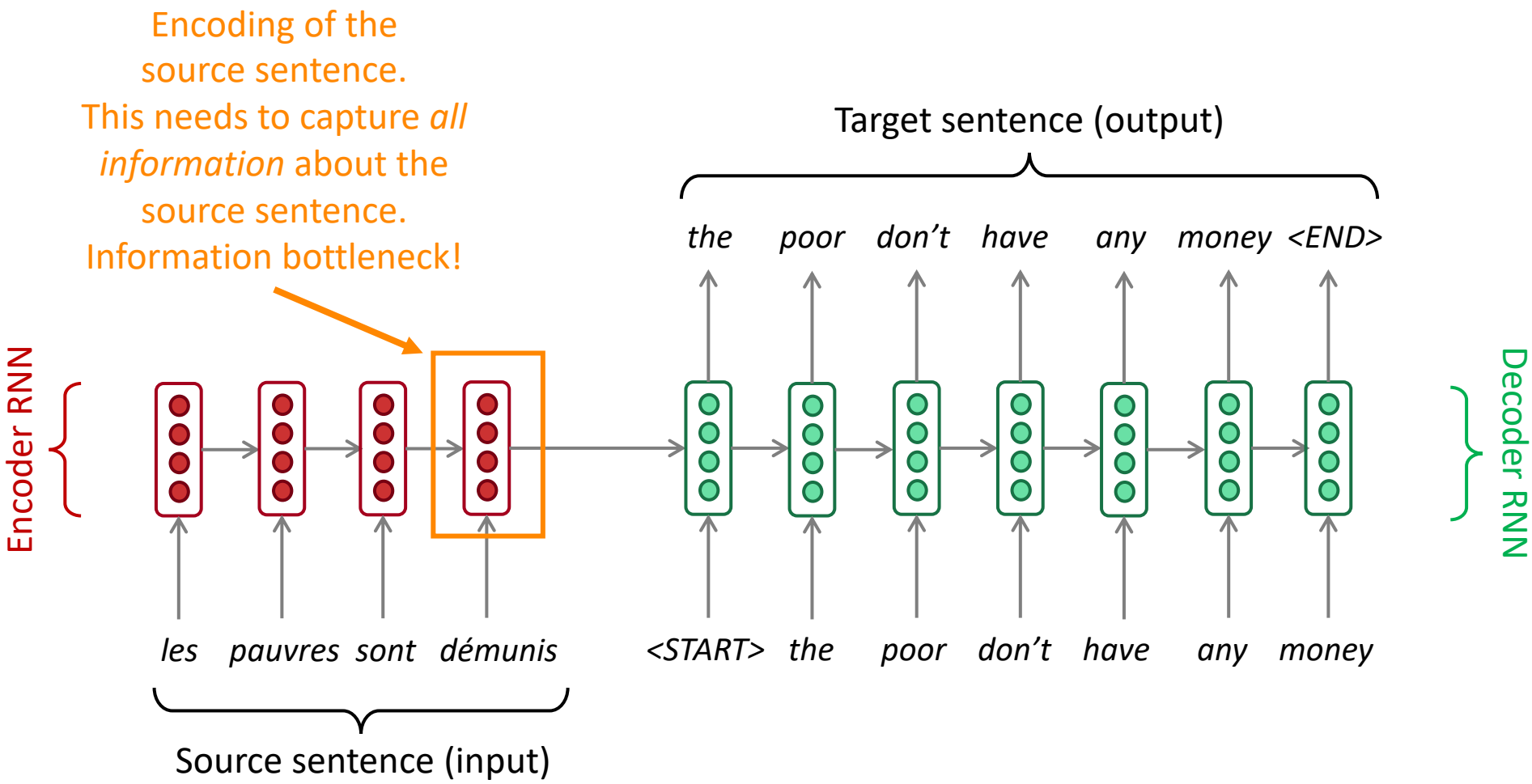
h = tanh(Wxh * x + Whh * h **+ Wih * v**)

Many difficulties remains in MT

- Out-of-vocabulary words
- Domain mismatch between train and test data
- Maintaining context over longer text
- Low-resource language pairs
- Using common sense is still hard
- NMT picks up biases in training data
- ……..

- **ATTENTION**

Sequence-to-sequence: the bottleneck problem



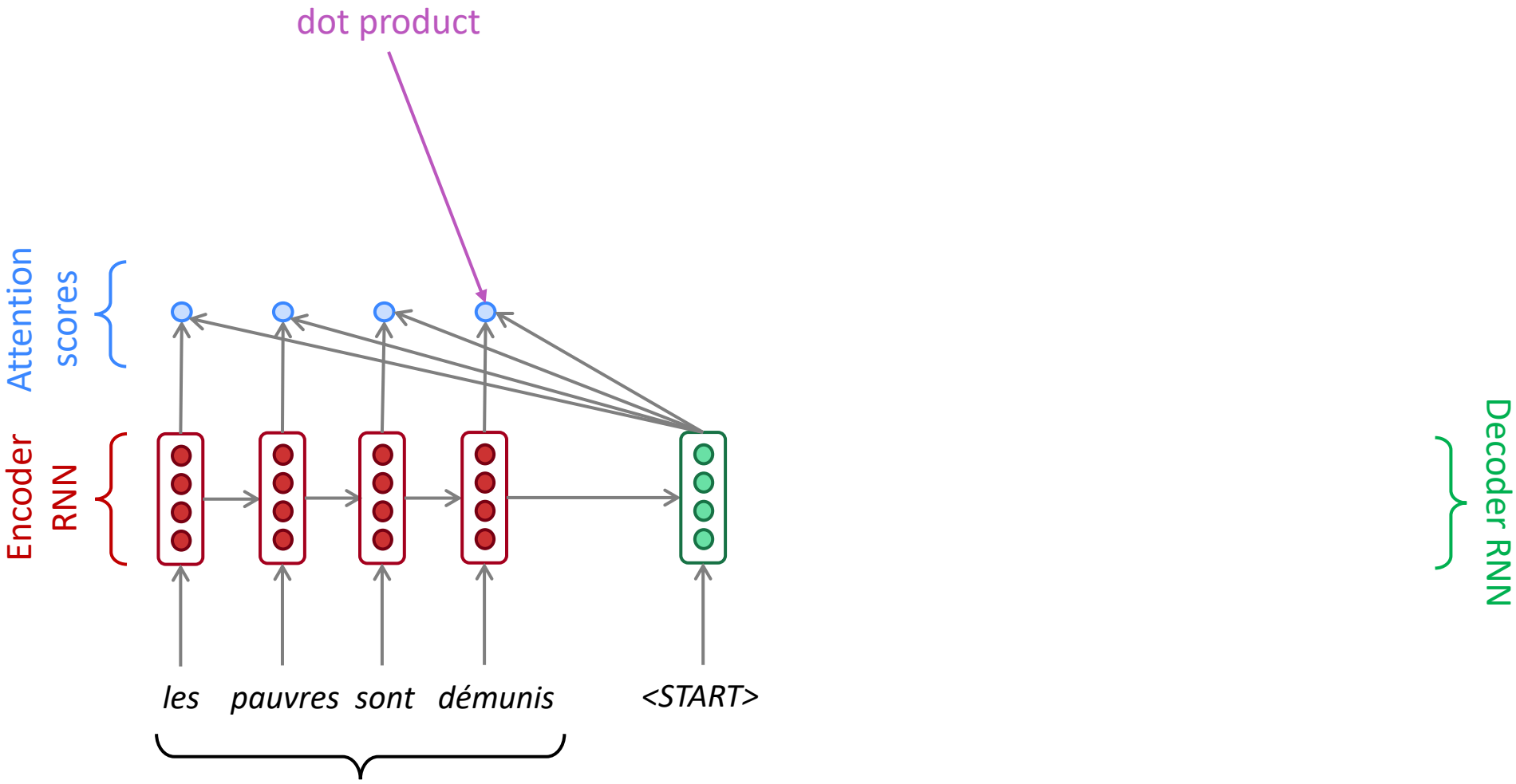
Attention

- **Attention** provides a solution to the bottleneck problem.

