Neural Machine Translation

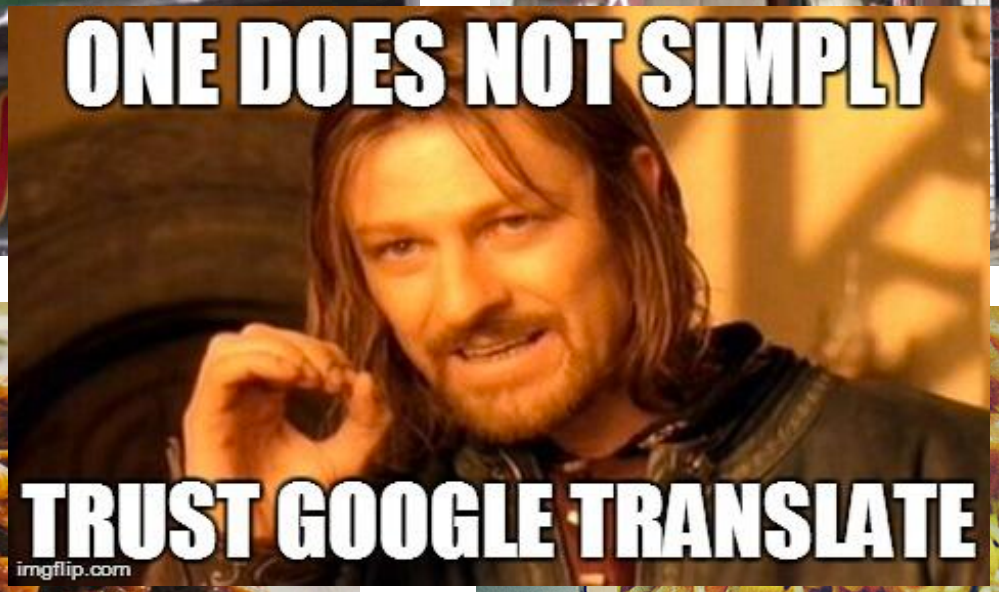
Thang Luong

Tutorial @ Stanford NLP lunch

(Thanks to Chris Manning, Abigail See, and Russell Stewart for comments and discussions.)

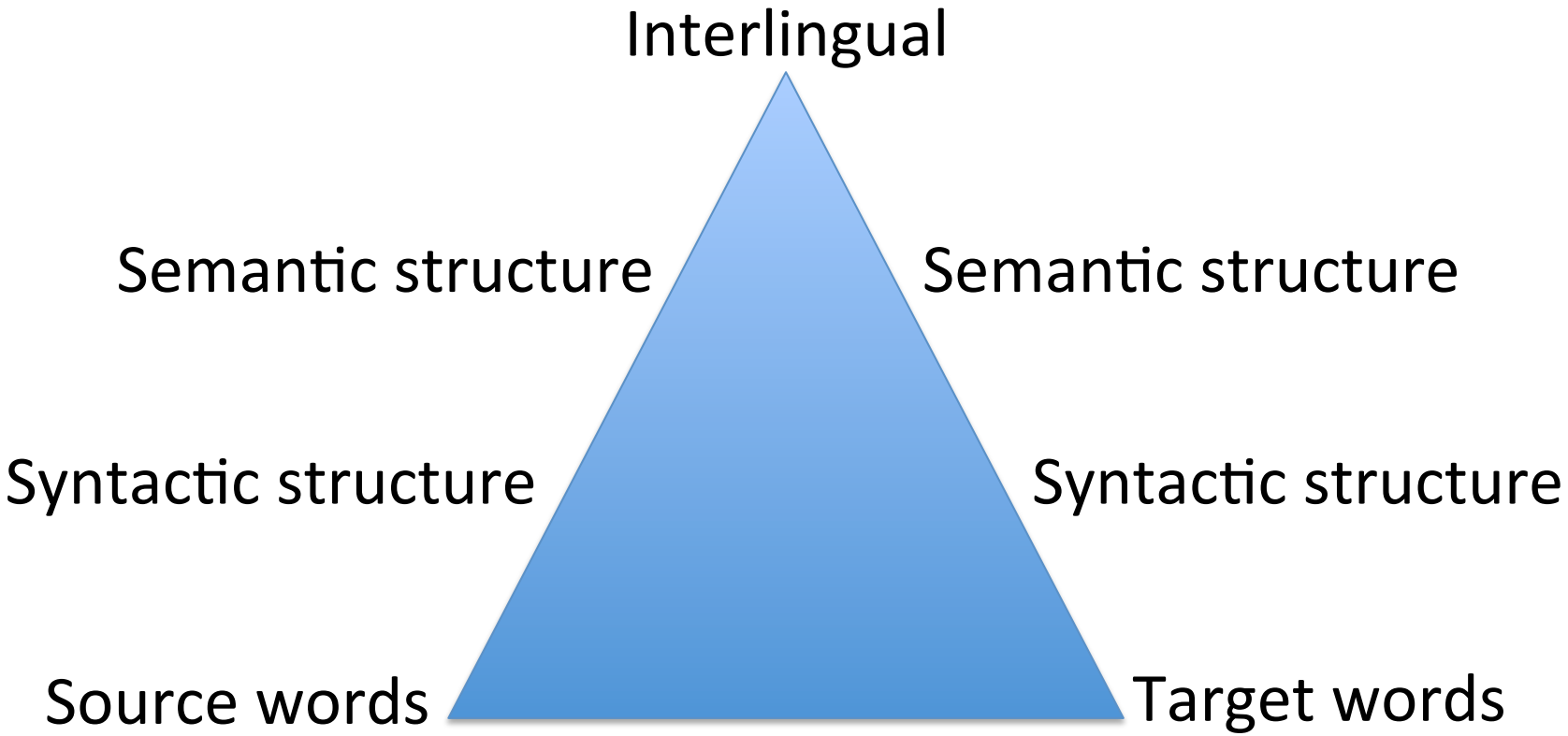
Sixi roasted husband

Meat Muscle Stupid Bean Sprouts



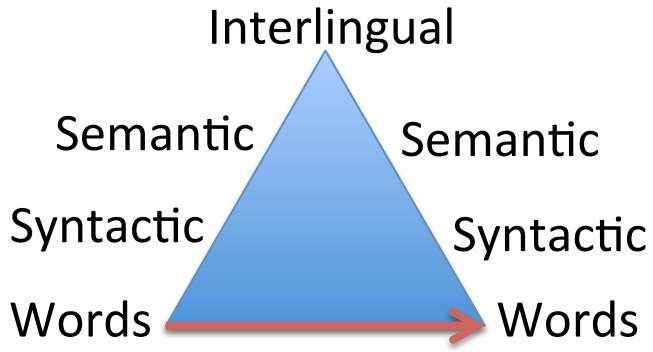
Sixi roasted husband

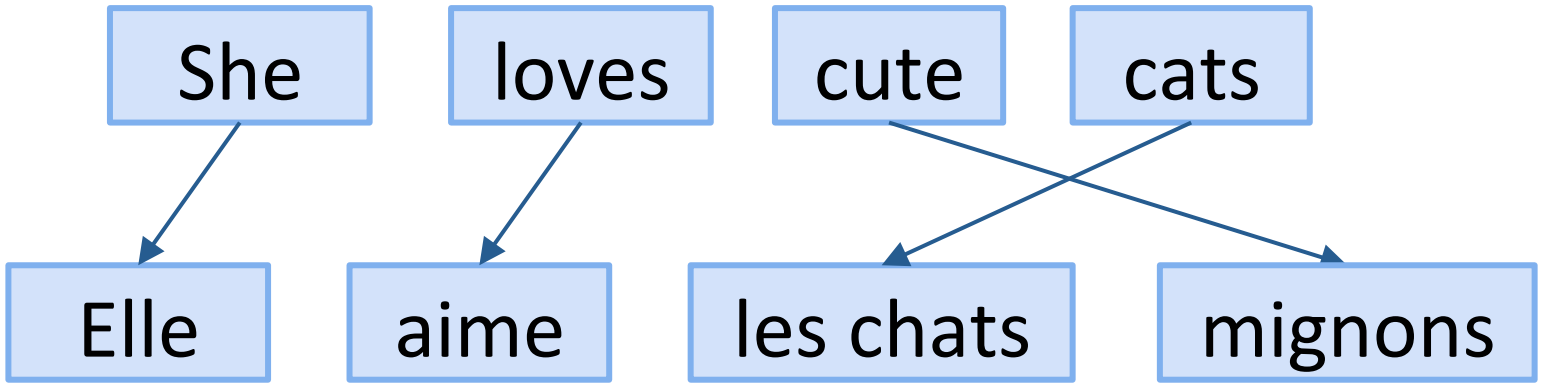
Meat Muscle Stupid Bean Sprouts



The Vauquois Diagram (1968)

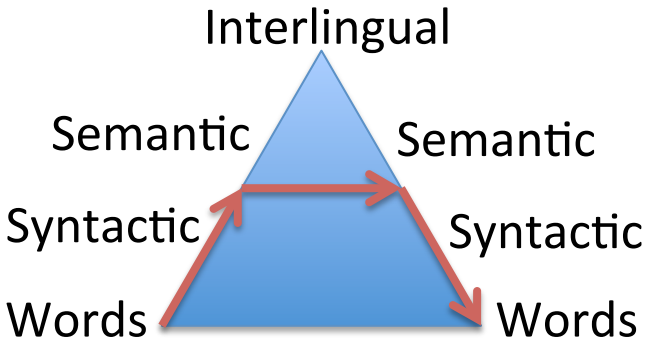
Direct approach

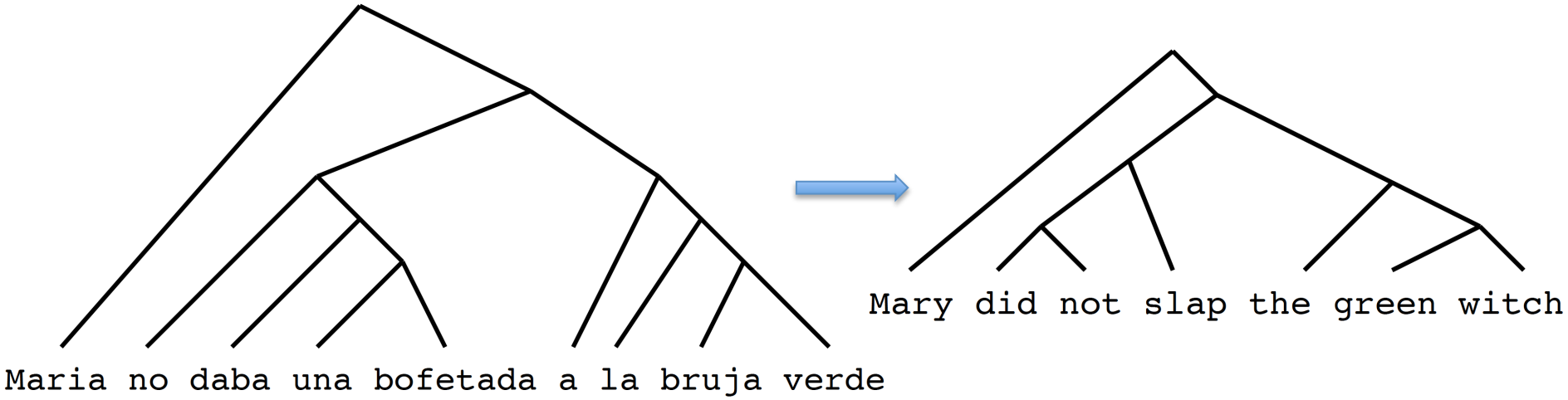




Translate locally!

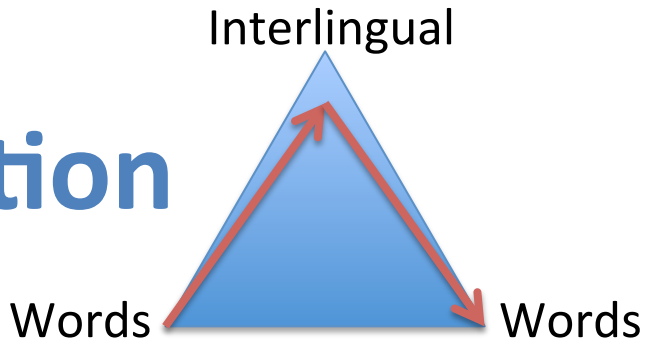
Transfer approach





Complex, slow, need lots of resources!

Neural Machine Translation



- Simple
  - Skip intermediate structures.
- Generalization
  - With distributed representations.

Outline

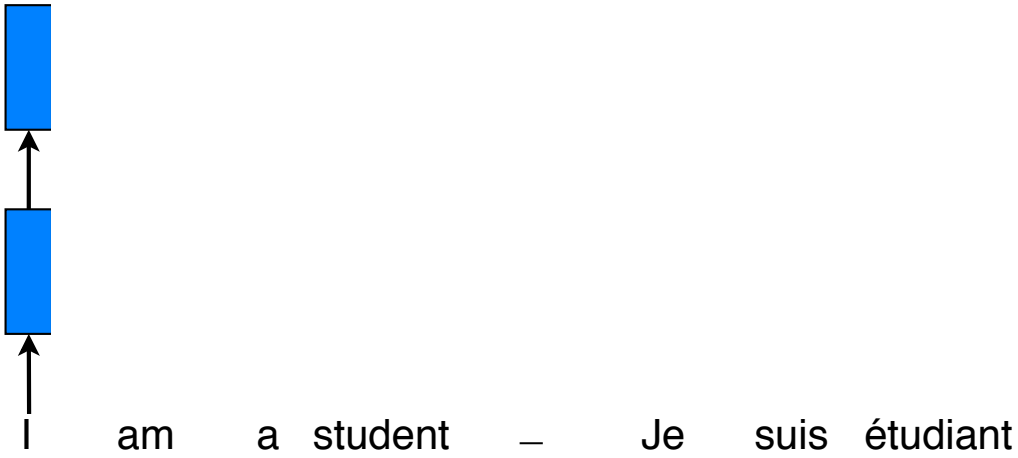
- NMT basics (Sutskever et al., 2014)
  - Architecture.
  - Recurrent units.
  - Backpropagation.
- Attention mechanism (Bahdanau et al., 2015)

Neural Machine Translation (NMT)

I am a student _ Je suis étudiant

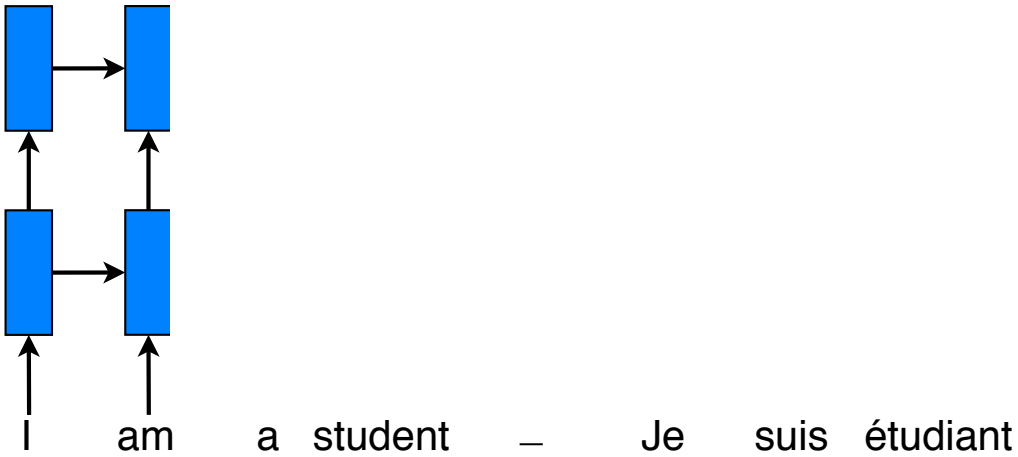
- Big RNNs trained end-to-end.

Neural Machine Translation (NMT)



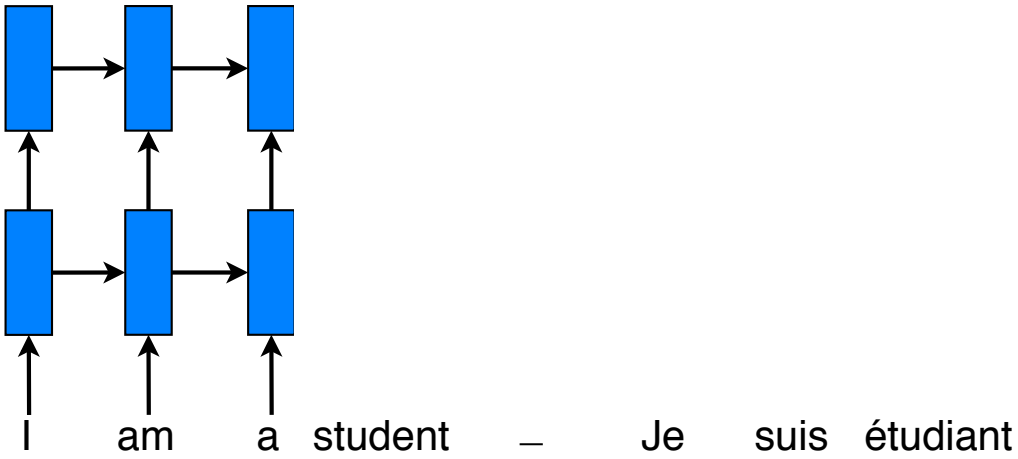
- Big RNNs trained end-to-end.

Neural Machine Translation (NMT)



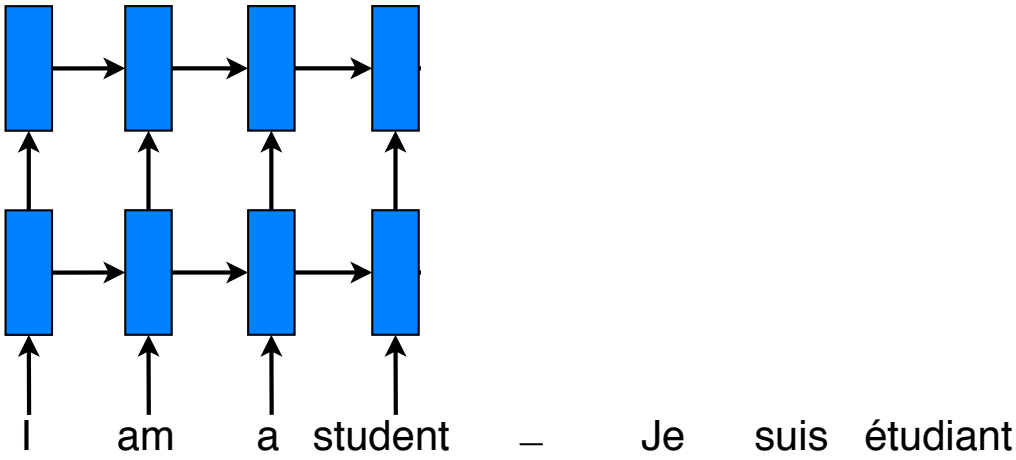
- Big RNNs trained end-to-end.

Neural Machine Translation (NMT)



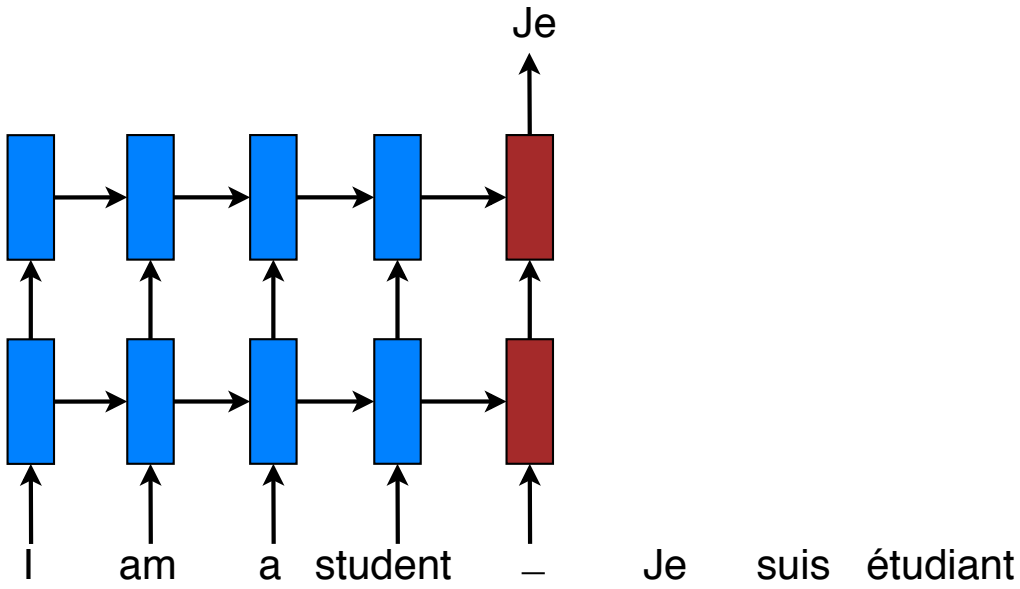
- Big RNNs trained end-to-end.

