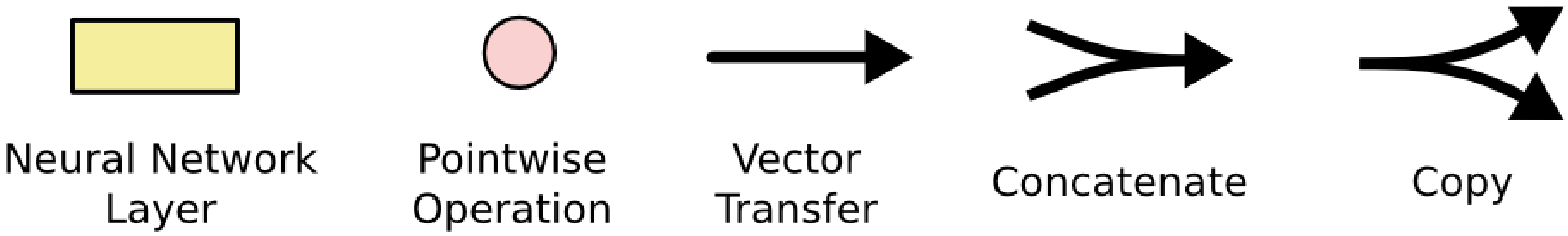
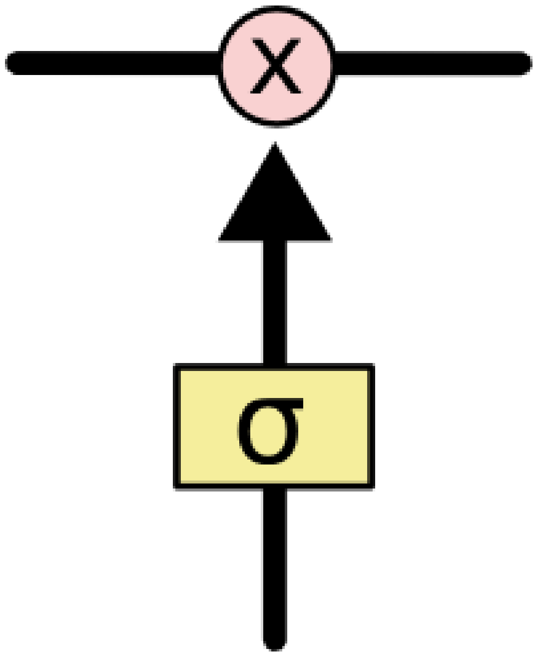
Neural Machine Translation

Manish Shrivastava





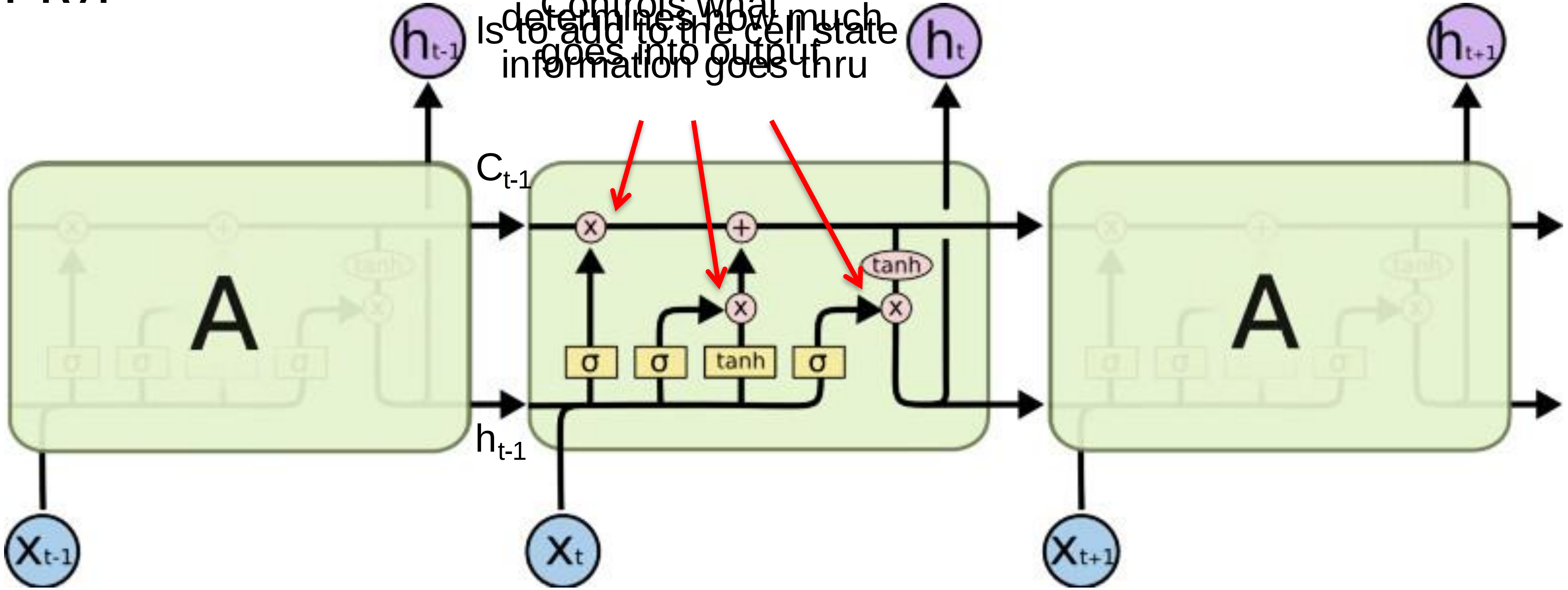
The sigmoid layer outputs numbers between 0-1 determine how much each component should be let through. Pink X gate is point-wise multiplication.

LSTM

Output gate
Controls what
goes into output

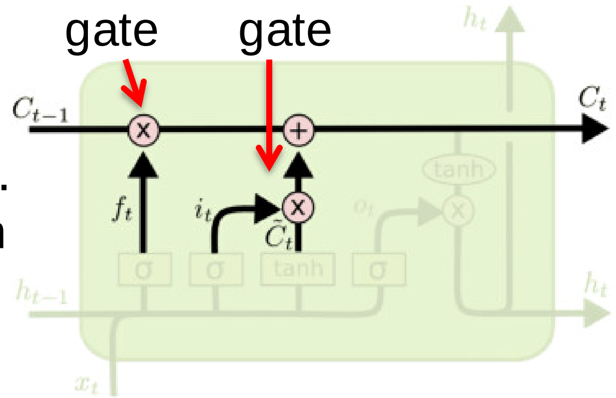
This sigmoid gate
determines how much
information goes thru

This decides what info
Is to add to the cell state



Forget gate

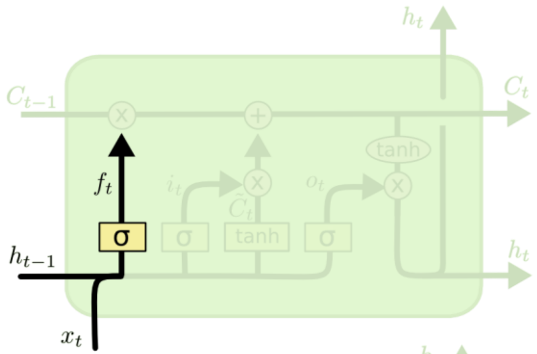
input gate



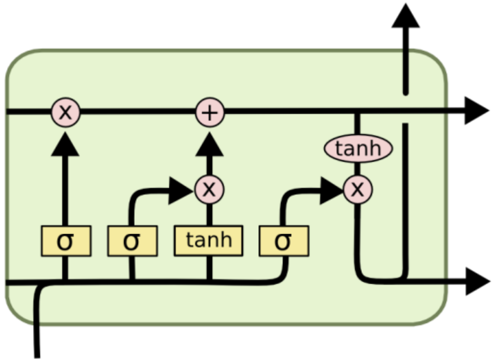
$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

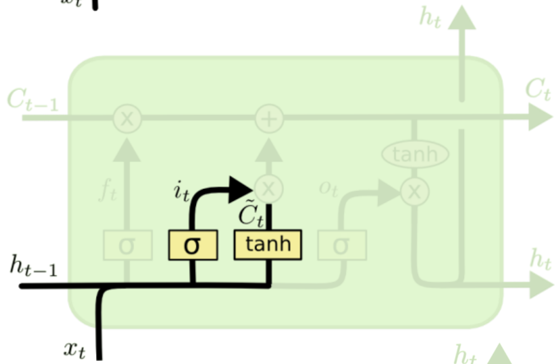
The core idea is this cell state C_t, it is changed slowly, with only minor linear interactions. It is very easy for information to flow along it unchanged.

Why sigmoid or tanh:
Sigmoid: 0,1 gating as switch.
Vanishing gradient problem in LSTM is handled already.
ReLU replaces tanh ok?



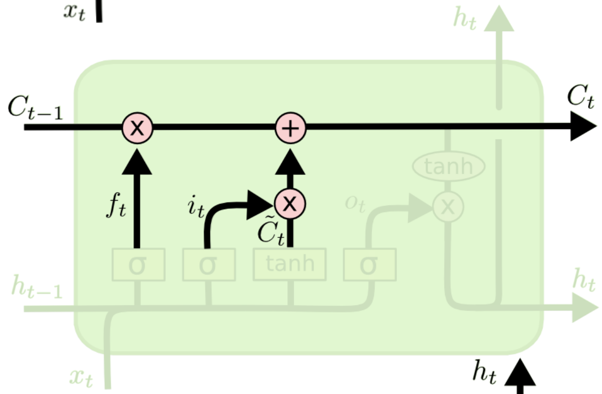
$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \ + \ b_f\right)$$





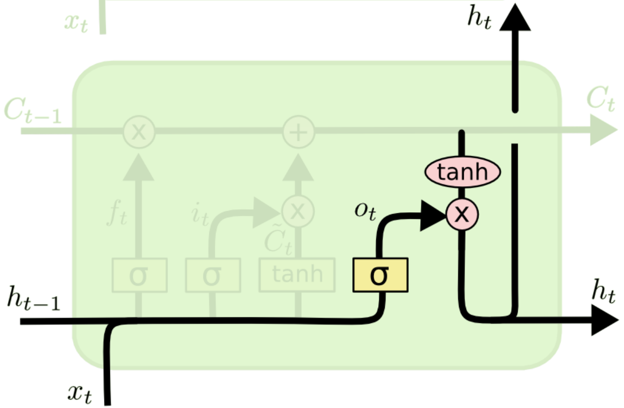
$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \ + \ b_C)$$

i_t decides what component is to be updated.
C'$_t$ provides change contents



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

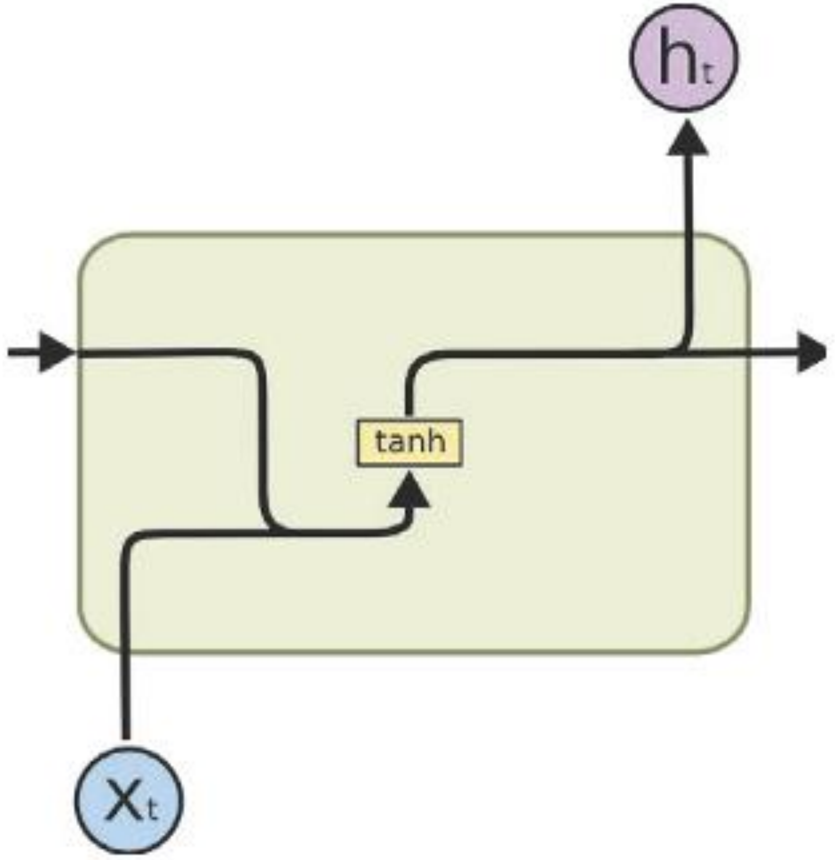
Updating the cell state



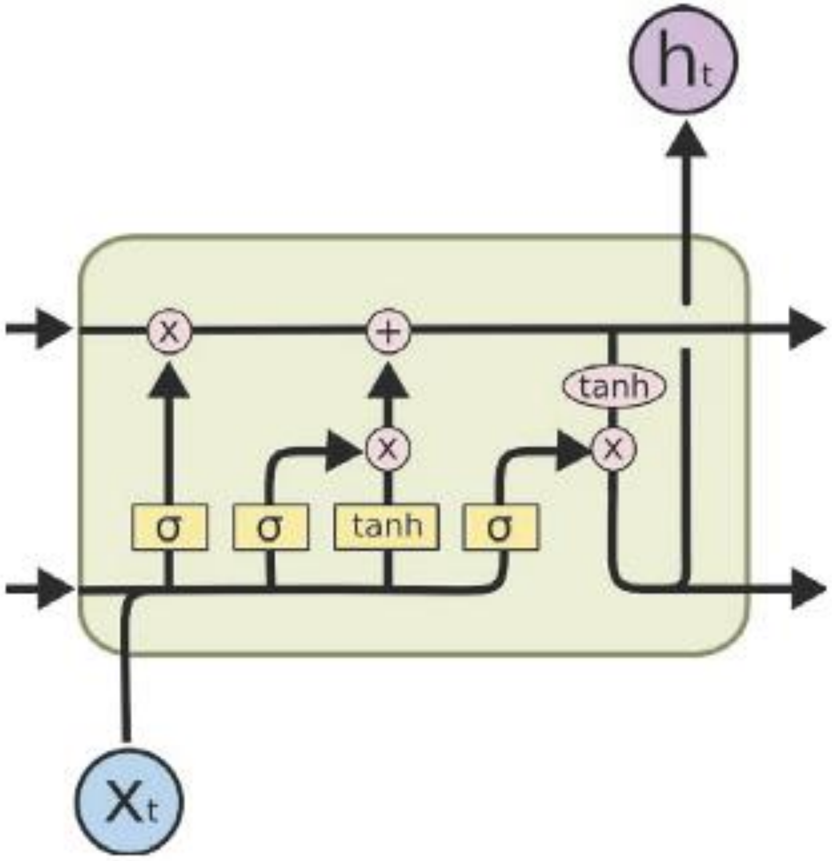
$$o_t = \sigma\left(W_o \ [\, h_{t-1}, x_t] \ + \ b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

Decide what part of the cell state to output

RNN vs LSTM

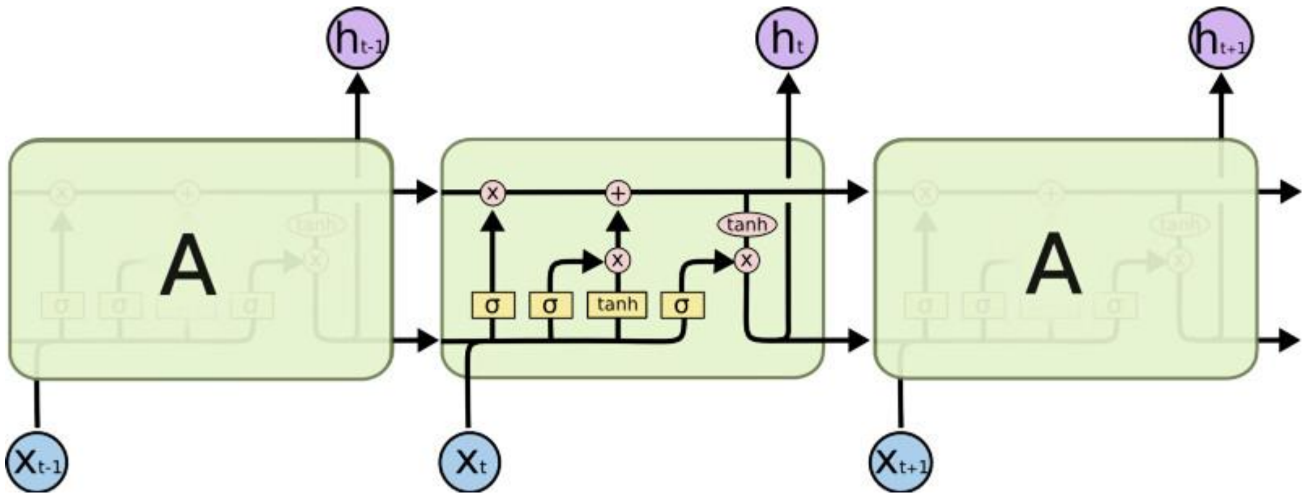


(a) RNN

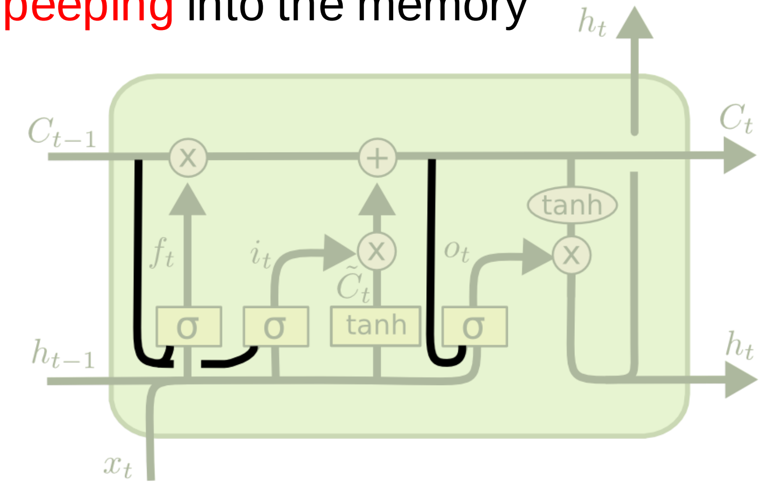


(b) LSTM

Peephole LSTM

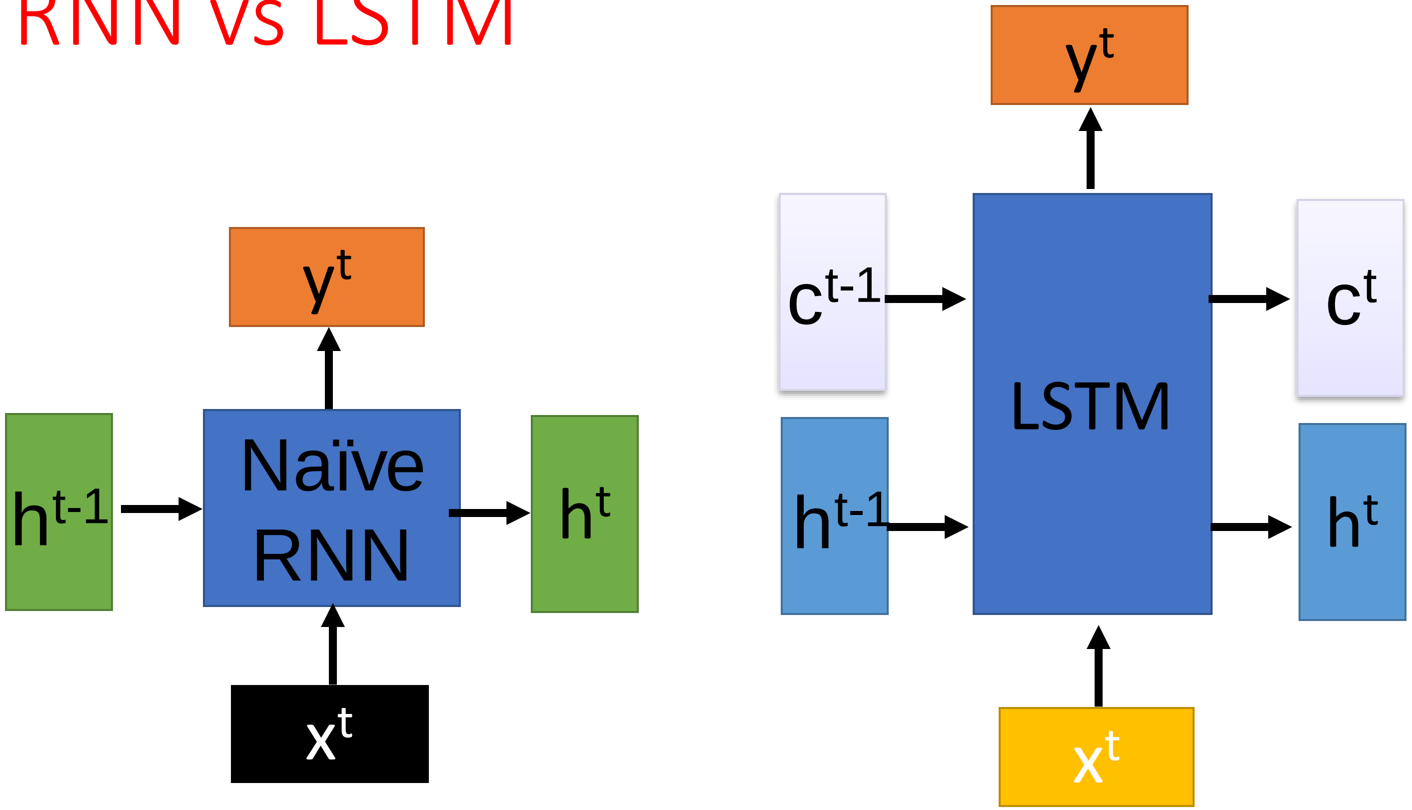


Allows "peeping into the memory"



$$f_t = \sigma\left(W_f \cdot [\boldsymbol{C_{t-1}}, h_{t-1}, x_t] \ + \ b_f\right)$$
$$i_t = \sigma\left(W_i \cdot [\boldsymbol{C_{t-1}}, h_{t-1}, x_t] \ + \ b_i\right)$$
$$o_t = \sigma\left(W_o \cdot [\boldsymbol{C_t}, h_{t-1}, x_t] \ + \ b_o\right)$$

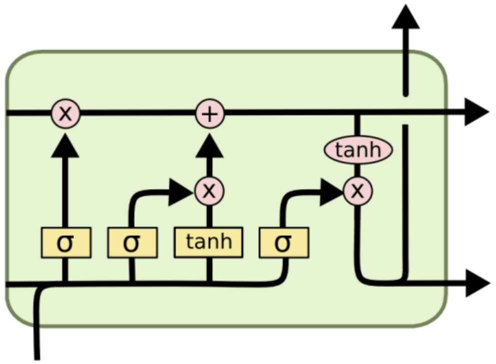
Naïve RNN vs LSTM



c changes slowly ➡ c^t is c^{t-1} added by something

h changes faster ➡ h^t and h^{t-1} can be very different

These 4 matrix computation should be done concurrently.