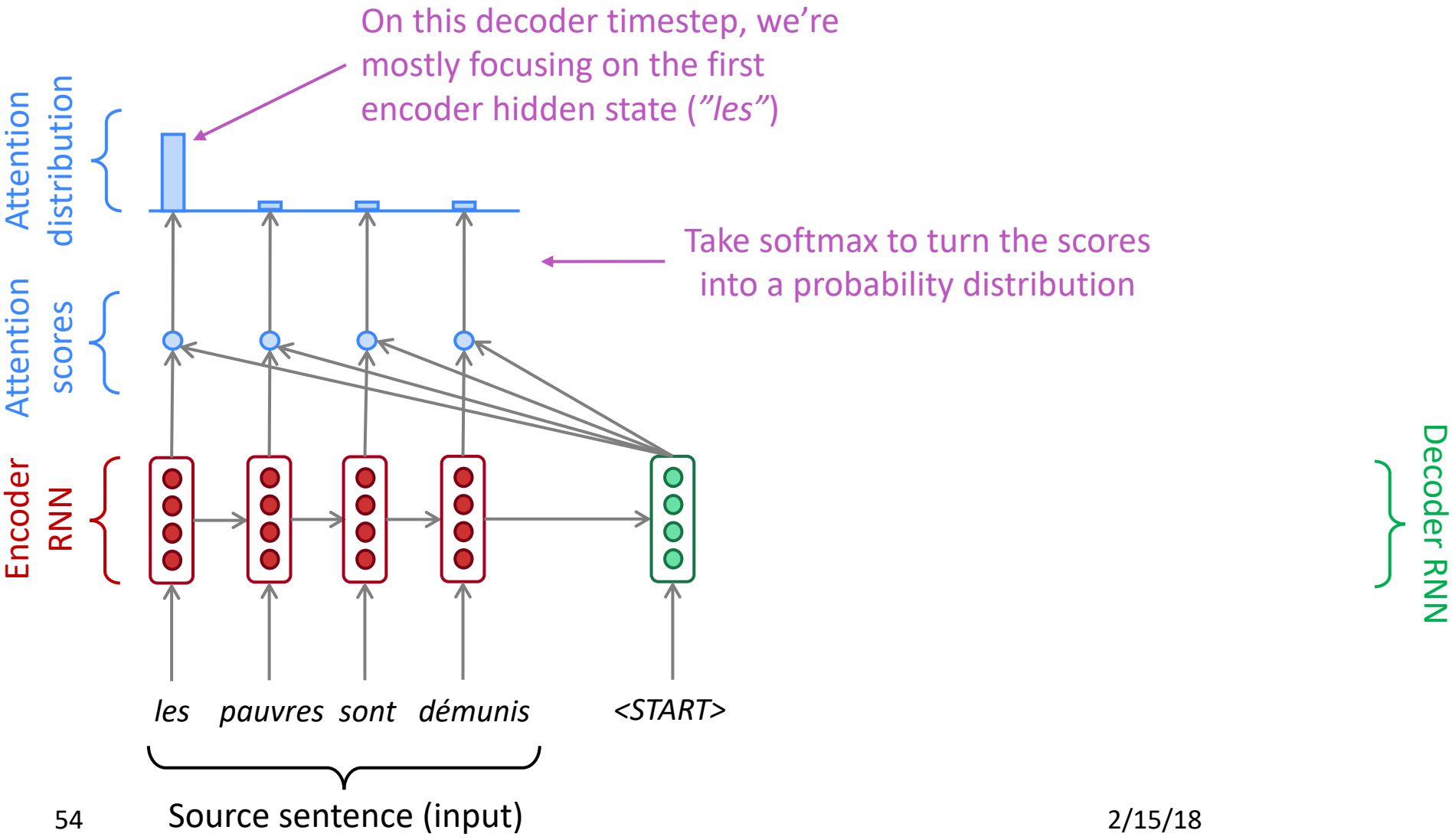
- Core idea: on each step of the decoder, *focus on a particular part* of the source sequence

- First we will show via diagram (no equations), then we will show with equations