Neural Machine Translation (NMT)



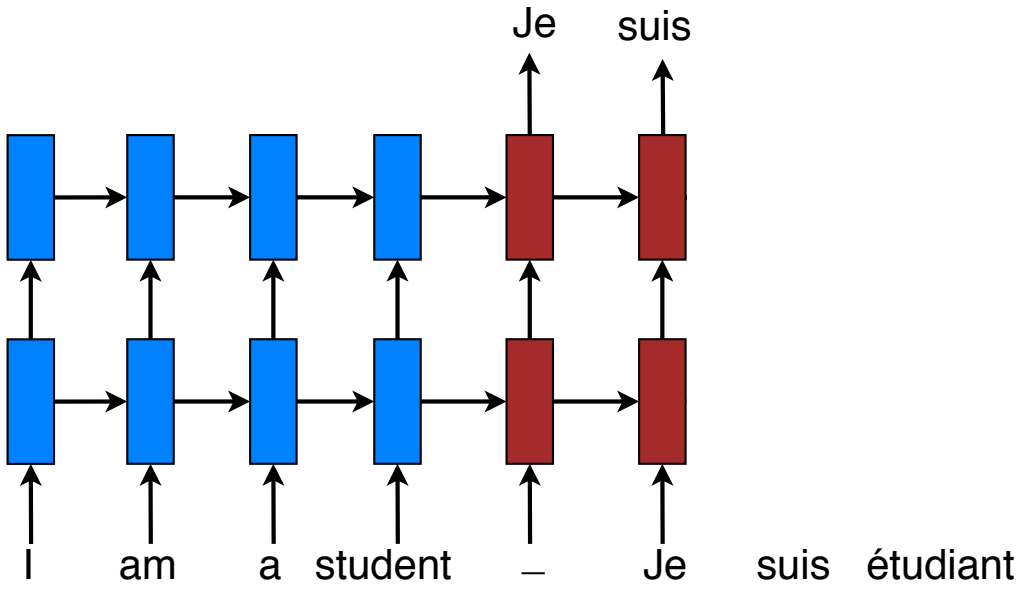
- Big RNNs trained end-to-end.

Neural Machine Translation (NMT)



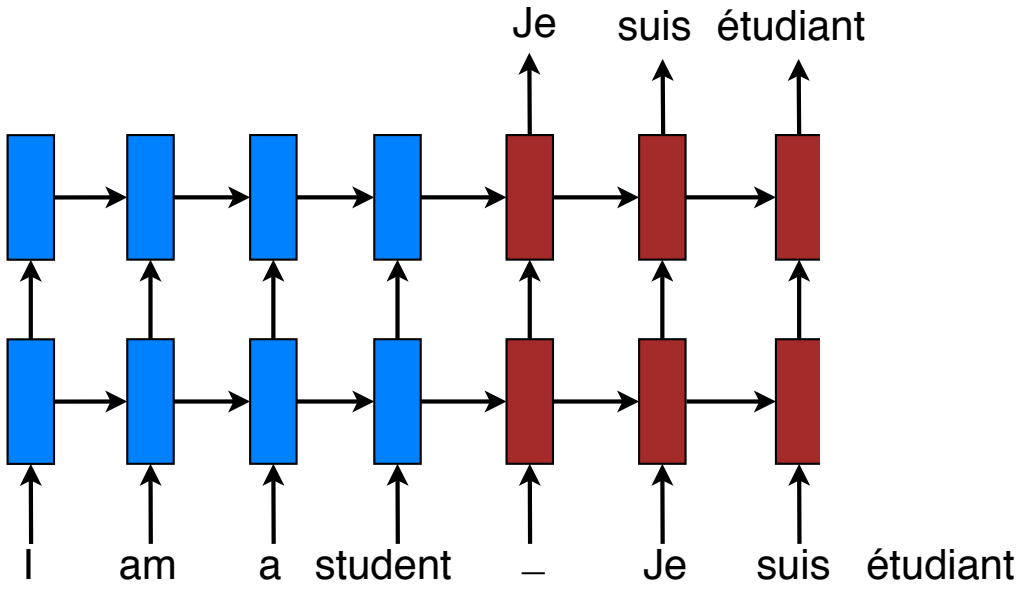
- Big RNNs trained end-to-end: encoder-decoder.

Neural Machine Translation (NMT)



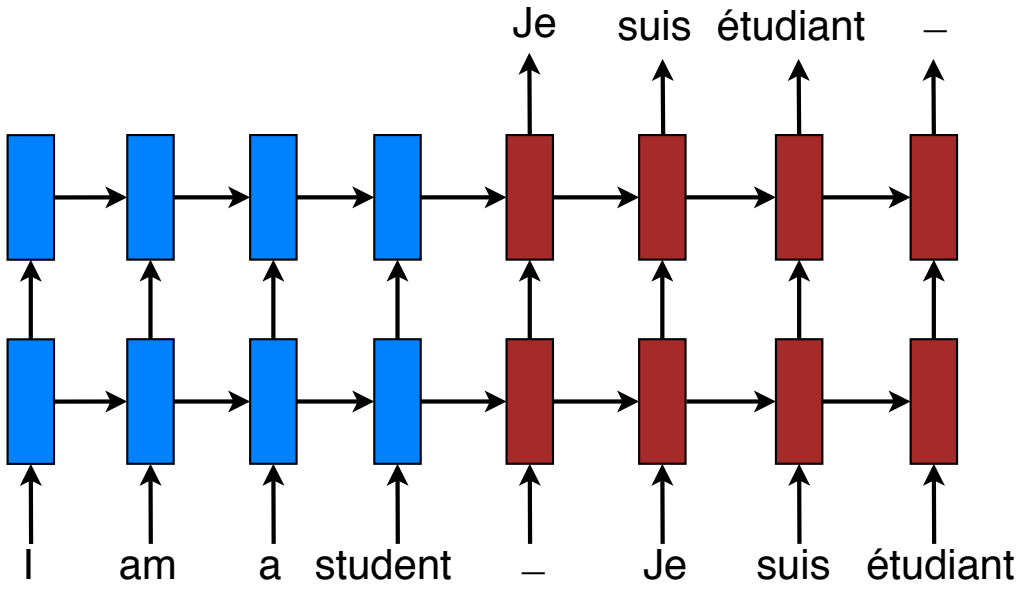
- Big RNNs trained end-to-end: encoder-decoder.

Neural Machine Translation (NMT)



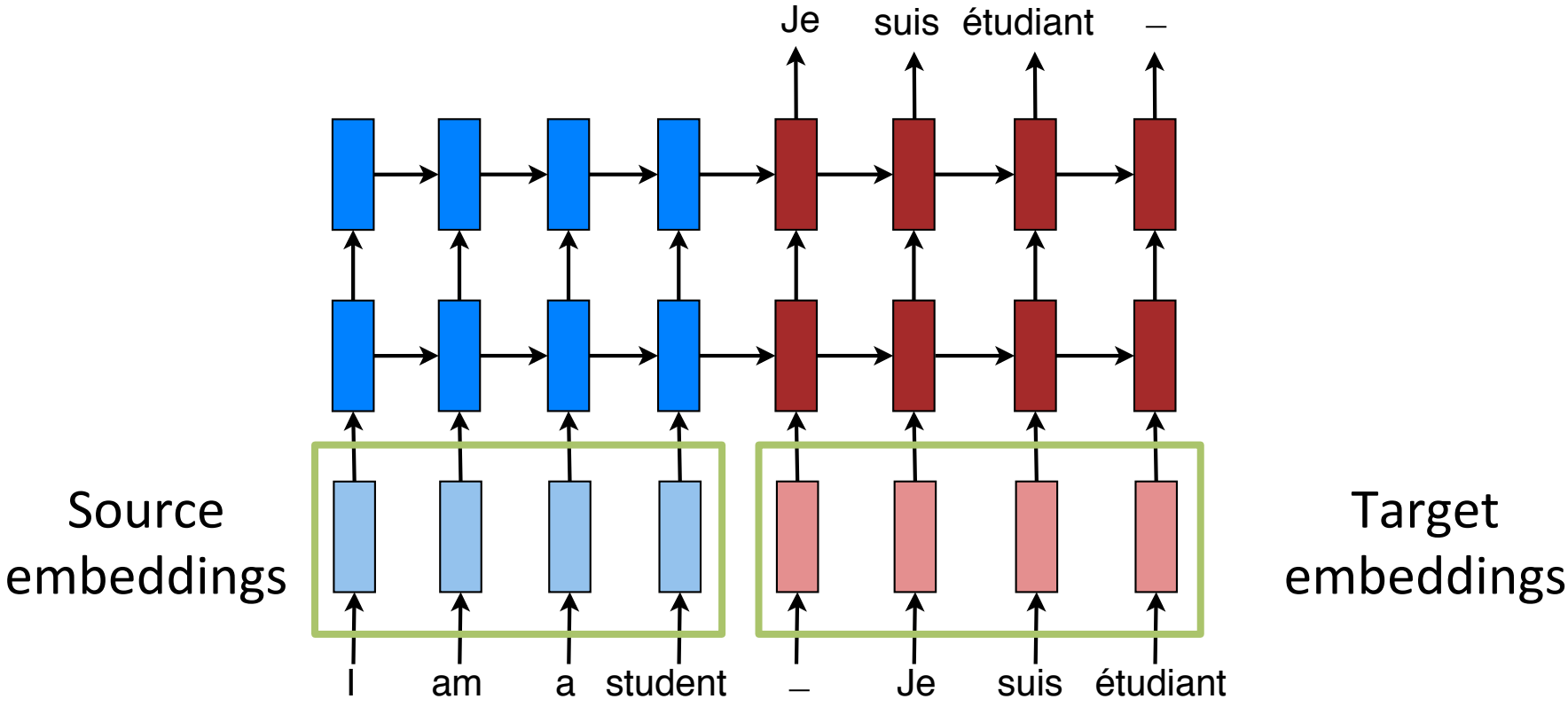
- Big RNNs trained end-to-end: encoder-decoder.

Neural Machine Translation (NMT)



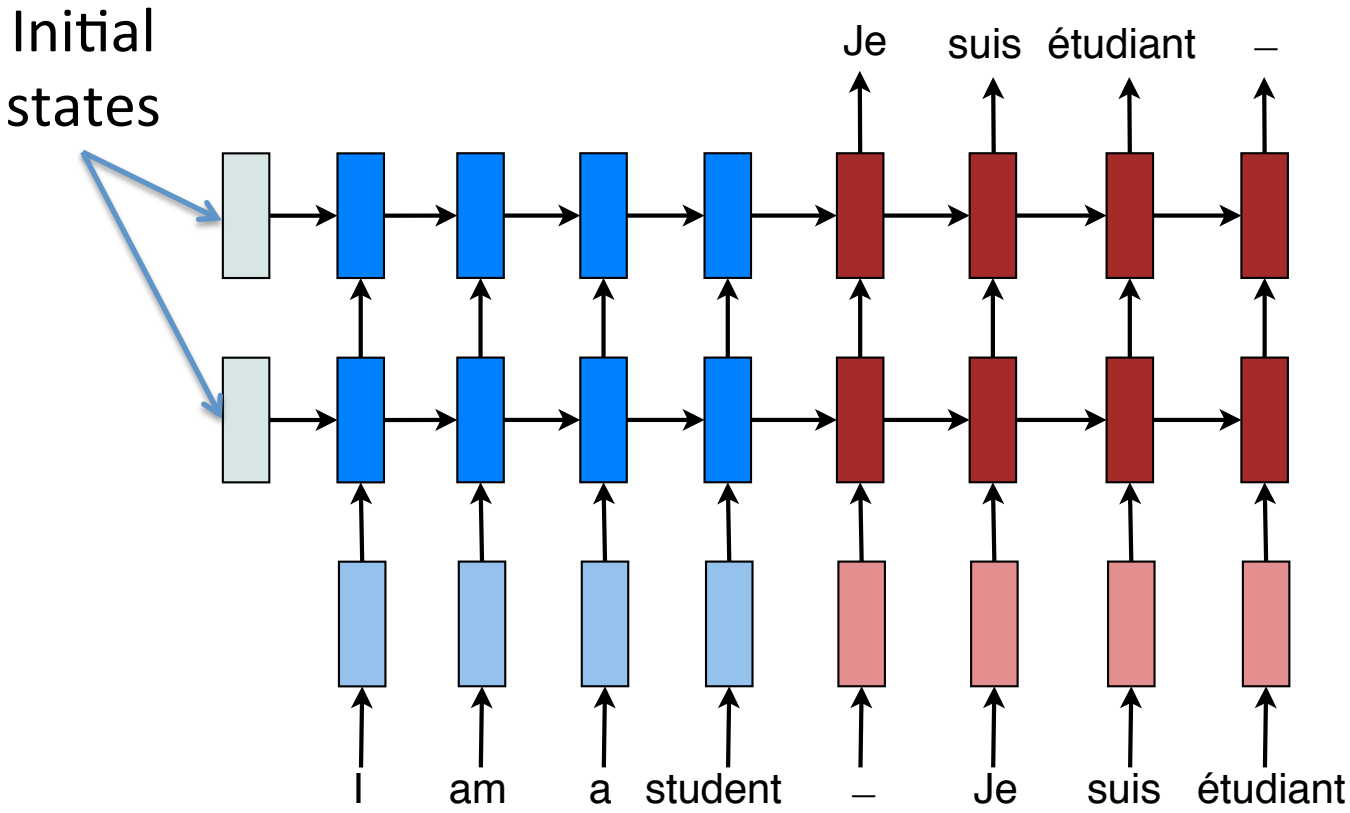
- Big RNNs trained end-to-end: encoder-decoder.

Word Embeddings



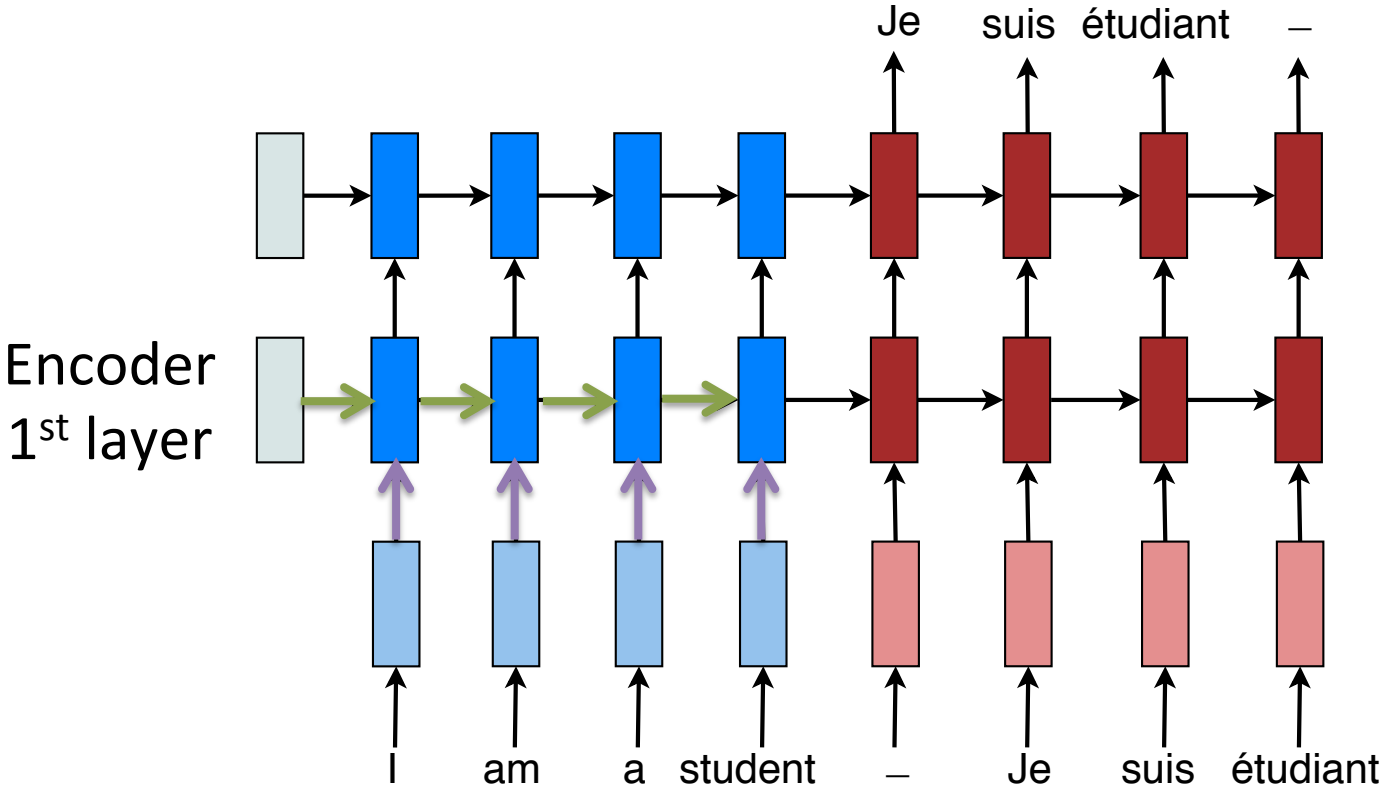
- Randomly initialized, one for each language.

Recurrent Connections



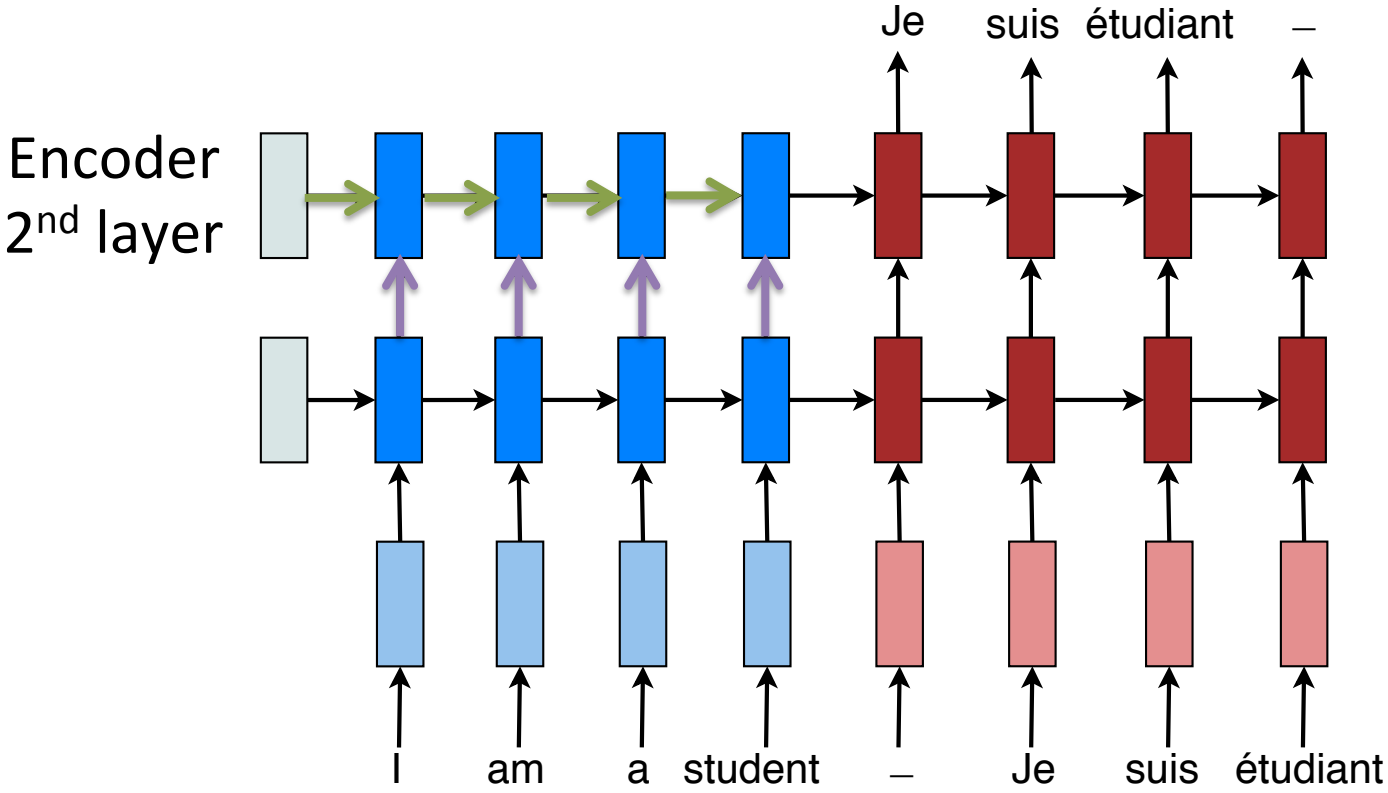
- Often set to 0.

Recurrent Connections



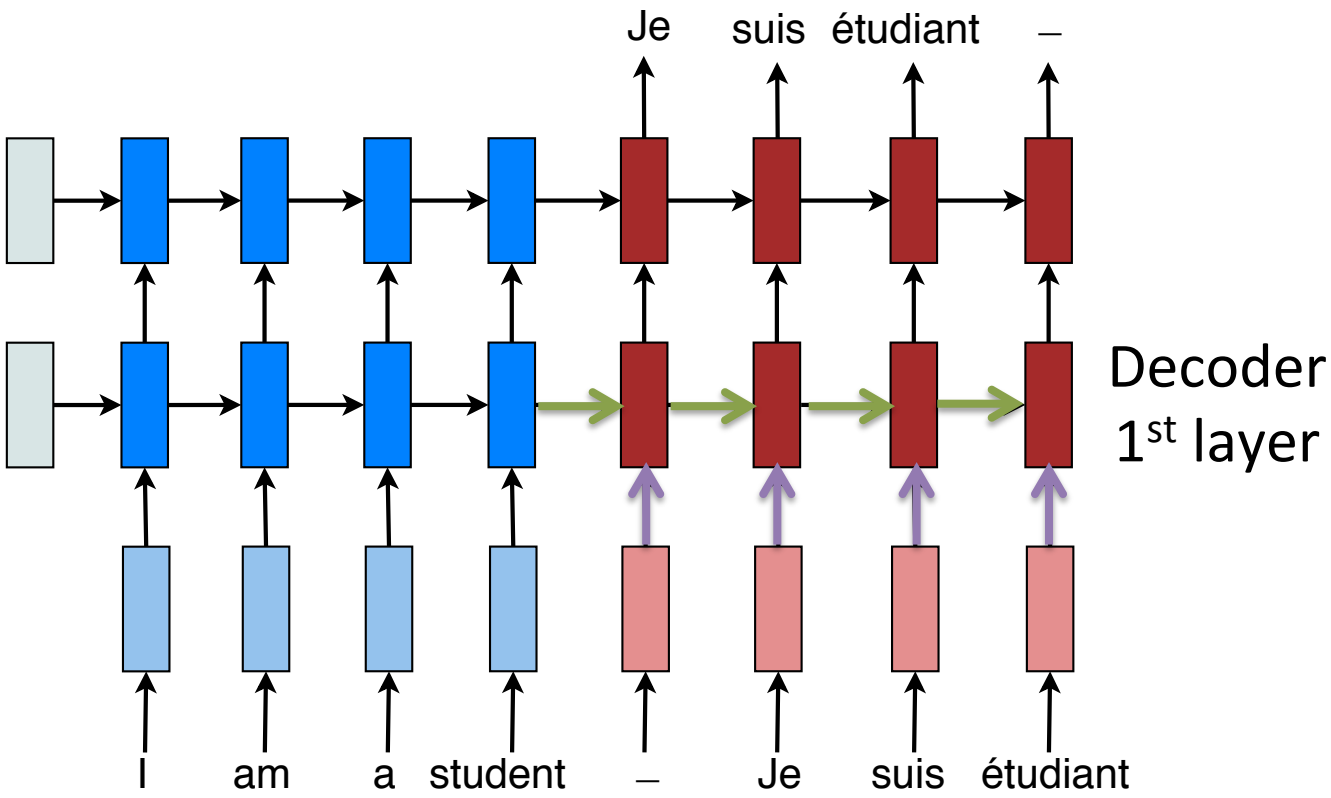
- Different across layers and encoder / decoder.