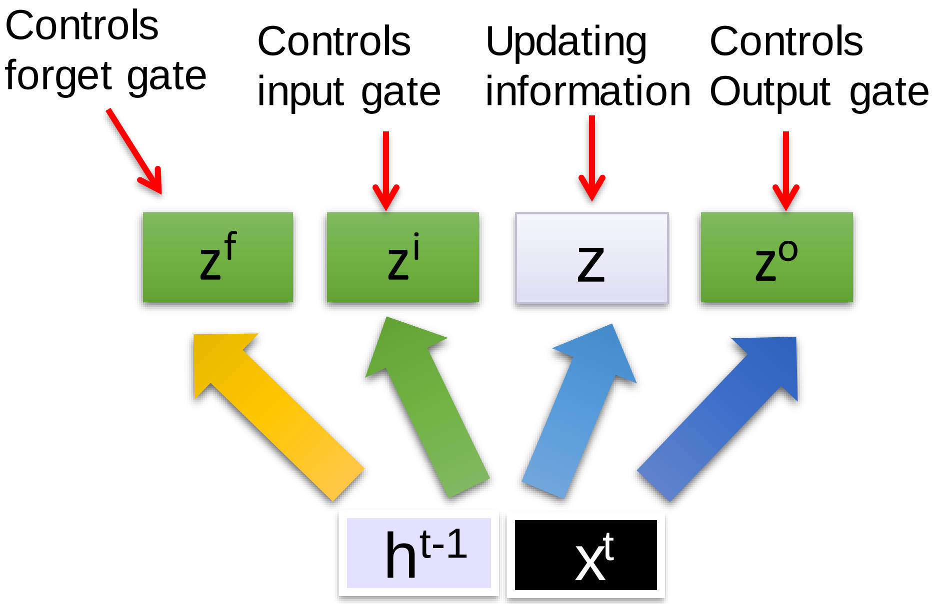
c^{t-1}

$$z = tanh\left(W \begin{bmatrix} x^t \\ h^{t-1} \end{bmatrix} \right)$$

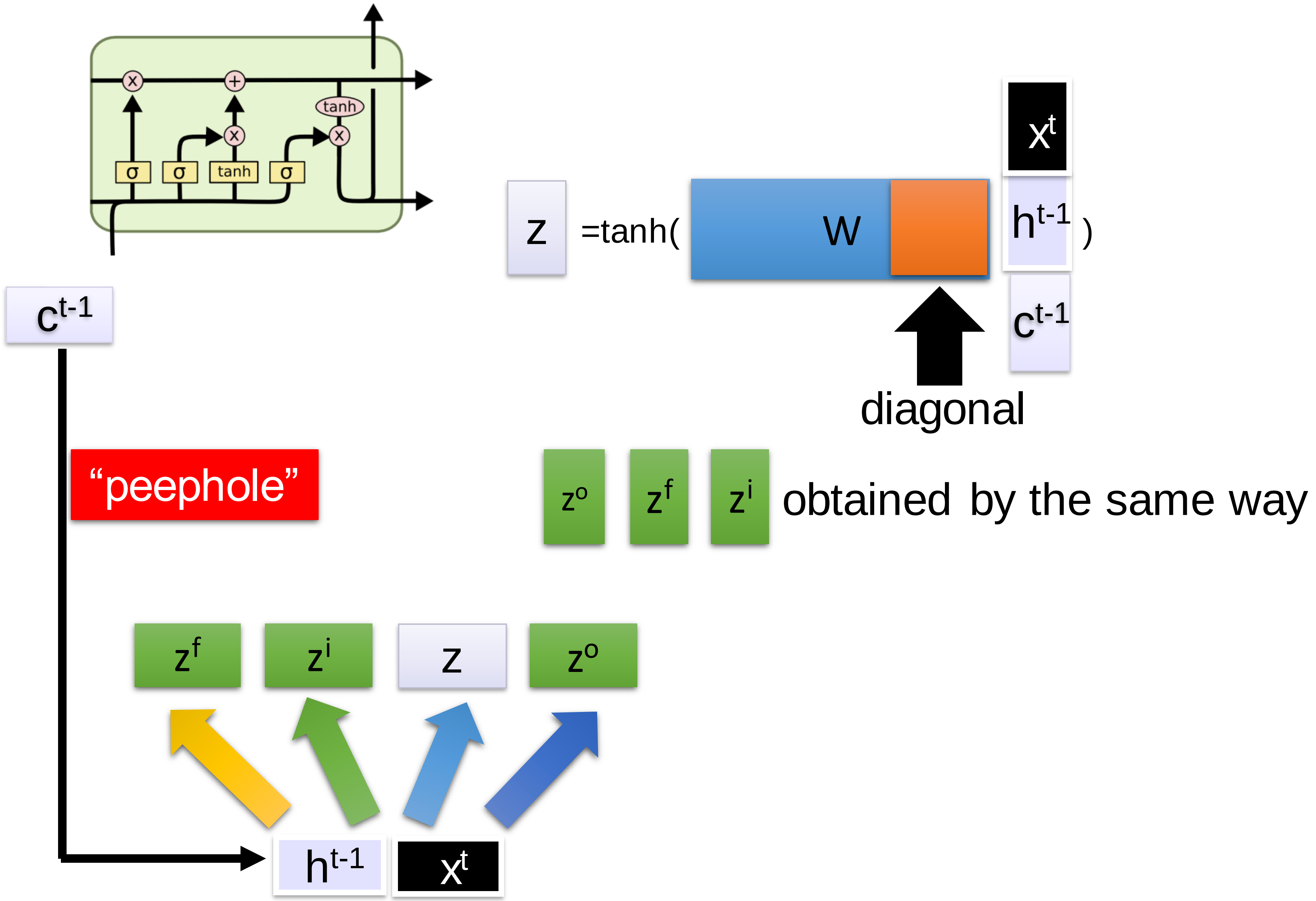
$$z^i = \sigma\left(W^i \begin{bmatrix} x^t \\ h^{t-1} \end{bmatrix} \right)$$

$$z^f = \sigma\left(W^f \begin{bmatrix} x^t \\ h^{t-1} \end{bmatrix} \right)$$

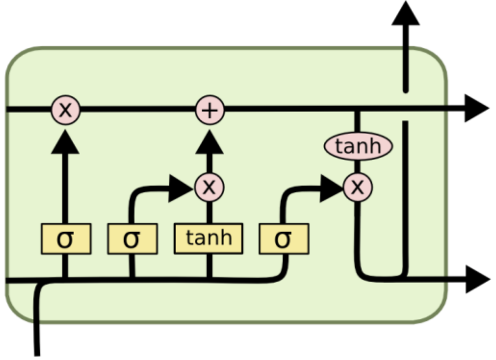
$$z^o = \sigma\left(W^o \begin{bmatrix} x^t \\ h^{t-1} \end{bmatrix} \right)$$



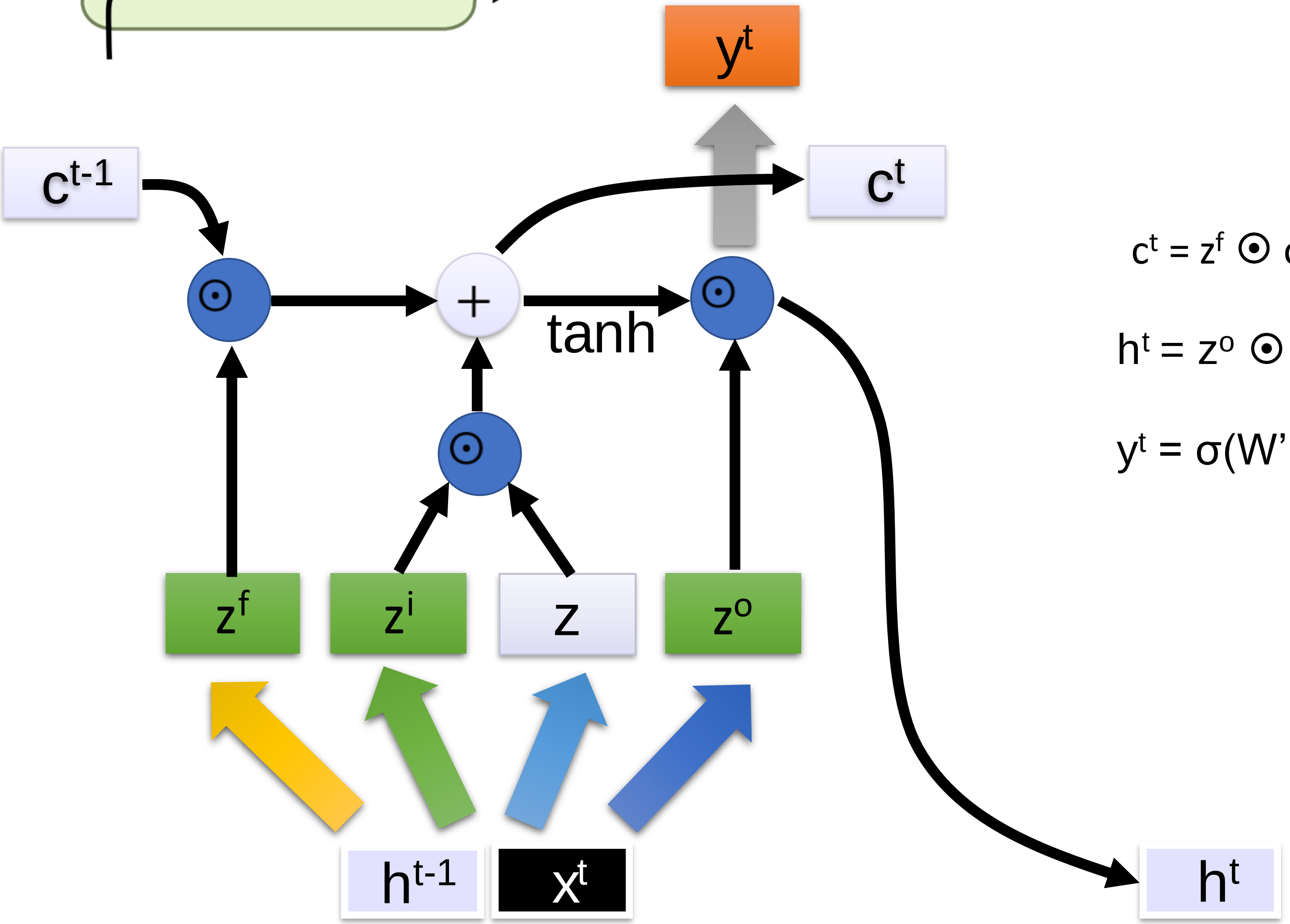
Information flow of LSTM



Information flow of LSTM



⊙ Element-wise multiply



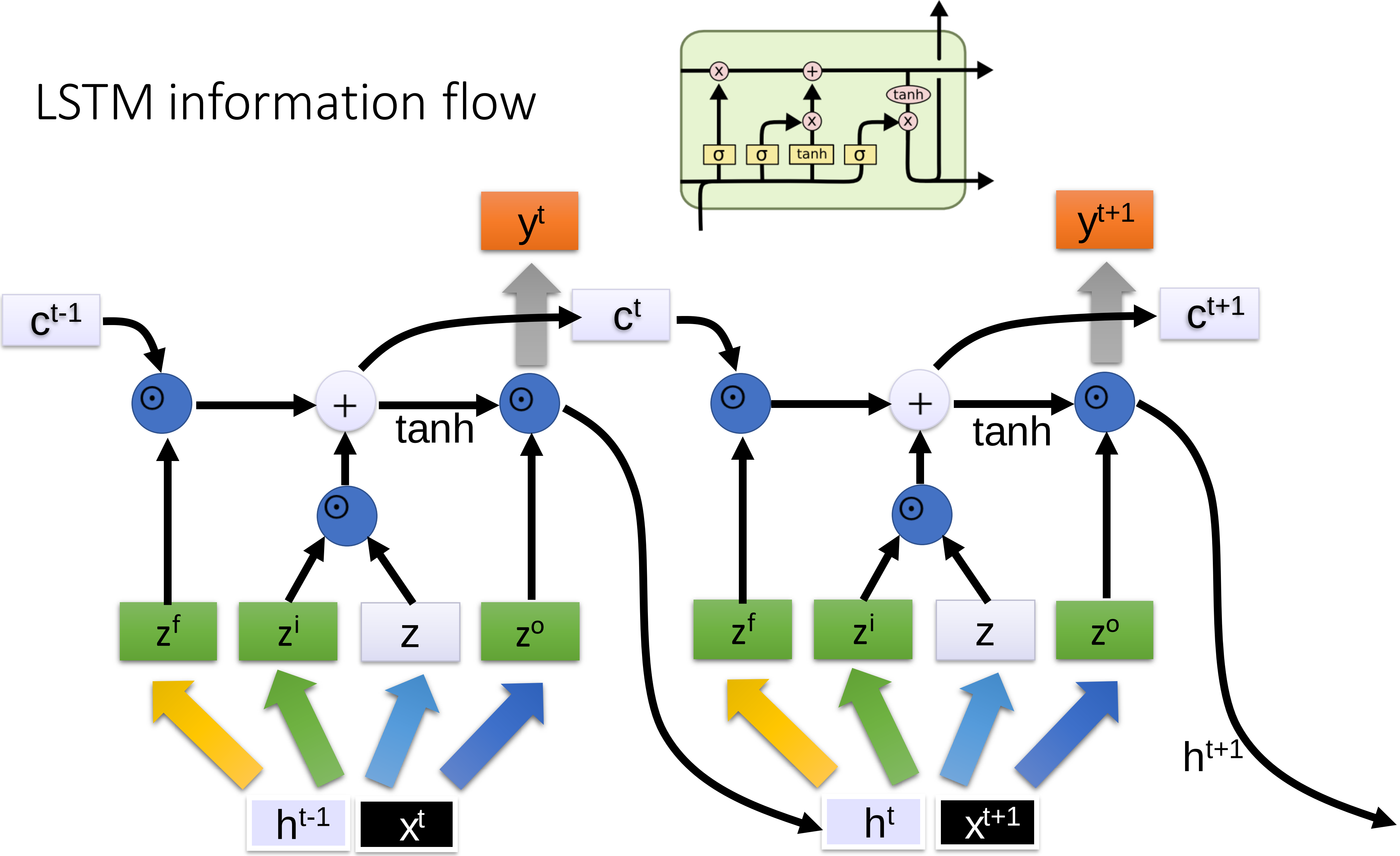
$$c^t = z^f \odot c^{t-1} + z^i \odot z$$

$$h^t = z^o \odot \tanh(c^t)$$

$$y^t = \sigma(W' h^t)$$

Information flow of LSTM

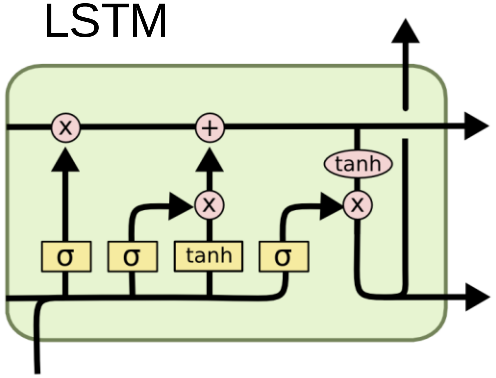
LSTM information flow

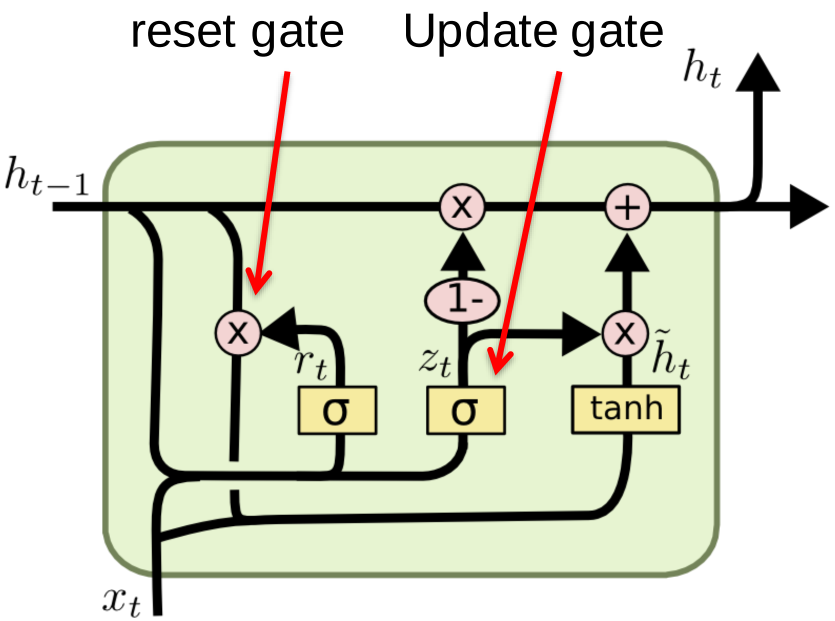


Information flow of LSTM

GRU – gated recurrent unit

(more compression)





$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$

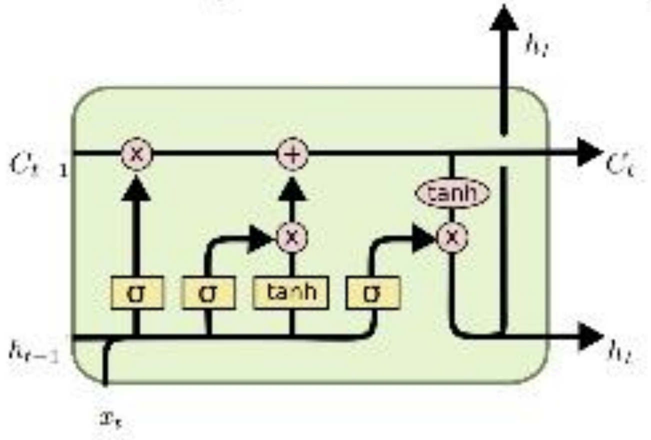
$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

It combines the forget and input into a single update gate.
It also merges the cell state and hidden state. This is simpler than LSTM. There are many other variants too.

