Text Mining and Natural Language Processing Using Deep Learning

Stacked, Bi-directional models, LSTM, GRUs, Encoder - Decoder architectures and applications, Attention Mechanism

Maheshkumar Duvvarapu

Slides by Dr.Manish Gupta

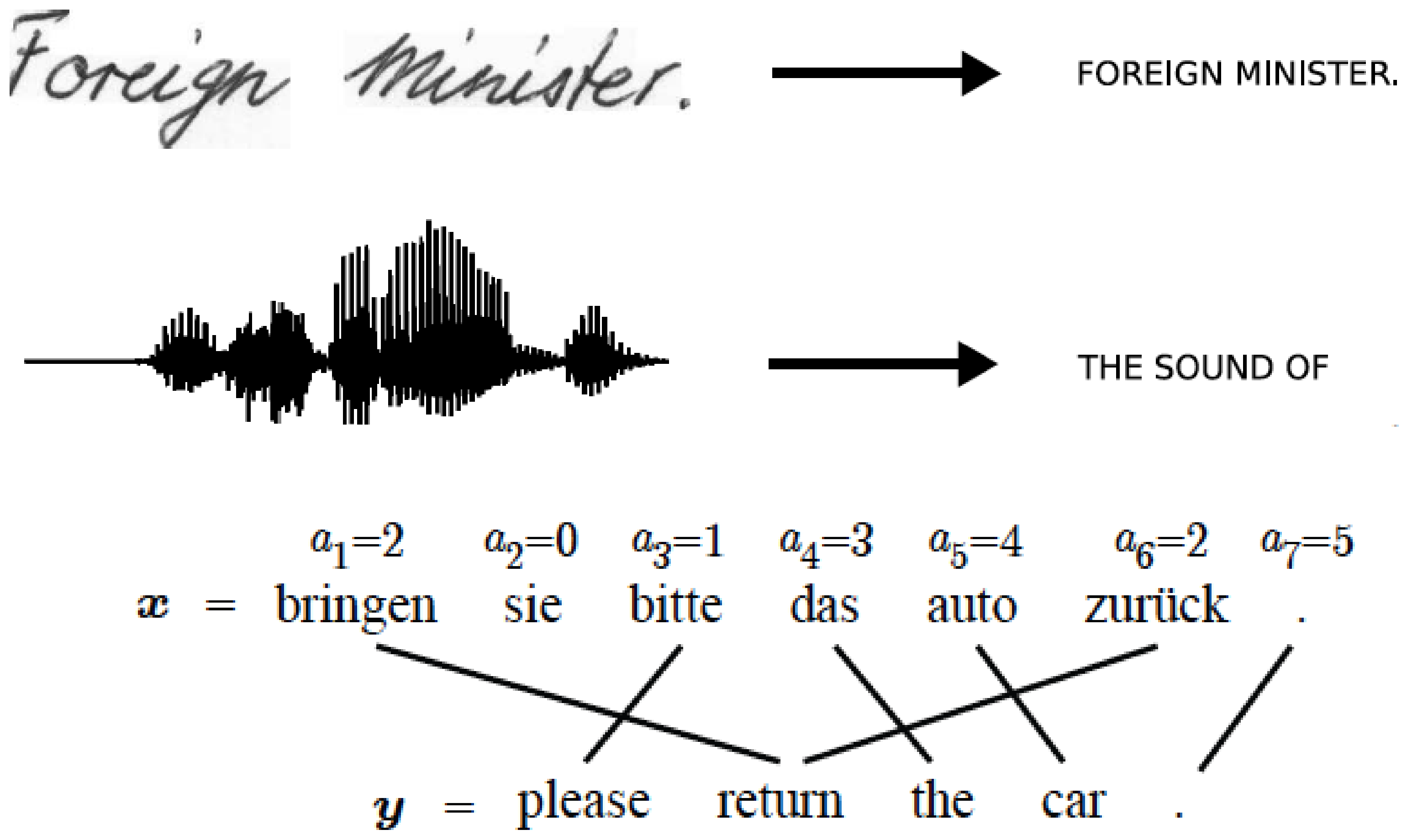
Agenda

- Traditional n-gram language models and NNLM
- RNNs
- Bidirectional RNNs, Stacked RNNs, Vanishing gradients problem
- LSTMs
- Encoder-decoder models
- Attention based models

Agenda

- **Traditional n-gram language models and NNLM**
- RNNs
- Bidirectional RNNs, Stacked RNNs, Vanishing gradients problem
- LSTMs

Sequences are everywhere…


Foreign Minister.
FOREIGN MINISTER.
THE SOUND OF
$a_1=2$ $a_2=0$ $a_3=1$ $a_4=3$ $a_5=4$ $a_6=2$ $a_7=5$
$\boldsymbol{x}$ = bringen sie bitte das auto zurück .
$\boldsymbol{y}$ = please return the car .

- We need a way to model such sequence data using neural networks.
- Humans don't start their thinking from scratch every second. As you read this essay, you understand each word based on your understanding of previous words. You don't throw everything away and start thinking from scratch again. Your thoughts have persistence.
- RNNs have loops in them which allow for information to persist.

Language Models

- A language model computes a probability for a sequence of words: $P(w_1, \dots, w_m)$
- Very useful for many tasks like
  - Next word prediction
    - Stocks plunged this morning, despite a cut in interest rates by the Federal Reserve, as Wall …
  - Spell checkers
    - They are leaving in about fifteen **minuets** to go to her house
  - Mobile auto-correct
    - He is trying to **fine** out.
  - Speech recognition
    - Theatre owners say **popcorn/unicorn** sales have doubled…
  - Automated essay grading
  - Machine translation
    - Word ordering: p(the cat is small) > p(small the is cat)
    - Word choice: p(walking home after school) > p(walking house after school)

Traditional Language Models

- Probability is usually conditioned on a window of n previous words:
  - $P(w_1, \dots, w_m) = \Pi_{i=1}^{m} P(w_i | w_1, \dots w_{i-1}) \approx \Pi_{i=1}^{m} P(w_i | w_{i-(n-1)}, \dots, w_{i-1})$
  - Markov assumption
- To estimate probabilities, compute for unigrams and bigrams (conditioning on one/two previous word(s):
  - Bigram models: $p(w_2 | w_1) = \frac{count(w_1, w_2)}{count(w_1)}$
  - Trigram models: $p(w_3 | w_1, w_2) = \frac{count(w_1, w_2, w_3)}{count(w_1, w_2)}$