Recurrent Connections



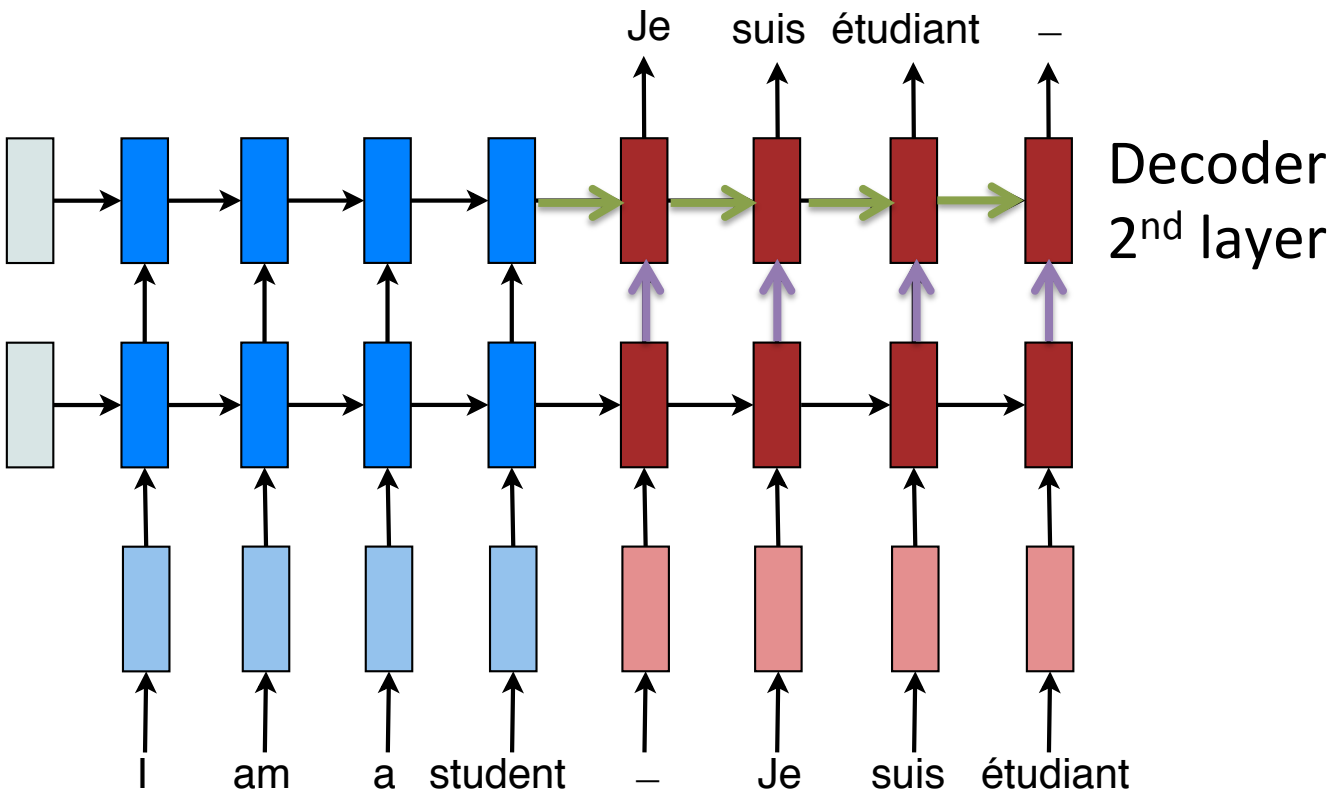
- Different across layers and encoder / decoder.

Recurrent Connections



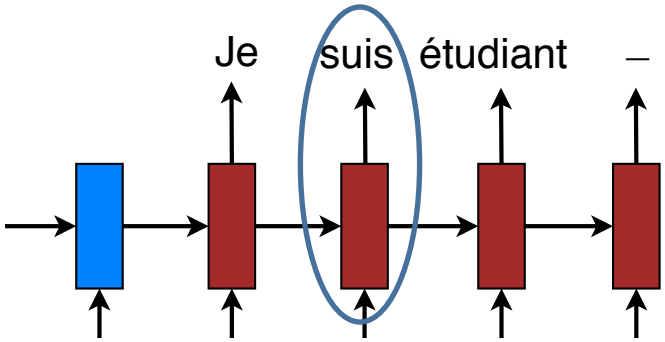
- Different across layers and encoder / decoder.

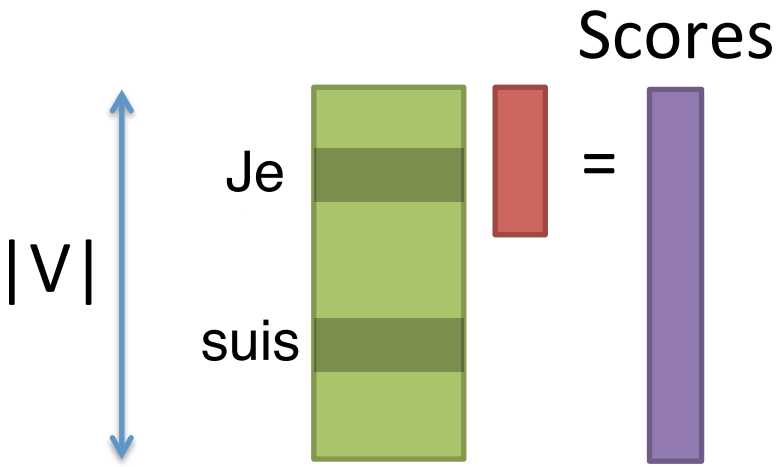
Recurrent Connections



- Different across layers and encoder / decoder.

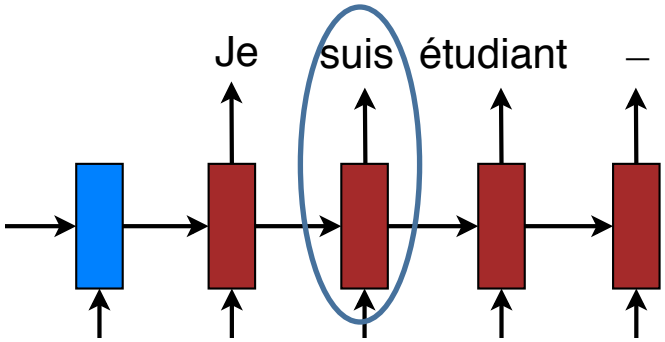
Softmax Layer

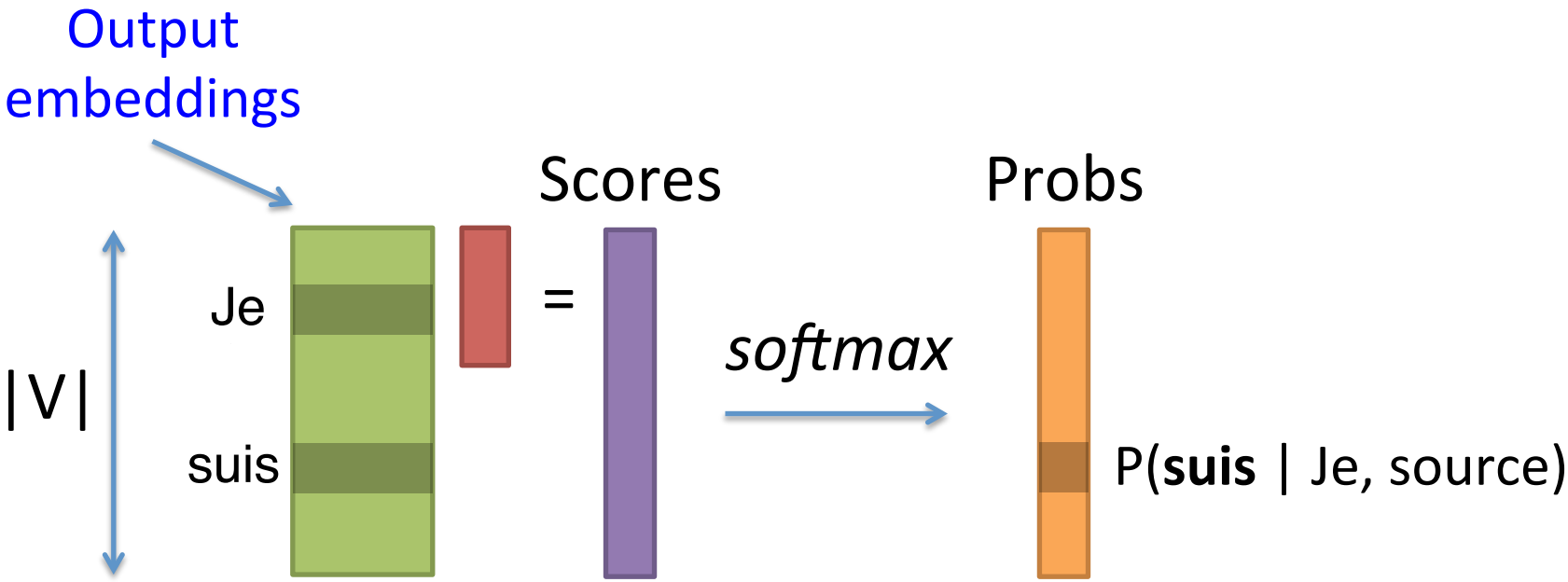




- Hidden states ↦ scores.

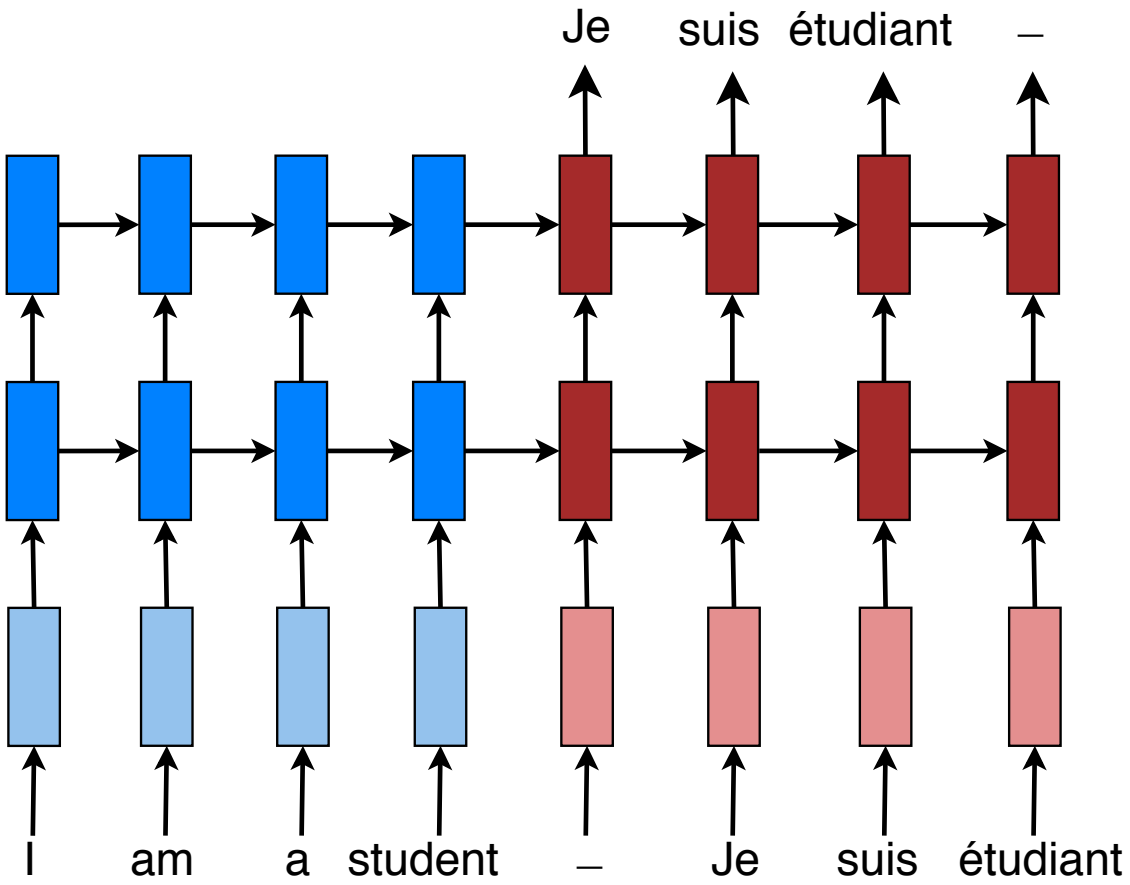
Softmax Layer



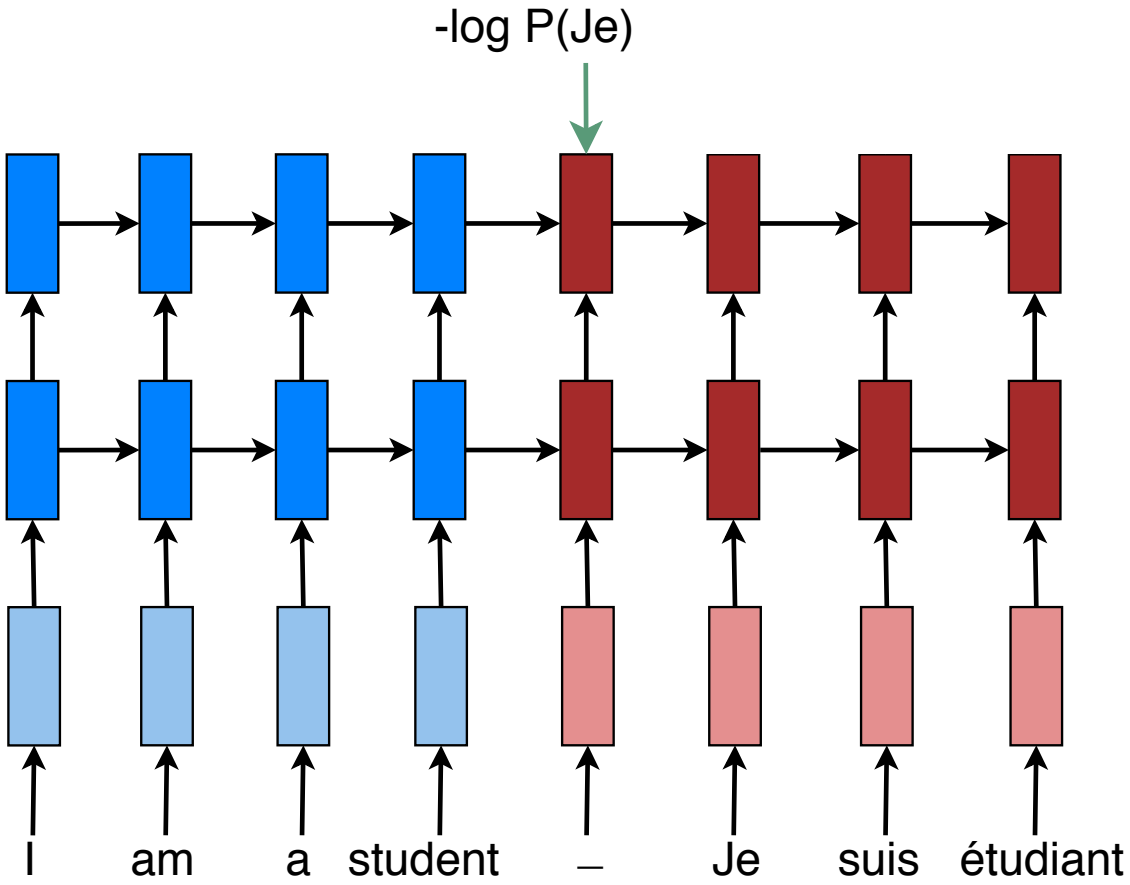


- Scores ↦ probabilities.