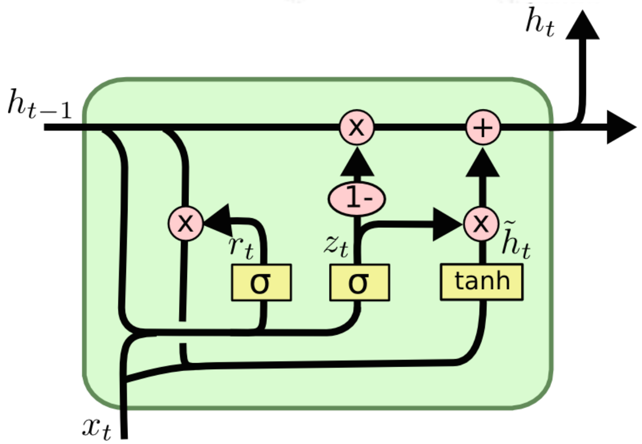
X,*: element-wise multiply

LSTM and GRU

- LSTM [Hochreiter&Schmidhuber97]



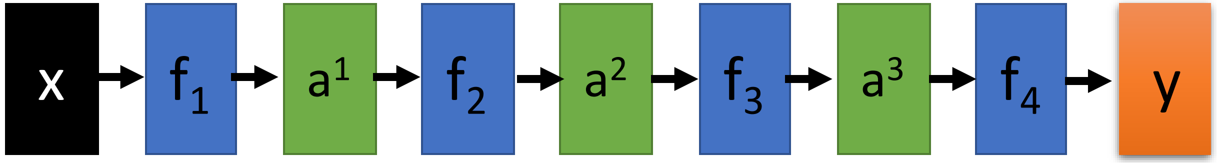
- GRU [Cho+14]



GRUs also takes x_t and h_{t-1} as inputs. They perform some calculations and then pass along h_t. What makes them different from LSTMs is that GRUs don't need the cell layer to pass values along. The calculations within each iteration insure that the h_t values being passed along either retain a high amount of old information or are jump-started with a high amount of new information.

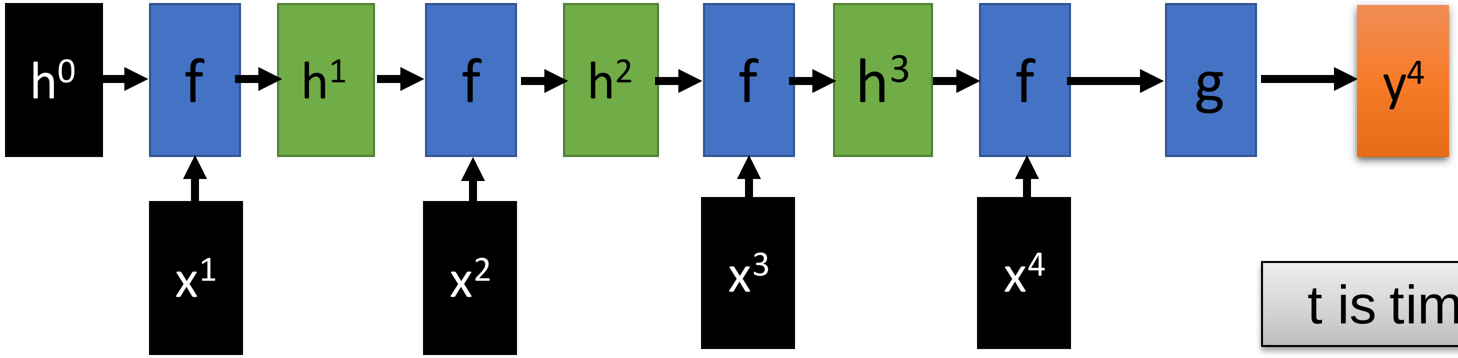
Feed-forward vs Recurrent Network

1. Feedforward network does not have input at each step
2. Feedforward network has different parameters for each layer



t is layer

$$a^t = f_t(a^{t-1}) = \sigma(W^t a^{t-1} + b^t)$$



t is time step

$$a^t = f(a^{t-1}, x^t) = \sigma(W^h a^{t-1} + W^i x^t + b^i)$$

We will turn the recurrent network 90 degrees.

GRU → Highway Network

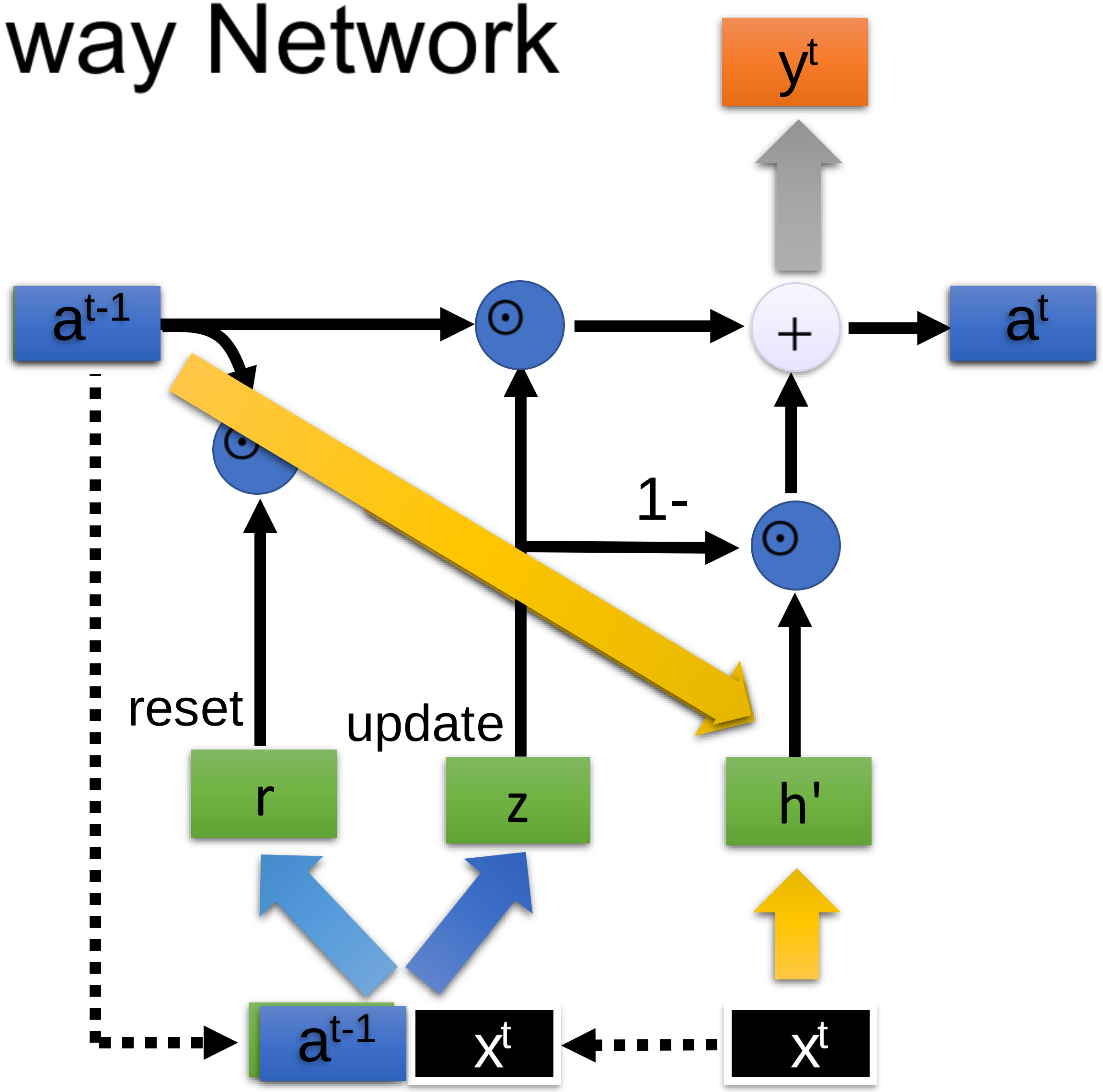
No input x^t at each step

No output y^t at each step

a^{t-1} is the output of the (t-1)-th layer

a^t is the output of the t-th layer

No reset gate



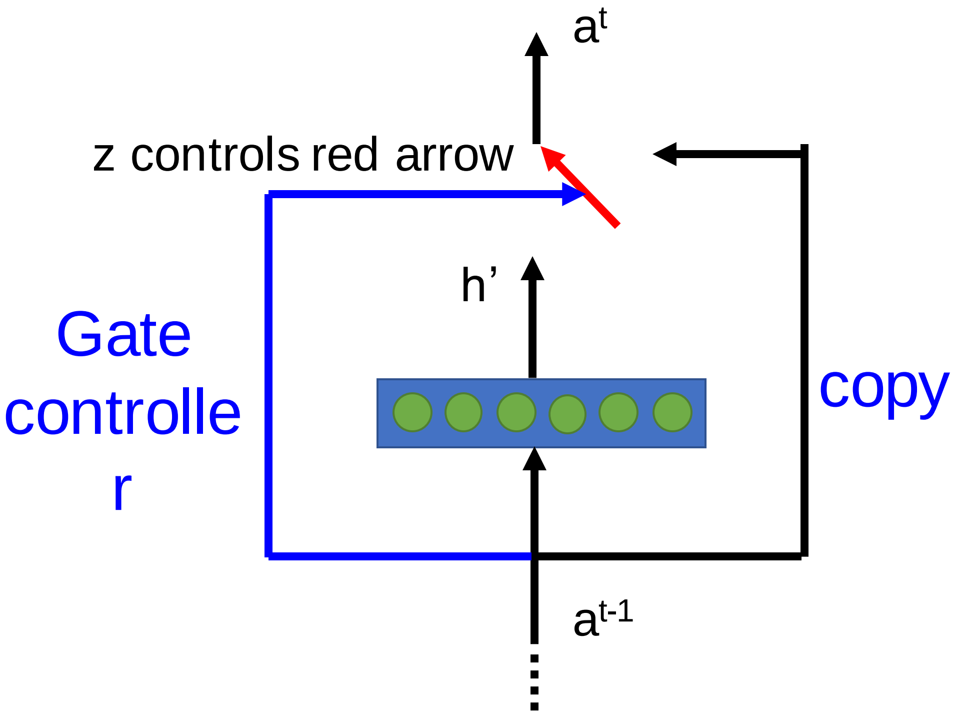
Highway Network

$h' = \sigma(Wa^{t-1})$

$z = \sigma(W'a^{t-1})$

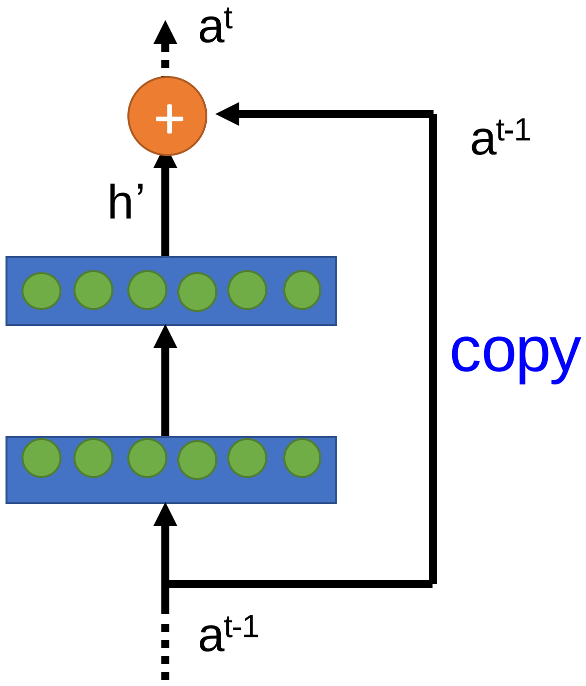
$a^t = z \odot a^{t-1} + (1-z) \odot h$

- **Highway Network**

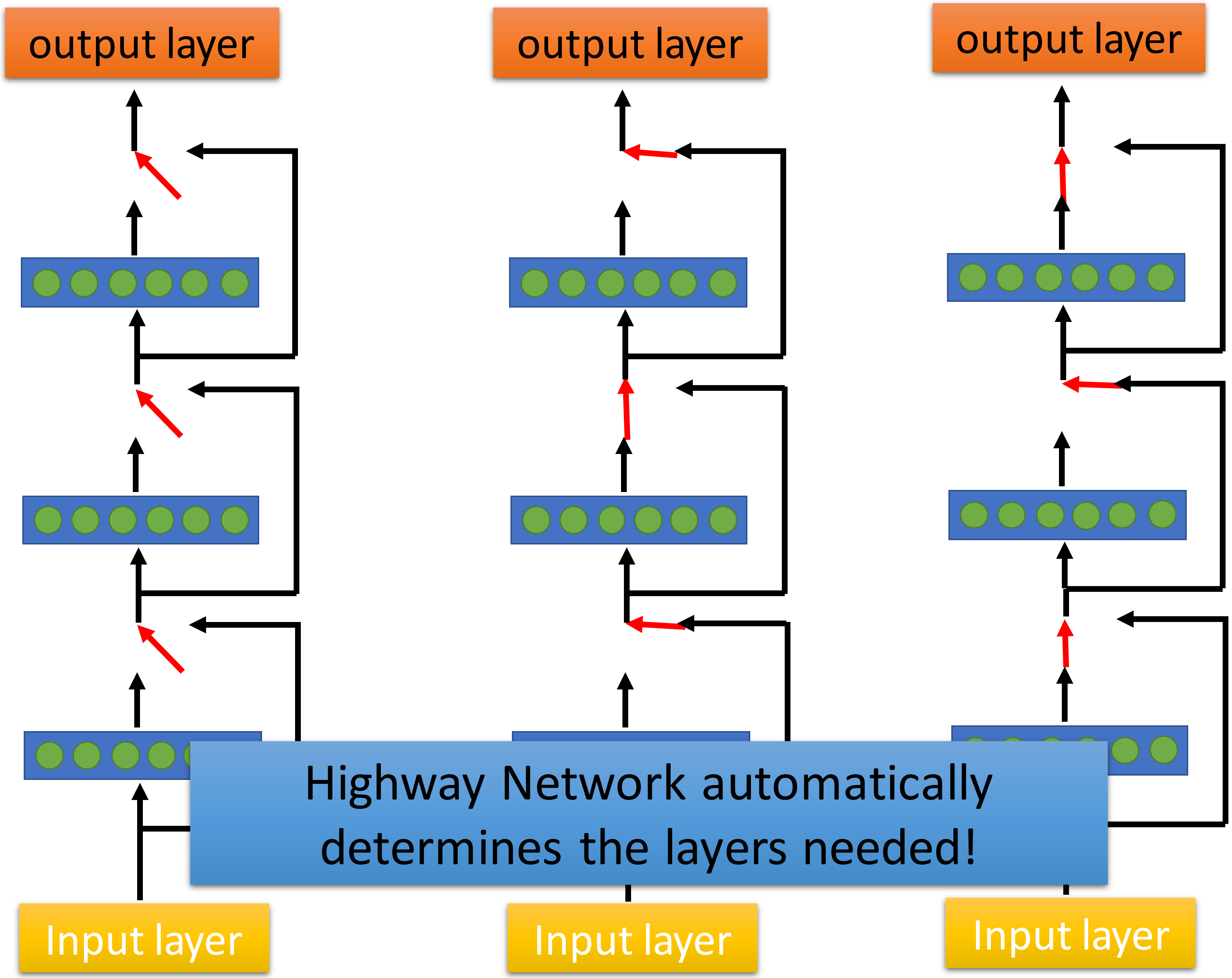


Training Very Deep Networks https://arxiv.org/pdf/1507.06228v2.pdf

- **Residual Network**

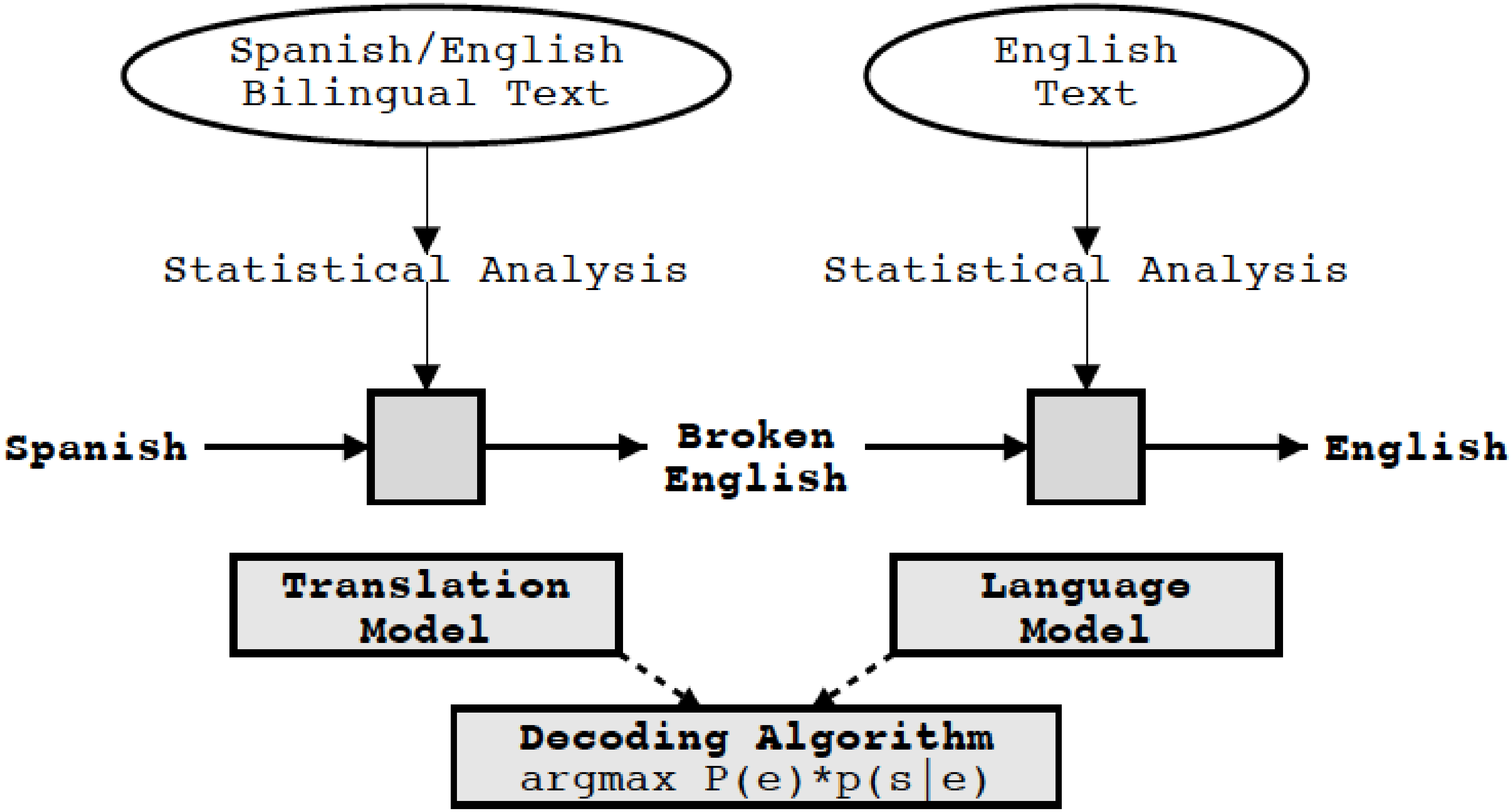


Deep Residual Learning for Image Recognition http://arxiv.org/abs/1512.03385



Highway Network automatically determines the layers needed!