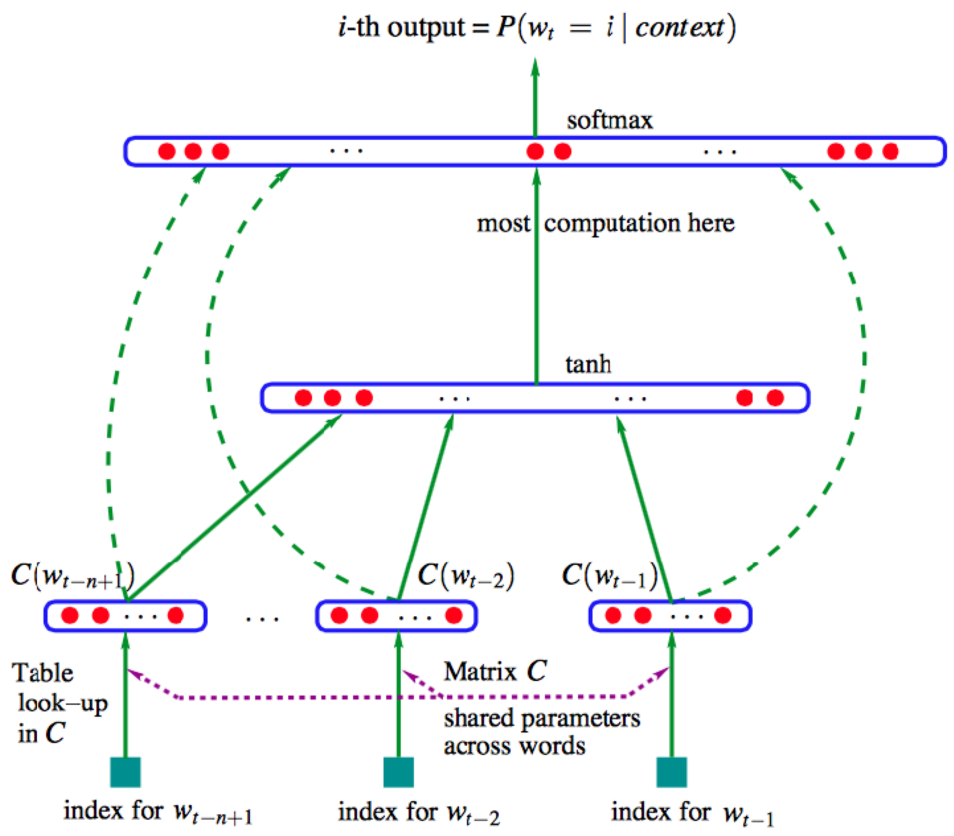
Traditional Language Models

- Performance improves as n for n-grams increases, and doing smoothing and backoff (e.g. if 4-gram not found, try 3-gram, etc).
- There are A LOT of n-grams!
- Gigantic RAM requirements!
- Heafield et al: “Using one machine with 140 GB RAM for 2.8 days, we built an unpruned model on 126 billion tokens”
- In some cases, the window of past consecutive n words may not be sufficient to capture the context.
  - For instance, consider a case where an article discusses the history of Spain and France and somewhere later in the text, it reads "The two countries went on a battle"; clearly the information presented in this sentence alone is not sufficient to identify the name of the two countries.

Heafield, Kenneth, Ivan Pouzyrevsky, Jonathan H. Clark, and Philipp Koehn. "Scalable modified Kneser-Ney language model estimation." In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 690-696. 2013.

Neural Language Model using MLPs

- Predict the i^{th} word based on a fixed size context.
- In all conventional language models, the memory requirements of the system grows exponentially with the window size n making it nearly impossible to model large word windows without running out of memory.
- Problem: Fixed window of context (i.e., n)
- Can there be a model which can handle contexts of different length?



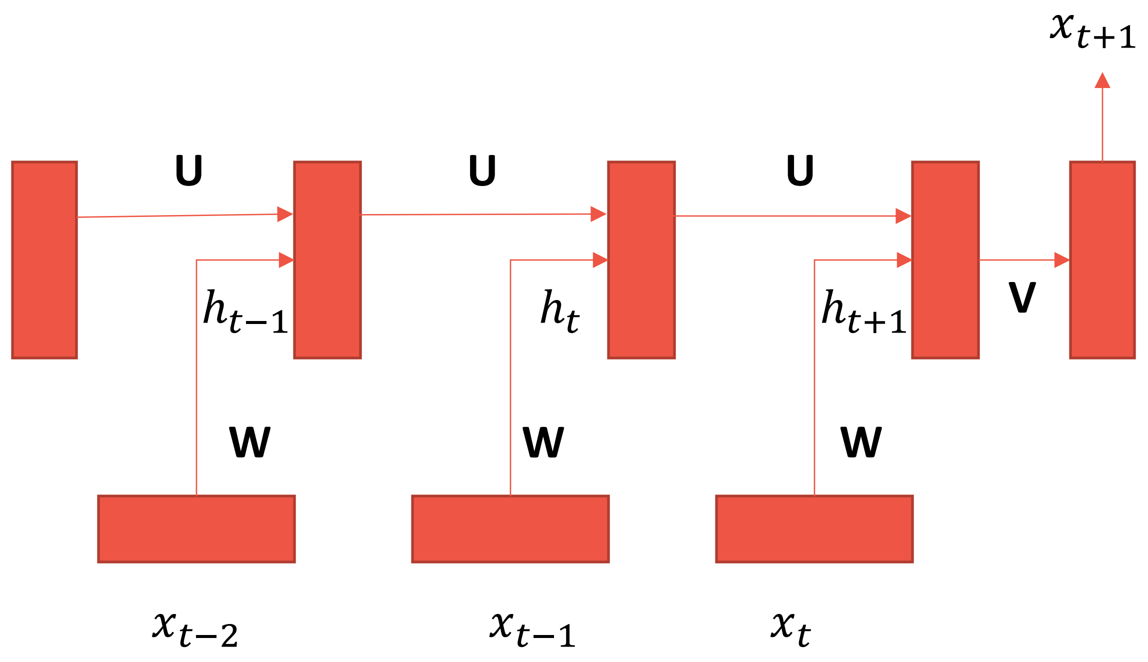
$$\hat{y} = softmax(W^{(2)} tanh(W^{(1)} x + b^{(1)}) + W^{(3)} x + b^{(3)})$$

Agenda

- Traditional n-gram language models and NNLM
- **RNNs**
- Bidirectional RNNs, Stacked RNNs, Vanishing gradients problem
- LSTMs

Recurrent Neural Networks

- RNNs tie the weights at each time step.
- Condition the neural network on all previous words.
- RAM requirement only scales with the number of words.



RNNs

$$\boxed{h_t} = \boxed{f_W}(\boxed{h_{t-1}}, \boxed{x_t})$$

new state

some function
with parameters W

old state

input vector at
some time step

$$h_t = f_W(h_{t-1}, x_t)$$

$$\downarrow$$

$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t)$$

$$y_t = W_{hy}h_t$$

- RNNs are called recurrent because they perform same task for every element of a sequence.
- Only thing that differs is the input at each time step.
- Output is dependent on previous computations.
- RNNs can be seen as an NN having “memory” about what has been calculated so far.

RNN Language Model

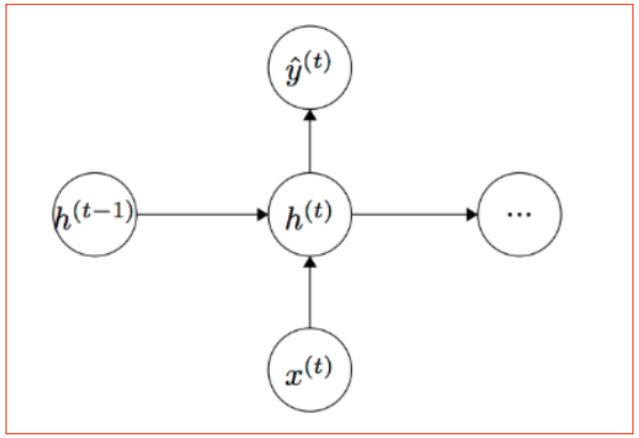
Given list of word **vectors**: $x_1, \ldots, x_{t-1}, x_t, x_{t+1}, \ldots, x_T$

At a single time step:

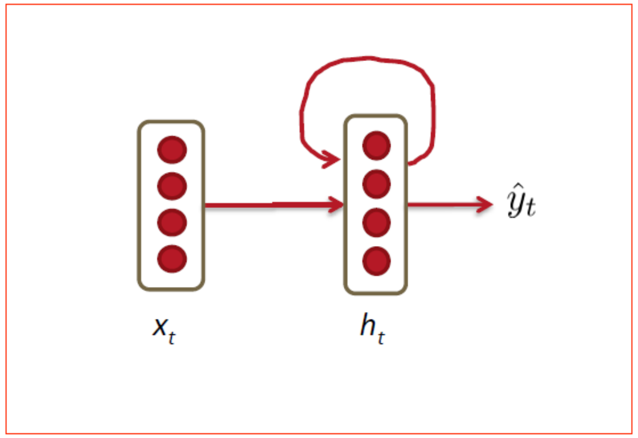
$$h_t = \sigma\left(W^{(hh)} h_{t-1} + W^{(hx)} x_{[t]}\right)$$