Training Loss

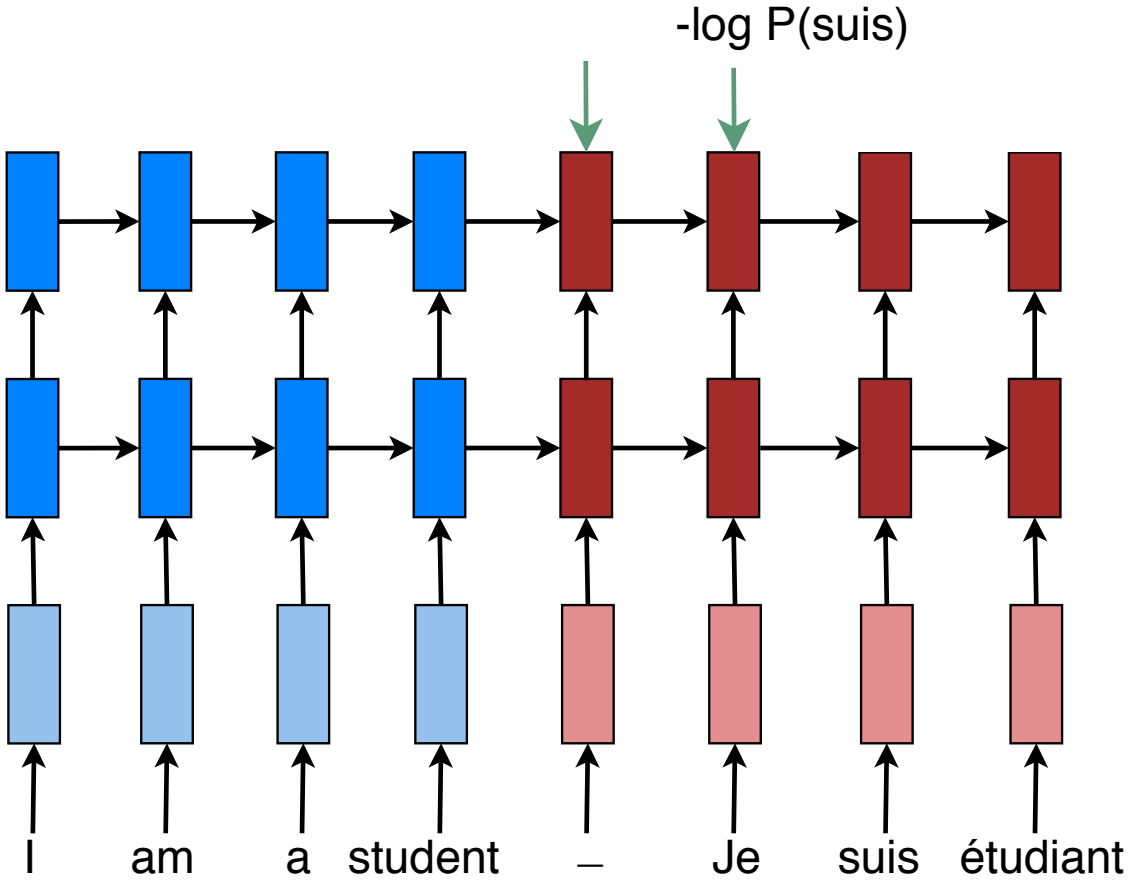


Training Loss



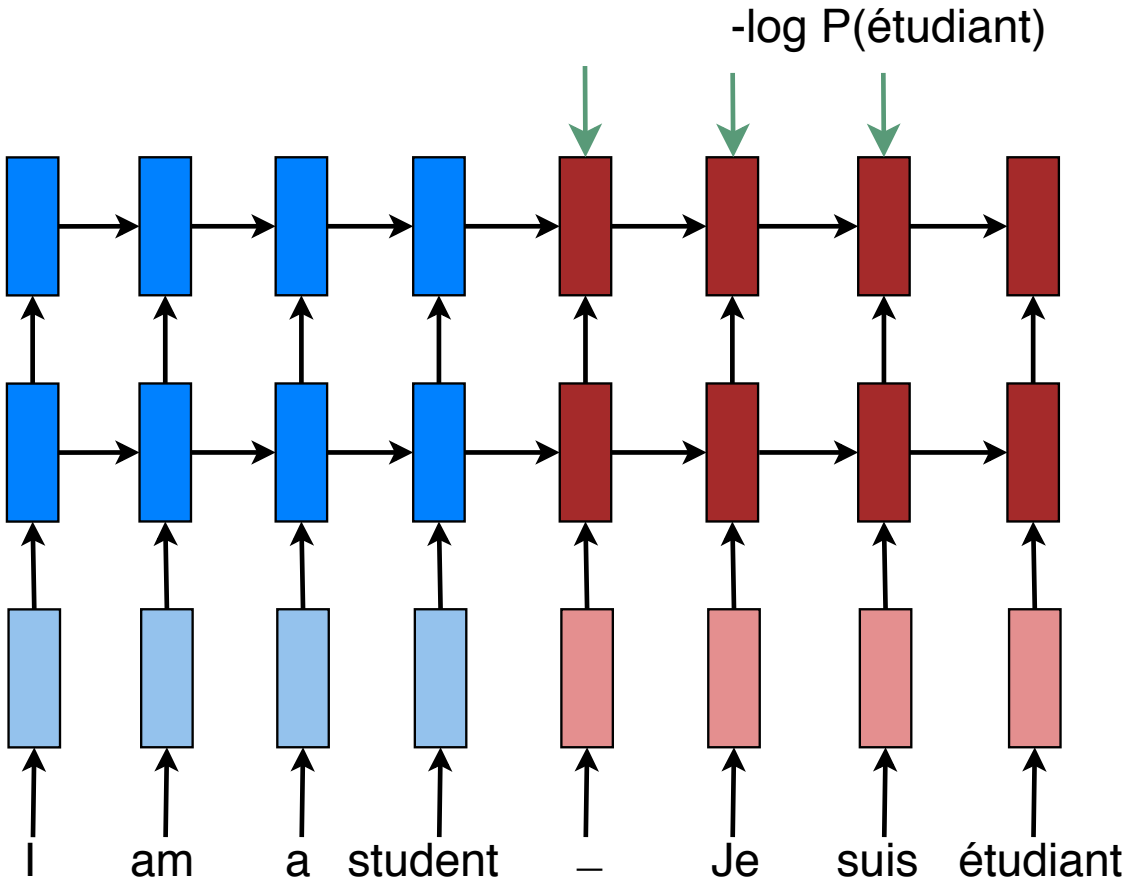
- Sum of all individual losses

Training Loss



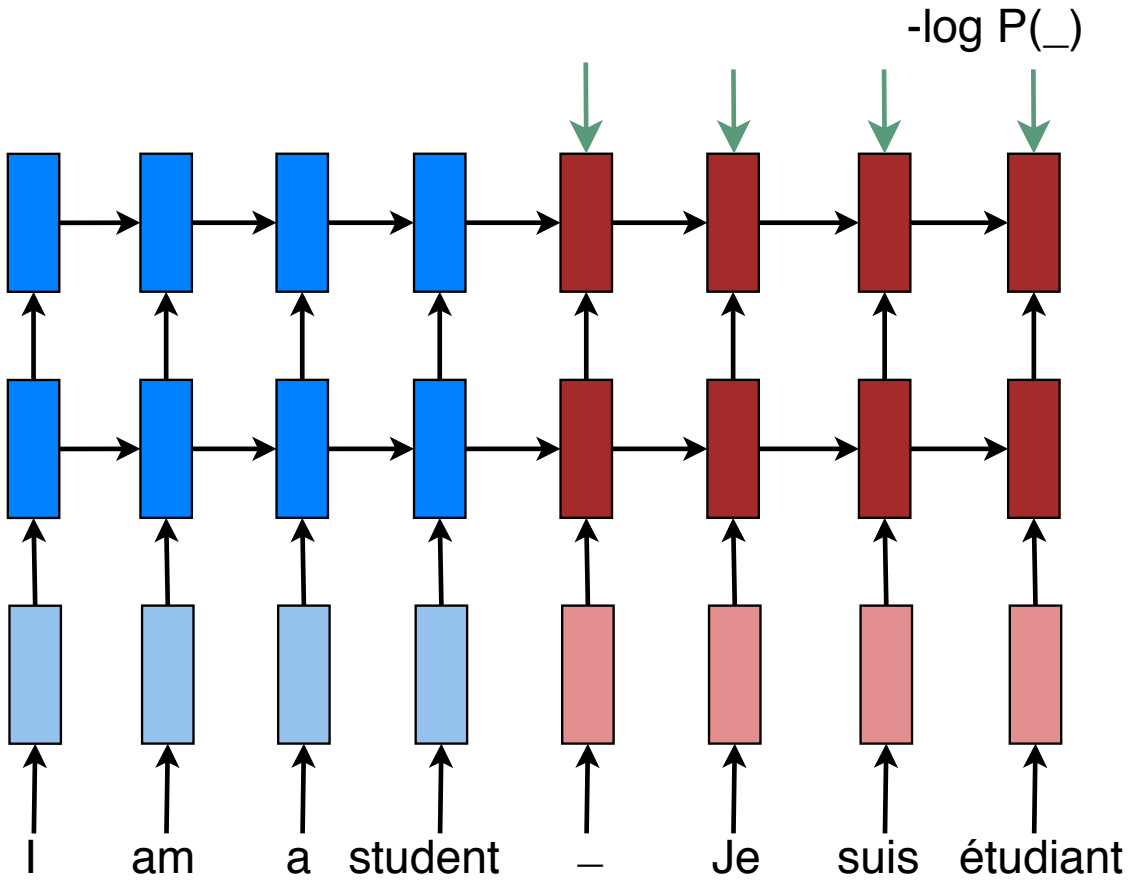
- Sum of all individual losses

Training Loss



- Sum of all individual losses

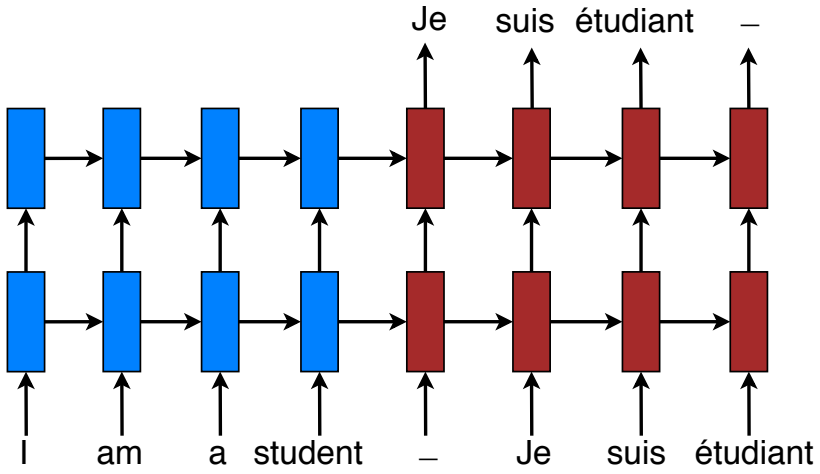
Training Loss



- Sum of all individual losses

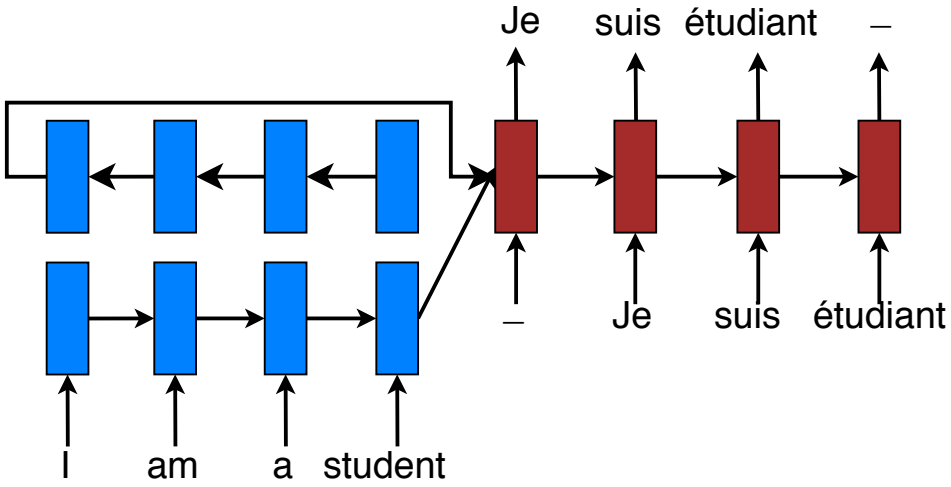
Deep RNNs

(Sutskever et al., 2014)



Bidirectional RNNs

(Bahdanau et al., 2015)



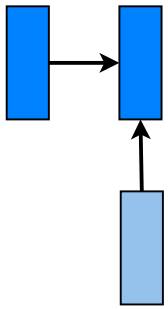
Summary

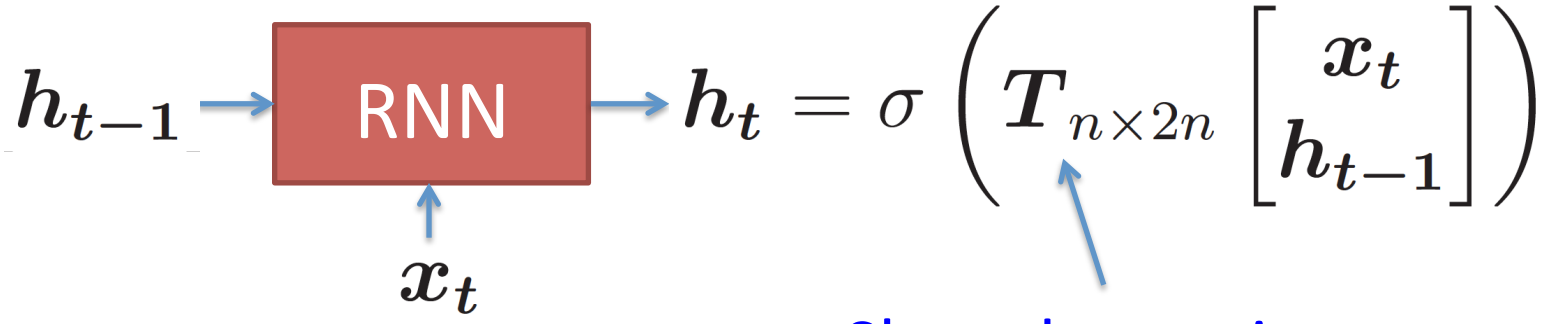
- Generalize well.
- Small memory.
- Simple decoder.

Outline

- NMT basics (Sutskever et al., 2014)
  - Architecture.
  - Recurrent units.
  - Backpropagation.
- Attention mechanism (Bahdanau et al., 2015)

Recurrent types – vanilla RNN





Vanishing gradient problem!

Vanishing gradients

$$\boldsymbol{h}_t = \sigma\left(\boldsymbol{W}_{\boldsymbol{xh}}\boldsymbol{x}_t + \boldsymbol{W}_{\boldsymbol{hh}}\boldsymbol{h}_{t-1}\right)$$

$$\frac{\partial \boldsymbol{h}_t}{\partial \boldsymbol{h}_{t-1}} = \operatorname{diag}\left(\sigma'(\ldots)\right) \boldsymbol{W}_{\boldsymbol{hh}}{}^{\top}$$

Chain Rule

$$\left\| \frac{\partial \boldsymbol{h}_t}{\partial \boldsymbol{h}_{t-1}} \right\| \leq \gamma \left\| \boldsymbol{W}_{\boldsymbol{hh}}{}^{\top} \right\| \leq \gamma \lambda_1$$

Bounding Rules

Bound $\left\|\operatorname{diag}\left(\sigma'(\ldots)\right)\right\|$ Largest singular value $\boldsymbol{W}_{\boldsymbol{hh}}{}^{\top}$

(Pascanu et al., 2013)

Vanishing gradients

$$\boldsymbol{h}_t = \sigma\left(\boldsymbol{W}_{\boldsymbol{xh}}\boldsymbol{x}_t + \boldsymbol{W}_{\boldsymbol{hh}}\boldsymbol{h}_{t-1}\right)$$

$$\frac{\partial \boldsymbol{h}_t}{\partial \boldsymbol{h}_{t-1}} = \operatorname{diag}\left(\sigma'(\ldots)\right) \boldsymbol{W}_{\boldsymbol{hh}}{}^{\top}$$

Chain Rule

$$\left\| \frac{\partial \boldsymbol{h}_t}{\partial \boldsymbol{h}_{t-1}} \right\| \leq \gamma \left\| \boldsymbol{W}_{\boldsymbol{hh}}{}^{\top} \right\| \leq \gamma \lambda_1$$

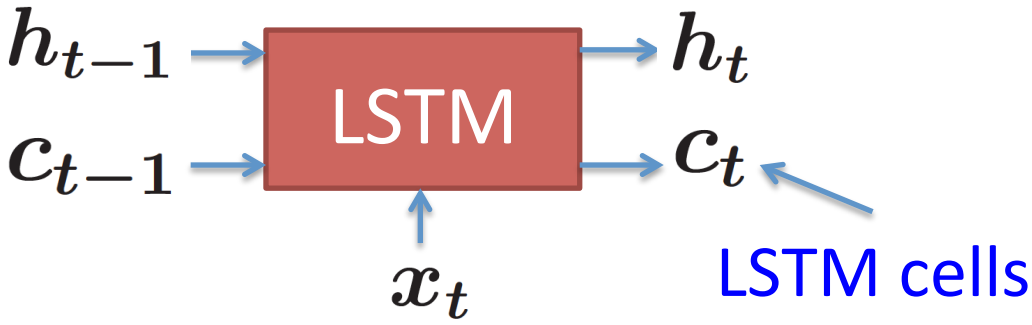
Bounding Rules

$$\left\| \frac{\partial \boldsymbol{h}_t}{\partial \boldsymbol{h}_{t-k}} \right\| \leq (\gamma \lambda_1)^k \rightarrow 0 \text{ if } \lambda_1 < \frac{1}{\gamma}$$

Sufficient Cond

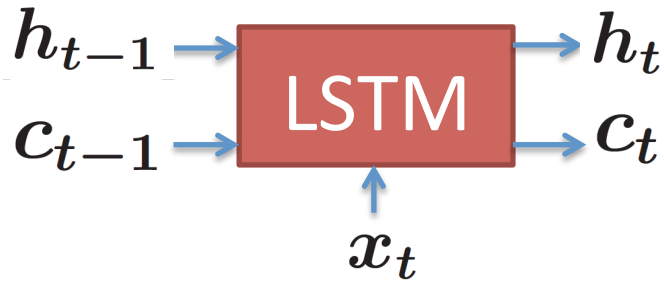
(Pascanu et al., 2013)

Recurrent types – LSTM



- Long-Short Term Memory (LSTM)
  - (Hochreiter & Schmidhuber, 1997)
- LSTM cells are additively updated
  - Make backprop through time easier.

Building LSTM



$$\begin{pmatrix} \\ \\ \hat{h}_t \end{pmatrix} = \begin{pmatrix} \\ \\ \tanh \end{pmatrix} \underset{n \times 2n}{T} \begin{bmatrix} x_t \\ h_{t-1} \end{bmatrix}$$

$$c_t = c_{t-1} + \hat{h}_t$$

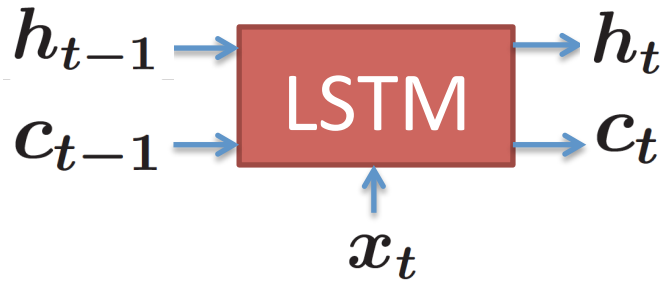
$$h_t = c_t$$

Nice gradients!

$$\frac{\partial c_t}{\partial c_{t-1}} = I$$

- A naïve version.

Building LSTM



Input gates

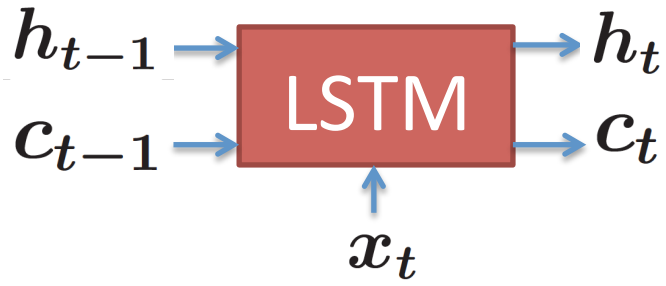
$$\begin{pmatrix} i_t \\ \\ \hat{h}_t \end{pmatrix} = \begin{pmatrix} \text{sigm} \\ \\ \tanh \end{pmatrix} T_{n \times 2n} \begin{bmatrix} x_t \\ h_{t-1} \end{bmatrix}$$

$$c_t = c_{t-1} + i_t \circ \hat{h}_t$$

$$h_t = c_t$$

- Add input gates: control input signal.

Building LSTM



Forget gates

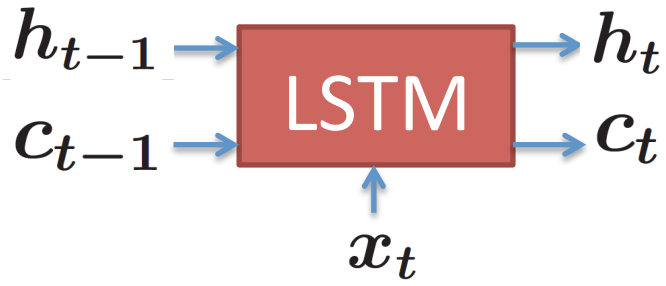
$$\begin{pmatrix} i_t \\ f_t \\ \\ \hat{h}_t \end{pmatrix} = \begin{pmatrix} \text{sigm} \\ \text{sigm} \\ \\ \tanh \end{pmatrix} T_{n \times 2n} \begin{bmatrix} x_t \\ h_{t-1} \end{bmatrix}$$

$$c_t = f_t \circ c_{t-1} + i_t \circ \hat{h}_t$$

$$h_t = c_t$$

- Add forget gates: control memory.

Building LSTM



Output gates

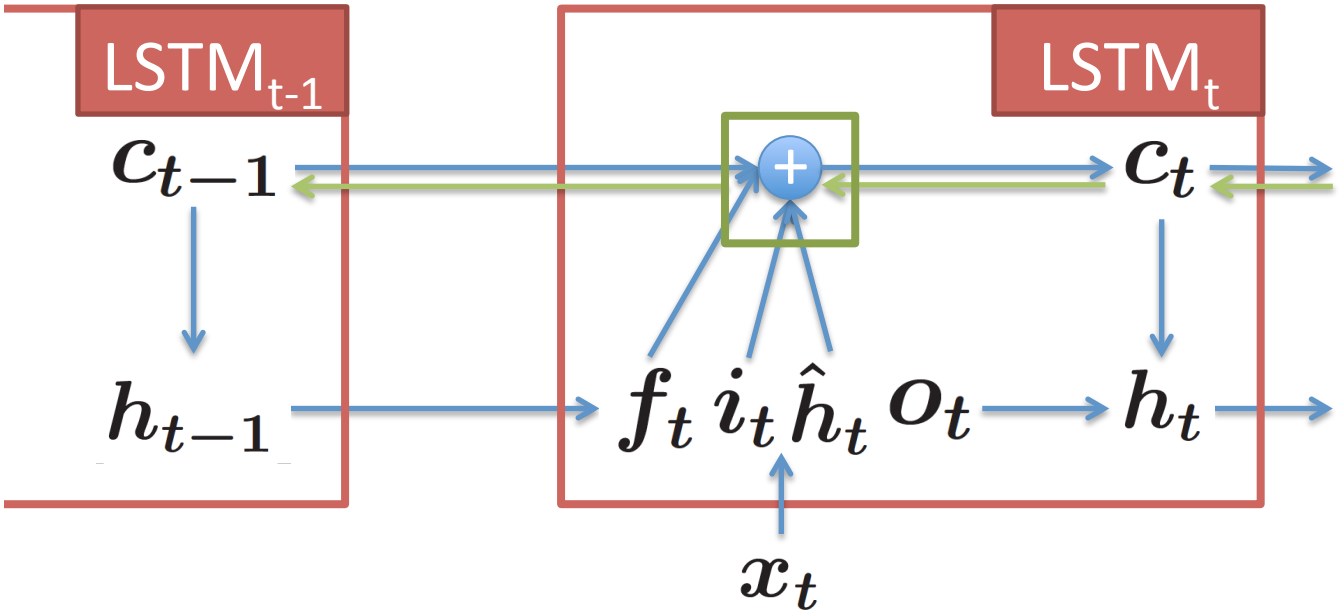
$$\begin{pmatrix} i_t \\ f_t \\ o_t \\ \hat{h}_t \end{pmatrix} = \begin{pmatrix} \text{sigm} \\ \text{sigm} \\ \text{sigm} \\ \tanh \end{pmatrix} T_{4n \times 2n} \begin{bmatrix} x_t \\ h_{t-1} \end{bmatrix}$$

$$c_t = f_t \circ c_{t-1} + i_t \circ \hat{h}_t$$

$$h_t = \boxed{o_t} \circ \tanh(c_t)$$

- Add output gates: extract information.
- (Zaremba et al., 2014).