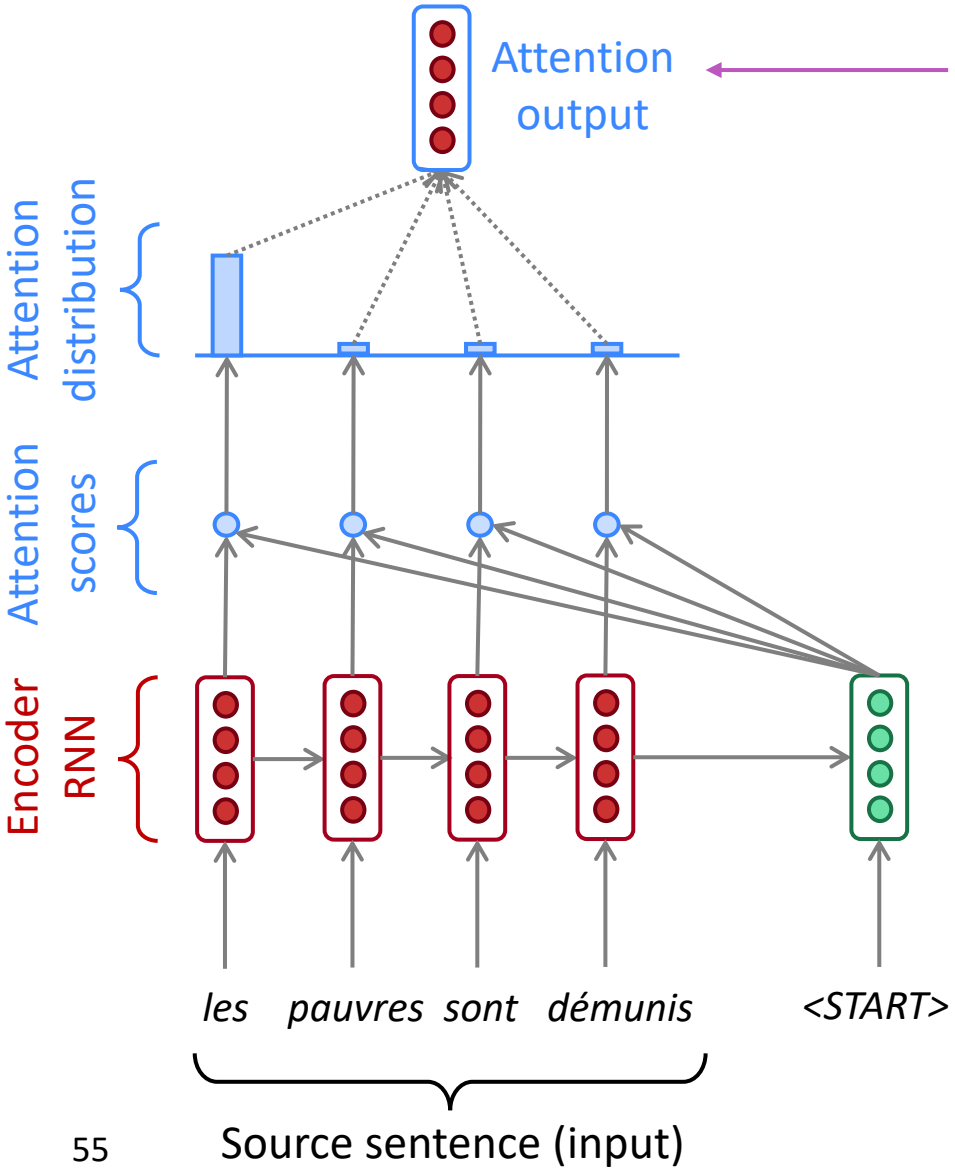
Sequence-to-sequence with attention



Sequence-to-sequence with attention



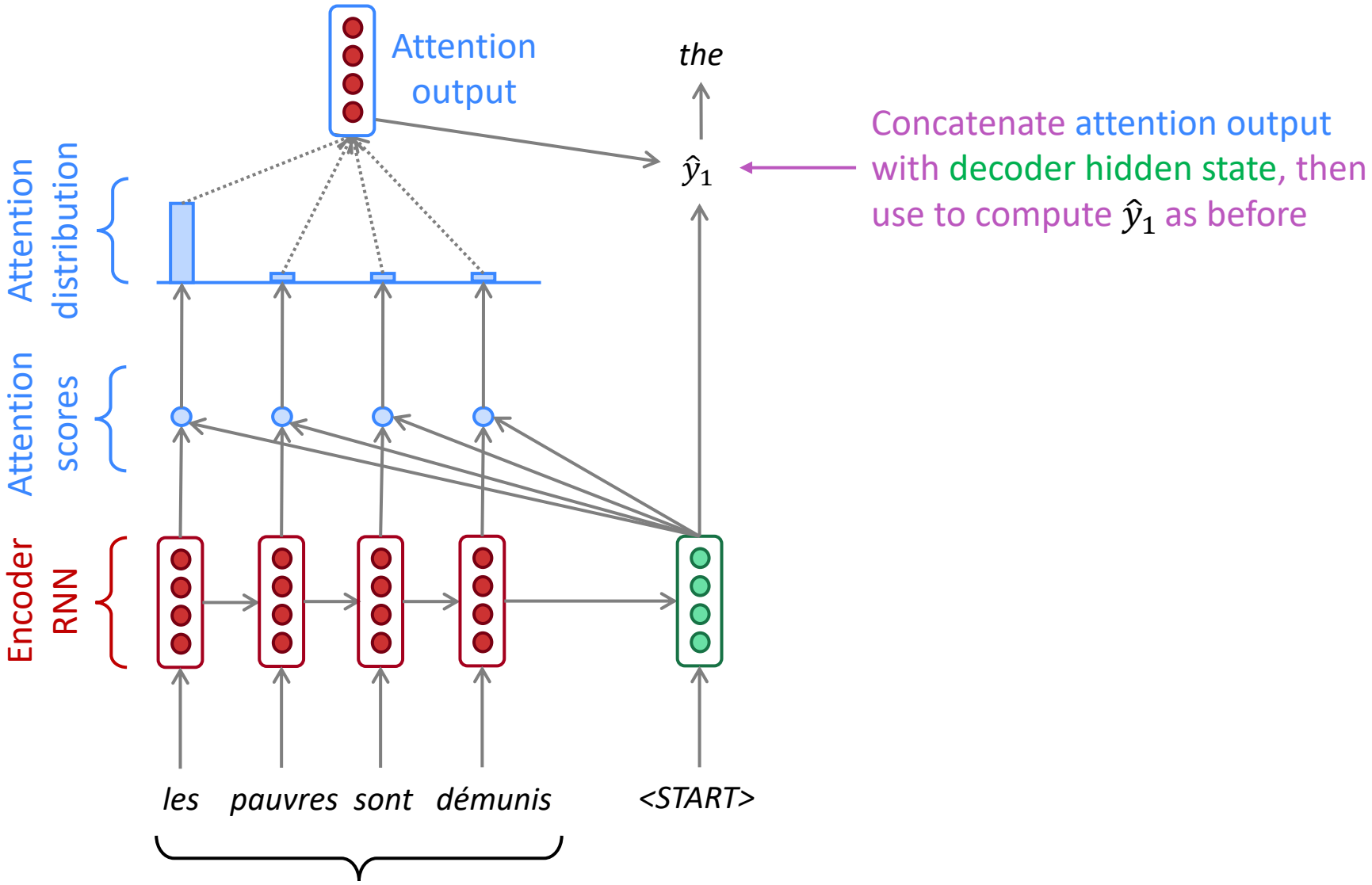