$$\hat{y}_t = \operatorname{softmax}\left(W^{(S)} h_t\right)$$

$$\hat{P}(x_{t+1} = v_j \mid x_t, \ldots, x_1) = \hat{y}_{t,j}$$

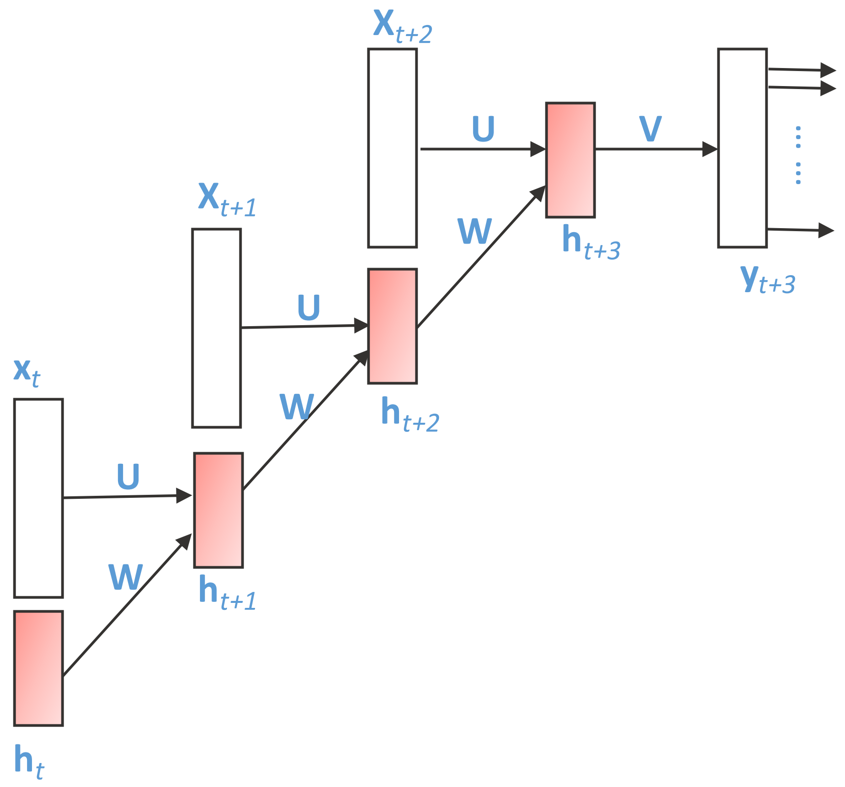


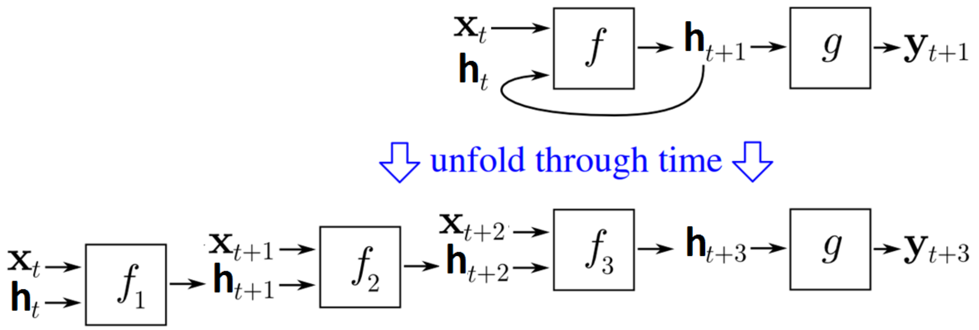
$\leftrightarrow$



Recurrent NN unfolds into a DNN over

- Condition the neural network on all previous words and tie the weights at each time step.





$$\begin{tikzcd} \mathbf{x}_t \arrow[r] & \boxed{f} \arrow[r] & \mathbf{h}_{t+1} \arrow[r] \arrow[ll, bend left, to path={}] & \boxed{g} \arrow[r] & \mathbf{y}_{t+1} \\ \mathbf{h}_t \arrow[ur] \end{tikzcd}$$

$$\begin{tikzcd} \mathbf{x}_t, \mathbf{h}_t \arrow[r] & \boxed{f_1} \arrow[r] & \mathbf{x}_{t+1}, \mathbf{h}_{t+1} \arrow[r] & \boxed{f_2} \arrow[r] & \mathbf{x}_{t+2}, \mathbf{h}_{t+2} \arrow[r] & \boxed{f_3} \arrow[r] & \mathbf{h}_{t+3} \arrow[r] & \boxed{g} \arrow[r] & \mathbf{y}_{t+3} \end{tikzcd}$$

$\boldsymbol{x}_t$ input at time t
$\mathbf{h}_t$ memory (state) at time t
$\mathbf{y}_t$ output at time t

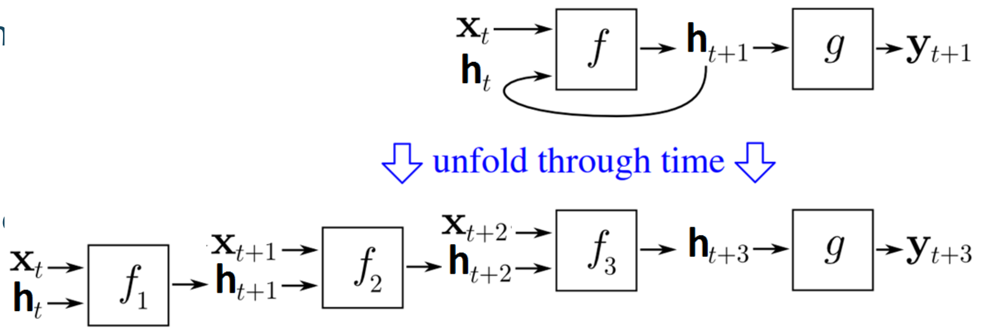
$$\mathbf{z}_t = \mathbf{U}\boldsymbol{x}_t + \mathbf{W}\mathbf{h}_{t-1}$$
$$\mathbf{h}_t = f(\mathbf{z}_t)$$
$$\mathbf{y}_t = g(\mathbf{V}\mathbf{h}_t)$$

where

$$f(z) = \frac{1}{1+\exp(-z)}, \quad g(z_m) = \frac{\exp(z_m)}{\sum_k \exp(z_k)}$$

Back propagation through time (BPTT)

```
Back_Propagation_Through_Time(x, y)   // x[t] is the input at time t. y[t] is th
   Unfold the network to contain k instances of f
   do until stopping criteria is met:
      h = the zero-magnitude vector; // h is the current context
      for t from 0 to n - 1        // t is time. n is the length of the training sequenc
         Set the network inputs to h, x[t], x[t+1], ..., x[t+k-1]
         p = forward-propagate the inputs over the whole unfolded network
         e = y[t+k] - p;          // error = target - prediction
         Back-propagate the error, e, back across the whole unfolded network
         Update all the weights in the network
         Average the weights in each instance of f together, so that each f is identical
         h = f(h);               // compute the context for the next time-step
```