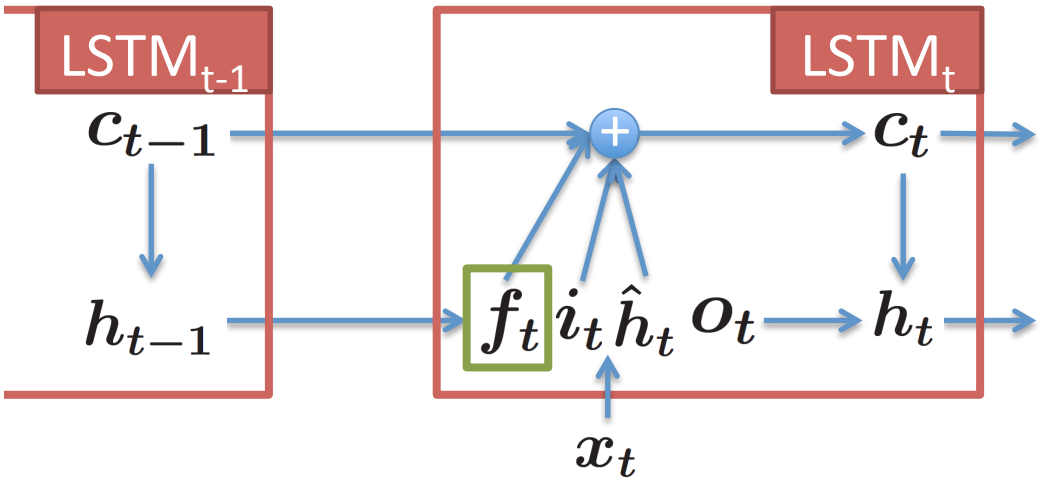
Why LSTM works?



- The additive operation is the key!
- Backpropation path through the cell is effective.

But forget gates can also be 0?

Important LSTM components?



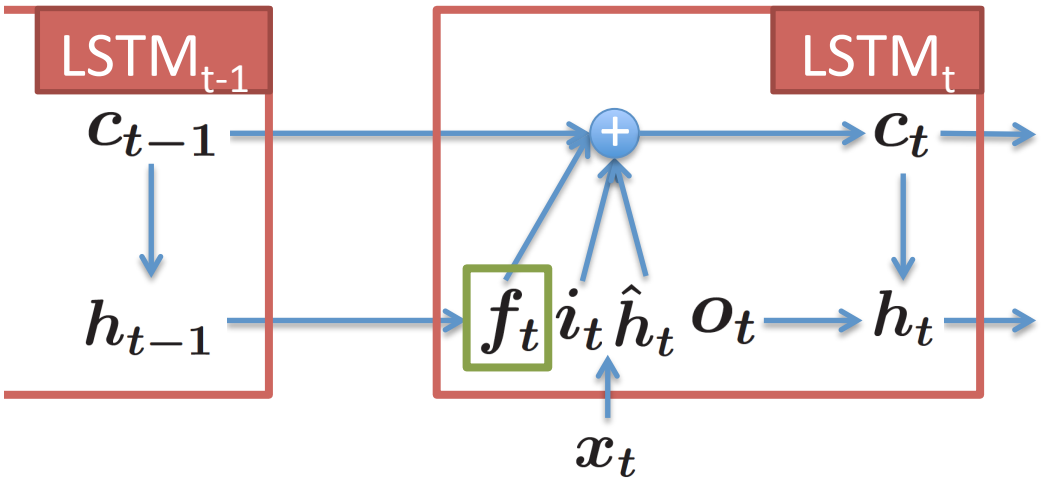
- (Jozefowicz et al., 2015): forget gate bias of 1.

$$c_t = f_t \circ c_{t-1} + i_t \circ \hat{h}_t$$

- (Greff et al., 2015): forget gates & output acts.

$$h_t = o_t \circ \tanh(c_t)$$

Important LSTM components?



- (Jozefowicz et al., 2015): forget gate bias of 1.

$$c_t = f_t \circ c_{t-1} + i_t \circ \hat{h}_t$$

- (Greff et al., 2015): forget gates & output acts.

Forget gates are important!

Other RNN units

- (Graves, 2013): revived LSTM.
  - Direct connections between cells and gates.
- Gated Recurrent Unit (GRU) – (Cho et al., 2014a)
  - No cells, same additive idea.
- LSTM vs. GRU: mixed results (Chung et al., 2015).

Encoder-decoder Summary

	Encoder	Decoder
(Sutskever et al., 2014) (Luong et al., 2015a) (Luong et al., 2015b)	Deep LSTM	Deep LSTM

Encoder-decoder Summary

	Encoder	Decoder
(Sutskever et al., 2014) (Luong et al., 2015a) (Luong et al., 2015b)	Deep LSTM	Deep LSTM
(Cho et al., 2014a) (Bahdanau et al., 2015) (Jean et al., 2015)	(Bidirectional) GRU	GRU

Encoder-decoder Summary

	Encoder	Decoder
(Sutskever et al., 2014) (Luong et al., 2015a) (Luong et al., 2015b)	Deep LSTM	Deep LSTM
(Cho et al., 2014a) (Bahdanau et al., 2015) (Jean et al., 2015)	(Bidirectional) GRU	GRU
(Kalchbrenner & Blunsom, 2013)	CNN	(Inverse CNN) RNN

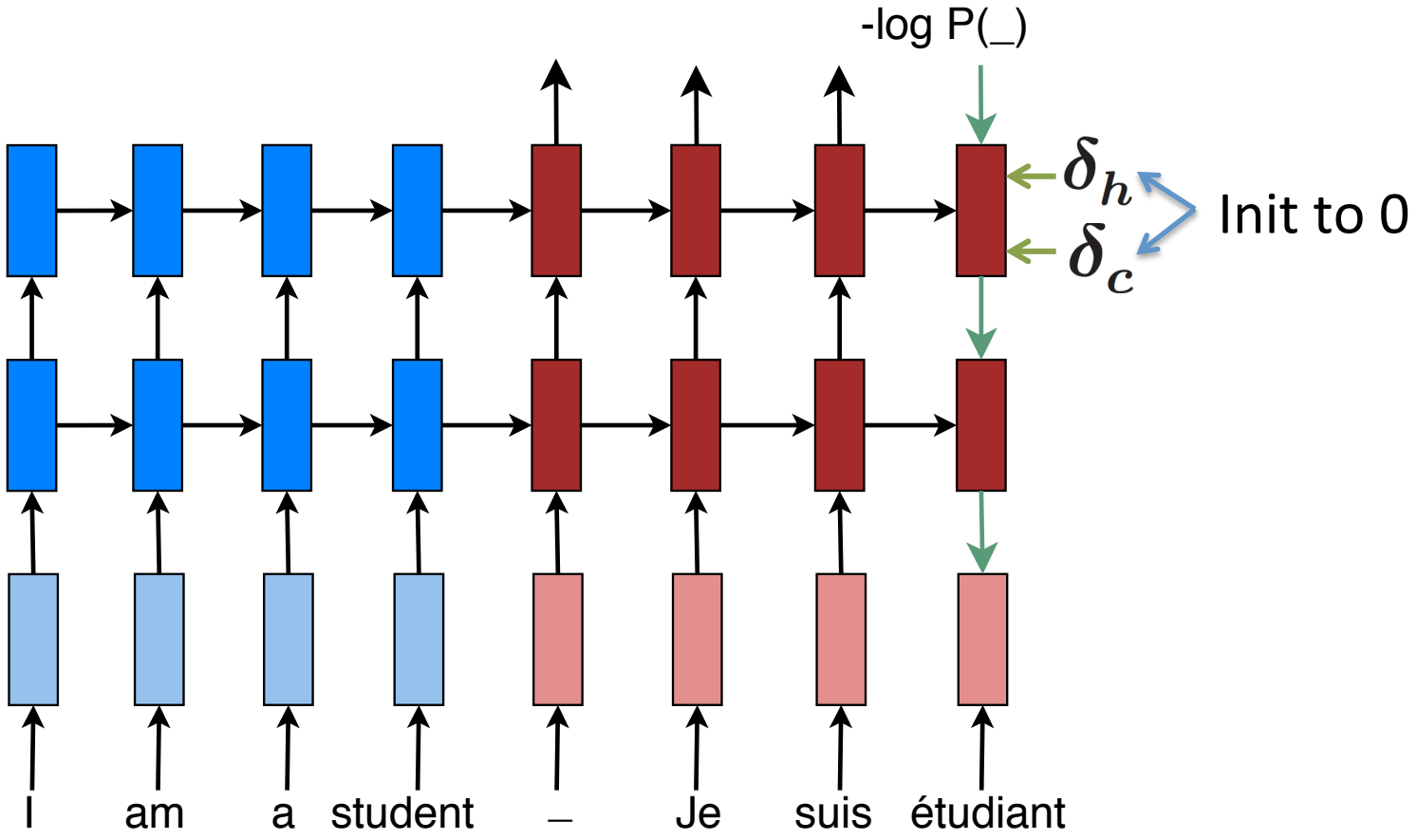
Encoder-decoder Summary

	Encoder	Decoder
(Sutskever et al., 2014) (Luong et al., 2015a) (Luong et al., 2015b)	Deep LSTM	Deep LSTM
(Cho et al., 2014a) (Bahdanau et al., 2015) (Jean et al., 2015)	(Bidirectional) GRU	GRU
(Kalchbrenner & Blunsom, 2013)	CNN	(Inverse CNN) RNN
(Cho et al., 2014b)	Gated Recursive CNN	GRU

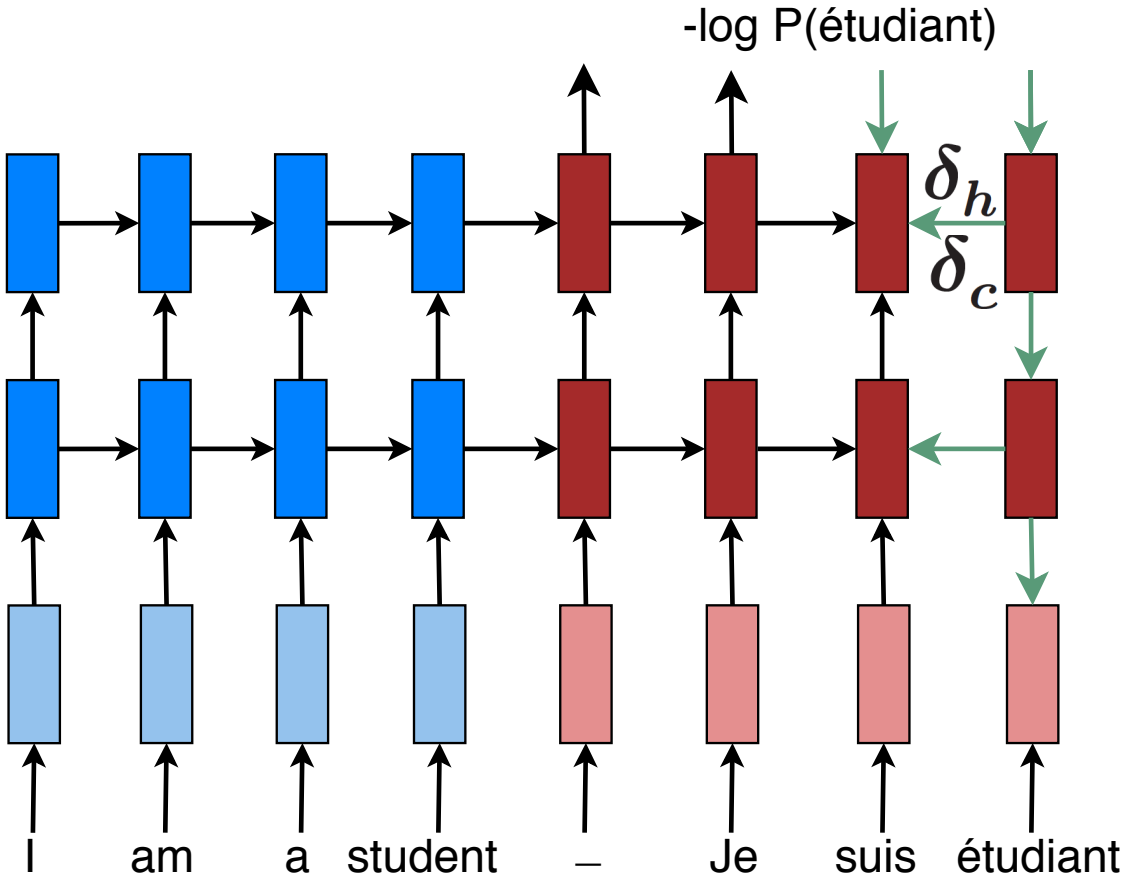
Outline

- NMT basics (Sutskever et al., 2014)
  - Architecture.
  - Recurrent units.
  - Backpropagation.
- Attention mechanism (Bahdanau et al., 2015)