Sequence-to-sequence with attention



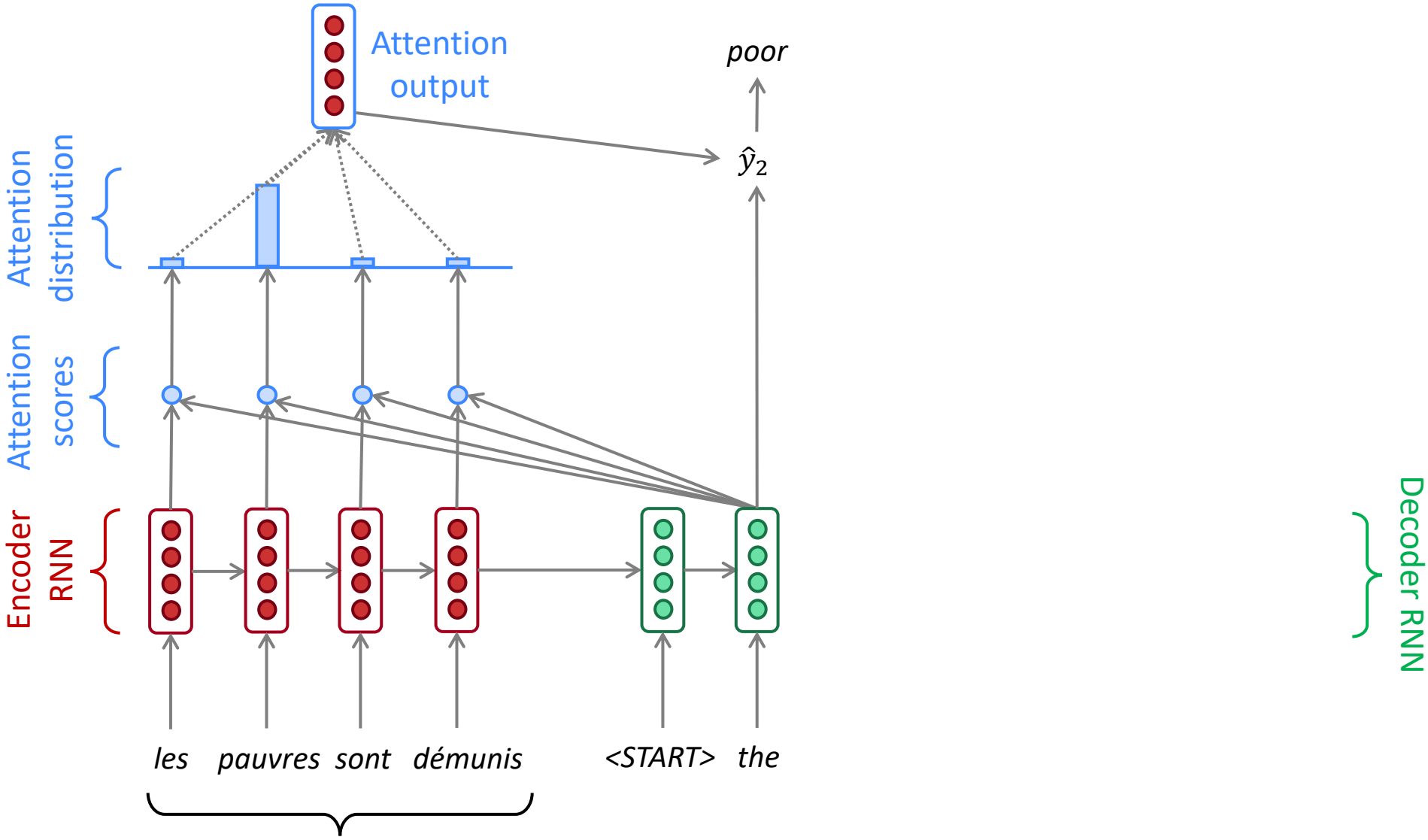
Use the attention distribution to take a **weighted sum** of the encoder hidden states.