Statistical Machine Translation

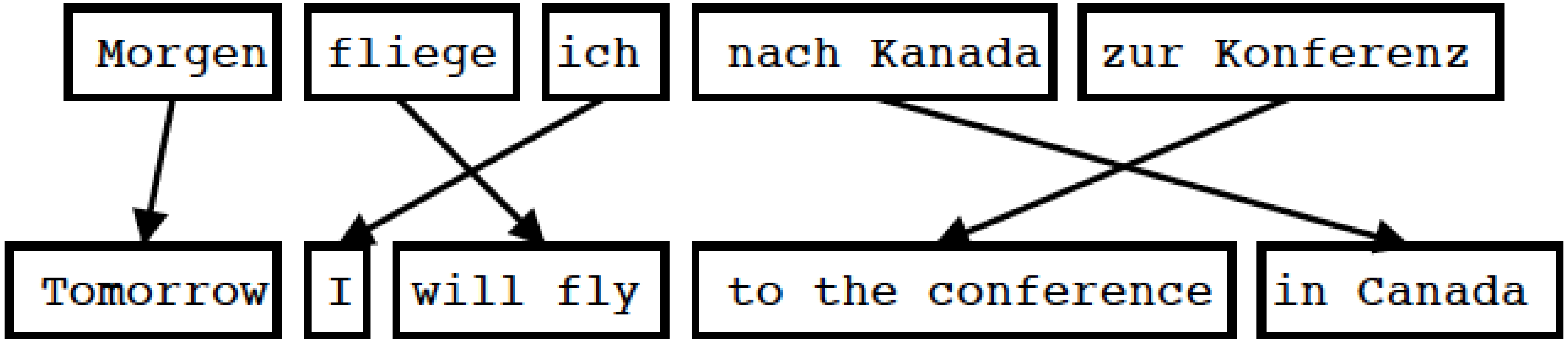


Knight and Koehn 2003

Statistical Machine Translation

- Translation model

- Input is Segmented in Phrases
- Each Phrase is Translated into English
- Phra



Koehn 2004

Statistical Machine Translation

- Language Model

Goal of the Language Model: Detect good English ***P****(e)*

Standard Te

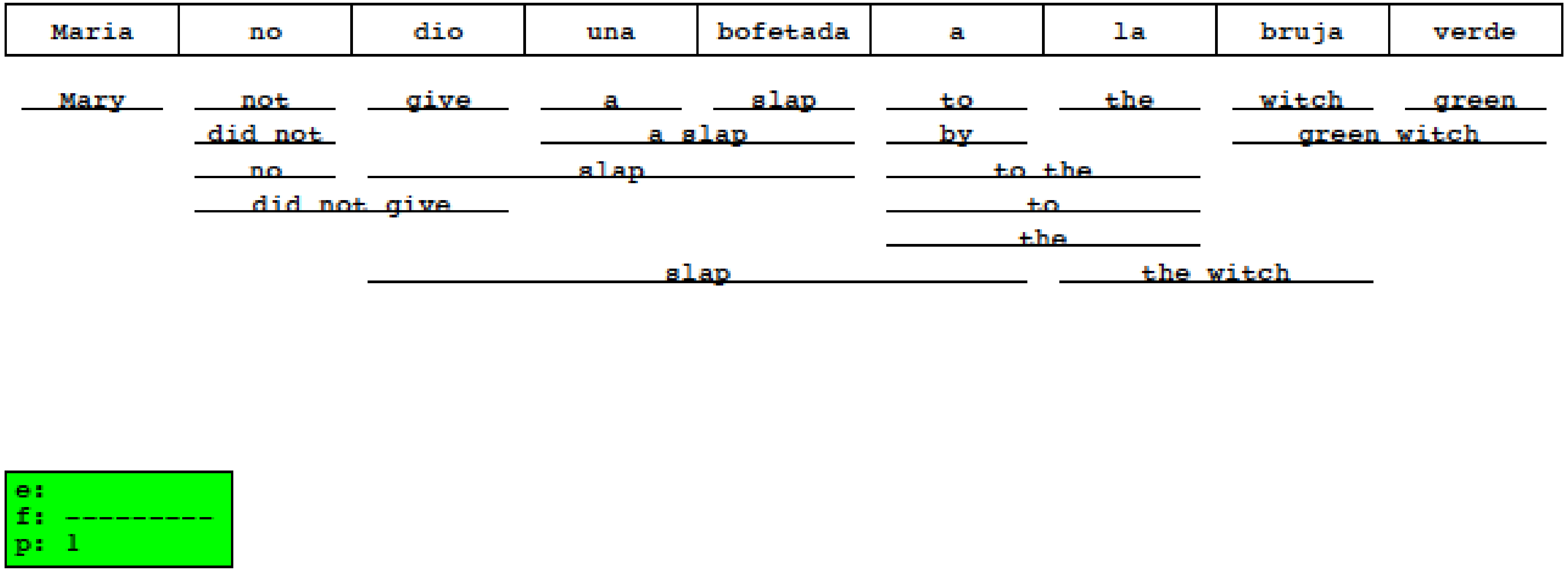
```
Mary did not slap the green witch

Mary  =>  p(Mary)
Mary did   =>  p(did|Mary)
Mary did not  =>  p(not|Mary did)
     did not slap  =>  p(slap|did not)
         not slap the  =>  p(the|not slap)
             slap the green  =>  p(green|slap the)
                  the green witch  =>  p(witch|the green)
```

Knight and Koehn 2003

Statistical Machine Translation

- Decoding

Goal of the decoding algorithm: Put models to work, perform the actual translation

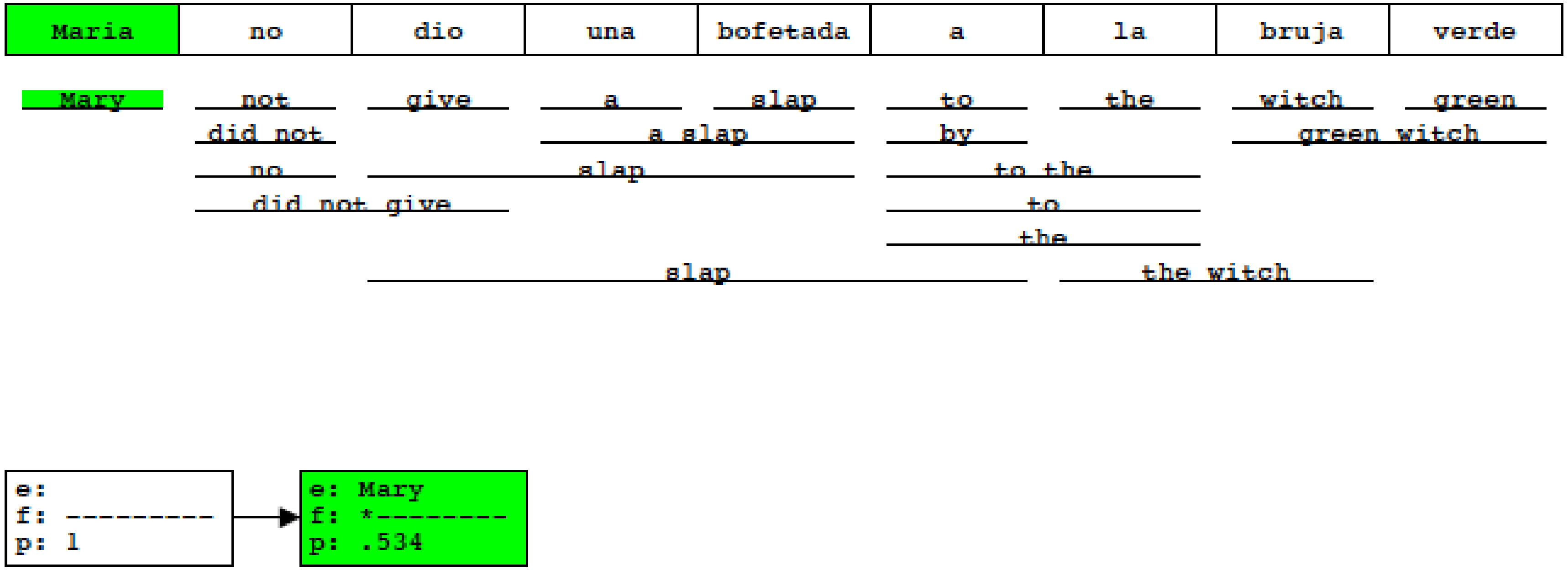


Koehn 2004

Statistical Machine Translation

- Decoding

Goal of the decoding algorithm: Put models to work, perform the actual translation

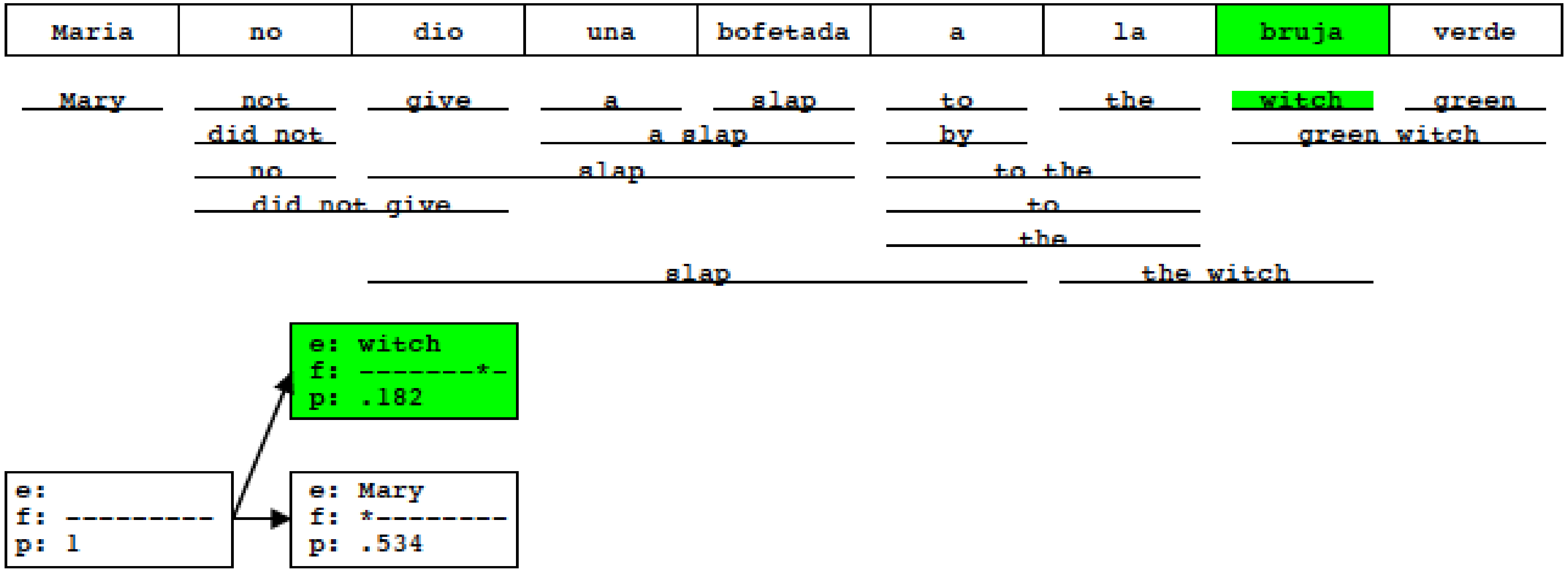


Koehn 2004

Statistical Machine Translation

- Decoding

Goal of the decoding algorithm: Put models to work, perform the actual translation

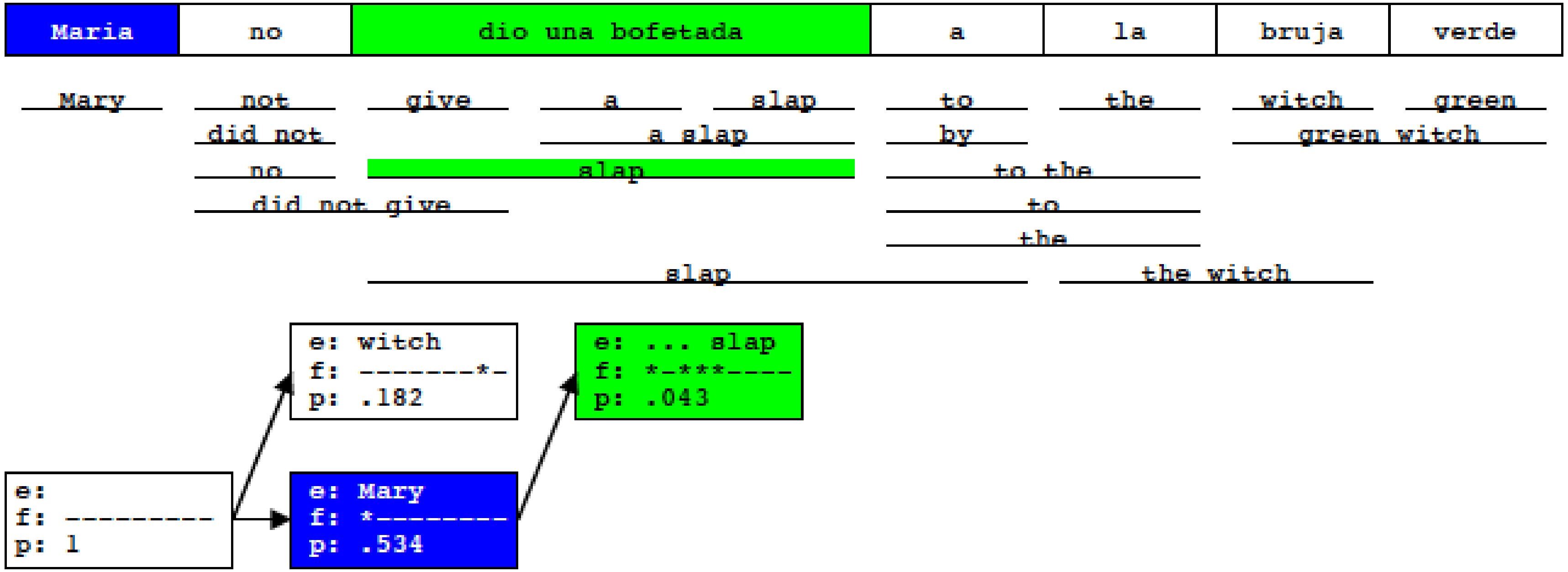


Koehn 2004

Statistical Machine Translation

- Decoding

Goal of the decoding algorithm: Put models to work, perform the actual translation

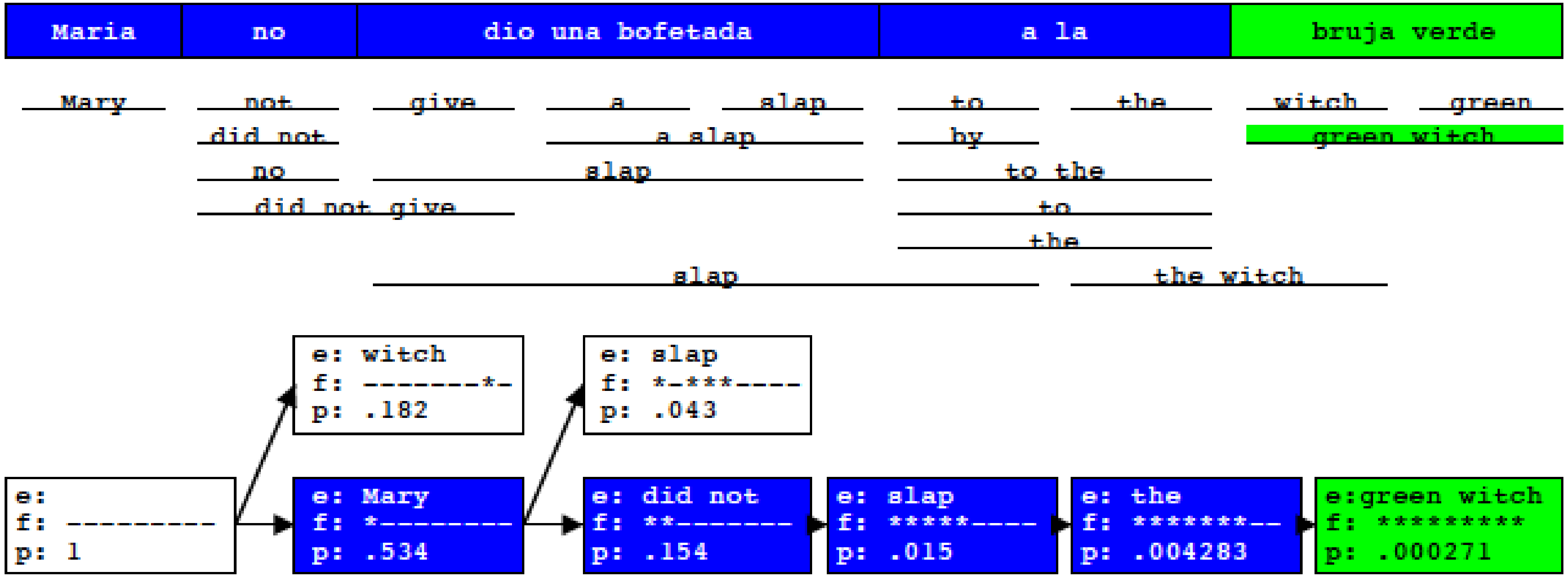


Koehn 2004

Statistical Machine Translation

- Decoding

Goal of the decoding algorithm: Put models to work, perform the actual translation

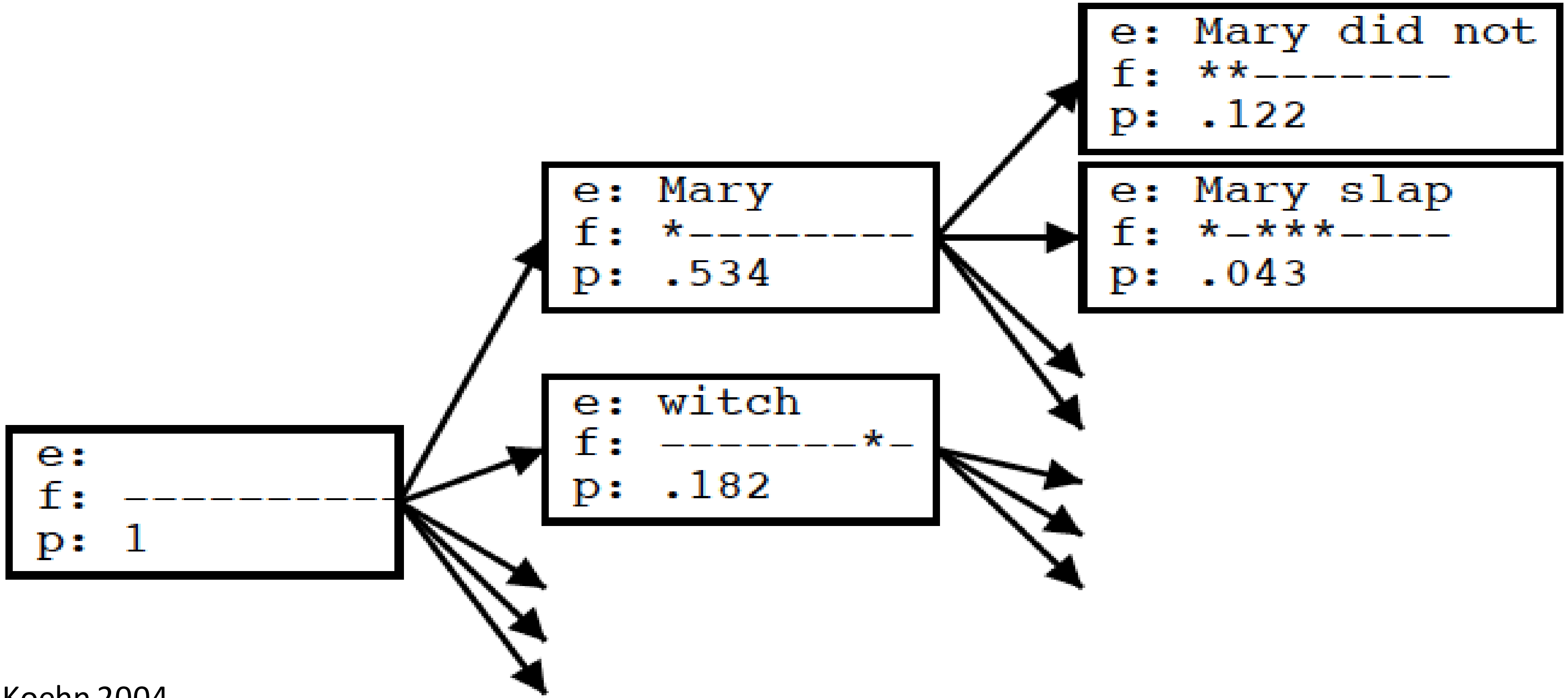


Koehn 2004

Statistical Machine Translation

- Decoding

Goal of the decoding algorithm: Put models to work, perform the actual translation



Koehn 2004

Statistical Machine Translation

- Decoding

Goal of the decoding algorithm: Put models to work, perform the actual translation