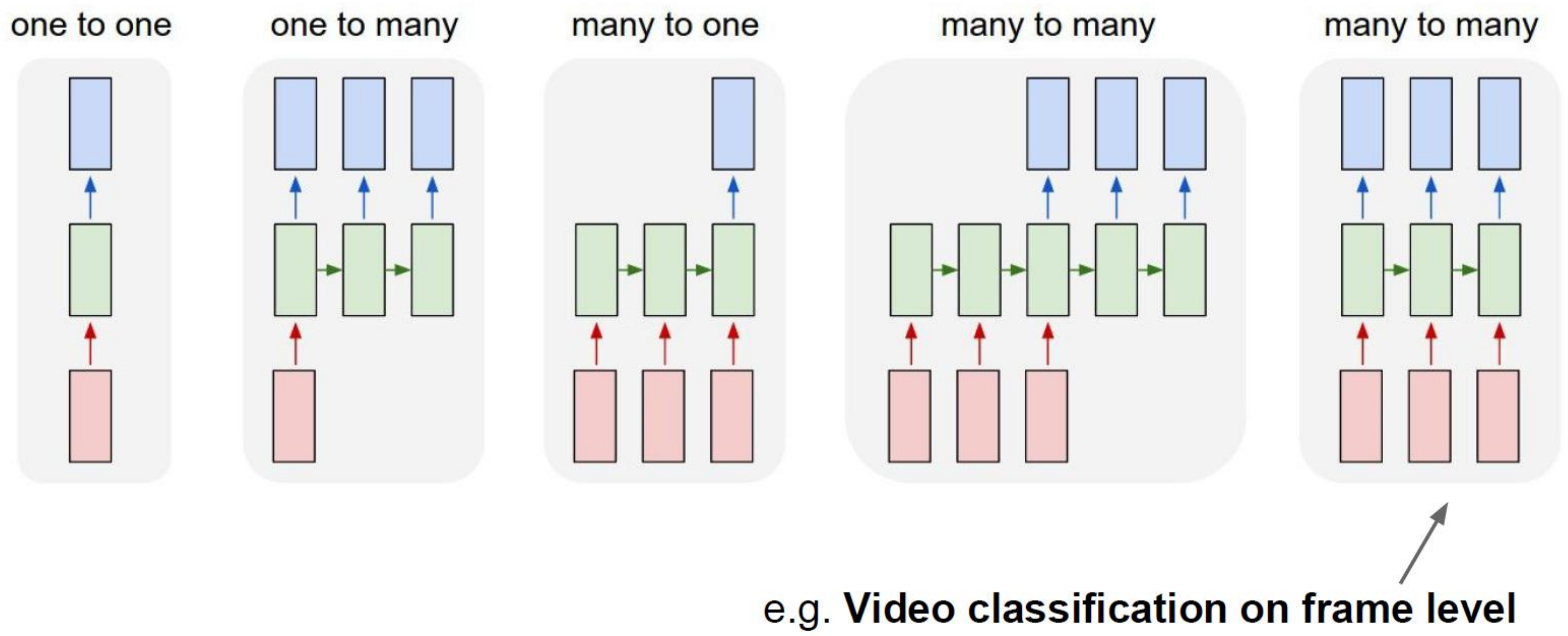


- Truncated BPTT is an approximation of full BPTT that is preferred for long sequences, since full BPTT's forward/backward cost per parameter update becomes very high over many time steps.
- The downside is that the gradient can only flow back so far due to that truncation, so the network can't learn dependencies that are as long as in full BPTT.

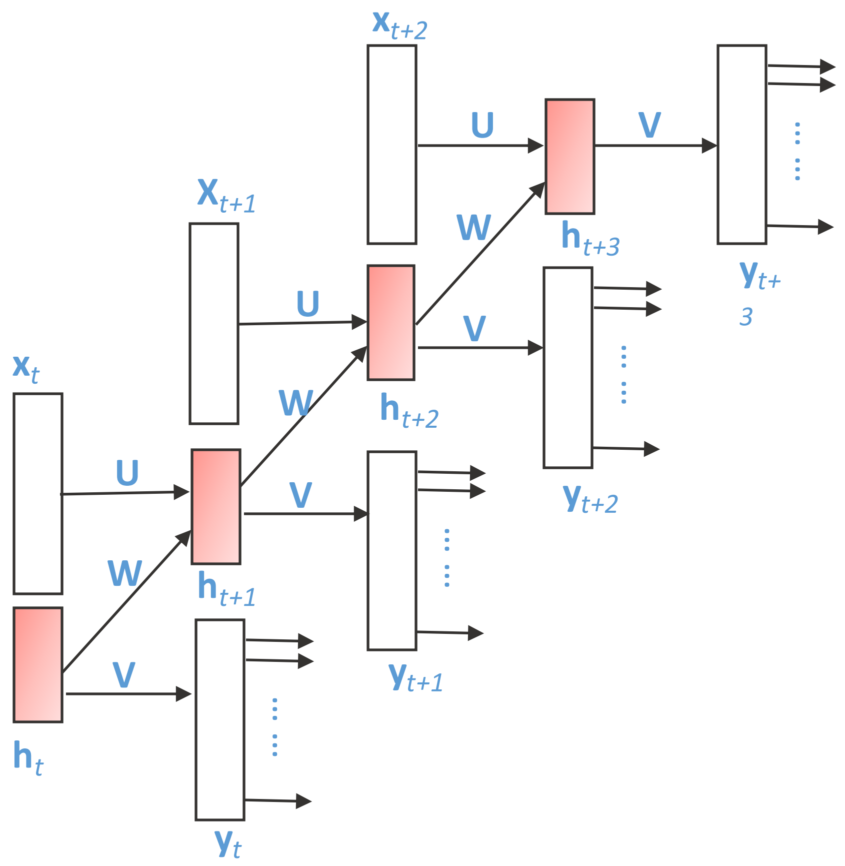
Recurrent Neural Networks Loss Computation

- $\hat{y} = R^{|V|}$ is a probability distribution over the vocabulary.
- Cross entropy loss function can be used for predicting words.
  - $J^{(t)}(\theta) = -\sum_{j=1}^{|V|} y_{t,j} \log \hat{y}_{t,j}$
- Over a dataset of T words
  - $J = -\frac{1}{T}\sum_{t=1}^{T}\sum_{j=1}^{|V|} y_{t,j} \log \hat{y}_{t,j}$
- Lower this value the better.