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

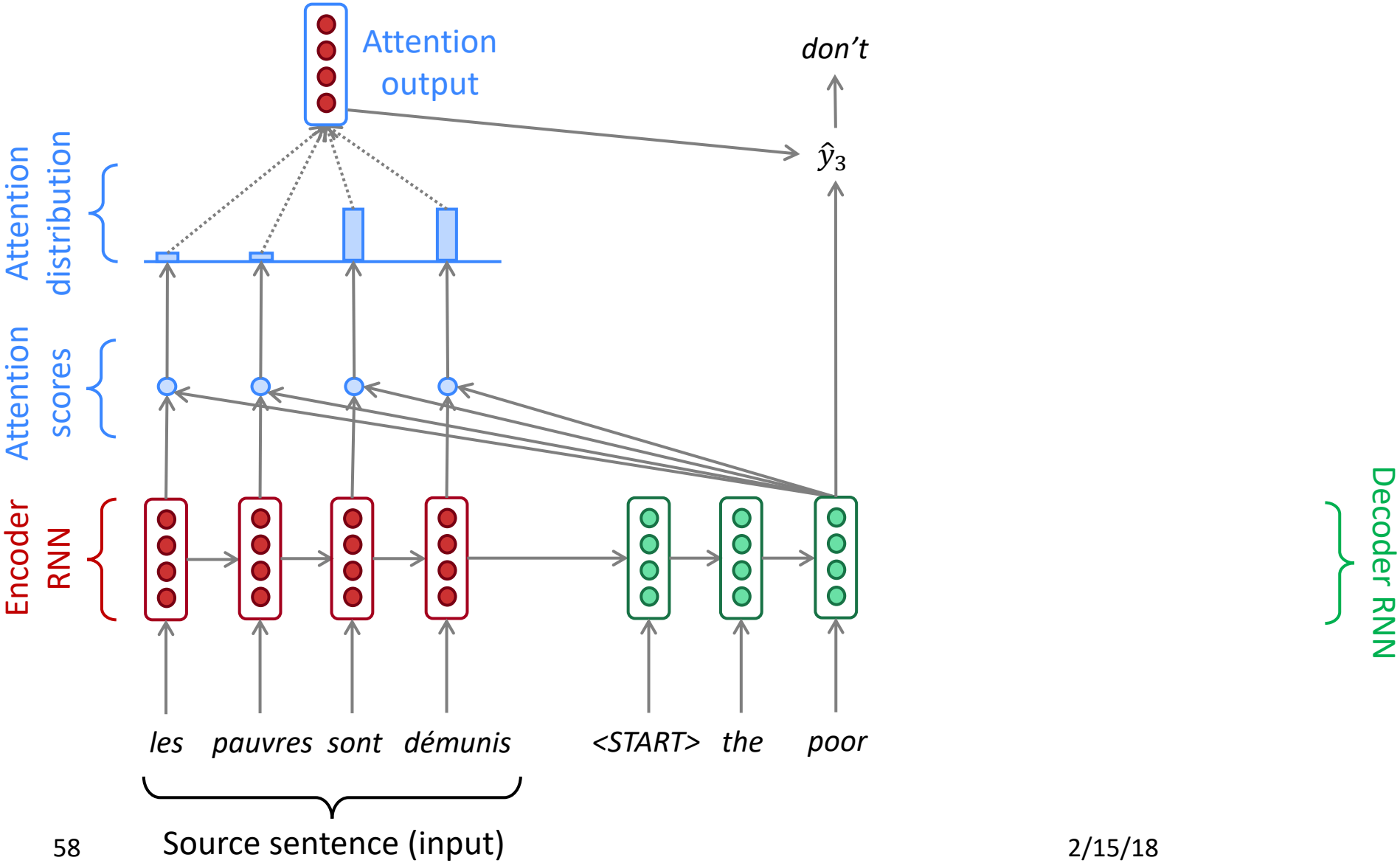
Sequence-to-sequence with attention



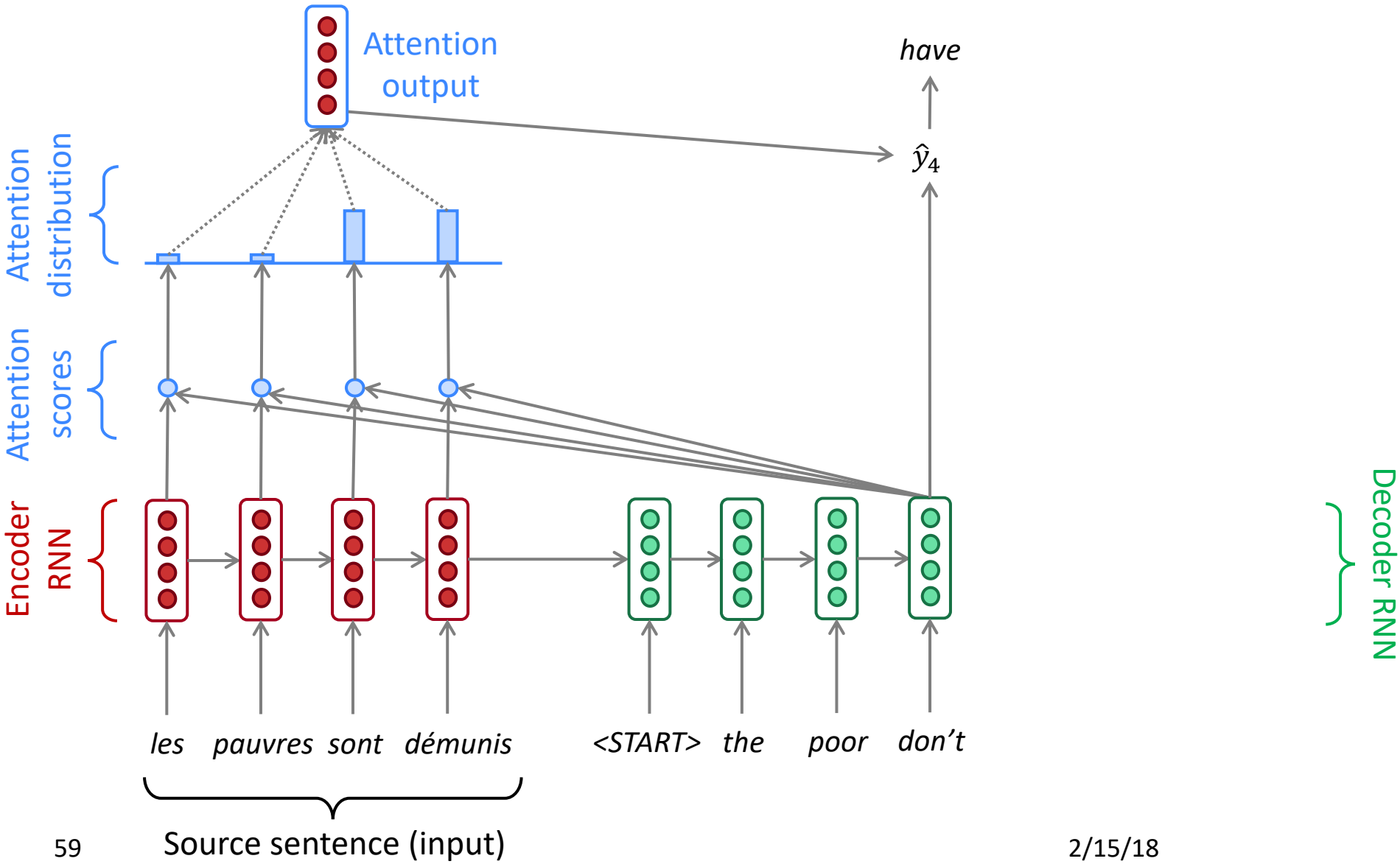
Sequence-to-sequence with attention



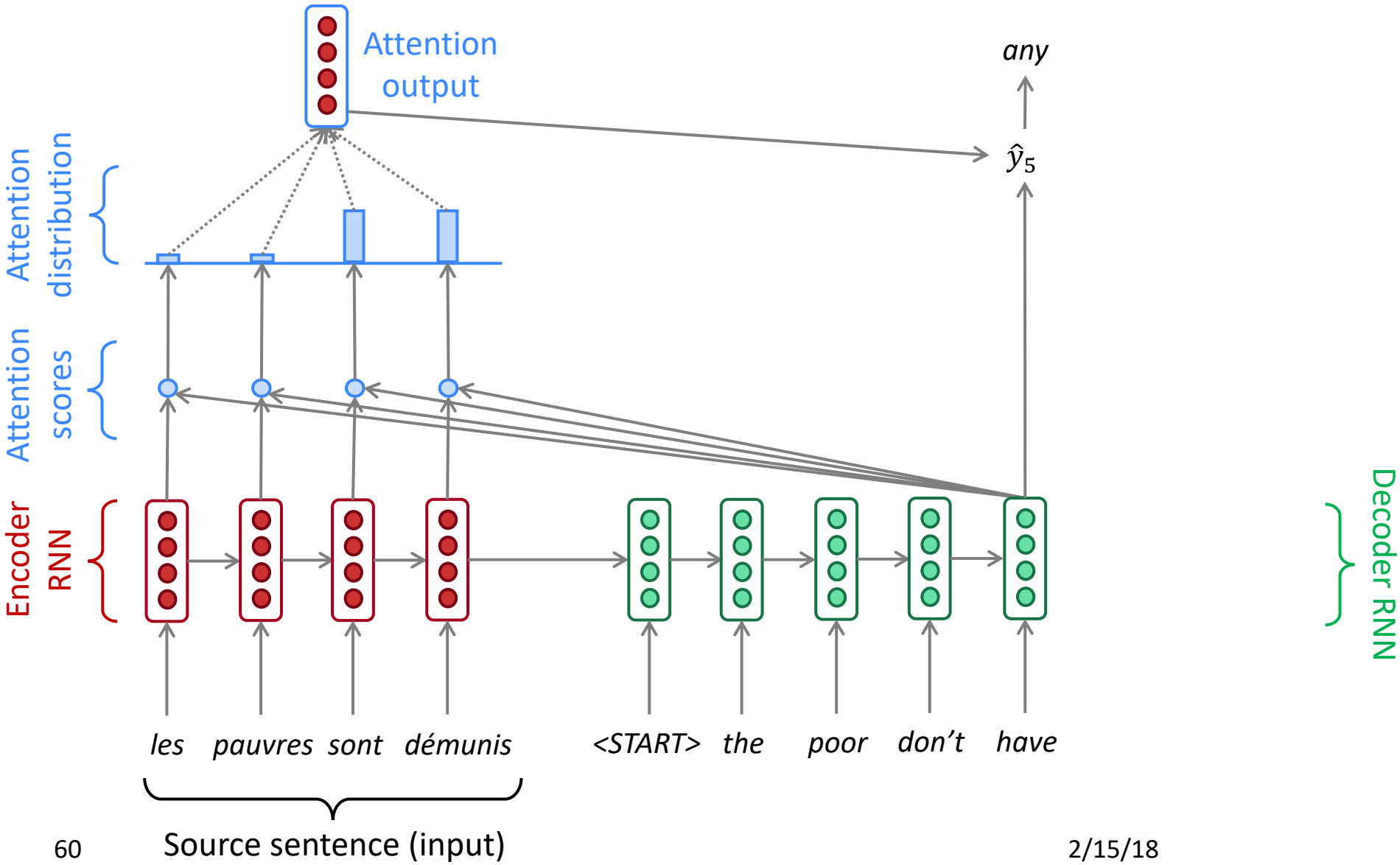
Sequence-to-sequence with attention



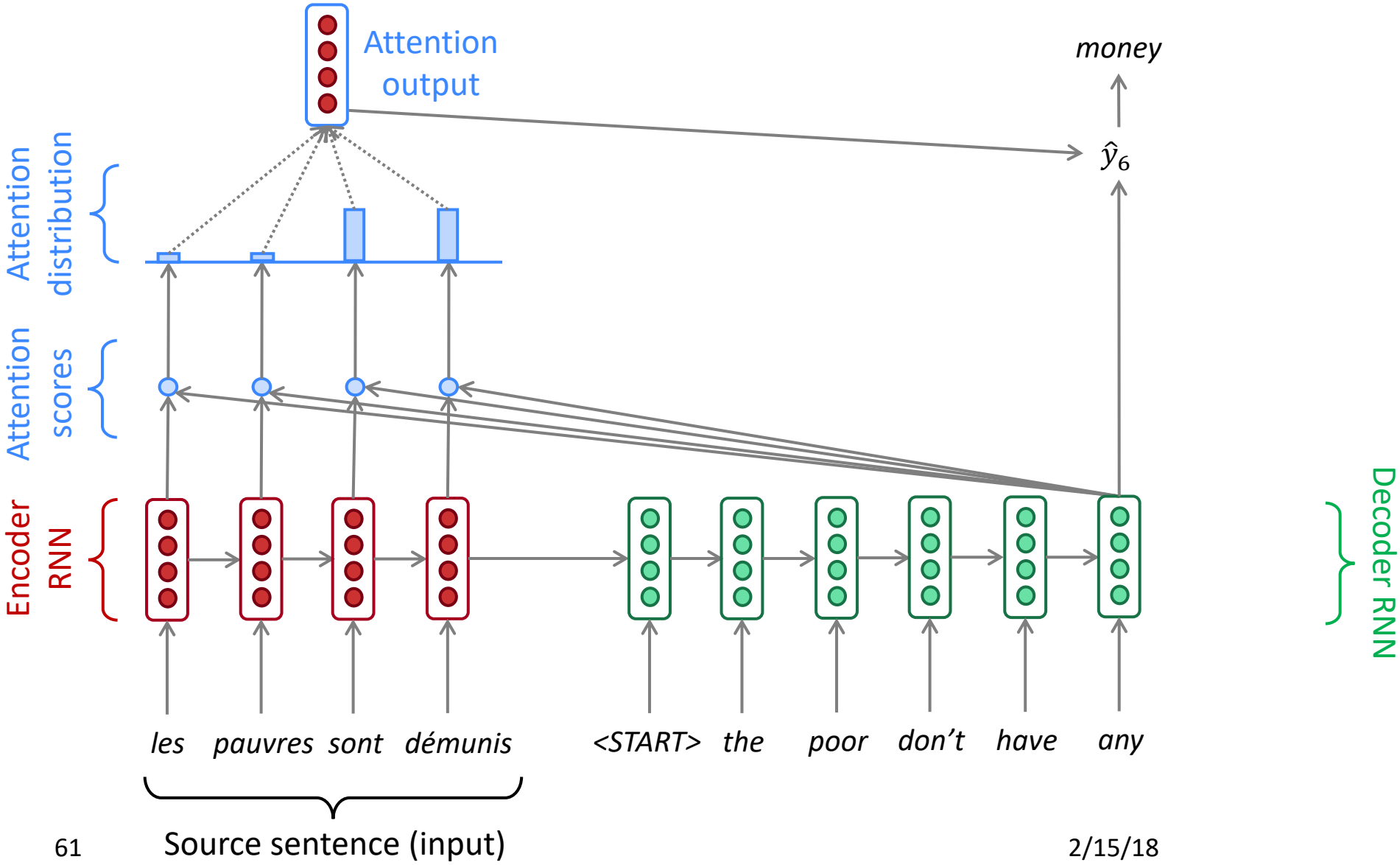
Sequence-to-sequence with attention



Sequence-to-sequence with attention



Sequence-to-sequence with attention



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)

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

- 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

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

There are *several* attention variants

- We have some *values* $\boldsymbol{h}_1, \ldots, \boldsymbol{h}_N \in \mathbb{R}^{d_1}$ and a *query* $\boldsymbol{s} \in \mathbb{R}^{d_2}$

- Attention always involves computing the *attention output* $\boldsymbol{a} \in \mathbb{R}^{d_1}$ (sometimes called the *context vector*) from the *attention scores* $\boldsymbol{e} \in \mathbb{R}^N$ (or *attention logits*) like so:

$$\alpha = \text{softmax}(\boldsymbol{e}) \in \mathbb{R}^N \qquad \text{(take softmax)}$$