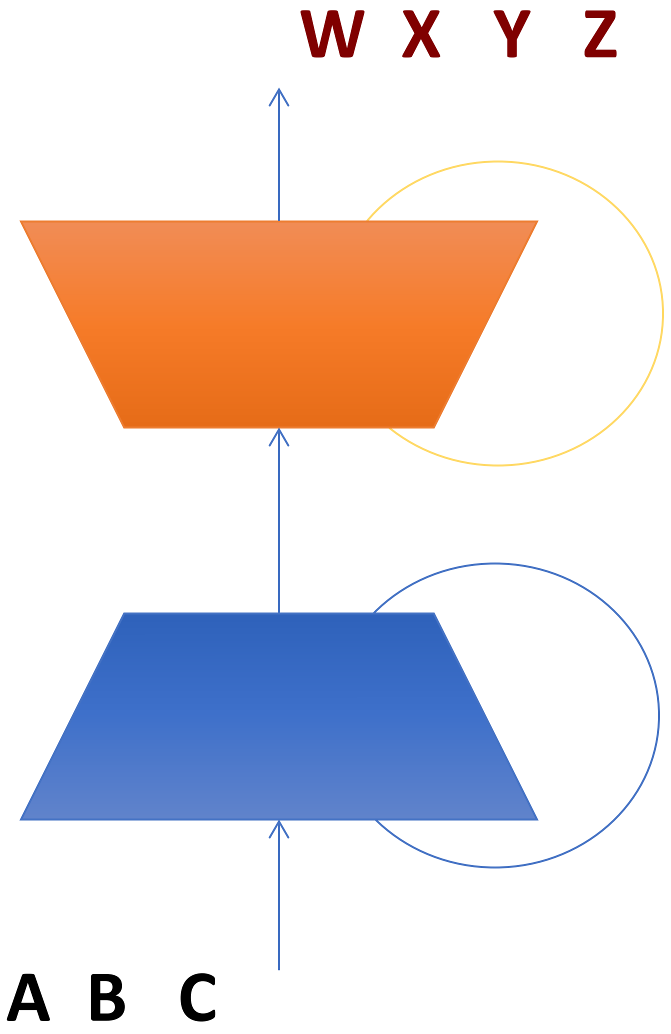
- Prune out Weakest Hypotheses
  - by absolute threshold (keep 100 best)
  - by relative cutoff
- Future Cost Estimation
  - compute expected cost of untranslated words