Recurrent Networks offer a lot of flexibility



Many-to-many case



CNN+RNN for image captioning

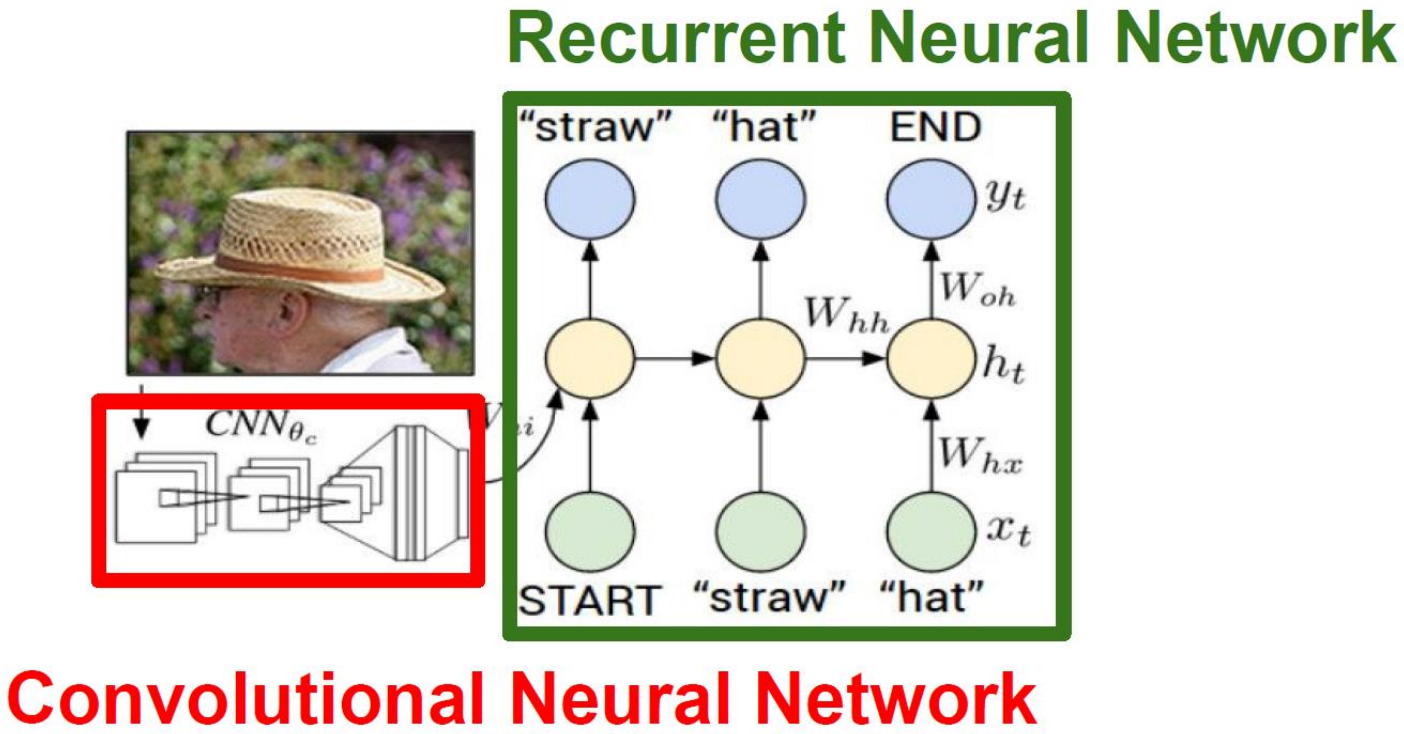


Image Captioning

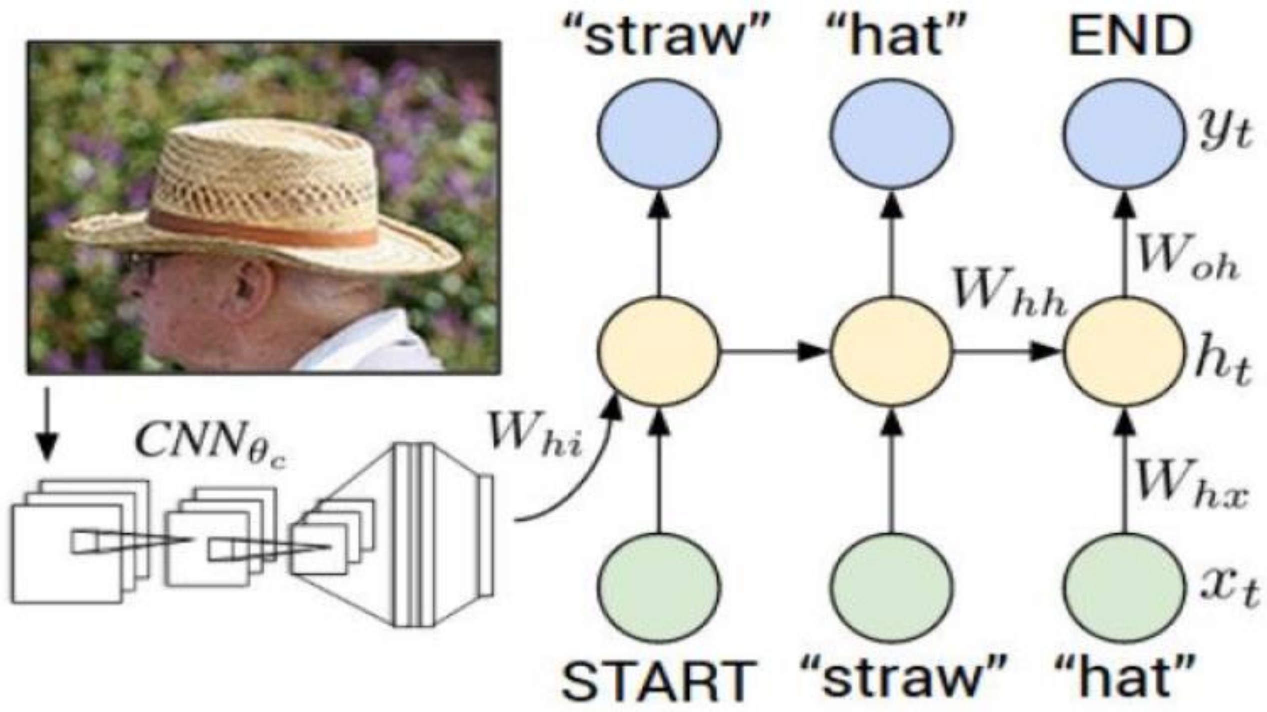
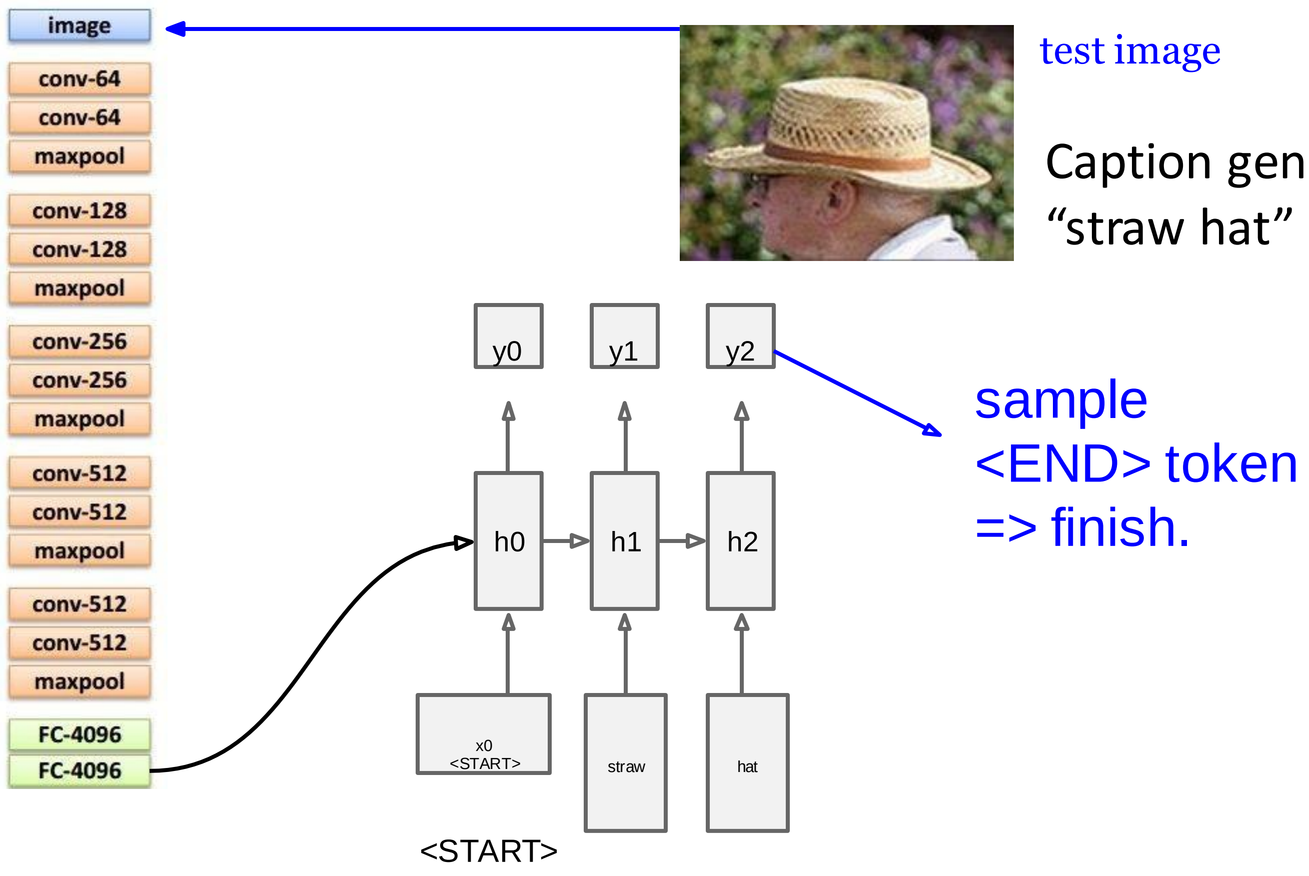


Image Captioning



Caption generated: "straw hat"

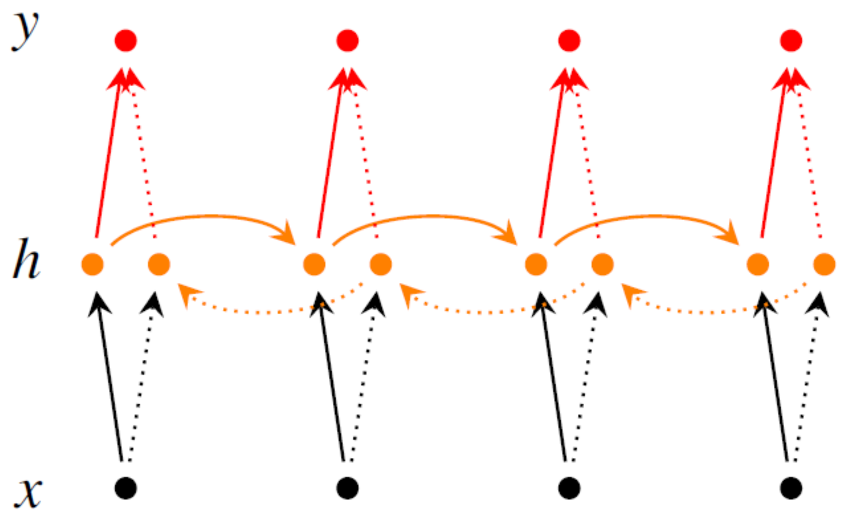
Adapted from Andrej Karpathy

Agenda

- Traditional n-gram language models and NNLM
- RNNs
- **Bidirectional RNNs, Stacked RNNs, Vanishing gradients problem**
- LSTMs