$$\boldsymbol{a} = \sum_{i=1}^{N} \alpha_i \boldsymbol{h}_i \in \mathbb{R}^{d_1} \qquad \text{(take weighted sum)}$$

- However, there are several ways you can compute $\boldsymbol{e} \in \mathbb{R}^N$

Attention variants

There are several ways you can compute $\boldsymbol{e} \in \mathbb{R}^N$ from $\boldsymbol{h}_1, \ldots, \boldsymbol{h}_N \in \mathbb{R}^{d_1}$ and $\boldsymbol{s} \in \mathbb{R}^{d_2}$:

- Basic dot-product attention: $\boldsymbol{e}_i = \boldsymbol{s}^T \boldsymbol{h}_i \in \mathbb{R}$
  - Note: this assumes $d_1 = d_2$
  - This is the version we saw earlier
- Multiplicative attention: $\boldsymbol{e}_i = \boldsymbol{s}^T \boldsymbol{W} \boldsymbol{h}_i \in \mathbb{R}$
  - Where $\boldsymbol{W} \in \mathbb{R}^{d_2 \times d_1}$ is a weight matrix
- Additive attention: $\boldsymbol{e}_i = \boldsymbol{v}^T \tanh(\boldsymbol{W}_1 \boldsymbol{h}_i + \boldsymbol{W}_2 \boldsymbol{s}) \in \mathbb{R}$
  - Where $\boldsymbol{W}_1 \in \mathbb{R}^{d_3 \times d_1}, \boldsymbol{W}_2 \in \mathbb{R}^{d_3 \times d_2}$ are weight matrices and $\boldsymbol{v} \in \mathbb{R}^{d_3}$ is a weight vector

More information: http://ruder.io/deep-learning-nlp-best-practices/index.html#attention