Sutskever et al.,2014

Sequence to Sequence Learning with Neural Networks

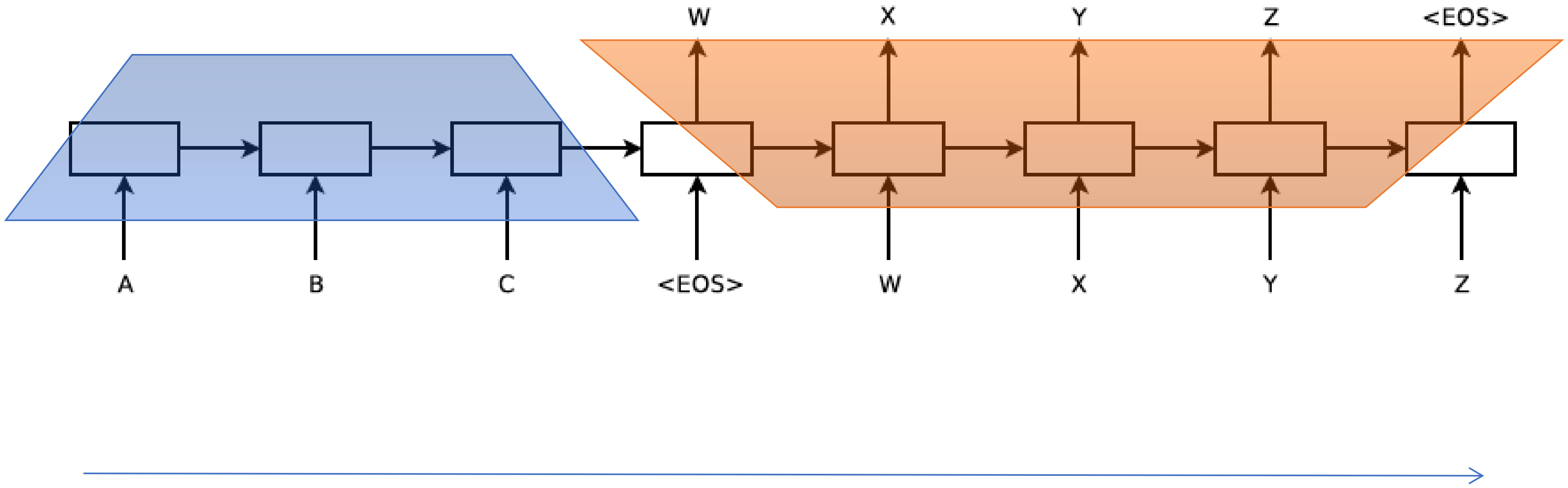
Neural Machine Translation

- Model



Neural Machine Translation

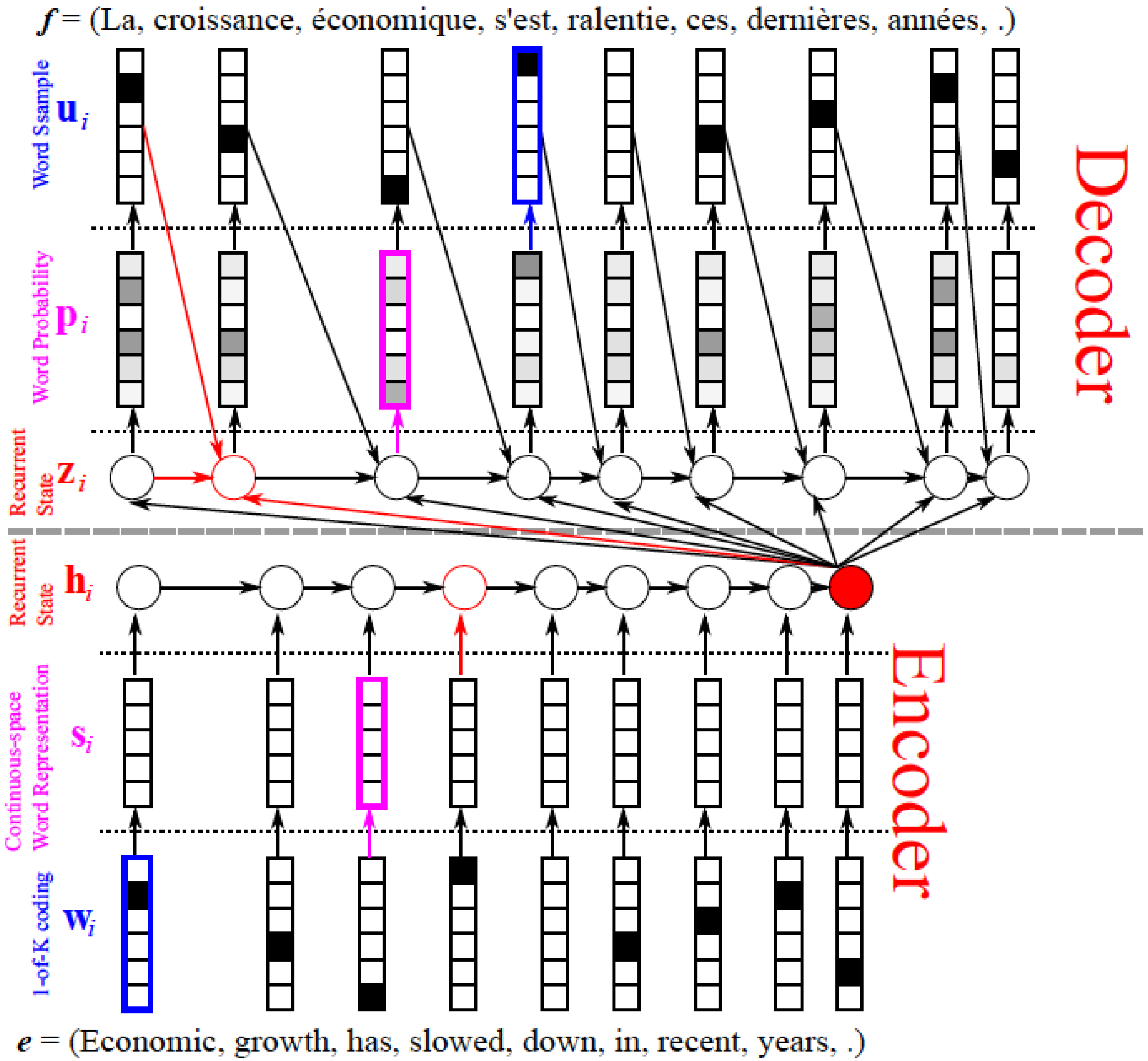
- Model



Sutskever et al. 2014

Neural Machine Translation

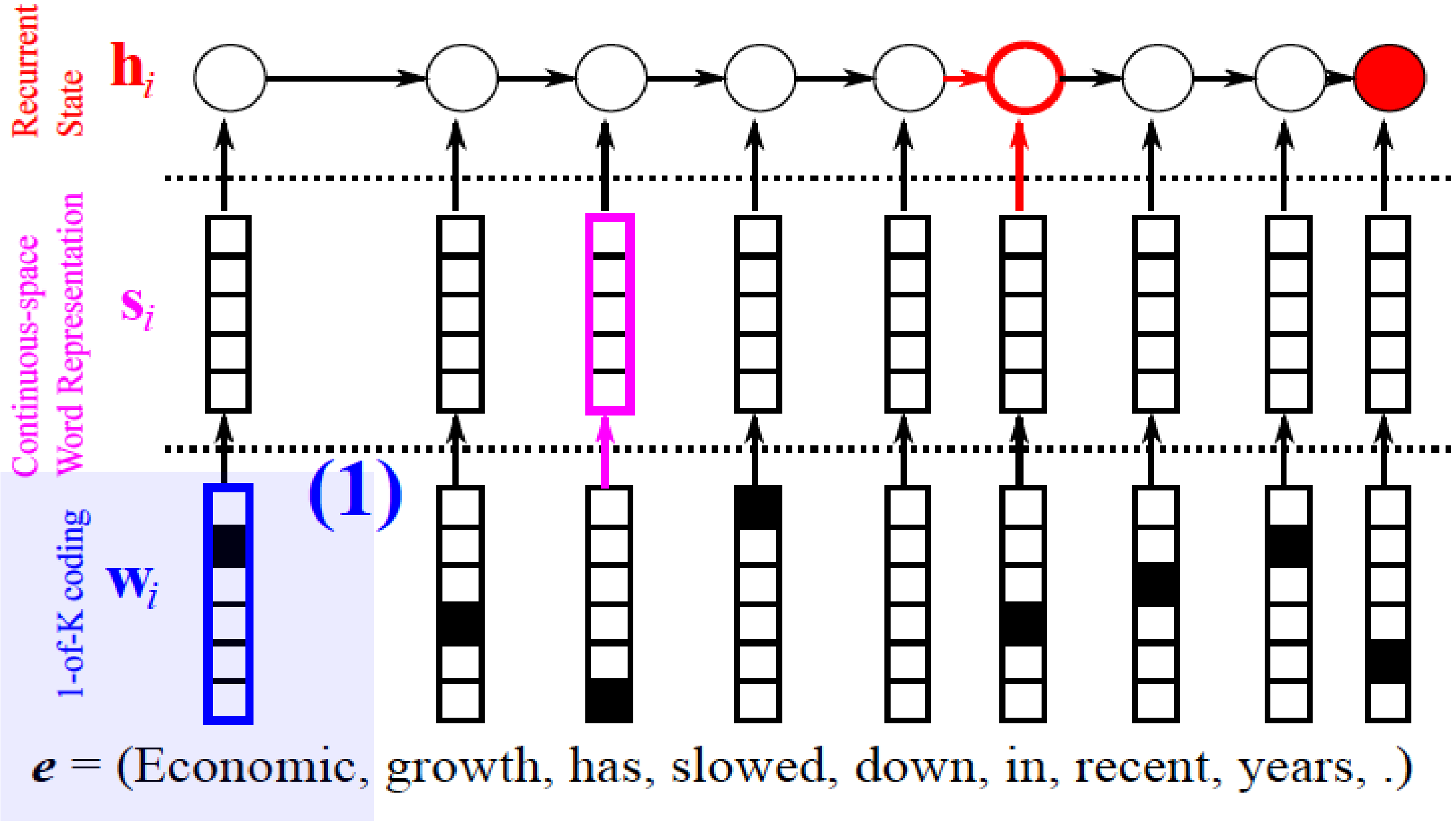
- Model-



Cho: From Sequence Modeling to Translation

Neural Machine Translation

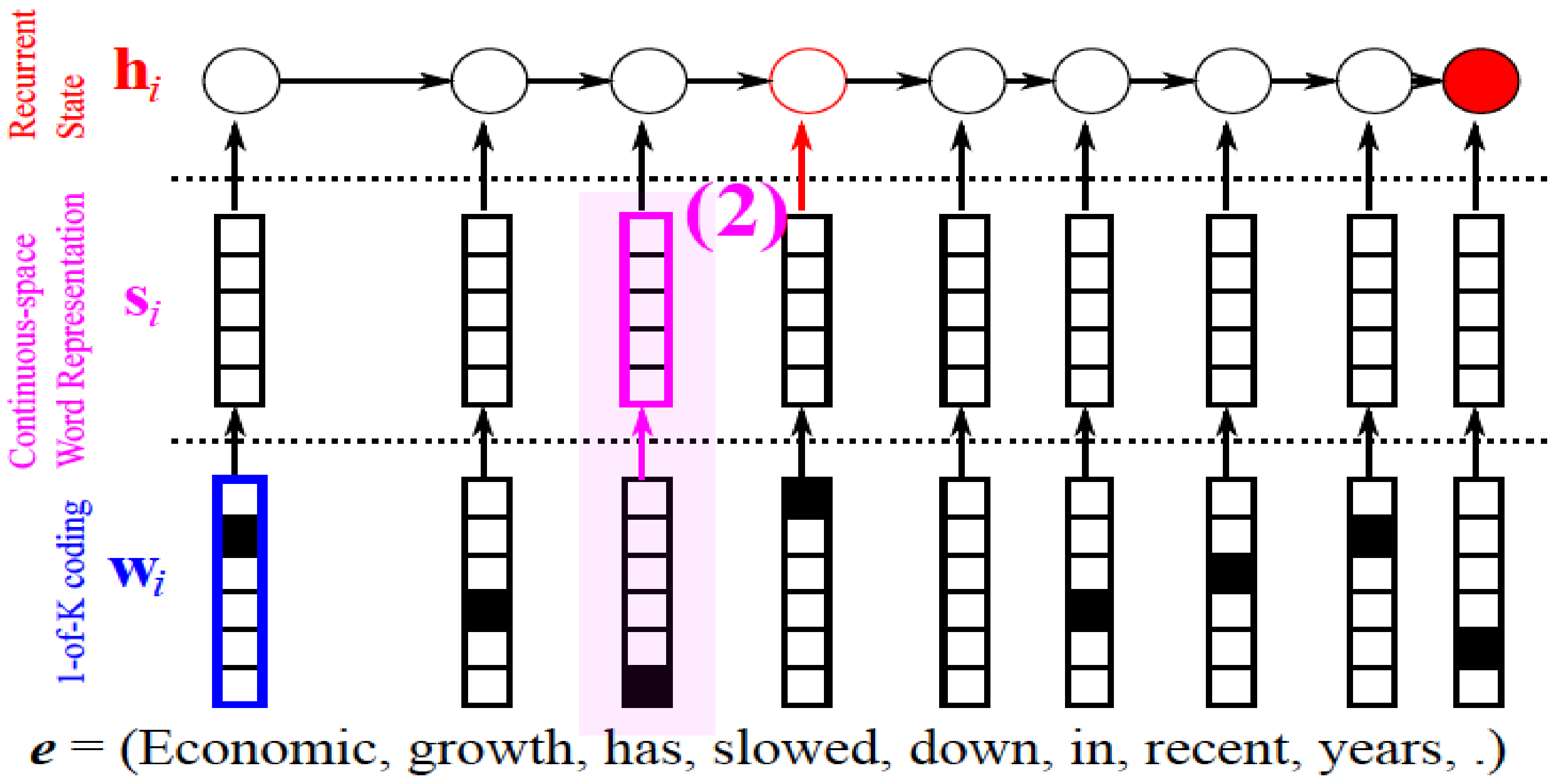
- Model- *encoder*



Cho: From Sequence Modeling to Translation

Neural Machine Translation

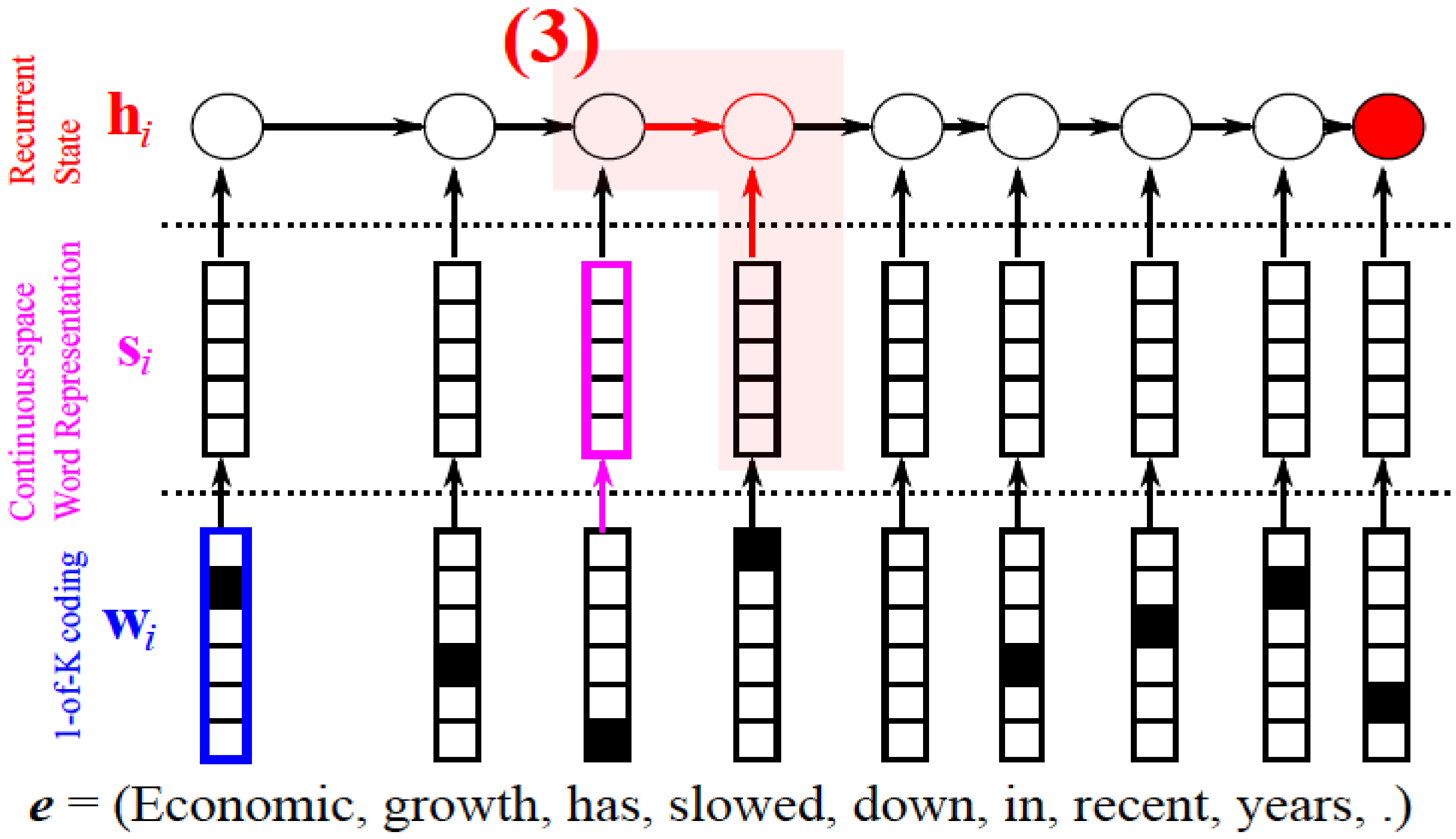
- Model- *encoder*



Cho: From Sequence Modeling to Translation

Neural Machine Translation

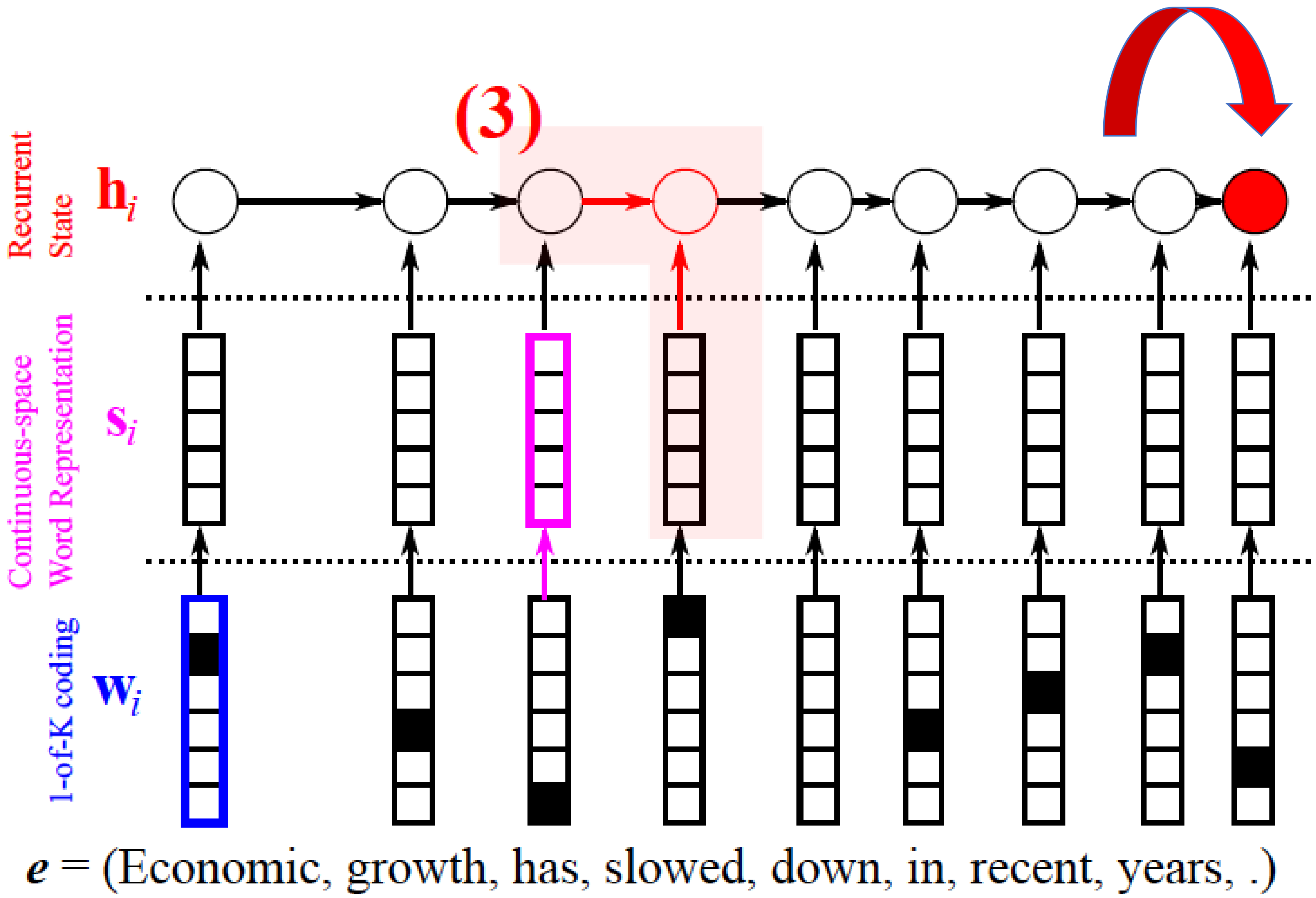
- Model- *encoder*



Cho: From Sequence Modeling to Translation

Neural Machine Translation

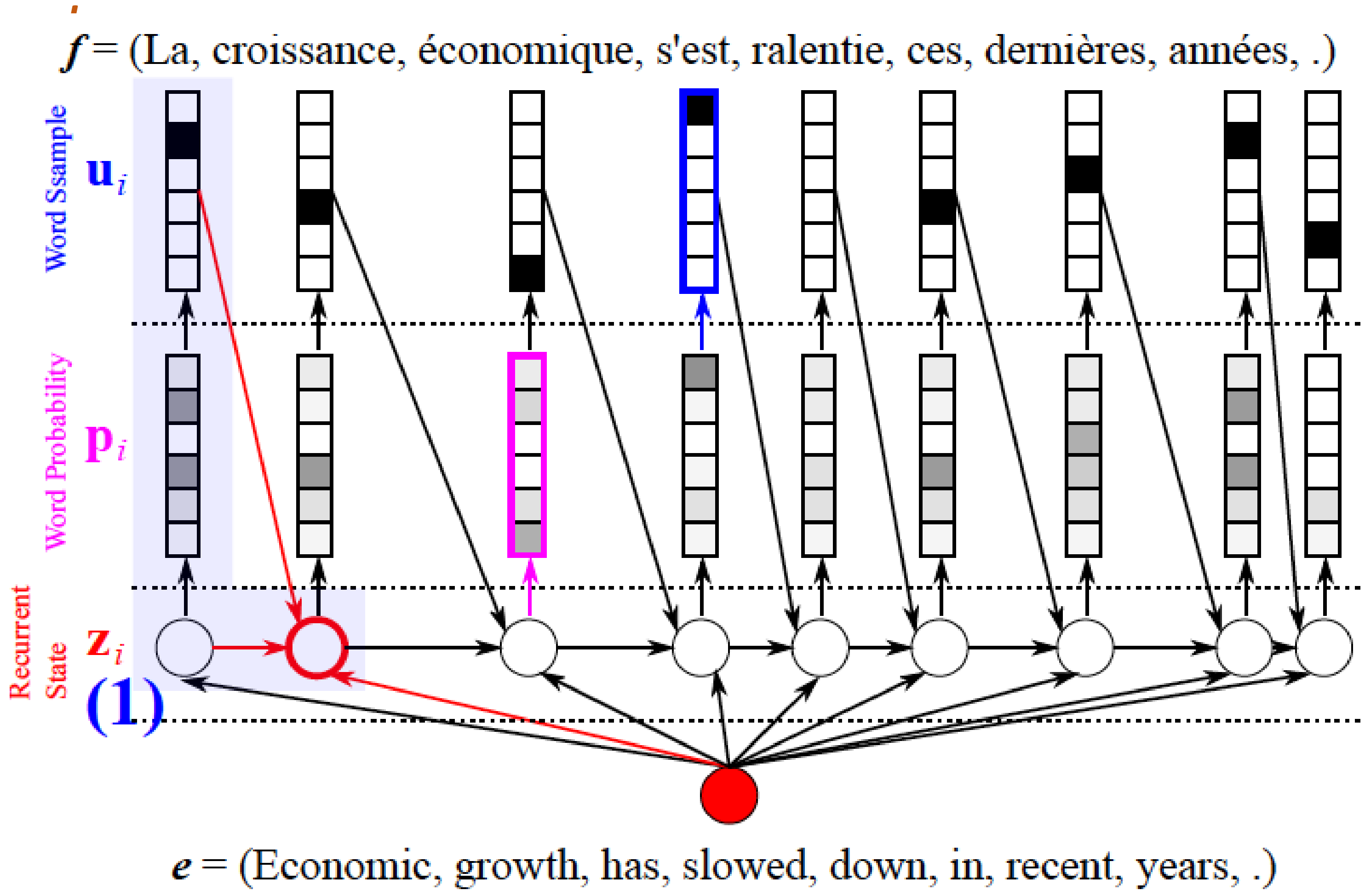
- Model-



Cho: From Sequence Modeling to Translation

Neural Machine Translation

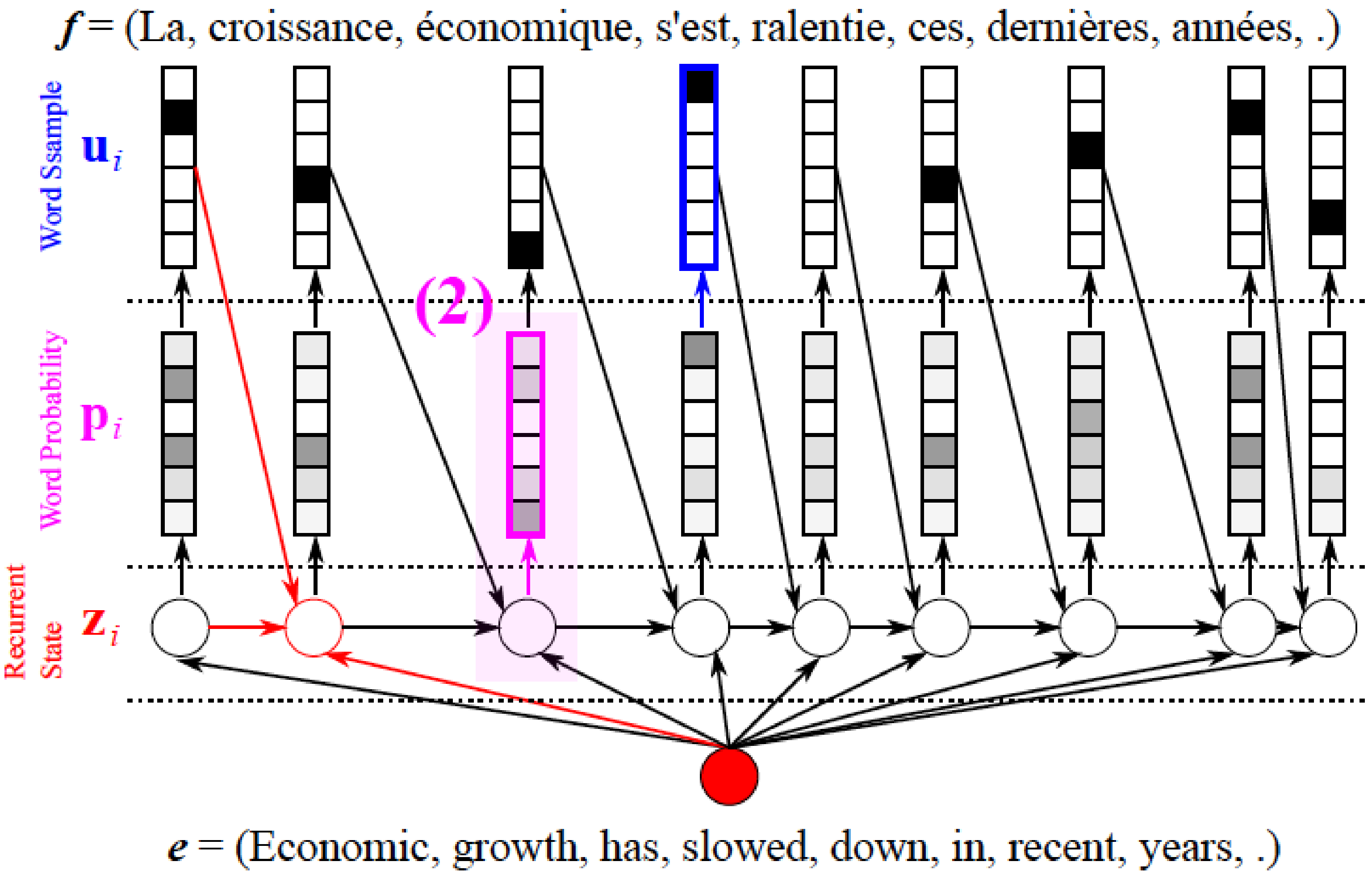
- Model- *de*



Cho: From Sequence Modeling to Translation

Neural Machine Translation

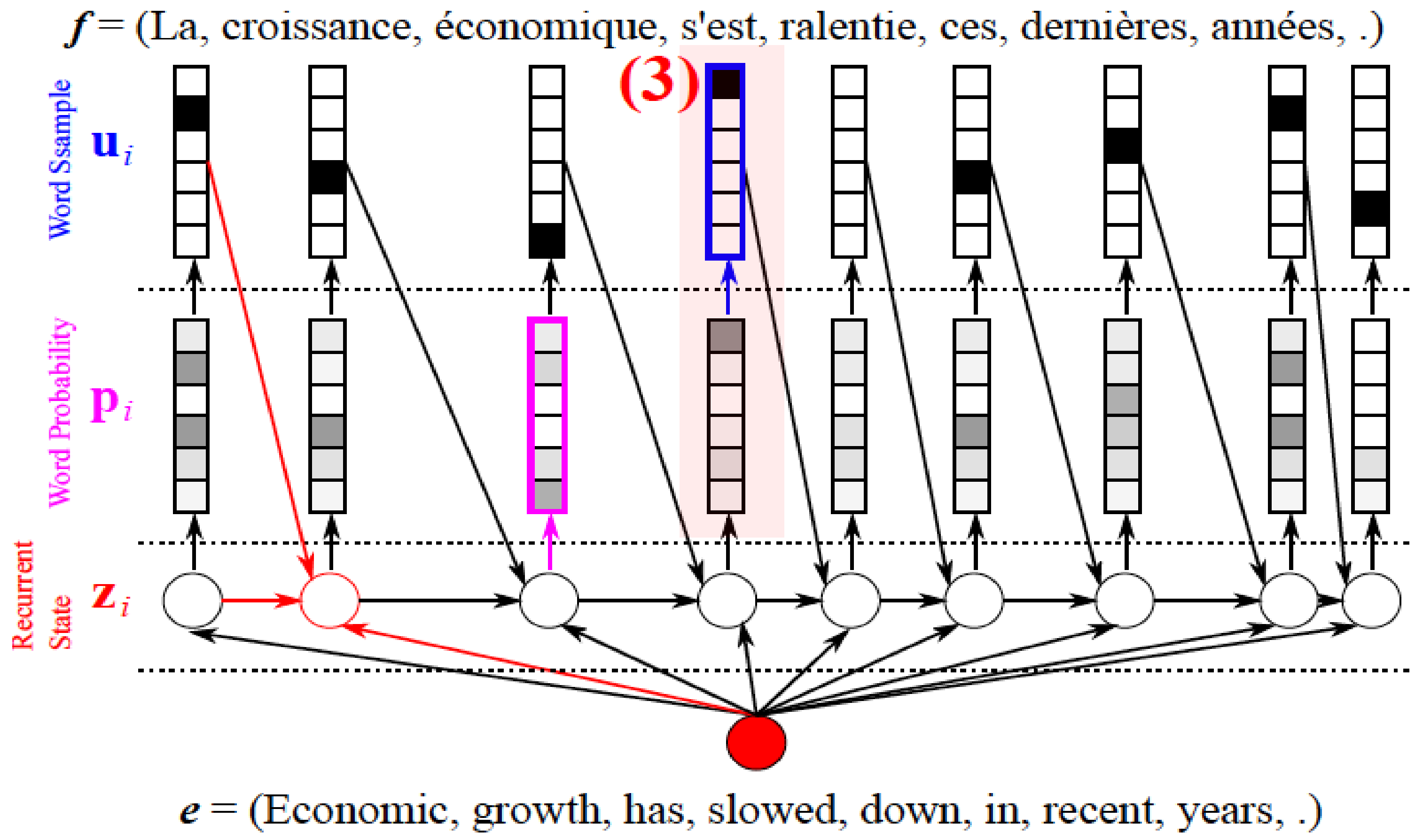
- Model- *de*



Cho: From Sequence Modeling to Translation

Neural Machine Translation

- Model- *decoder*



Cho: From Sequence Modeling to Translation

Neural Machine Translation

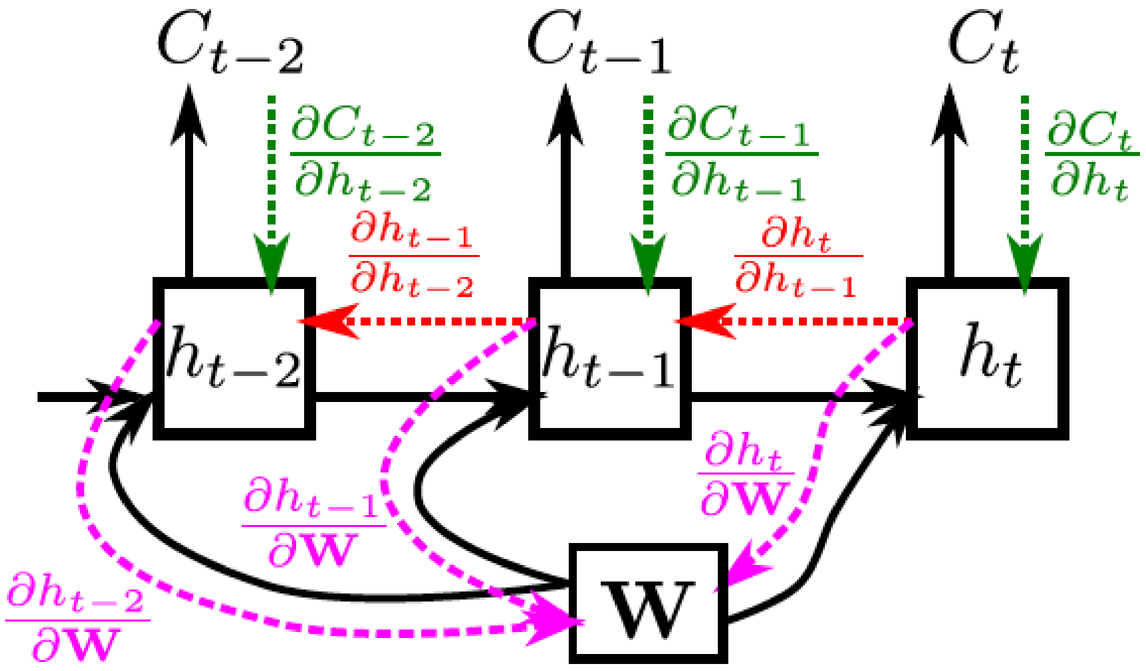
- RNN

$$
\begin{aligned}
h_t &= \operatorname{sigm}\left(W^{\mathrm{hx}} x_t + W^{\mathrm{hh}} h_{t-1}\right) \\
y_t &= W^{\mathrm{yh}} h_t
\end{aligned}
$$

Neural Machine Translation

- RNN

Vanishing gradient

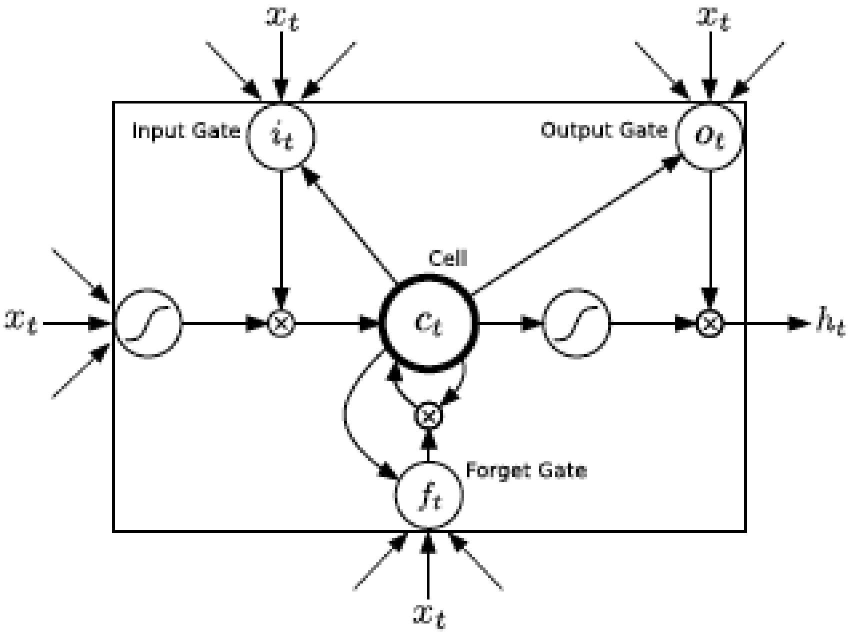


$$\frac{\partial C_t}{\partial \mathbf{W}} = \sum_{t'=1}^{t} \frac{\partial C_t}{\partial h_t} \frac{\partial h_t}{\partial h_{t'}} \frac{\partial h_{t'}}{\partial \mathbf{W}}, \text{ where } \frac{\partial h_t}{\partial h_{t'}} = \prod_{k=t'+1}^{t} \frac{\partial h_k}{\partial h_{k-1}}$$