Bidirectional RNNs

Problem: For classification you want to incorporate information from words both preceding and following



$$\overrightarrow{h}_t = f(\overrightarrow{W}x_t + \overrightarrow{V}\overrightarrow{h}_{t-1} + \overrightarrow{b})$$

$$\overleftarrow{h}_t = f(\overleftarrow{W}x_t + \overleftarrow{V}\overleftarrow{h}_{t+1} + \overleftarrow{b})$$

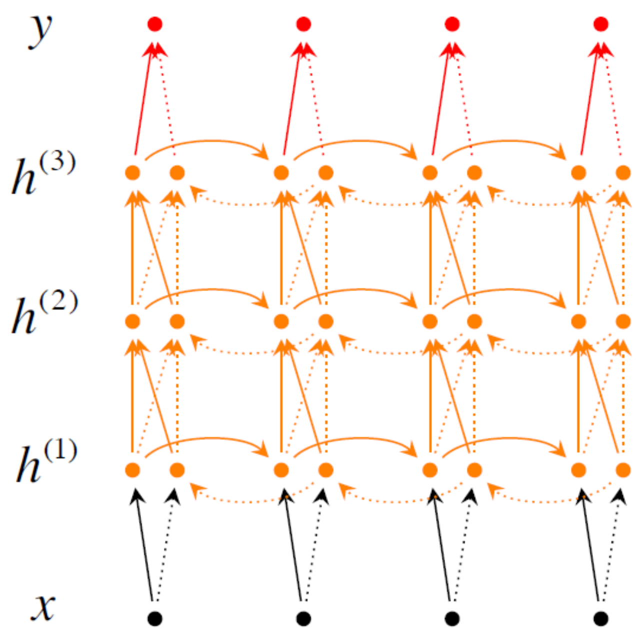
$$y_t = g(U[\overrightarrow{h}_t;\overleftarrow{h}_t] + c)$$

$h = [\overrightarrow{h};\overleftarrow{h}]$ now represents (summarizes) the past and future around a single token.

To maintain two hidden layers at any time, this network consumes twice as much memory space for its weight and bias parameters.

Schuster, Mike, and Kuldip K. Paliwal. "Bidirectional recurrent neural networks." *IEEE transactions on Signal Processing* 45, no. 11 (1997): 2673-2681.

Deep Bidirectional RNNs



$$\overrightarrow{h}_t^{(i)} = f(\overrightarrow{W}^{(i)} h_t^{(i-1)} + \overrightarrow{V}^{(i)} \overrightarrow{h}_{t-1}^{(i)} + \overrightarrow{b}^{(i)})$$

$$\overleftarrow{h}_t^{(i)} = f(\overleftarrow{W}^{(i)} h_t^{(i-1)} + \overleftarrow{V}^{(i)} \overleftarrow{h}_{t+1}^{(i)} + \overleftarrow{b}^{(i)})$$

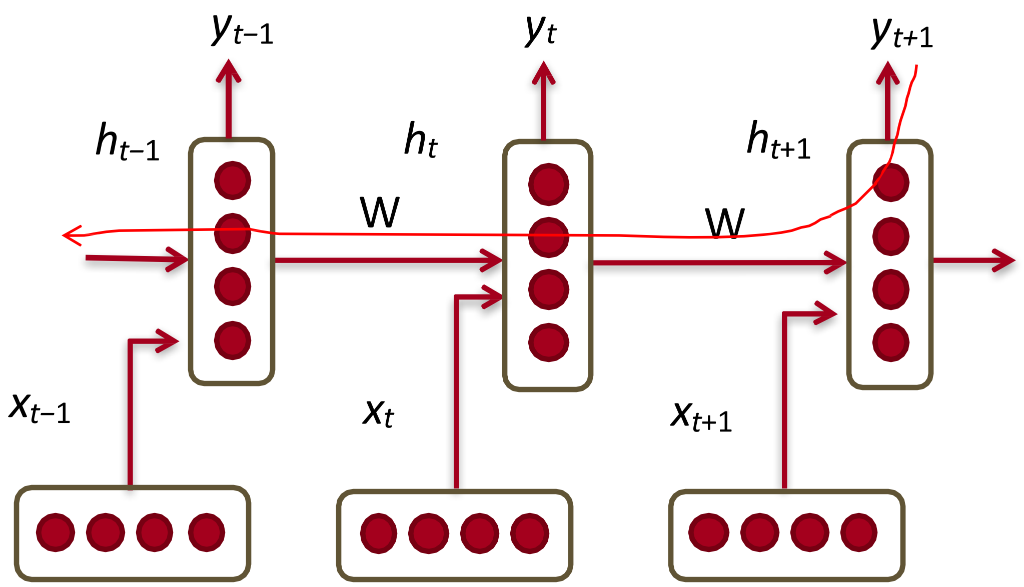
$$y_t = g(U[\overrightarrow{h}_t^{(L)}; \overleftarrow{h}_t^{(L)}] + c)$$

Each memory layer passes an intermediate sequential representation to the next.

At time-step t each intermediate neuron receives one set of parameters from the previous time-step (in the same RNN layer), and two sets of parameters from the previous RNN hidden layer; one input comes from the left-to-right RNN and the other from the right-to-left RNN.

The vanishing gradient problem

- The error at a time step ideally can tell a previous time step from many steps away to change during backpropagation.
- But we’re multiplying together many values between 0 and 1



The vanishing gradient problem

- In the case of language modeling or question answering, words from time steps far away are not taken into consideration when training to predict the next word
- Example:
- Jane walked into the room. John walked in too. It was late in the day. Jane said hi to __________
- Basic RNN are not very good at capturing very long-term dependencies because of vanishing gradient problem. Output $\hat{y}$ is mainly influenced by values close to $\hat{y}$.

*The **cat**, which actually ate …… , **was** full.*

*The **cats**, which actually ate …… , **were** full.*

Agenda

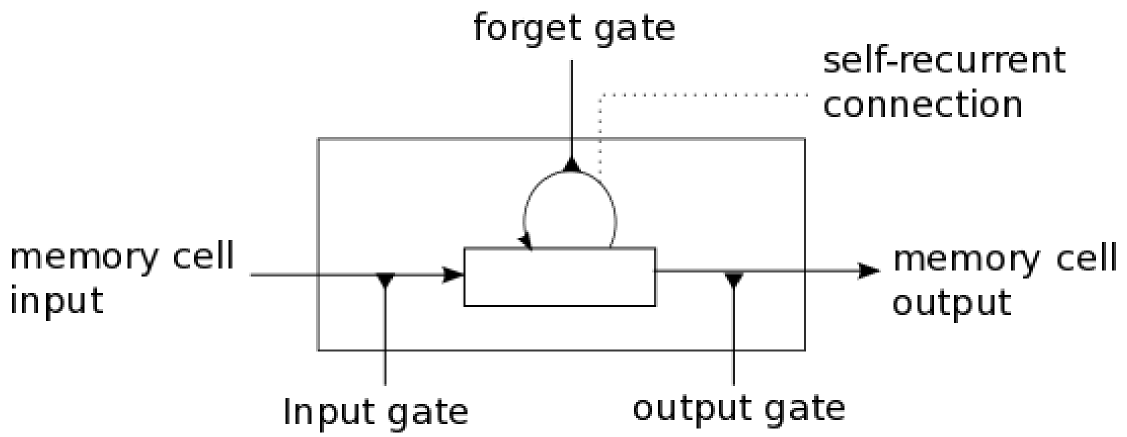
- Traditional n-gram language models and NNLM
- RNNs
- Bidirectional RNNs, Stacked RNNs, Vanishing gradients problem
- **LSTMs**

Memory based models

- Better units than RNNs.
- Beyond the extensions discussed so far, RNNs have been found to perform better with the use of more complex units for activation.
- Although RNNs can theoretically capture long-term dependencies, they are very hard to actually train to do this.
- Main ideas
  - keep around memories to capture long distance dependencies.
  - allow error messages to flow at different strengths depending on the inputs.

Long short-term memory (LSTM)

- Intuition: memory cells can keep information intact, unless inputs makes them forget it or overwrite it with new input.
- Cell can decide to output this information or just store it.



Hochreiter, Sepp, and Jürgen Schmidhuber. "Long short-term memory." *Neural computation* 9, no. 8 (1997): 1735-1780.