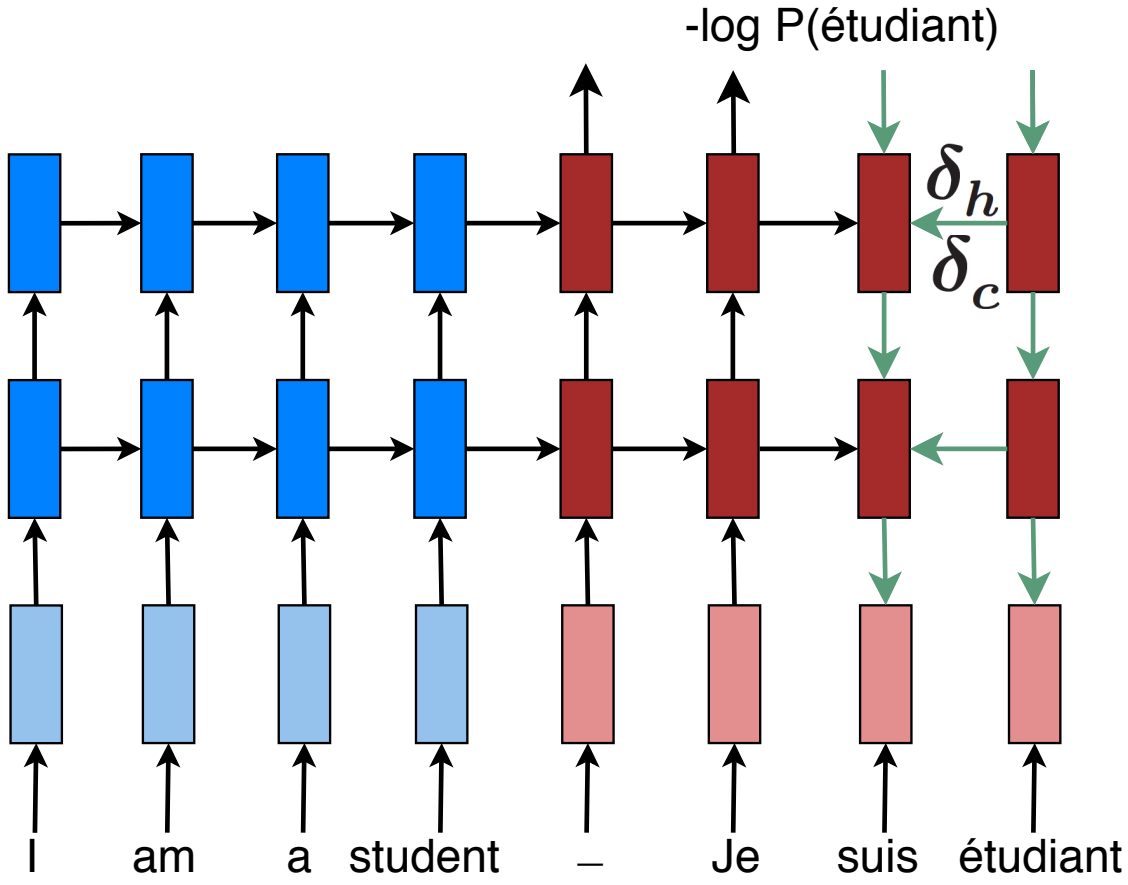
Backpropagation Through Time



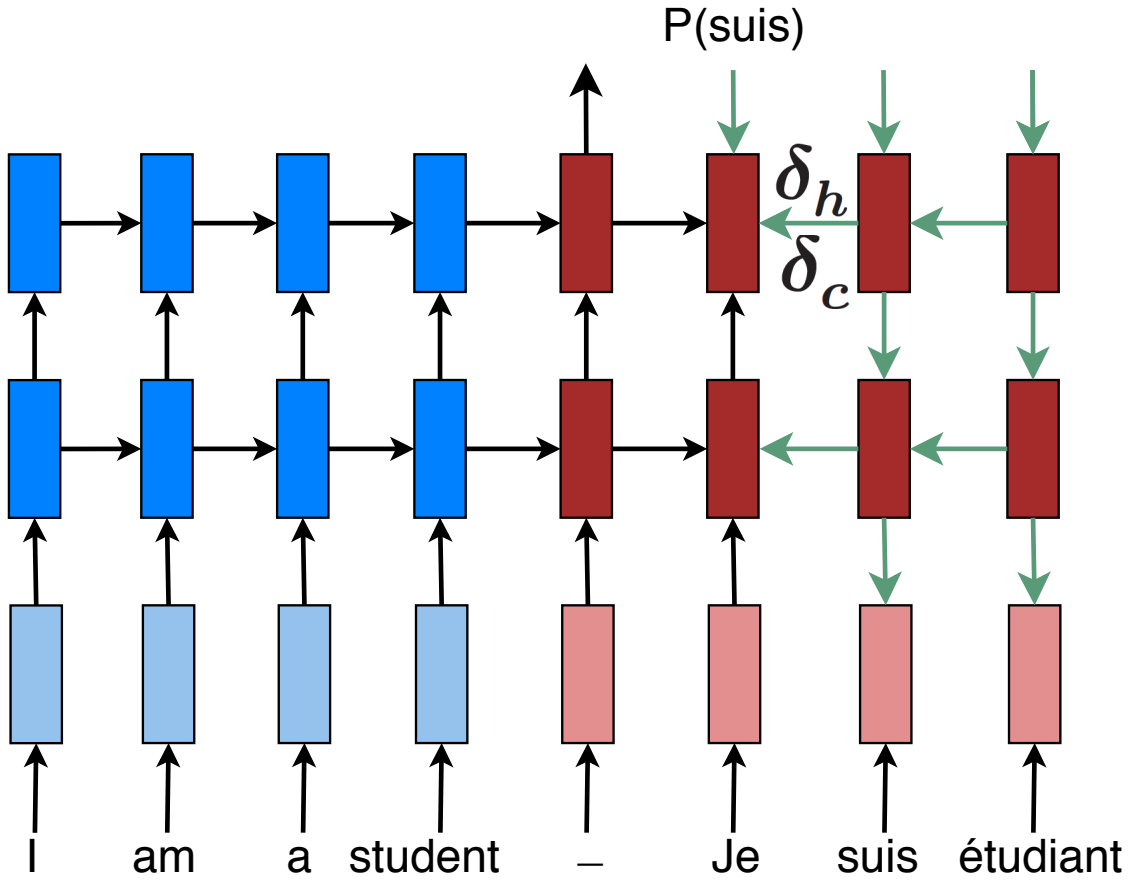
Backpropagation Through Time



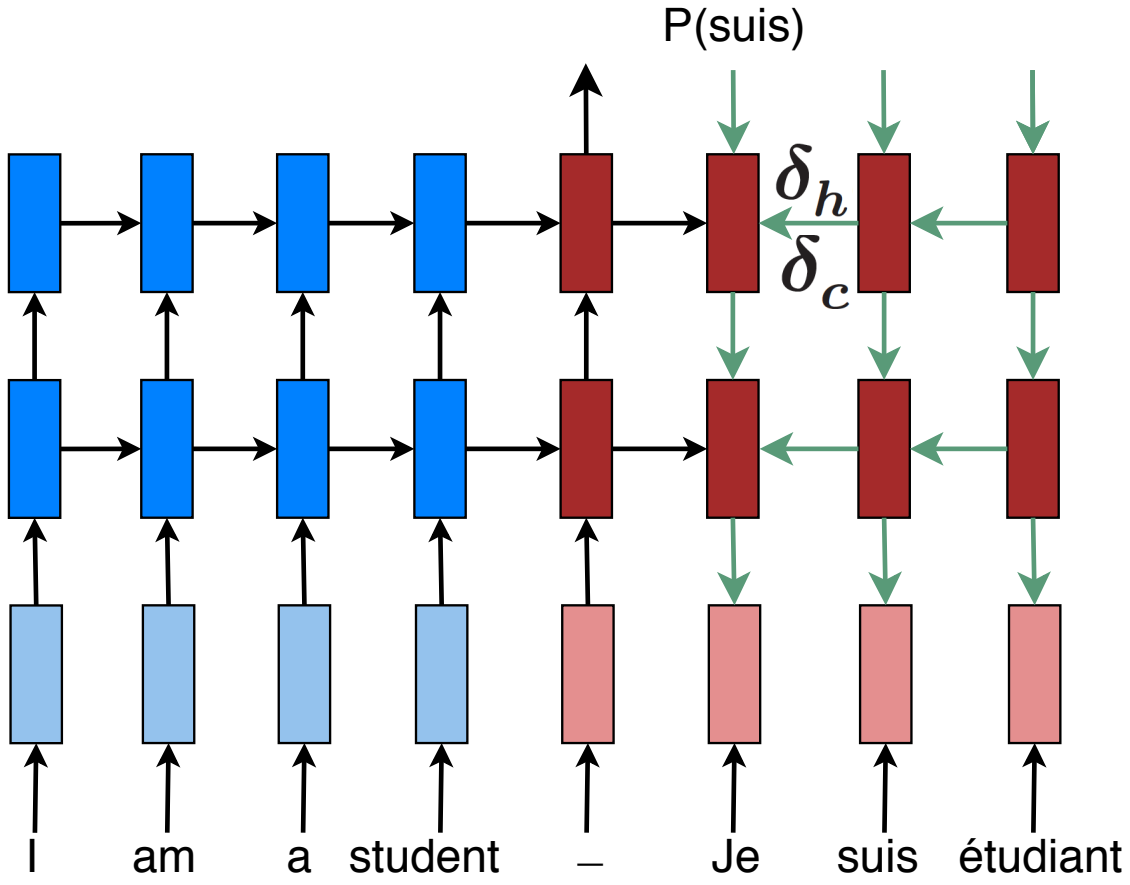
Backpropagation Through Time



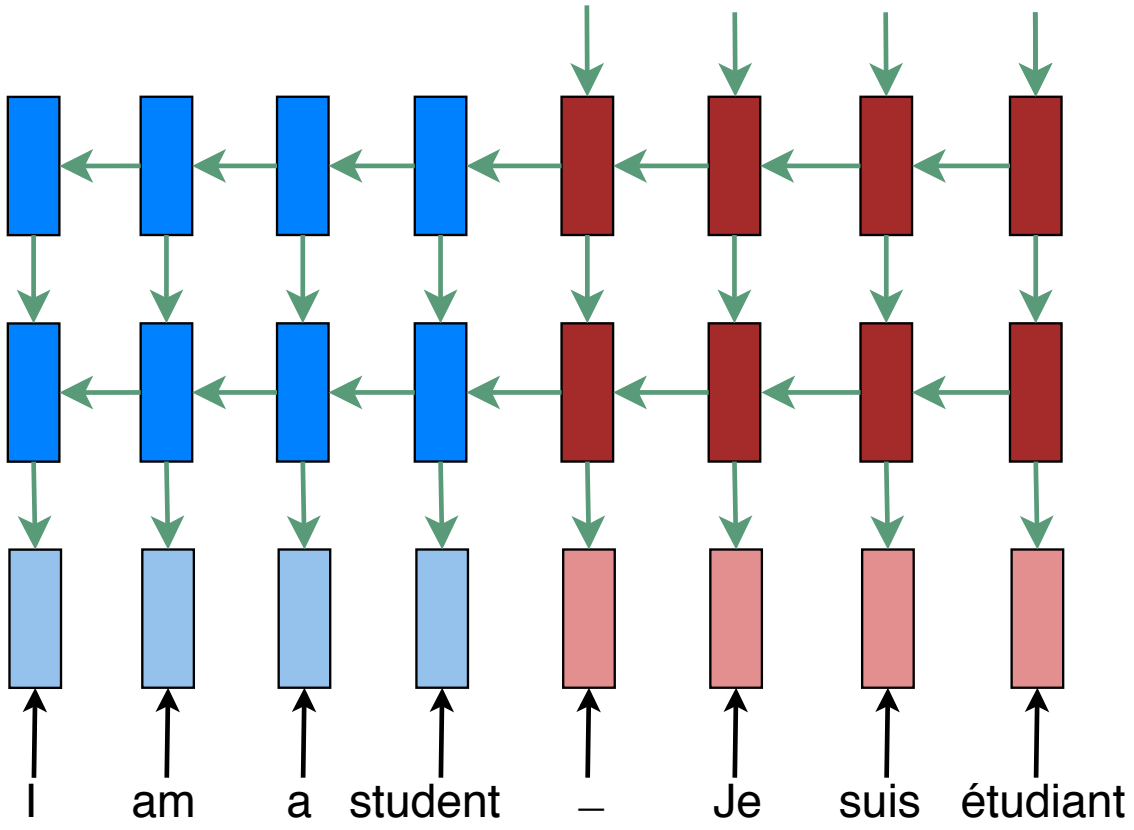
Backpropagation Through Time



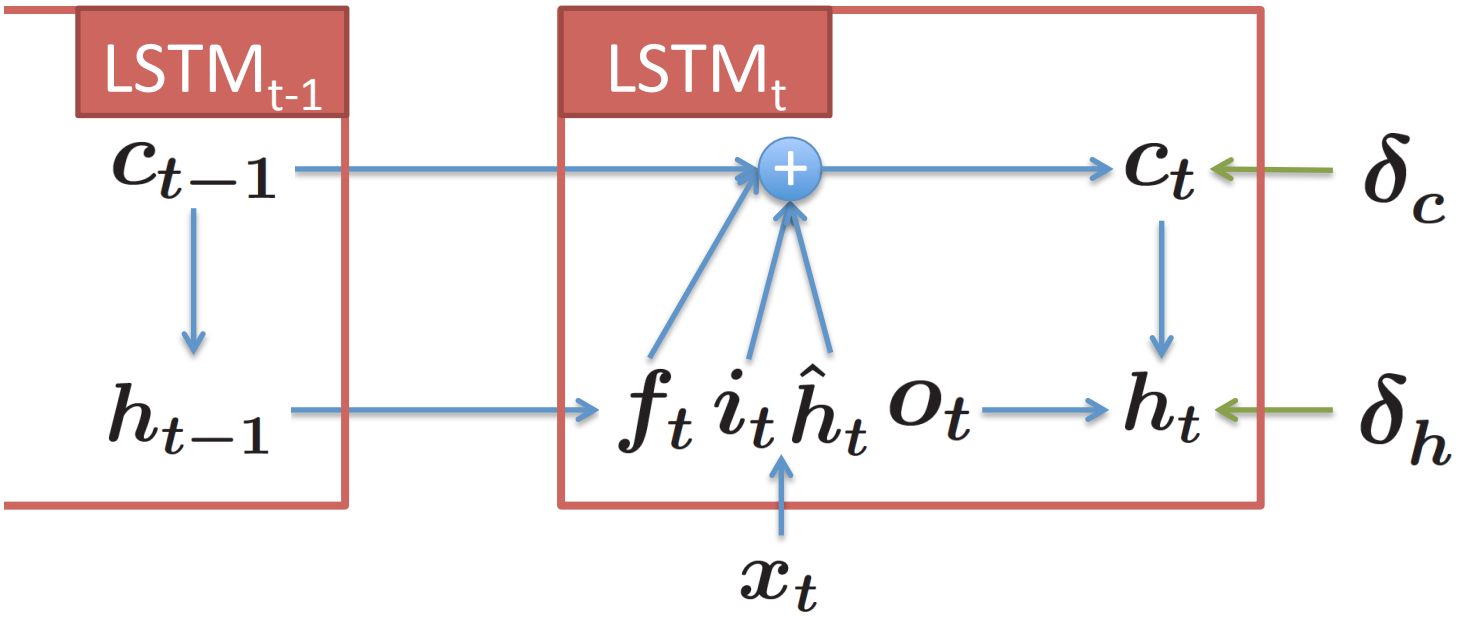
Backpropagation Through Time



Backpropagation Through Time

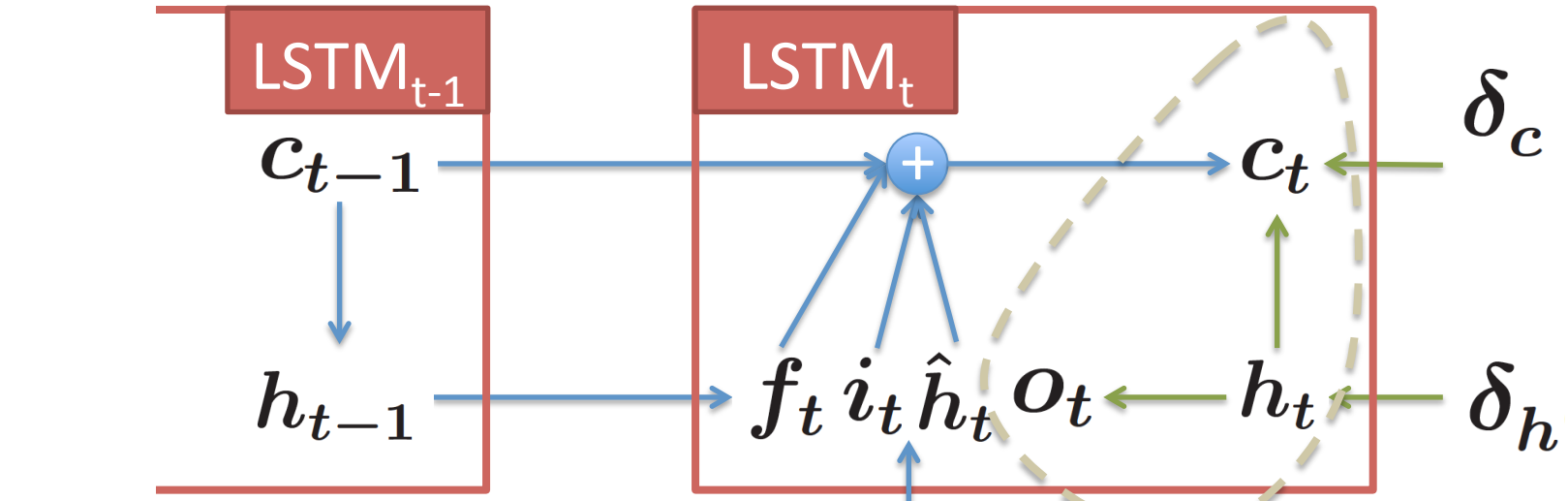


LSTM Backpropagation



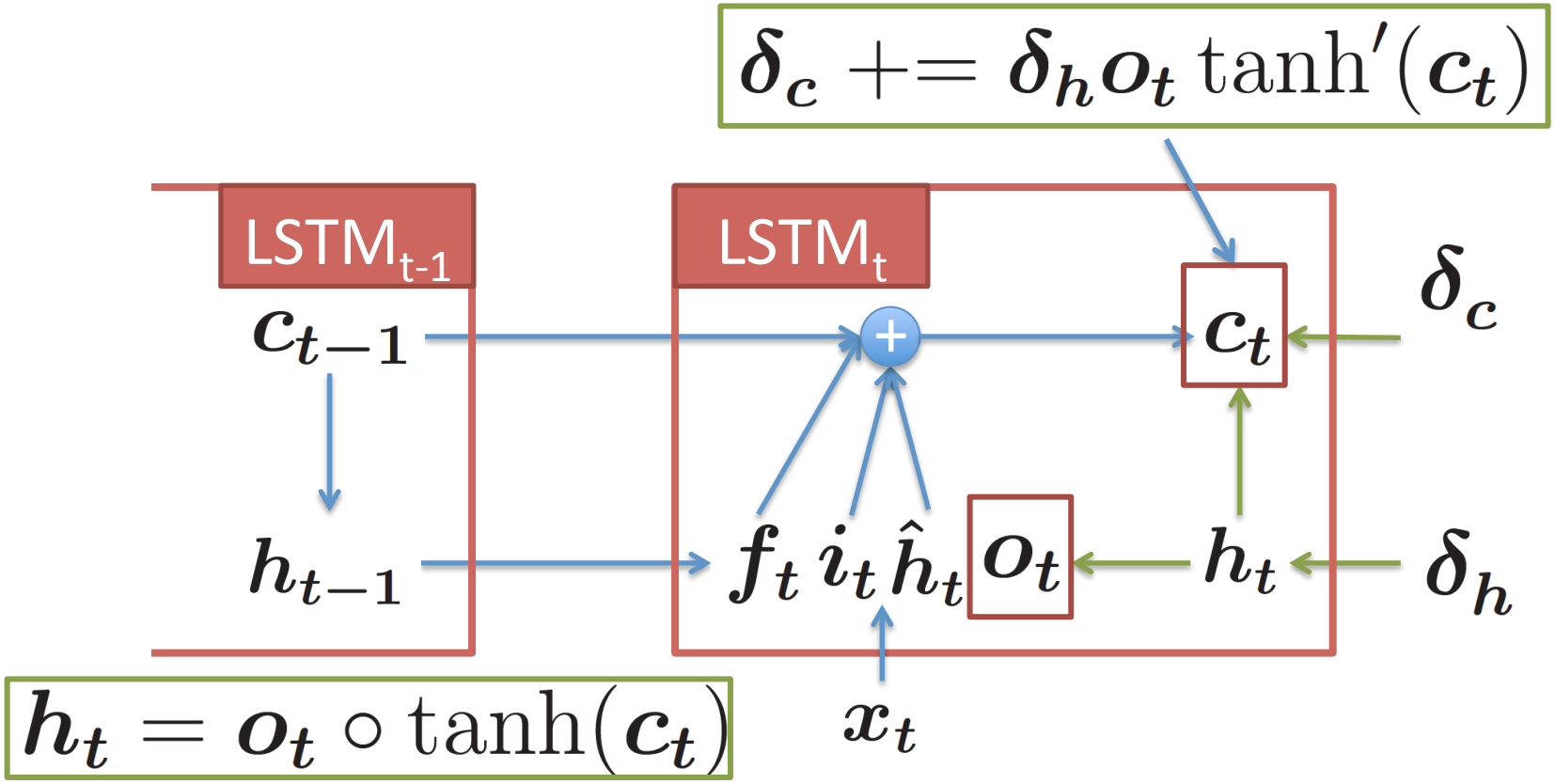
- Deltas sent back from the top layers.

LSTM Backpropagation – *Context*



$$h_t = o_t \circ \tanh(c_t)$$

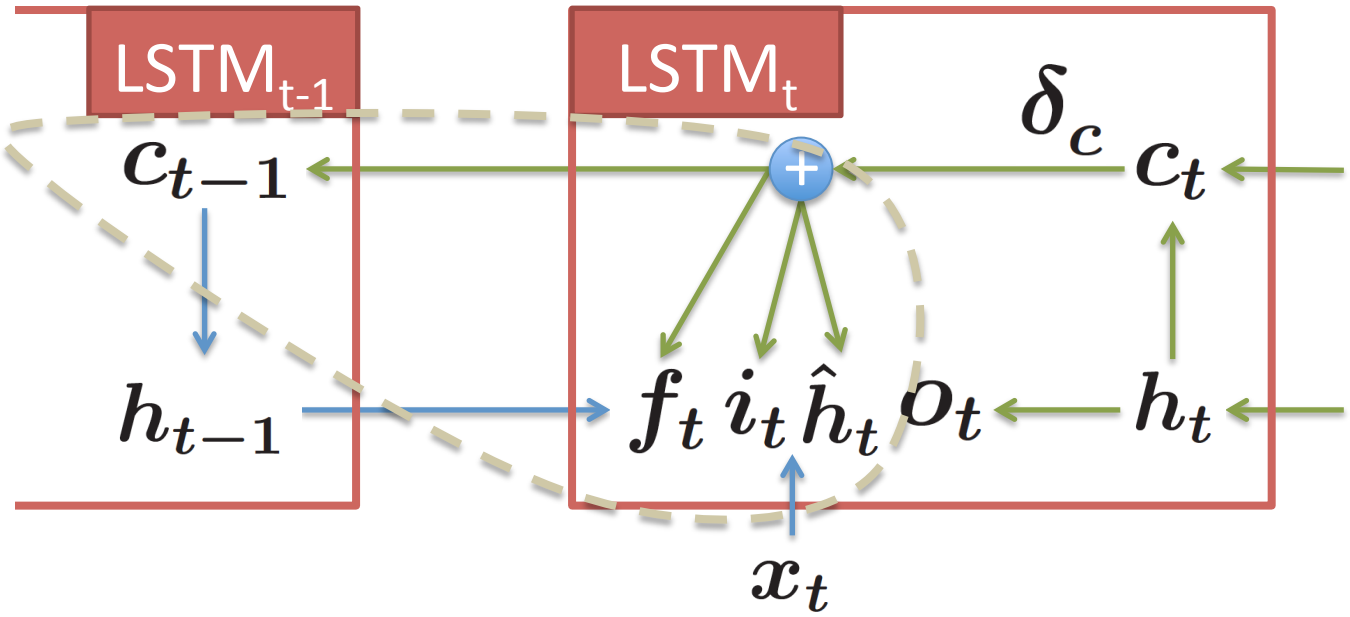
LSTM Backpropagation – *Context*



- Complete context vector gradient.

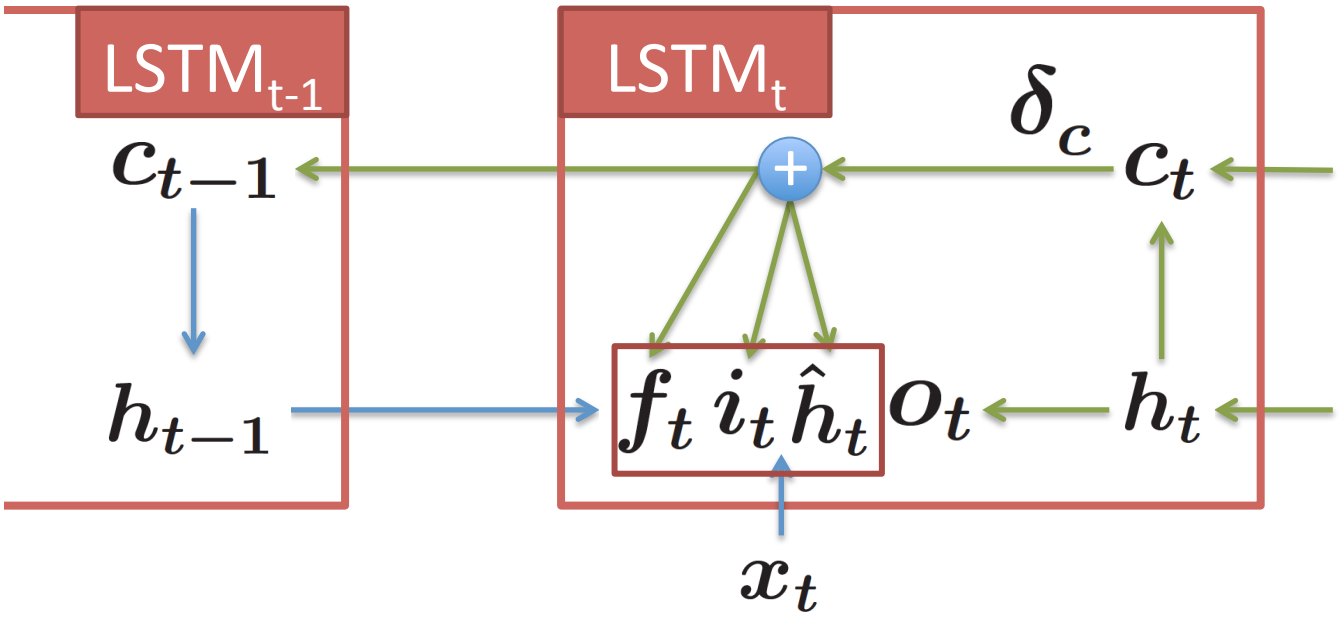
LSTM Backpropagation – *Context*

$$c_t = f_t \circ c_{t-1} + i_t \circ \hat{h}_t$$



LSTM Backpropagation – *Context*

$$c_t = f_t \circ c_{t-1} + i_t \circ \hat{h}_t$$

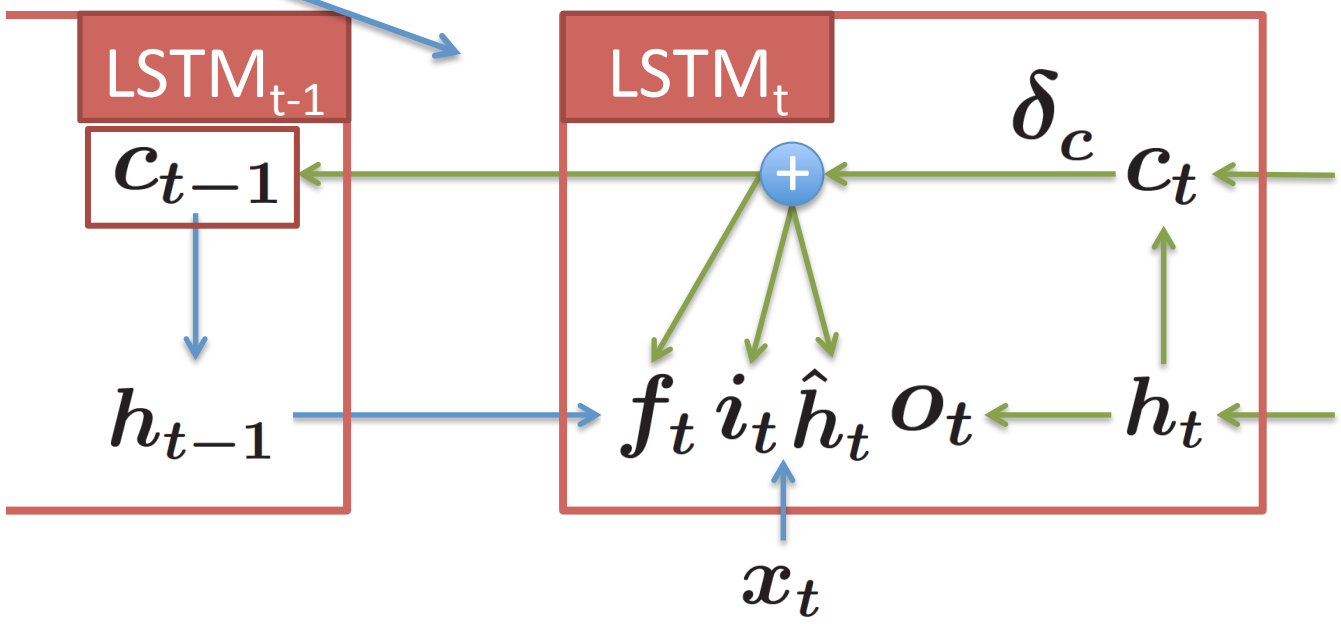


- First, use δ_c to compute gradients for $f_t\ i_t\ \hat{h}_t$.