Cho: From Sequence Modeling to Translation

Neural Machine Translation

- LSTM



Graves 2013

Neural Machine Translation

- LSTM

Problem: Exploding gradient

Neural Machine Translation

- LSTM

Problem: Exploding gradient

Solution: Scaling gradient

Sequence to Sequence

- Results

BLEU score (Bilingual Evaluation Understudy)

Candidate	the	the	the	the	the	the	the
Reference 1	the	cat	is	on	the	mat	
Reference 2	there	is	a	cat	on	the	mat

P = m/w= 7/7 = 1

Papineni et al. 2002

Sequence to Sequence

- Results

BLEU score (Bilingual Evaluation Understudy)

Candidate	the	the	the	the	the	the	the
Reference 1	the	cat	is	on	the	mat	
Reference 2	there	is	a	cat	on	the	mat

P = 2/7

Papineni et al. 2002

Sequence to Sequence

- Results

Method	test BLEU score (ntst14)
Bahdanau et al. [2]	28.45
Baseline System [29]	33.30
Single forward LSTM, beam size 12	26.17
Single reversed LSTM, beam size 12	30.59
Ensemble of 5 reversed LSTMs, beam size 1	33.00
Ensemble of 2 reversed LSTMs, beam size 12	33.27
Ensemble of 5 reversed LSTMs, beam size 2	34.50
Ensemble of 5 reversed LSTMs, beam size 12	**34.81**

Sutskever et al. 2014

Sequence to Sequence

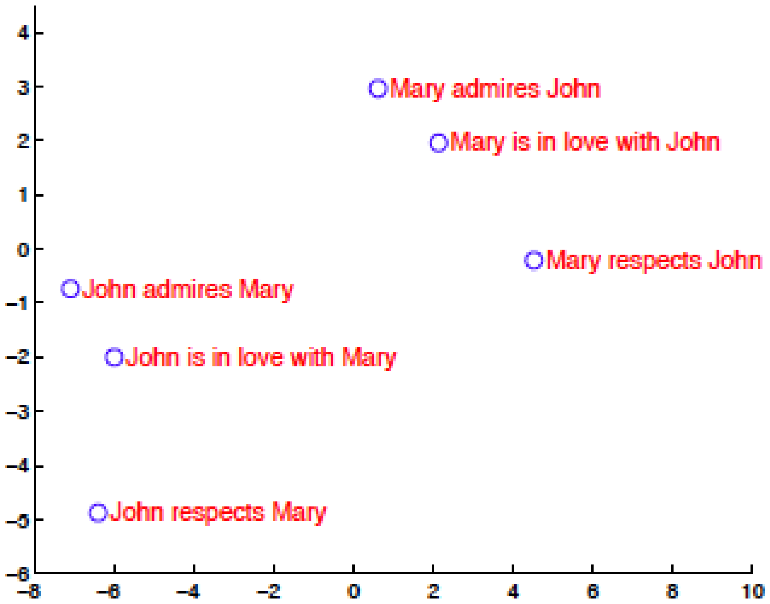
- Results

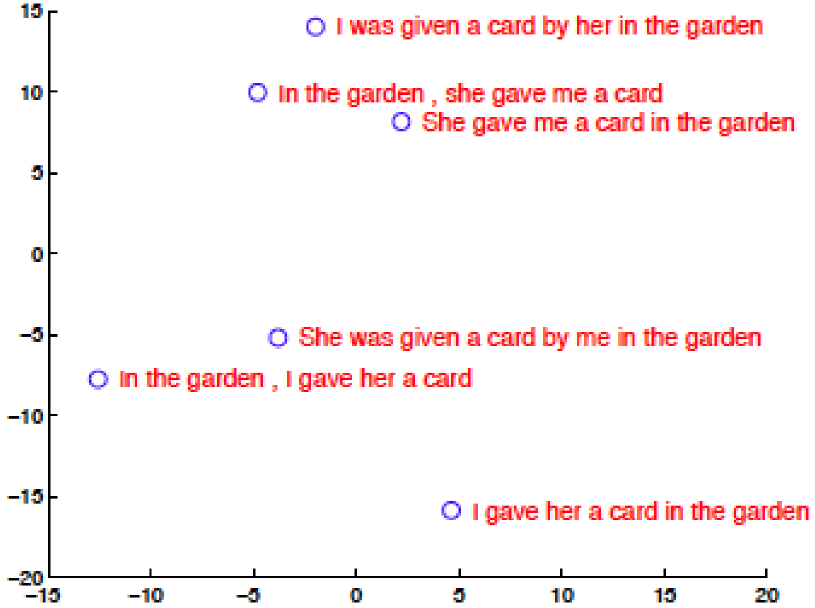
Method	test BLEU score (ntst14)
Baseline System [29]	33.30
Cho et al. [5]	34.54
Best WMT'14 result [9]	**37.0**
Rescoring the baseline 1000-best with a single forward LSTM	35.61
Rescoring the baseline 1000-best with a single reversed LSTM	35.85
Rescoring the baseline 1000-best with an ensemble of 5 reversed LSTMs	**36.5**
Oracle Rescoring of the Baseline 1000-best lists	$\sim$45

Sutskever et al. 2014

Sequence to Sequence

- Model Analysis

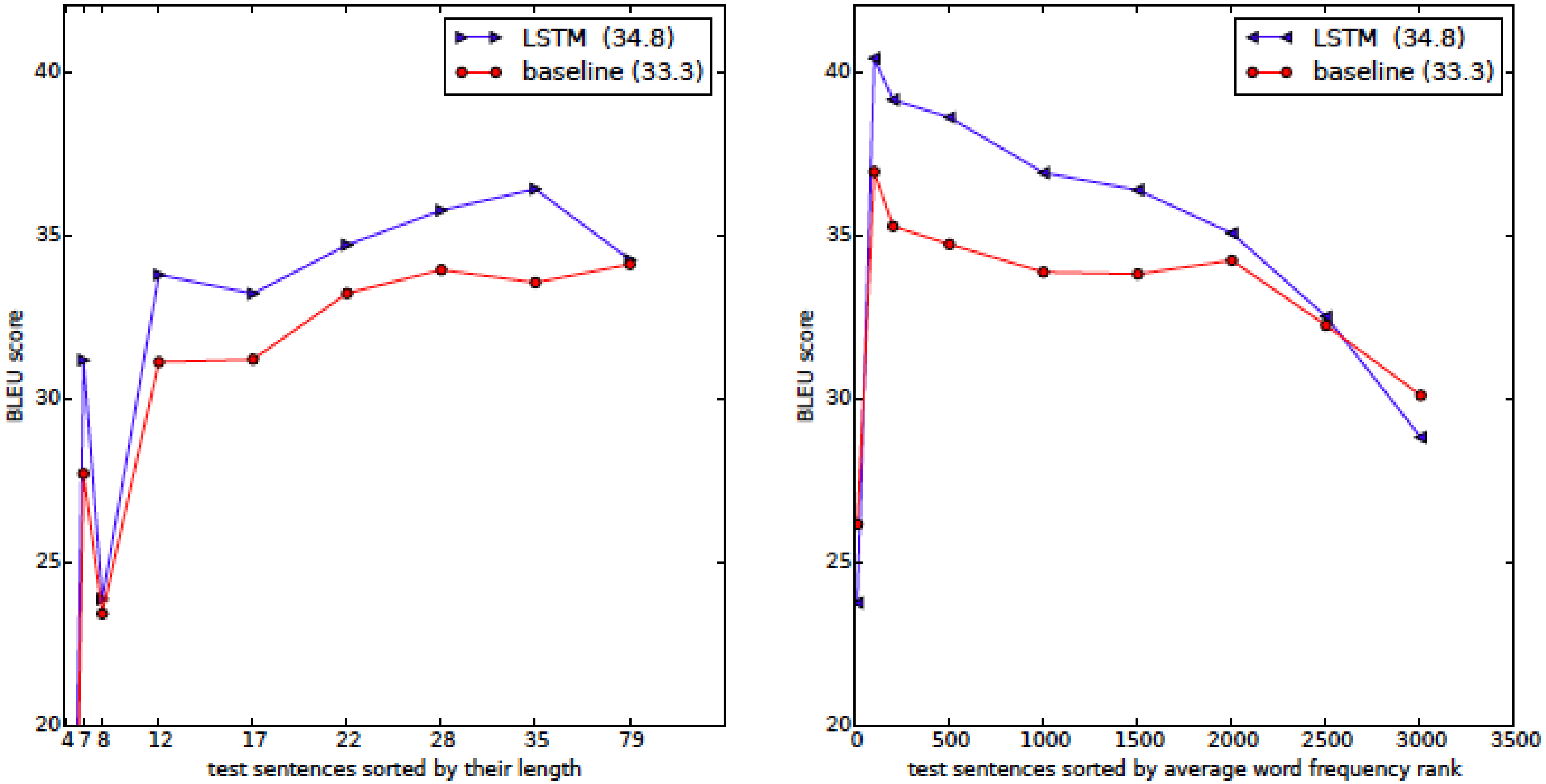




Sutskever et al. 2014

Sequence to Sequence

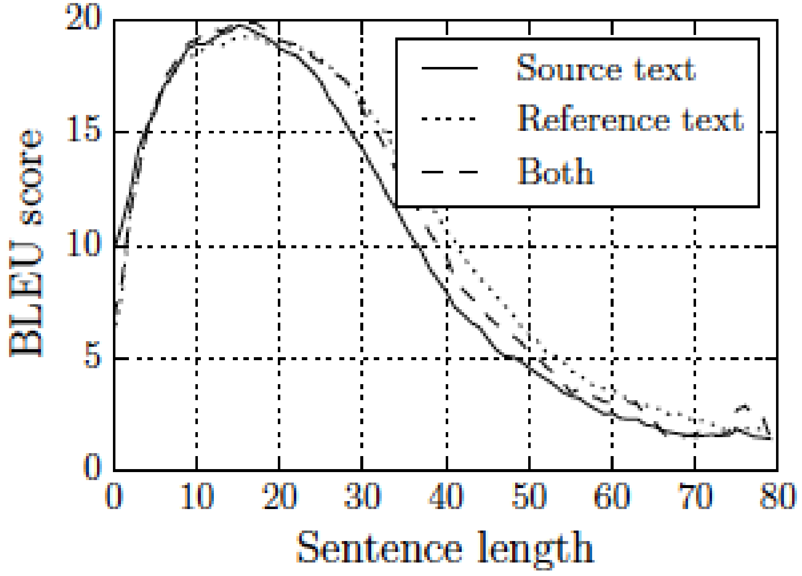
- Lon



Sutskever et al. 2014

Sequence to Sequence

- Long sentences



Cho et al. 2014

Bahdanau et al.,2014

Neural Machine Translation by Jointly Learning to Align and Translate

Sequence to Sequence

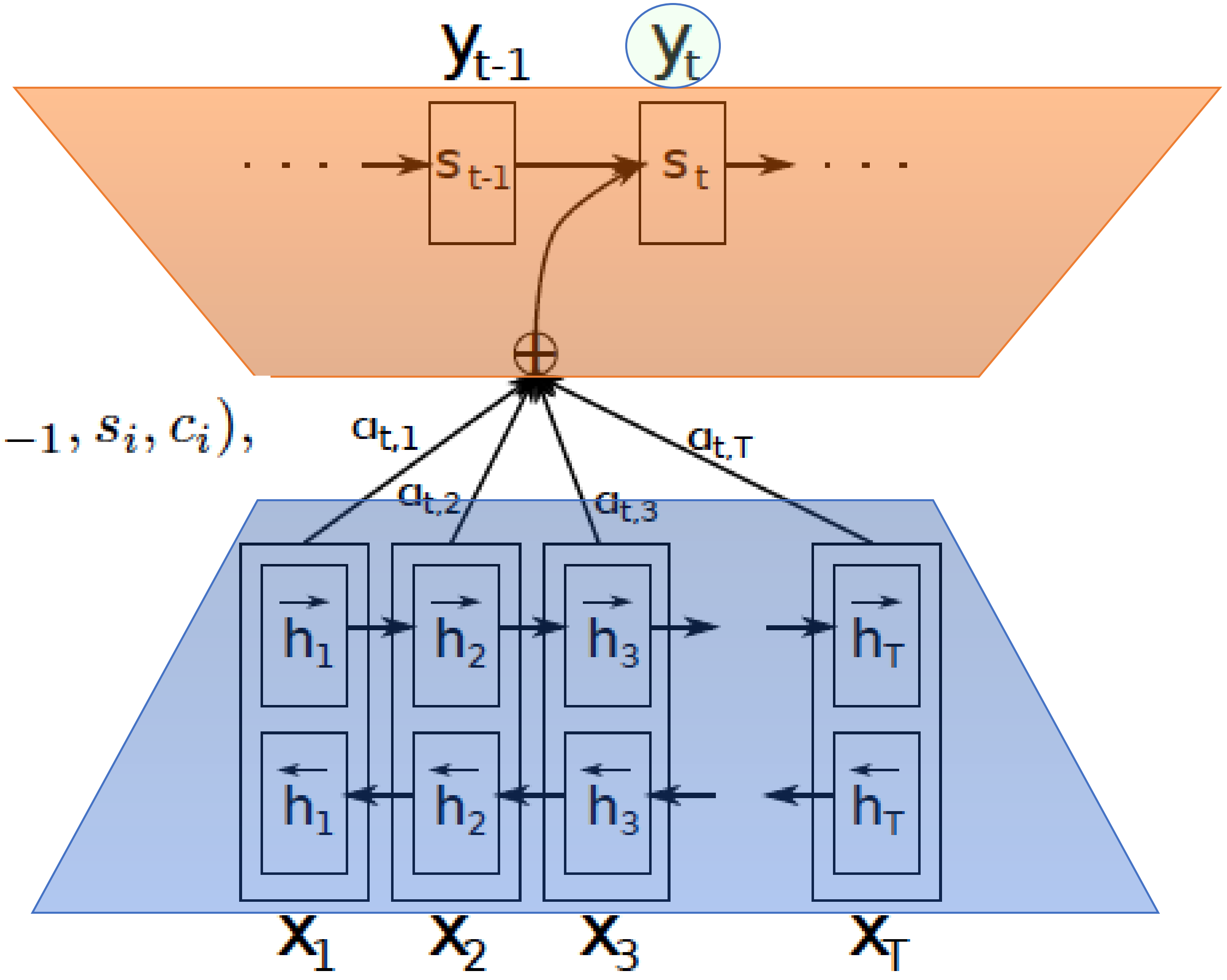
- Long sentences

Fixed length representation maybe the cause

Jointly Learning to Align and Translate

- Attention mechanism

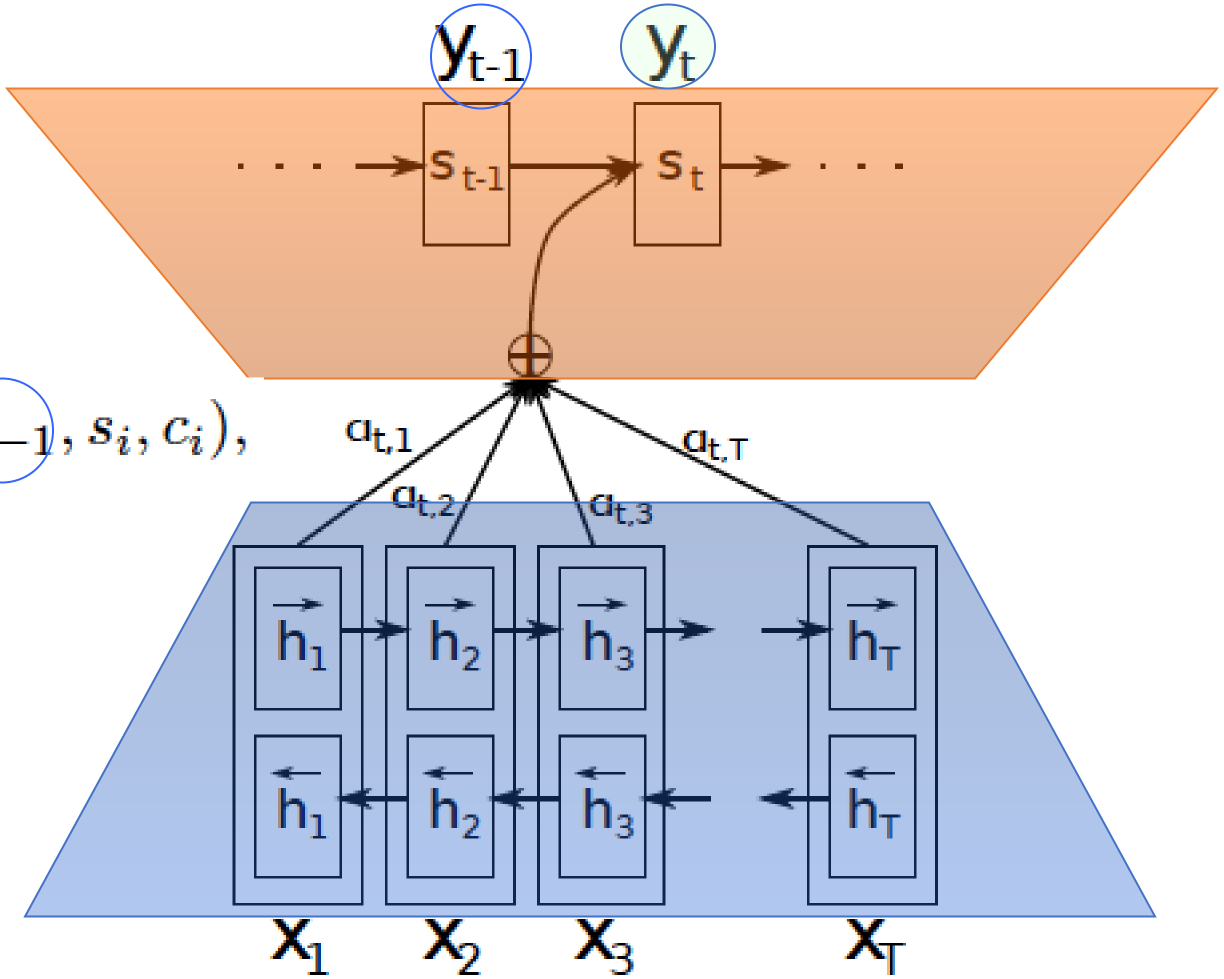
$$p(y_i|y_1, \ldots, y_{i-1}, \mathrm{x}) = g(y_{i-1}, s_i, c_i),$$



Jointly Learning to Align and Translate

- Attention mechanism

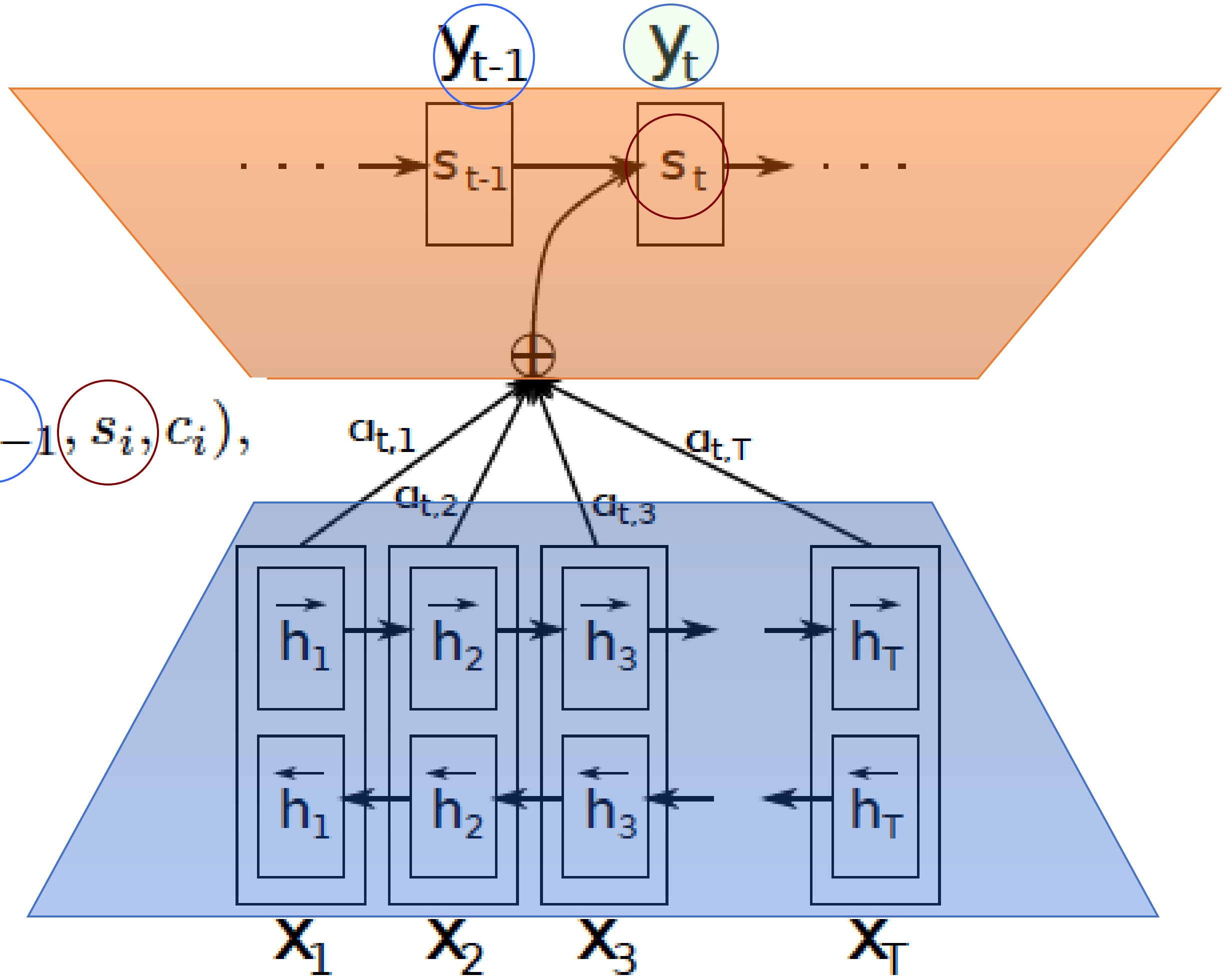
$$p(y_i|y_1, \ldots, y_{i-1}, \mathrm{x}) = g(y_{i-1}, s_i, c_i),$$