Long Short-term memories (LSTMs)

- We can make the units even more complex
- Allow each time step to modify
  - Input gate (current cell matters) $i_t = \sigma\left(W^{(i)}x_t + U^{(i)}h_{t-1}\right)$
  - Forget (gate 0, forget past) $f_t = \sigma\left(W^{(f)}x_t + U^{(f)}h_{t-1}\right)$
  - Output (how much cell is exposed) $o_t = \sigma\left(W^{(o)}x_t + U^{(o)}h_{t-1}\right)$
  - New memory cell $\tilde{c}_t = \tanh\left(W^{(c)}x_t + U^{(c)}h_{t-1}\right)$
- Final memory cell: $c_t = f_t \circ c_{t-1} + i_t \circ \tilde{c}_t$
- Final hidden state: $h_t = o_t \circ \tanh(c_t)$

Deep RNNs vs Deep LSTMs

RNN:

$$\boxed{h_t^l = \tanh W^l \begin{pmatrix} h_t^{l-1} \\ h_{t-1}^l \end{pmatrix}}$$

$h \in \mathbb{R}^n$. $\quad W^l \ [n \times 2n]$

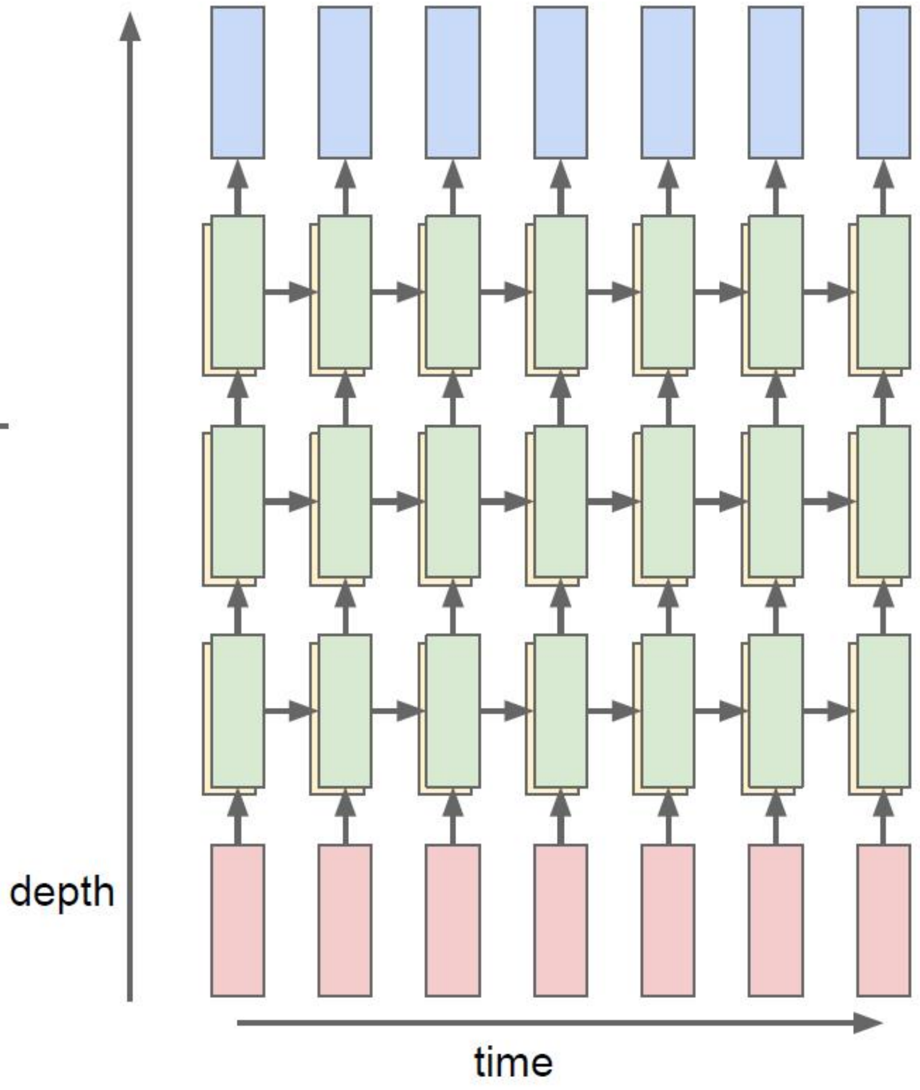
LSTM:

$W^l \ [4n \times 2n]$

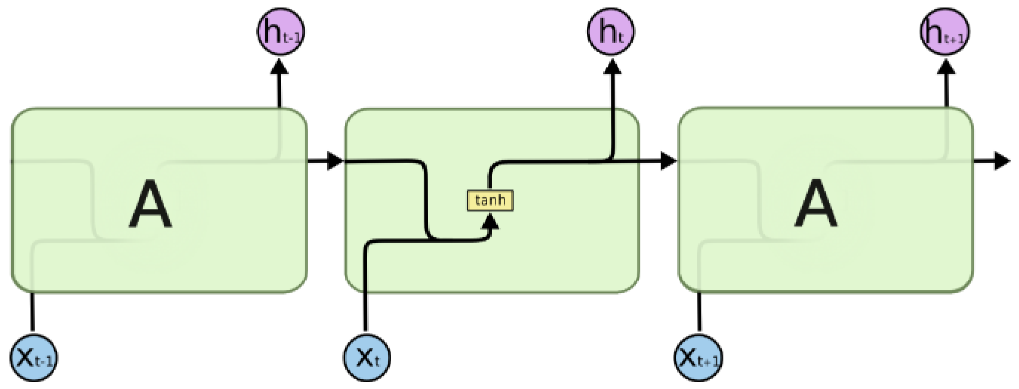
$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \text{sigm} \\ \text{sigm} \\ \text{sigm} \\ \tanh \end{pmatrix} W^l \begin{pmatrix} h_t^{l-1} \\ h_{t-1}^l \end{pmatrix}$$

$$c_t^l = f \odot c_{t-1}^l + i \odot g$$

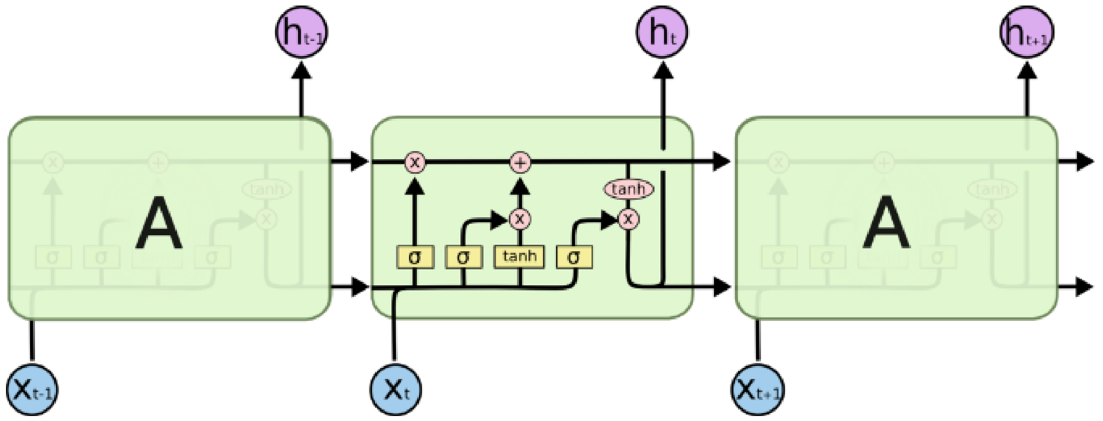
$$h_t^l = o \odot \tanh(c_t^l)$$



RNN vs LSTM



The repeating module in a standard RNN contains a single layer.



The repeating module in an LSTM contains four interacting layers.