The First Attention Paper

Published as a conference paper at ICLR 2015

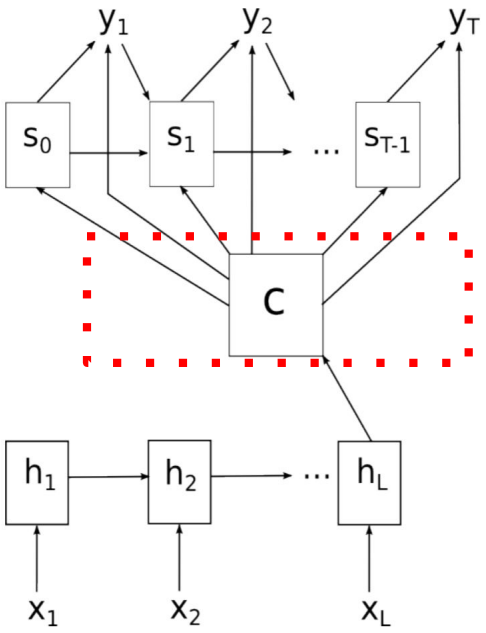
NEURAL MACHINE TRANSLATION BY JOINTLY LEARNING TO ALIGN AND TRANSLATE

Dzmitry Bahdanau
Jacobs University Bremen, Germany

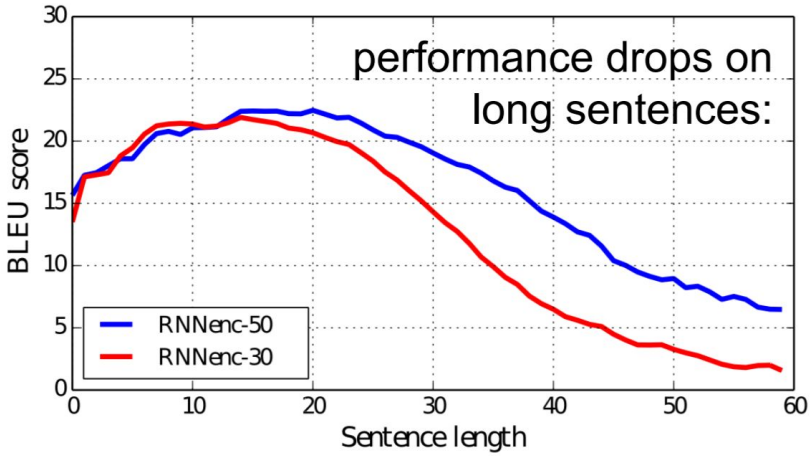
KyungHyun Cho **Yoshua Bengio***
Université de Montréal

Motivation

Existing encoder-decoder frameworks



Fixed-length representation is a potential bottleneck for long sentences

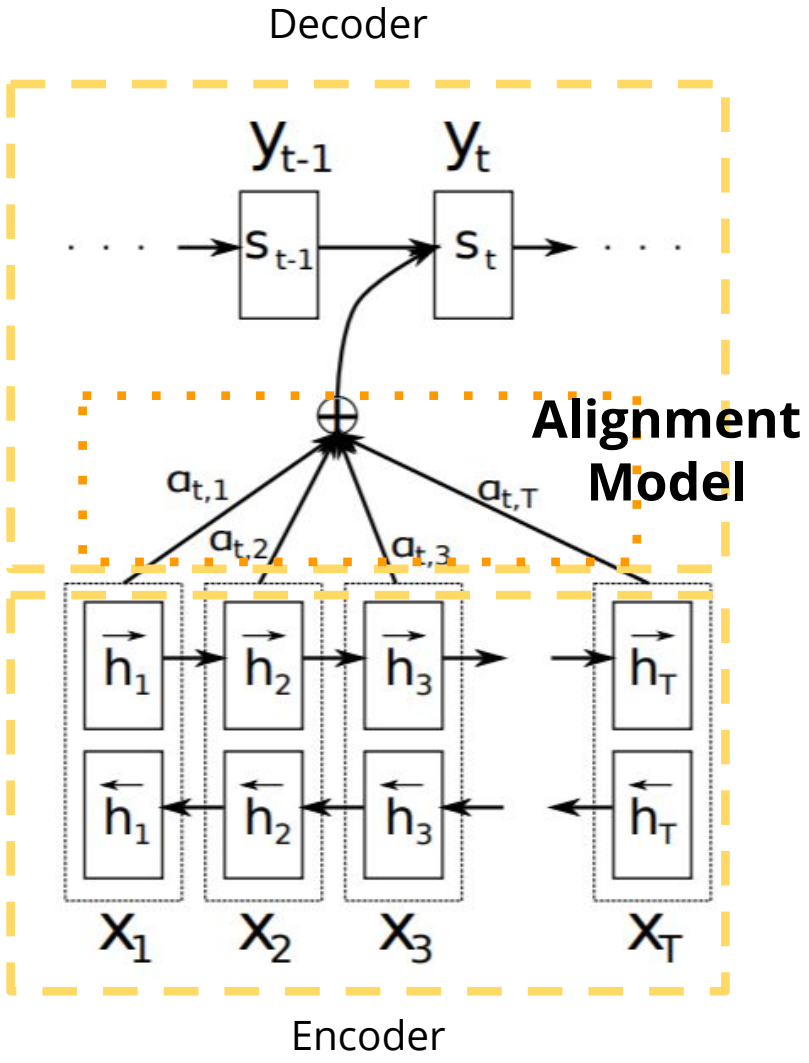


Evaluated on news-test-2014 from WMT '14 English-French parallel corpora

Method

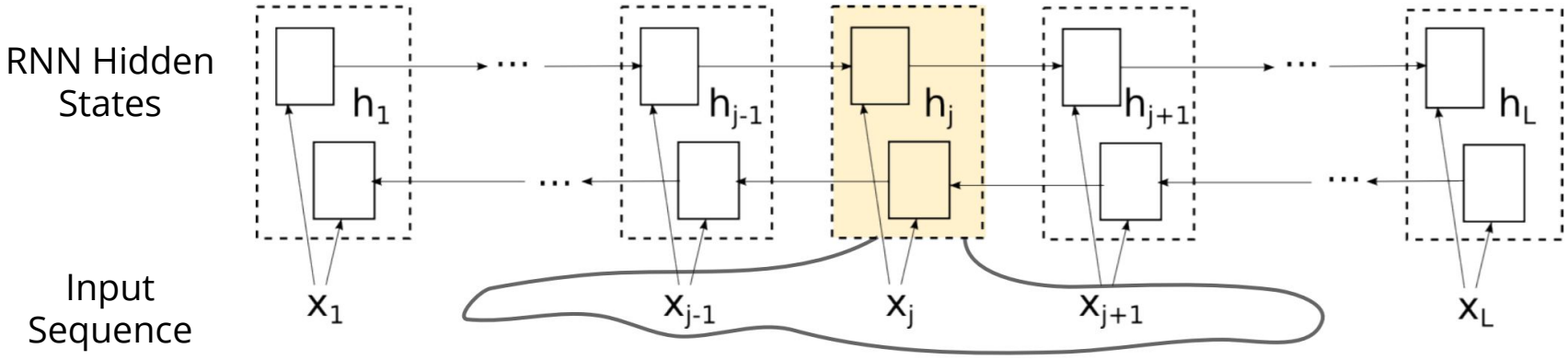
For each generated word y_t in the translation, soft-searches for a set of positions $(1, \ldots, T)$ in a source sentence $\boldsymbol{x}=(x_1, \ldots, x_T)$ where the most relevant information is concentrated.

The predicted target word y_t is based on the context vectors $\mathbf{c}_t=\sum\alpha_{t,j}\,\boldsymbol{h}_j$ associated with these source positions and all the previous generated target words s_{t-1}, y_{t-1}.