Jointly Learning to Align and Translate

- Attention mechanism

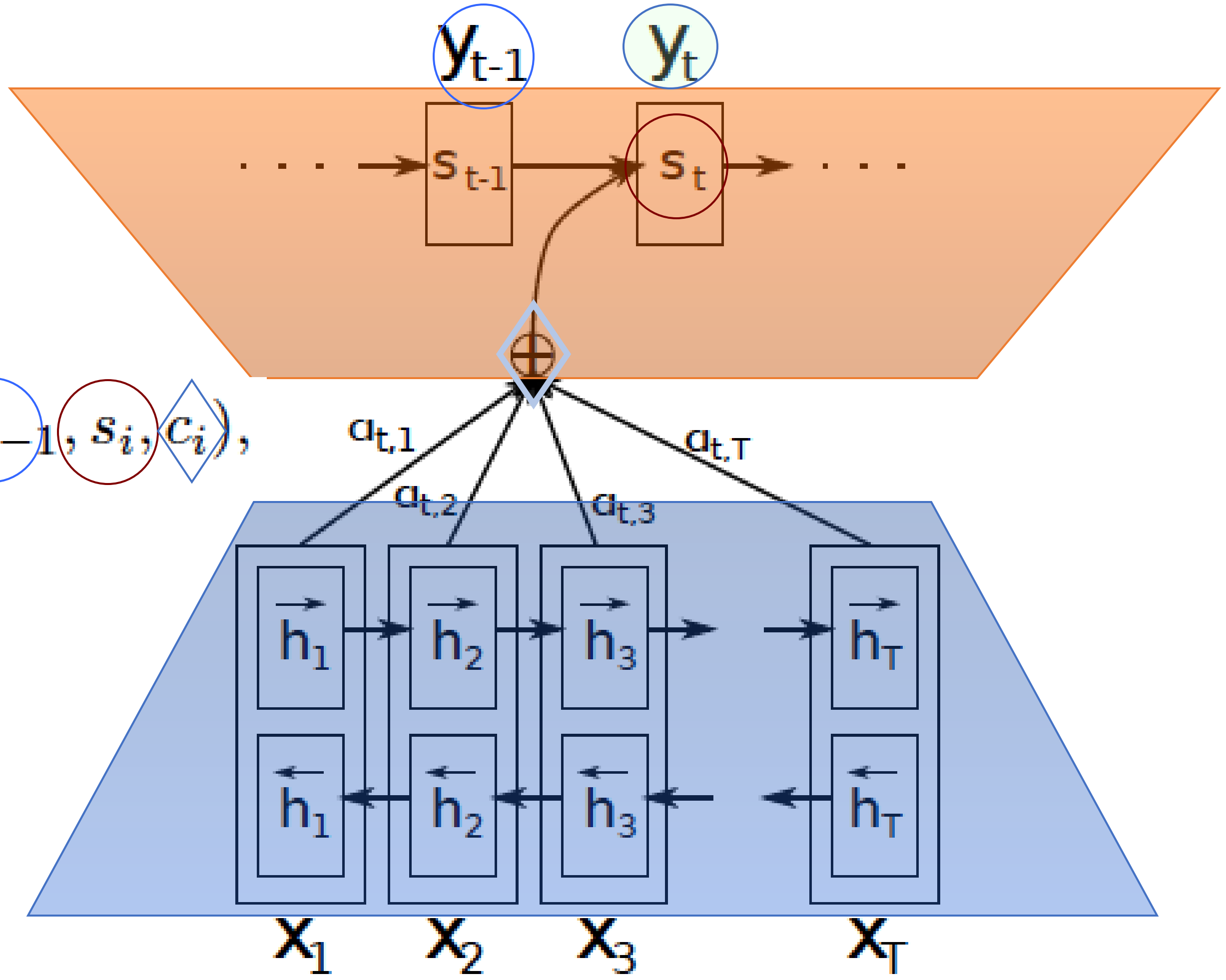
$$p(y_i|y_1,\ldots,y_{i-1},\mathbf{x}) = g(y_{i-1}, s_i, c_i),$$



Jointly Learning to Align and Translate

- Attention mechanism

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

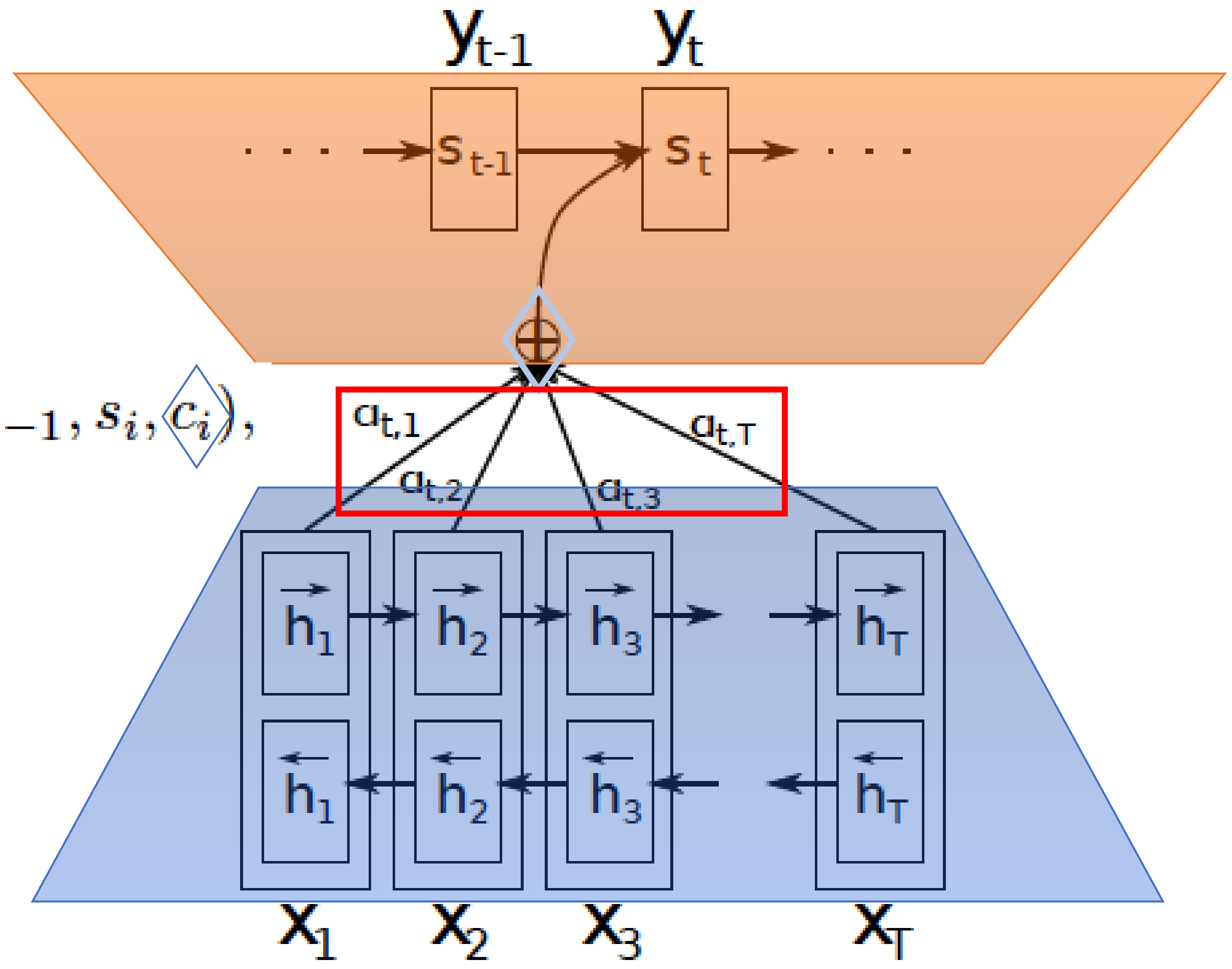


Jointly Learning to Align and Translate

- Attention mechanism

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

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



Jointly Learning to Align and Translate

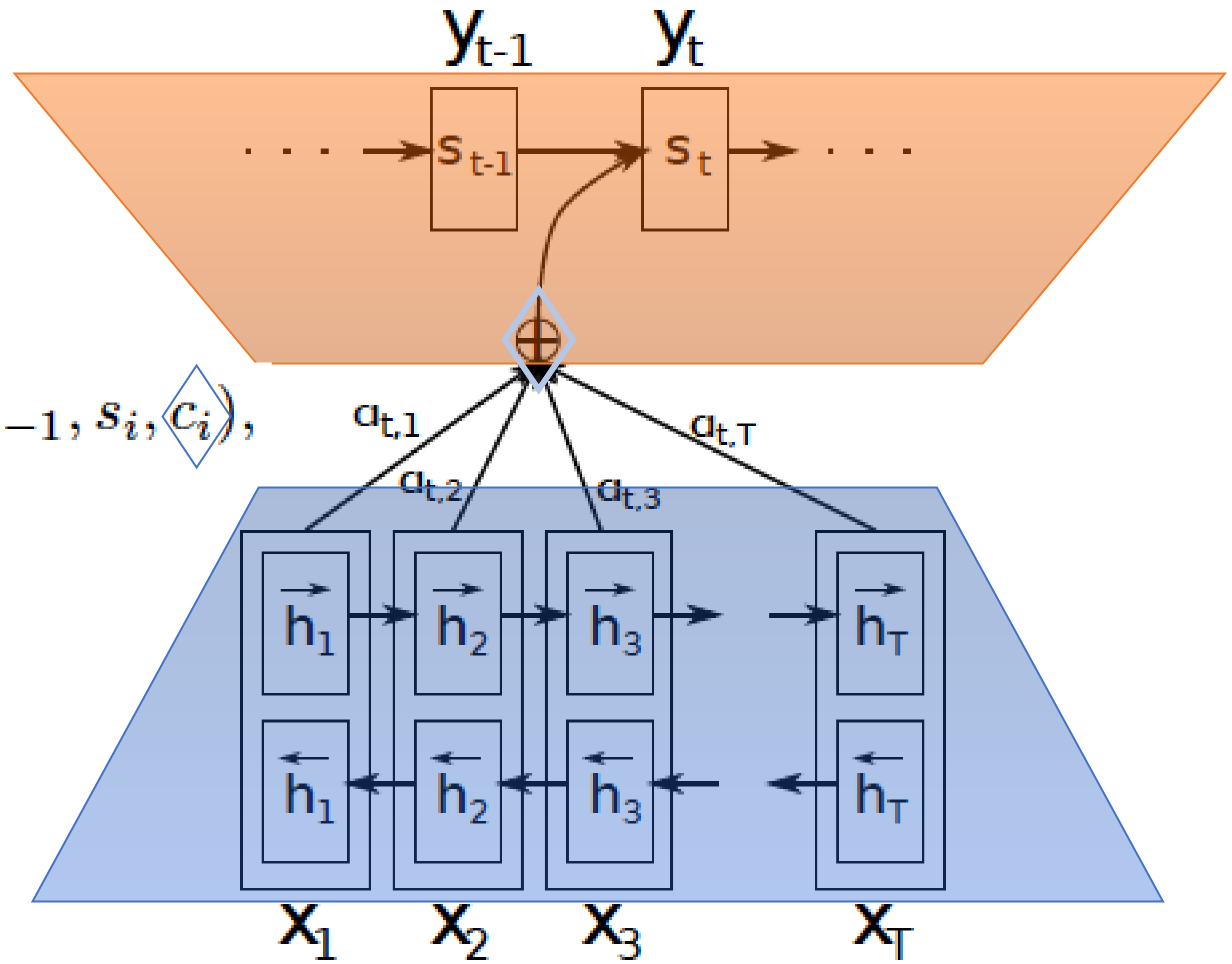
- Attention mechanism

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

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

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

$$e_{ij} = a(s_{i-1}, h_j)$$



Jointly Learning to Align and Translate

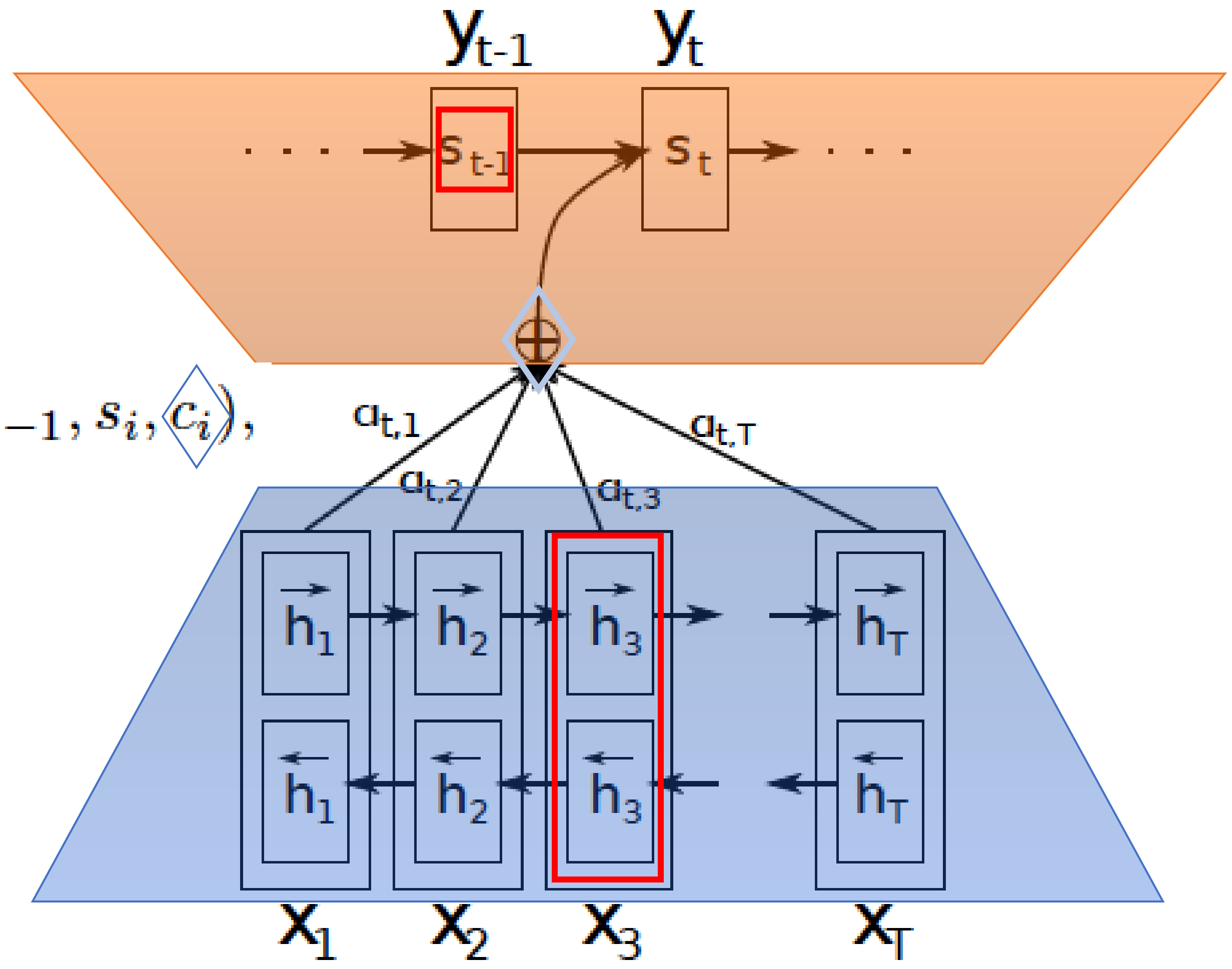
- Attention mechanism

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

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

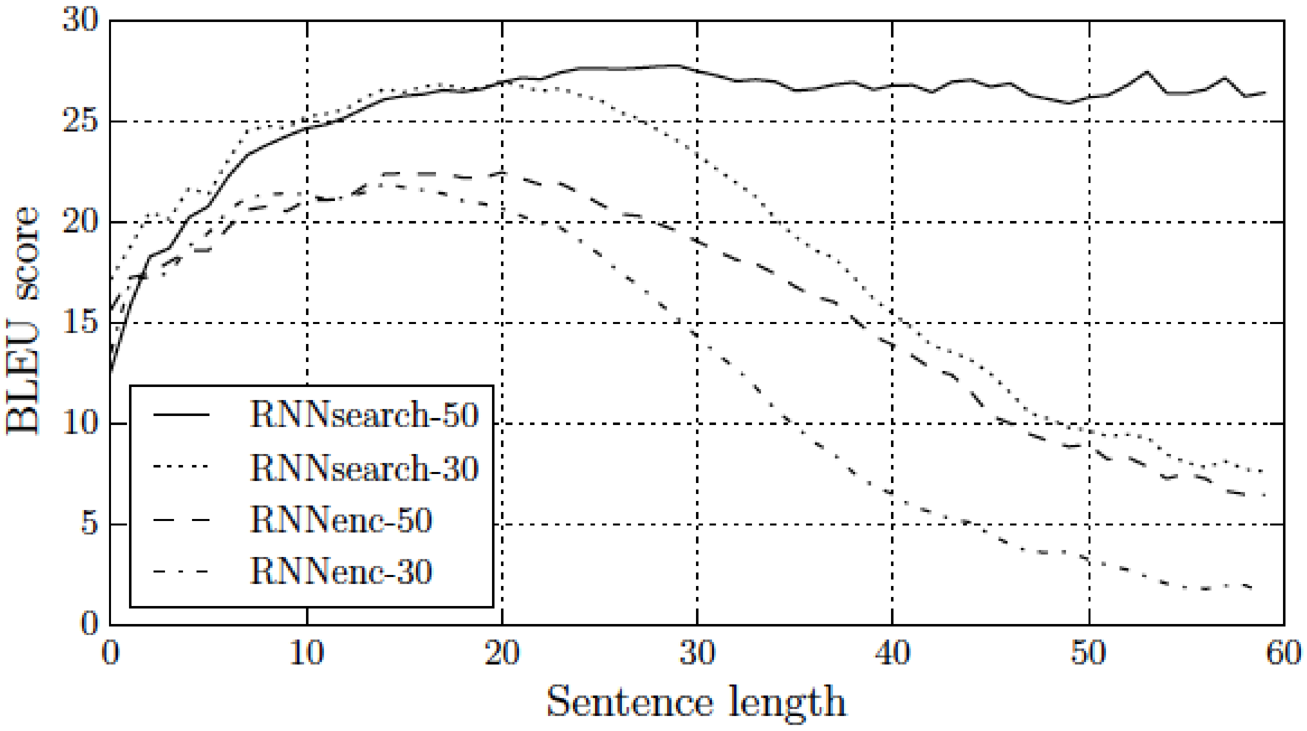
$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})},$$

$$\boxed{e_{ij}} = a(s_{i-1}, h_j)$$



Jointly Learning to Align and Translate

- Long sent



Cho et al. 2014