LSTM Backpropagation – *Context*

$$\delta_c = \delta_c \circ f_t$$

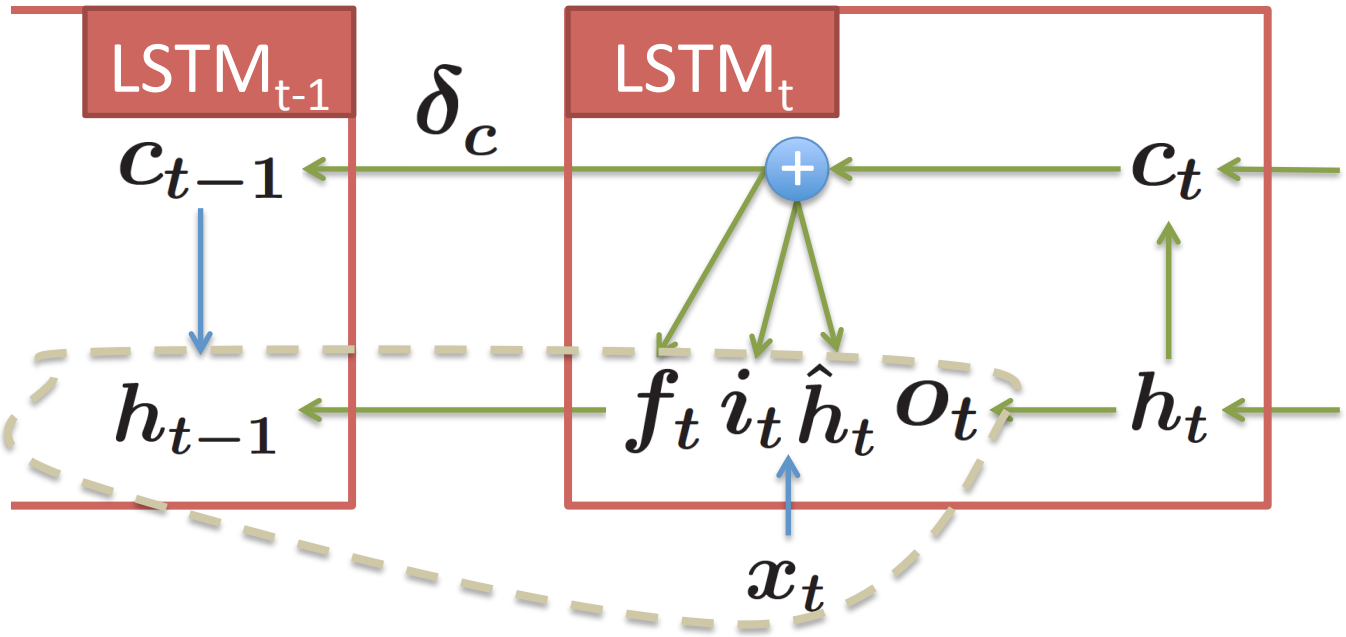
$$c_t = f_t \circ c_{t-1} + i_t \circ \hat{h}_t$$



- Then, update δ_c.

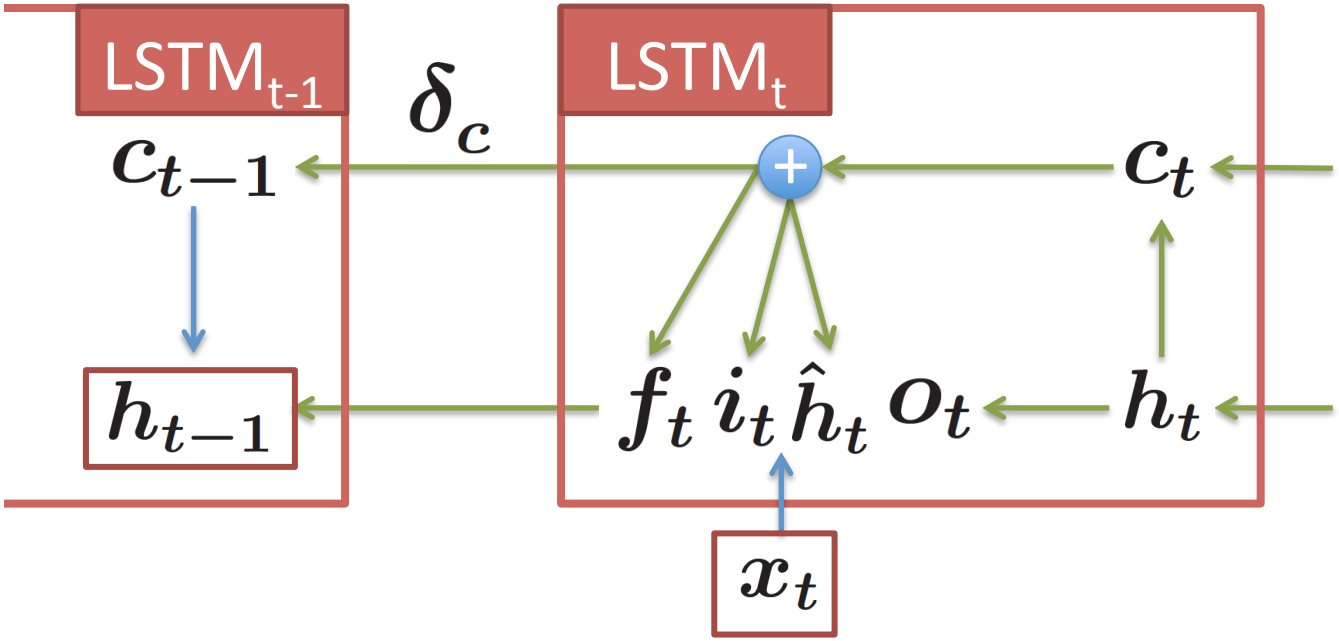
LSTM Backprop

$$\begin{pmatrix} i_t \\ f_t \\ o_t \\ \hat{h}_t \end{pmatrix} = \begin{pmatrix} \text{sigm} \\ \text{sigm} \\ \text{sigm} \\ \tanh \end{pmatrix} T_{4n \times 2n} \begin{bmatrix} x_t \\ h_{t-1} \end{bmatrix}$$



LSTM Backprop

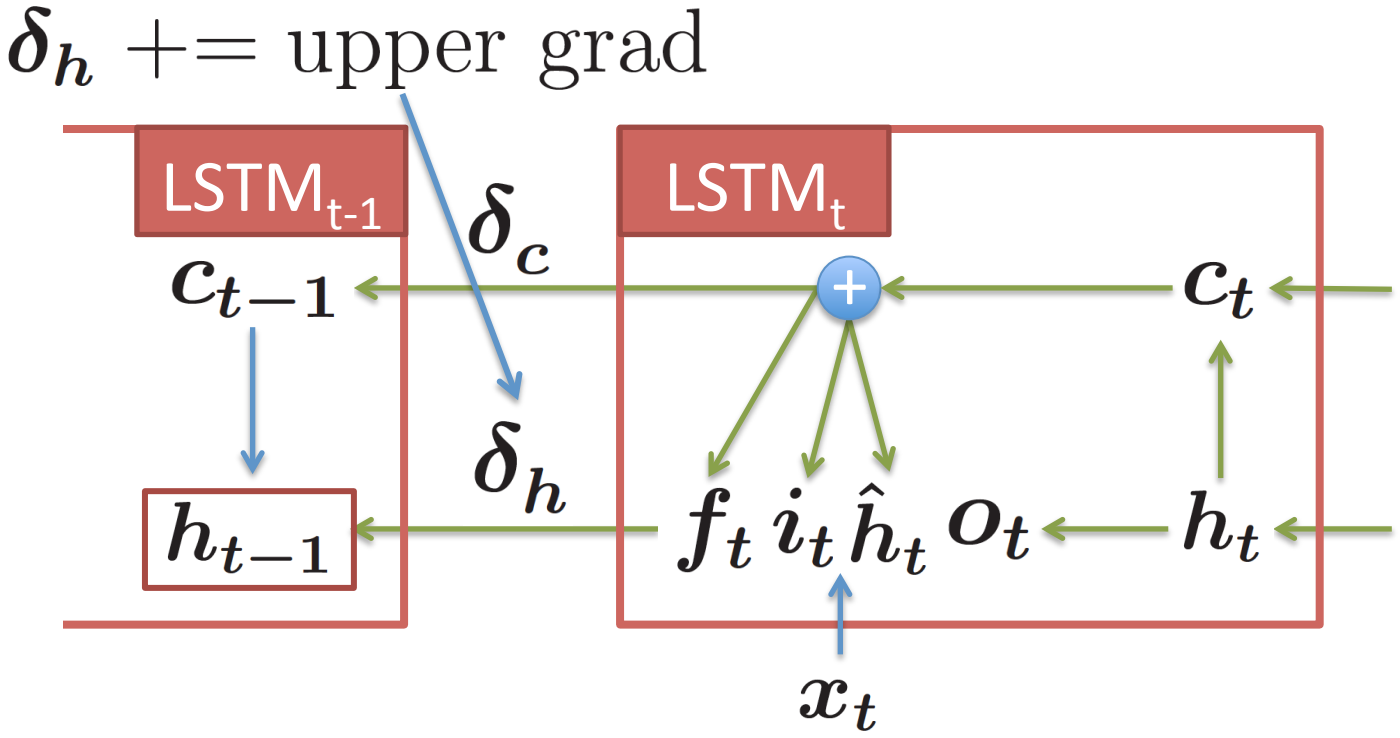
$$\begin{pmatrix} i_t \\ f_t \\ o_t \\ \hat{h}_t \end{pmatrix} = \begin{pmatrix} \text{sigm} \\ \text{sigm} \\ \text{sigm} \\ \tanh \end{pmatrix} T_{4n \times 2n} \begin{bmatrix} x_t \\ h_{t-1} \end{bmatrix}$$



- Compute gradients for $T_{4n \times 2n}$, x_t, h_{t-1}.

LSTM Backprop

$$\begin{pmatrix} i_t \\ f_t \\ o_t \\ \hat{h}_t \end{pmatrix} = \begin{pmatrix} \text{sigm} \\ \text{sigm} \\ \text{sigm} \\ \tanh \end{pmatrix} T_{4n \times 2n} \begin{bmatrix} x_t \\ h_{t-1} \end{bmatrix}$$



- Add gradients from the loss / upper layers.

Summary

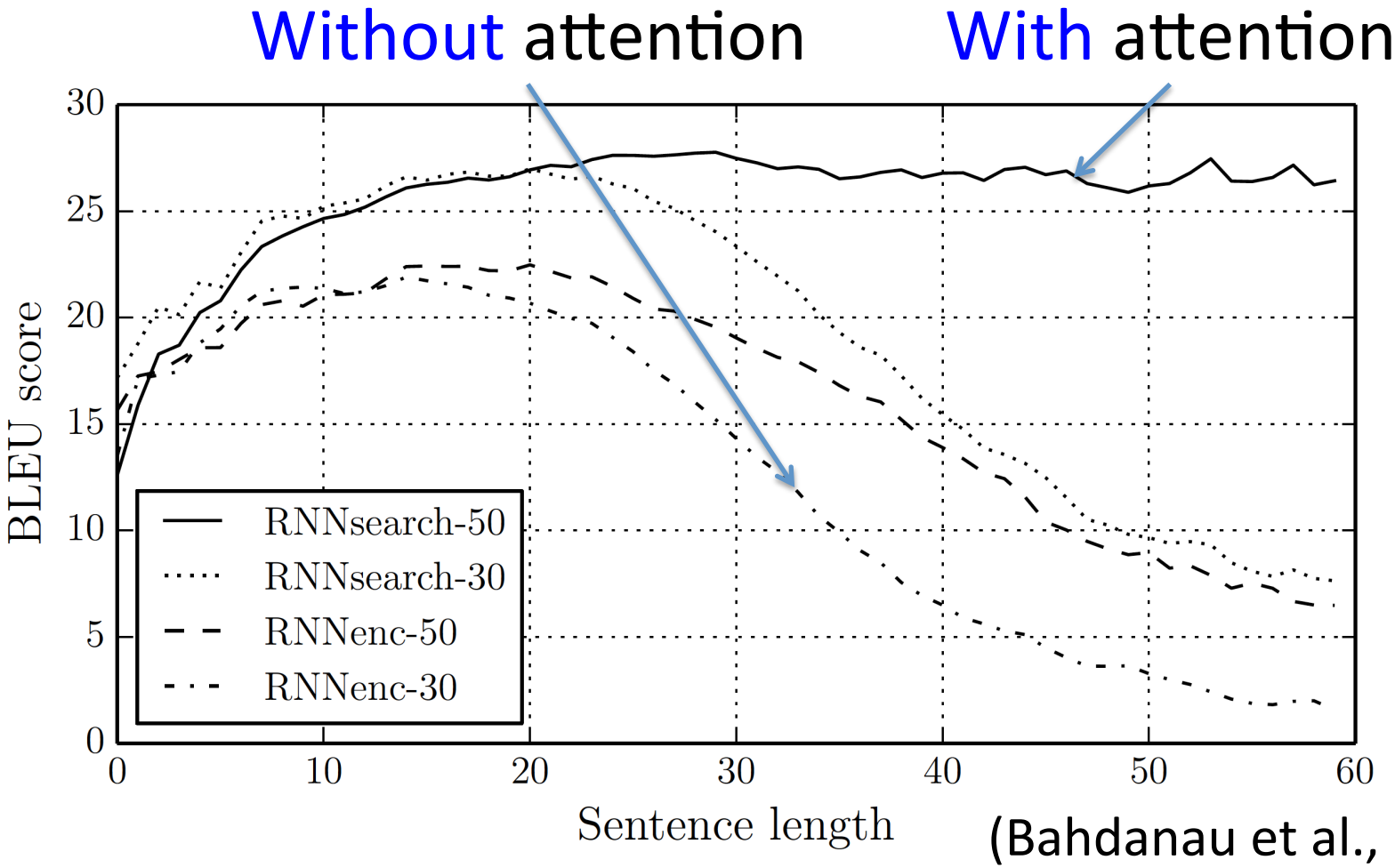
- LSTM backpropagation is nasty.

- But it will be much easier if:
  - Know your matrix calculus!
  - Pay attention to $\boldsymbol{\delta}_{\boldsymbol{c}}$ and $\boldsymbol{\delta}_{\boldsymbol{h}}$.

Outline

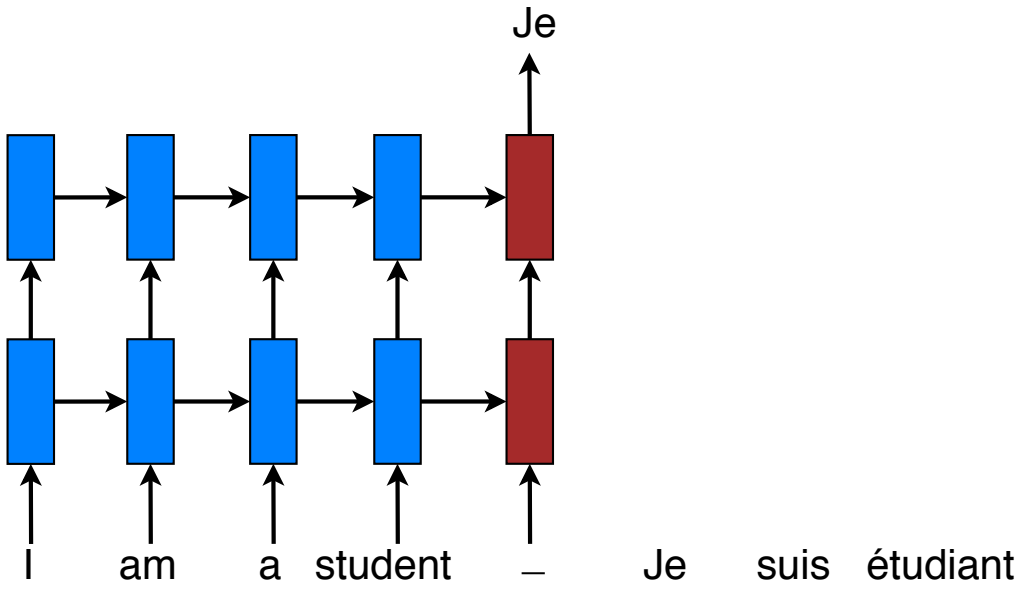
- NMT basics
- Attention mechanism (Bahdanau et al., 2015)

Sentence Length Problem



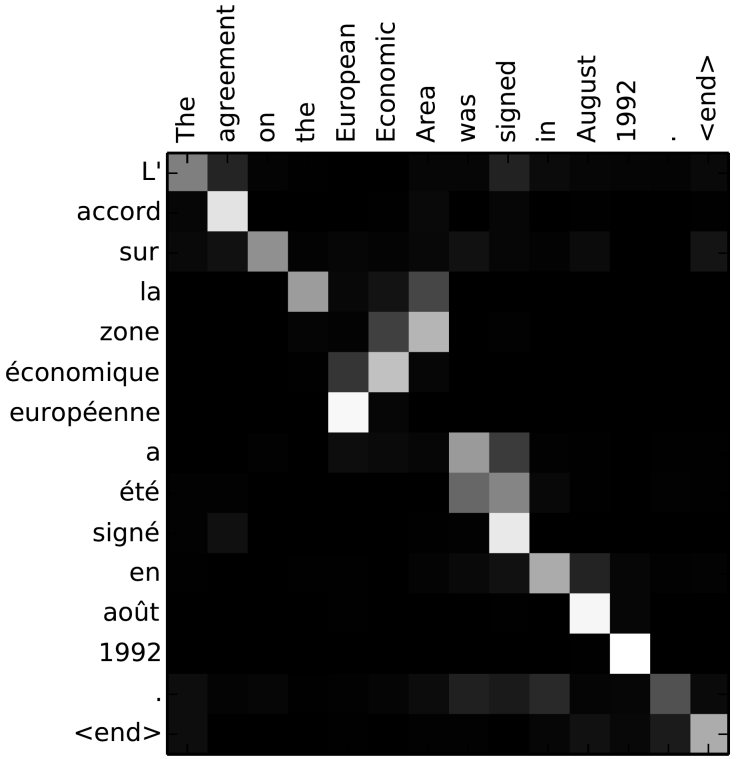
(Bahdanau et al., 2015)

Attention Mechanism



- Problem: Markovian process.
- Solution: random access memory
  - A memory of source hidden states.
  - cf. Neural Turing Machine (Graves et al., 2014).

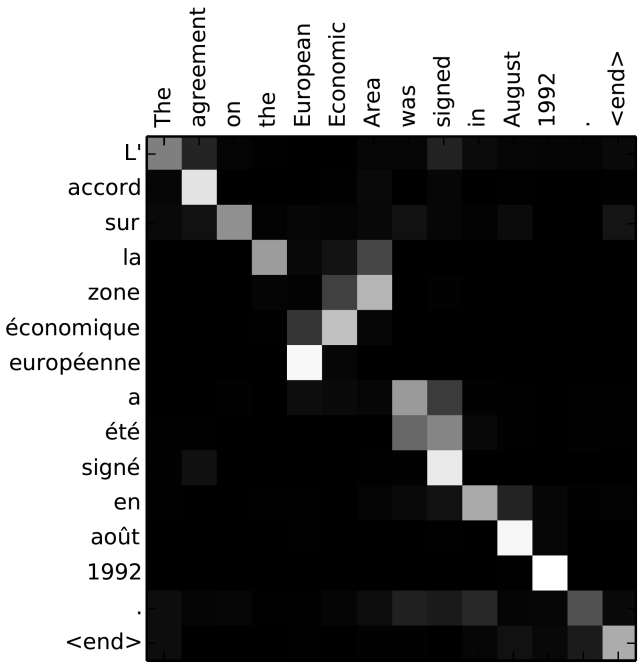
Attention mechanism



(Bahdanau et al., 2015)

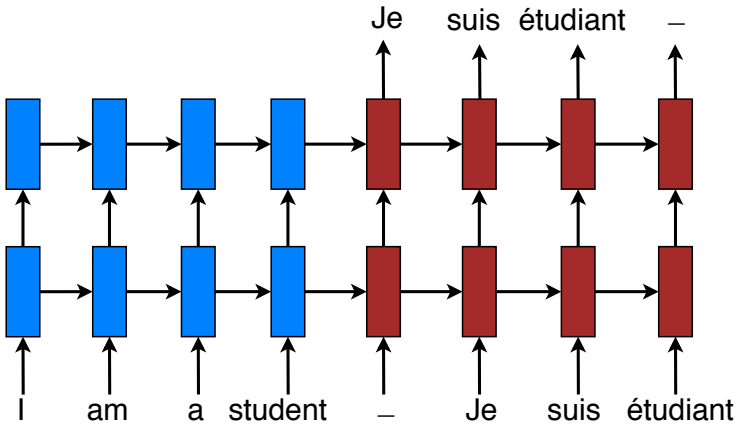
- Recent innovation in deep learning:
  - Control problem (Mnih et al., 14)
  - Speech recognition (Chorowski et al., 15)
  - Image captioning (Xu et al., 15)

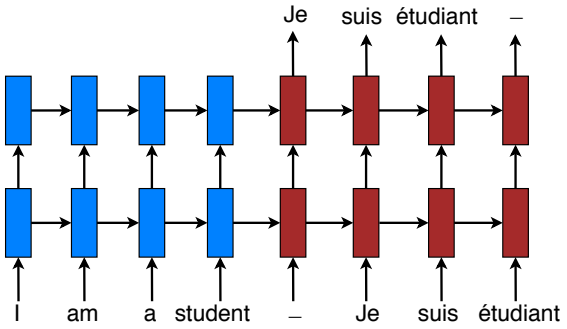
Simplified Attention
(Bahdanau et al., 2015)

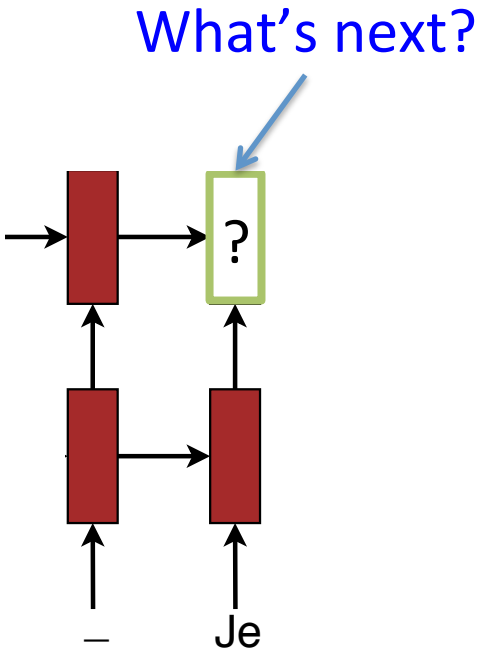


+

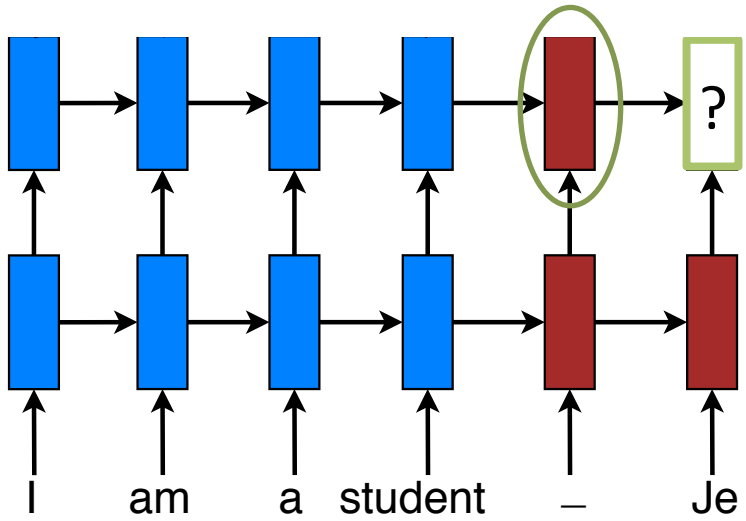
Deep LSTM
(Sutskever et al., 2014)




Je
suis
étudiant
–
I
am
a
student
–
Je
suis
étudiant


What's next?
?
–
Je

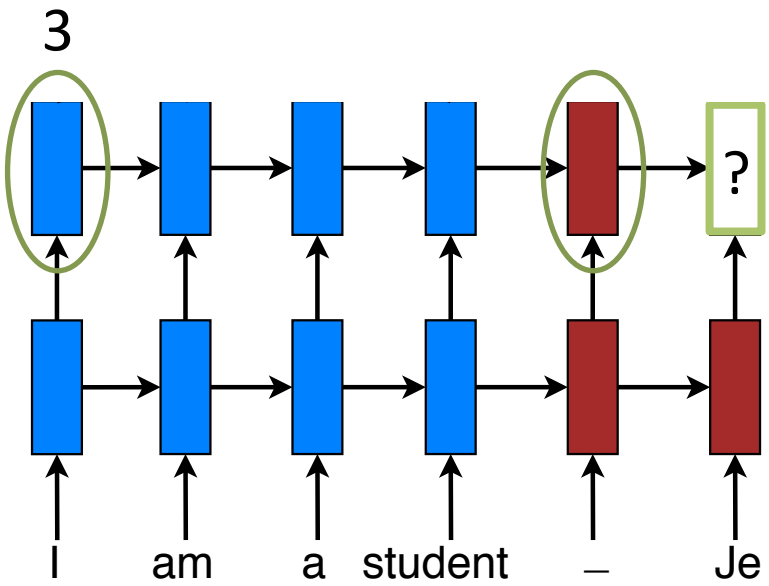
Attention Mechanism – *Scoring*



- Compare target and source hidden states.