Encoder - Bidirectional RNN

- Map input sentence $\boldsymbol{x}$ to a sequence of annotations $\boldsymbol{h}$
- Each annotation summarizes the preceding words and the following words



$$\mathbf{x} = (x_1, \cdots, x_{T_x}), \xrightarrow{h_t = f(x_t, h_{t-1})} \begin{matrix} (\overrightarrow{h}_1, \cdots, \overrightarrow{h}_{T_x}) \\ (\overleftarrow{h}_1, \cdots, \overleftarrow{h}_{T_x}) \end{matrix} \xrightarrow{\text{Concat}} \boldsymbol{h} = (h_1, \cdots, h_{T_x})$$

where $h_j = \left[\overrightarrow{h}_j^\top ; \overleftarrow{h}_j^\top\right]^\top$

Decoder - Alignment Model

- Compute alignment α_{ij}

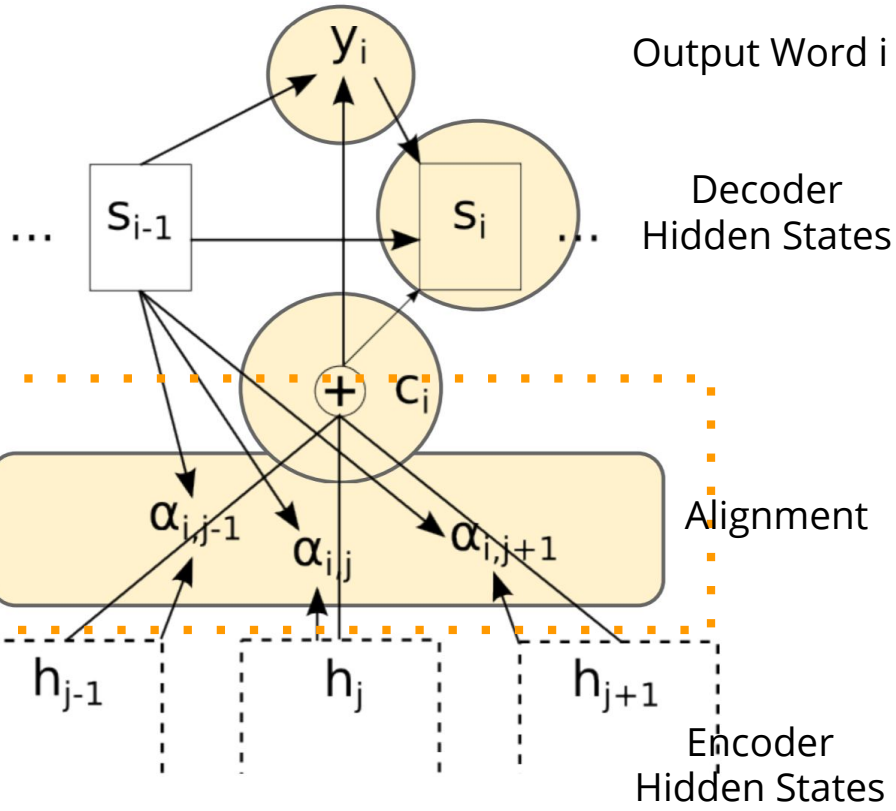
$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})}$$

How well the inputs around position j and the output at position i match

Use simple feedforward NN to compute e_{ij} based on s_{i-1} and h_j

$$e_{ij} = v^T \tanh(W s_{i-1} + V h_j)$$

where v, W, V are trainable weights



Decoder

- Compute context c_i

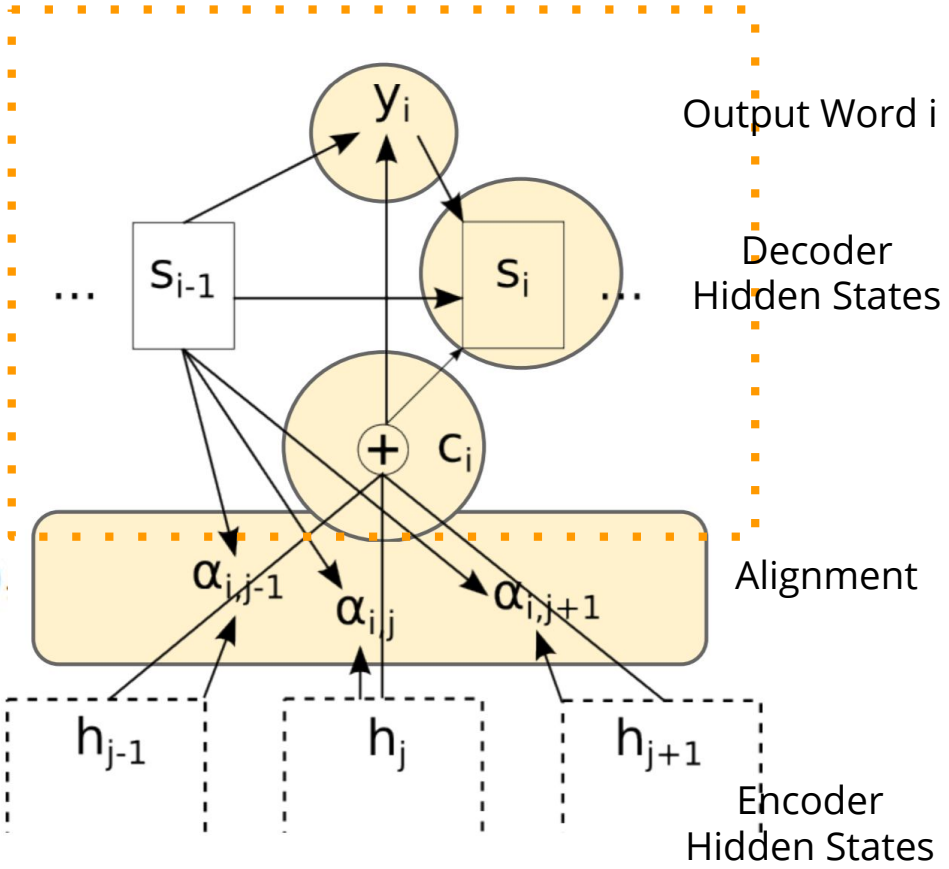
$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j$$

- Compute new decoder state s_i

$$s_i = f(s_{i-1}, y_{i-1}, c_i)$$

- Generate new output y_i

$$\text{argmax} \quad p(y_i | y_1, \ldots, y_{i-1}, \mathbf{x}) = g(y_{i-1}, s_i, c_i)$$



Various score functions

Effective Approaches to Attention-based Neural Machine Translation

Minh-Thang Luong **Hieu Pham** **Christopher D. Manning**
Computer Science Department, Stanford University, Stanford, CA 94305
{lmthang,hyhieu,manning}@stanford.edu

$$\text{score}(\boldsymbol{h}_t, \bar{\boldsymbol{h}}_s) = \begin{cases} \boldsymbol{h}_t^\top \bar{\boldsymbol{h}}_s & \textit{dot} \\ \boldsymbol{h}_t^\top \boldsymbol{W_a} \bar{\boldsymbol{h}}_s & \textit{general} \\ \boldsymbol{v}_a^\top \tanh\left(\boldsymbol{W_a}[\boldsymbol{h}_t; \bar{\boldsymbol{h}}_s]\right) & \textit{concat} \end{cases}$$

Attention Is All You Need

Ashish Vaswani[*]
Google Brain
avaswani@google.com

Noam Shazeer[*]
Google Brain
noam@google.com

Niki Parmar[*]
Google Research
nikip@google.com

Jakob Uszkoreit[*]
Google Research
usz@google.com

Llion Jones[*]
Google Research
llion@google.com

Aidan N. Gomez[*] [†]
University of Toronto
aidan@cs.toronto.edu

Łukasz Kaiser[*]
Google Brain
lukaszkaiser@google.com

Illia Polosukhin[*] [‡]
illia.polosukhin@gmail.com

Transformer Network