Attention Mechanism – *Scoring*

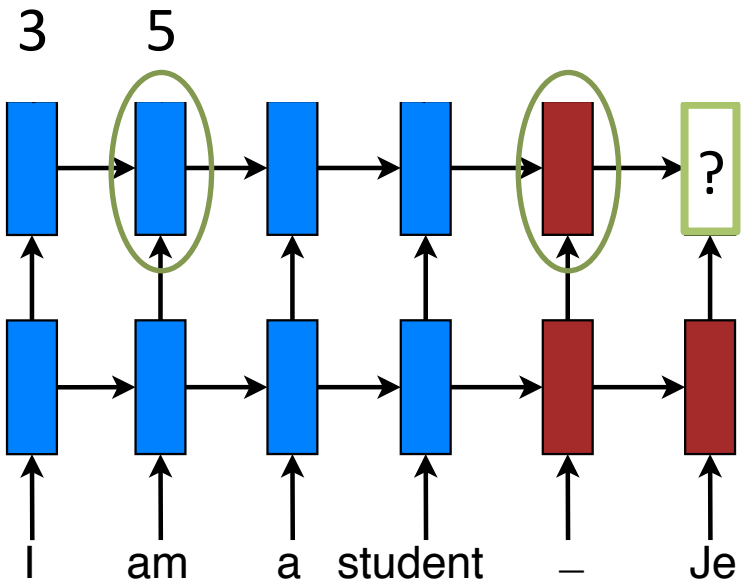
$$\text{score}(\boldsymbol{h}_t, \bar{\boldsymbol{h}}_s)$$



- Compare target and source hidden states.

Attention Mechanism – *Scoring*

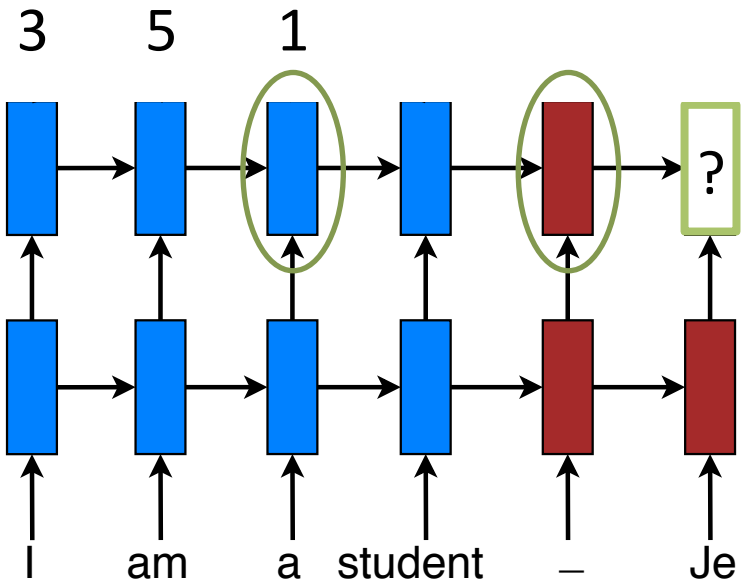
$$\text{score}(\boldsymbol{h}_t, \bar{\boldsymbol{h}}_s)$$



- Compare target and source hidden states.

Attention Mechanism – *Scoring*

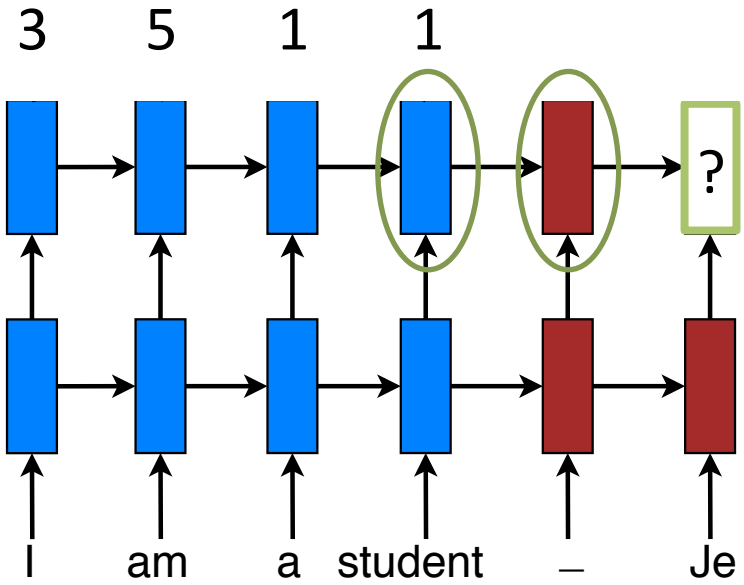
$$\text{score}(\boldsymbol{h}_t, \bar{\boldsymbol{h}}_s)$$



- Compare target and source hidden states.

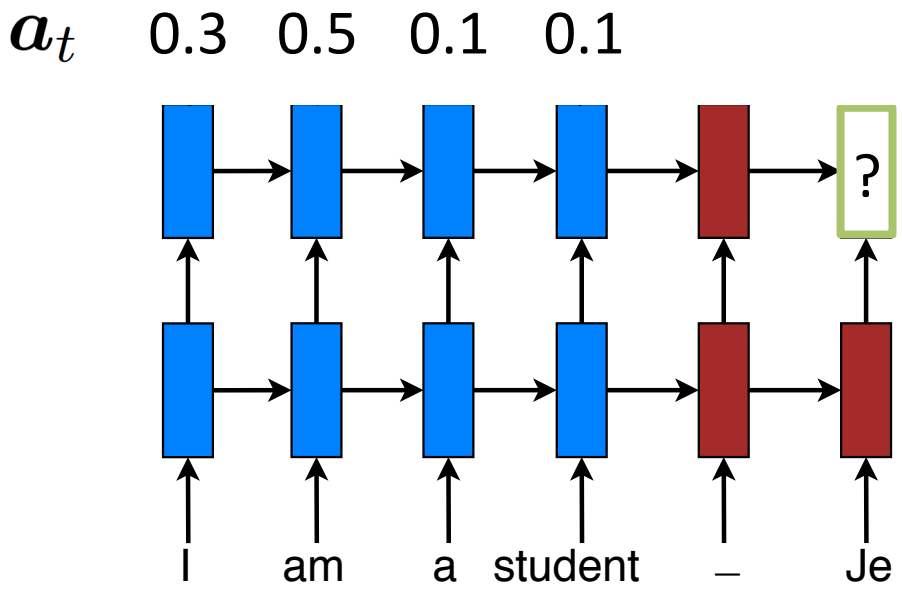
Attention Mechanism – *Scoring*

$$\text{score}(\boldsymbol{h}_t, \bar{\boldsymbol{h}}_s)$$



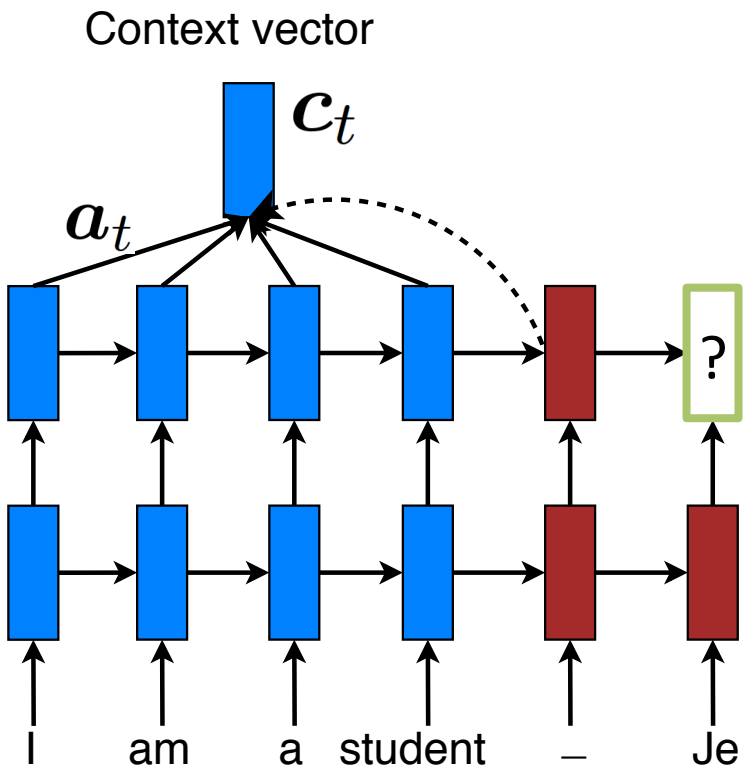
- Compare target and source hidden states.

Attention Mechanism – *Normalization*



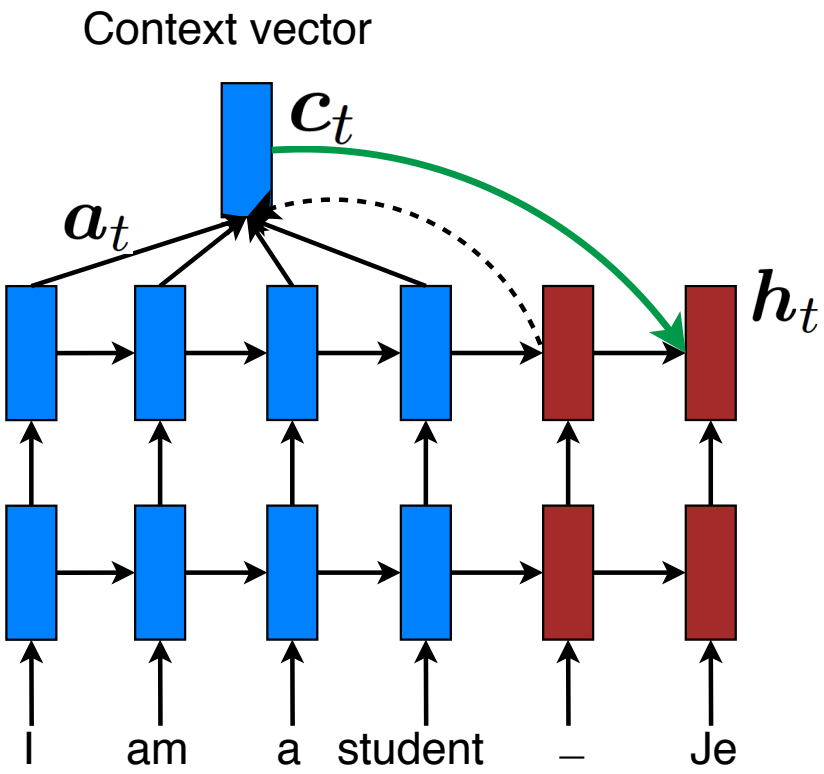
- Convert into alignment weights.

Attention Mechanism – *Context vector*



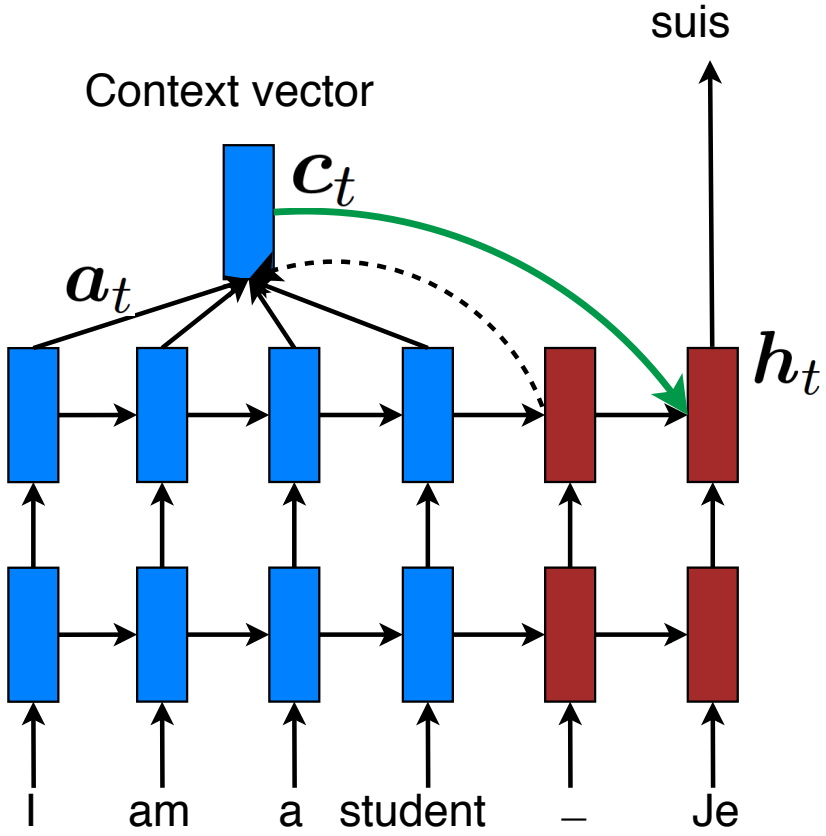
- Build context vector: weighted average.

Attention Mechanism – *Hidden state*



- Compute the next hidden state.

Attention Mechanism – *Predict*



- Predict the next word.

Attention Functions

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

(Bahdanau et al., 2015)

Attention Functions

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

(Luong et al., 2015b)

(Bahdanau et al., 2015)

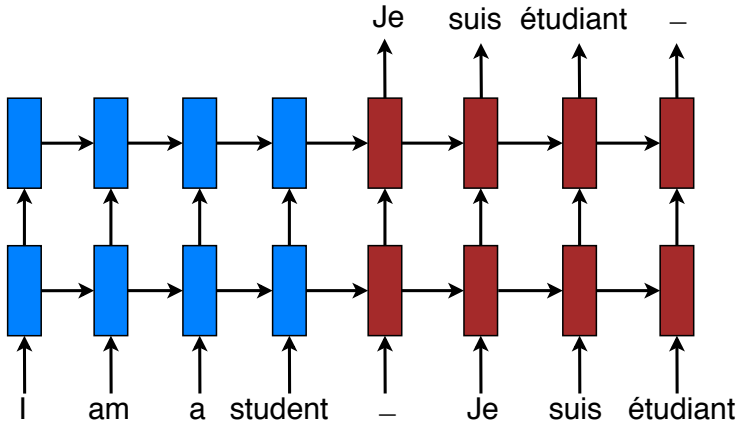
- Works so far for NMT!
- Concern: insensitive to locations
  - Similar weights for same source words
  - Bi-directional RNNs can help.

Other Attention Functions

- Content-based: $\boldsymbol{a}_t = Attend(\boldsymbol{h}_{t-1}, \bar{\boldsymbol{h}}_{1...S})$
- Location-based: $\boldsymbol{a}_t = Attend(\boldsymbol{h}_{t-1}, \boldsymbol{a}_{t-1})$
  - (Graves, 2013): hand-writing synthesis model.
- Hybrid: $\boldsymbol{a}_t = Attend(\boldsymbol{h}_{t-1}, \boldsymbol{a}_{t-1}, \bar{\boldsymbol{h}}_{1...S})$
  - (Chorowski et al., 2015) for speech recognition.

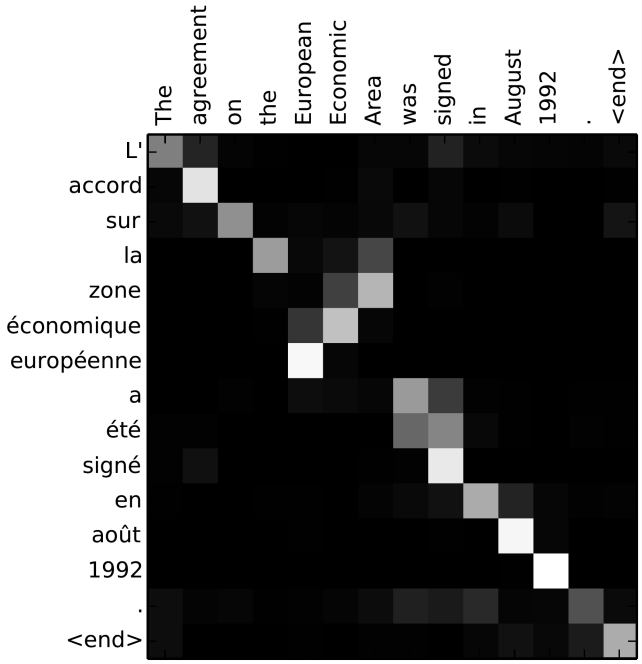
Summary

Deep LSTM
(Sutskever et al., 2014)



+

Simplified Attention
(Bahdanau et al., 2015)



References (1)

- [Bahdanau et al., 2015] Neural Translation by Jointly Learning to Align and Translate. http://arxiv.org/pdf/1409.0473.pdf
- [Cho et al., 2014a] Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. http://aclweb.org/anthology/D/D14/D14-1179.pdf
- [Cho et al., 2014b] On the Properties of Neural Machine Translation: Encoder–Decoder Approaches. http://www.aclweb.org/anthology/W14-4012
- [Chorowski et al., 2015] Attention-Based Models for Speech Recognition. http://arxiv.org/pdf/1506.07503v1.pdf
- [Chung et al., 2015] Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. http://arxiv.org/pdf/1412.3555.pdf
- [Graves, 2013] Generating Sequences With Recurrent Neural Networks. http://arxiv.org/pdf/1308.0850v5.pdf
- [Graves, 2014] Neural Turing Machine. http://arxiv.org/pdf/1410.5401v2.pdf.
- [Hochreiter & Schmidhuber, 1997] Long Short-term Memory. http://deeplearning.cs.cmu.edu/pdfs/Hochreiter97_lstm.pdf

References (2)

- [Kalchbrenner & Blunsom, 2013] Recurrent Continuous Translation Models. http://nal.co/papers/KalchbrennerBlunsom_EMNLP13
- [Luong et al., 2015a] Addressing the Rare Word Problem in Neural Machine Translation. http://www.aclweb.org/anthology/P15-1002
- [Luong et al., 2015b] Effective Approaches to Attention-based Neural Machine Translation. https://aclweb.org/anthology/D/D15/D15-1166.pdf
- [Mnih et al., 2014] Recurrent Models of Visual Attention. http://papers.nips.cc/paper/5542-recurrent-models-of-visual-attention.pdf
- [Pascanu et al., 2013] On the difficulty of training Recurrent Neural Networks. http://arxiv.org/pdf/1211.5063v2.pdf
- [Xu et al., 2015] Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. http://jmlr.org/proceedings/papers/v37/xuc15.pdf
- [Sutskever et al., 2014] Sequence to Sequence Learning with Neural Networks. http://papers.nips.cc/paper/5346-sequence-to-sequence-learning-with-neural-networks.pdf
- [Zaremba et al., 2015] Recurrent Neural Network Regularization. http://arxiv.org/pdf/1409.2329.pdf