Generating poetry with RNNs

at first:

```
tyntd-iafhatawiaoihrdemot  lytdws  e ,tfti, astai f ogoh eoase rrranbyne 'nhthnee e
plia tklrgd t o idoe ns,smtt   h ne etie h,hregtrs nigtike,aoaenns lng
```

train more

```
"Tmont thithey" fomesscerliund
Keushey. Thom here
sheulke, anmerenith ol sivh I lalterthend Bleipile shuwy fil on aseterlome
coaniogennc Phe lism thond hon at. MeiDimorotion in ther thize."
```

train more

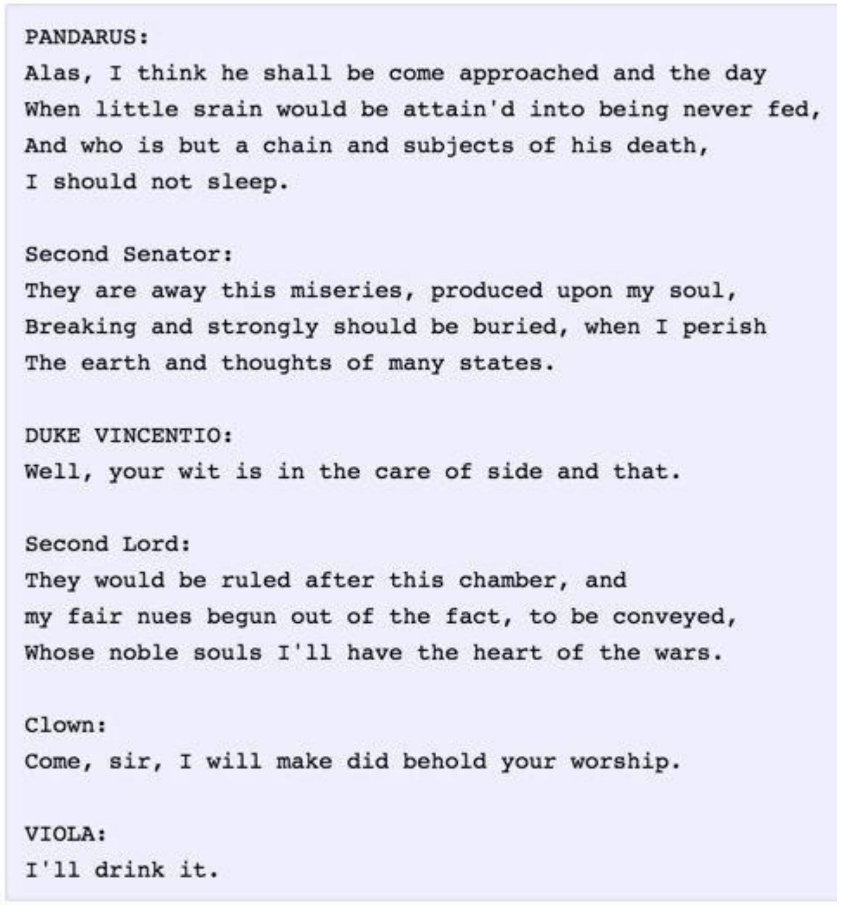
```
Aftair fall unsuch that the hall for Prince Velzonski's that me of
her hearly, and behs to so arwage fiving were to it beloge, pavu say falling misfort
how, and Gogition is so overelical and ofter.
```

train more

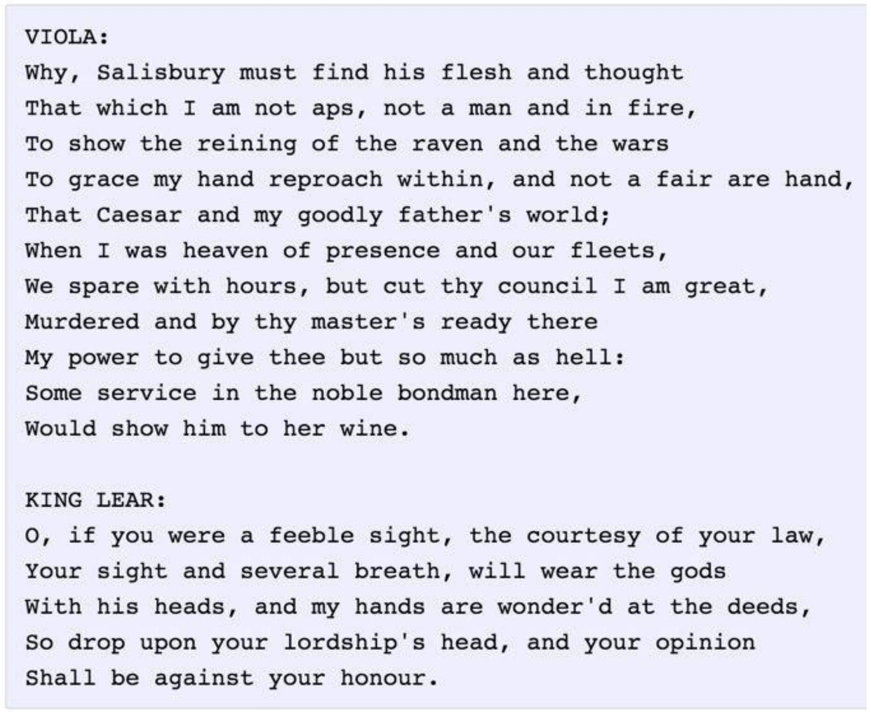
```
"Why do what that day," replied Natasha, and wishing to himself the fact the
princess, Princess Mary was easier, fed in had oftened him.
Pierre aking his soul came to the packs and drove up his father-in-law women.
```

More info: http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Generating poetry with RNNs

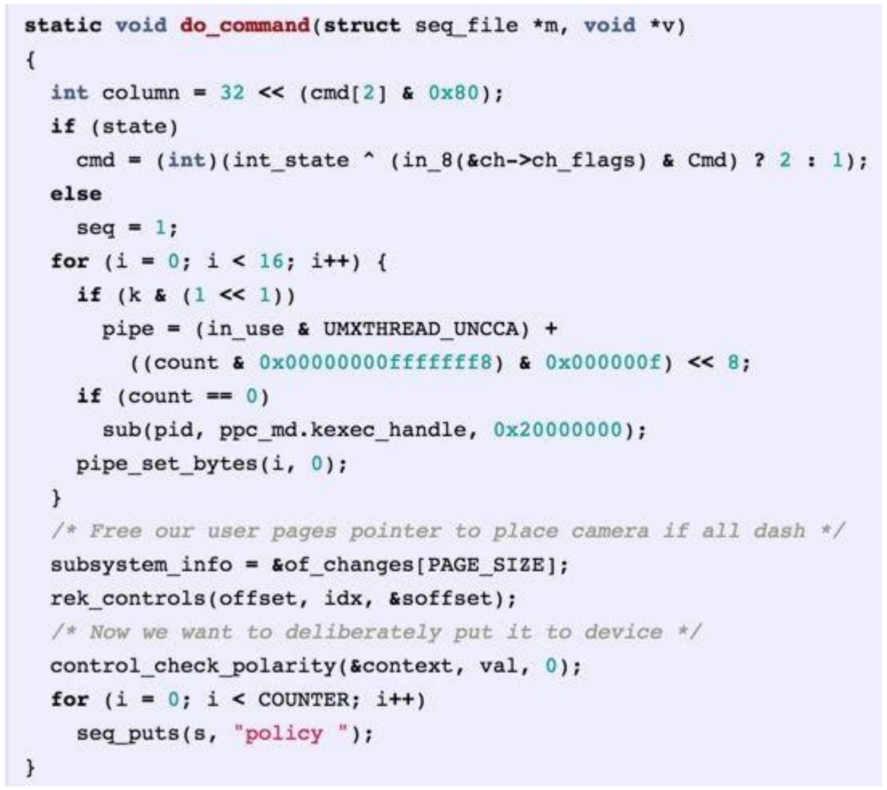


```
PANDARUS:
Alas, I think he shall be come approached and the day
When little srain would be attain'd into being never fed,
And who is but a chain and subjects of his death,
I should not sleep.

Second Senator:
They are away this miseries, produced upon my soul,
Breaking and strongly should be buried, when I perish
The earth and thoughts of many states.

DUKE VINCENTIO:
Well, your wit is in the care of side and that.

Second Lord:
They would be ruled after this chamber, and
my fair nues begun out of the fact, to be conveyed,
Whose noble souls I'll have the heart of the wars.

Clown:
Come, sir, I will make did behold your worship.

VIOLA:
I'll drink it.
```



```
VIOLA:
Why, Salisbury must find his flesh and thought
That which I am not aps, not a man and in fire,
To show the reining of the raven and the wars
To grace my hand reproach within, and not a fair are hand,
That Caesar and my goodly father's world;
When I was heaven of presence and our fleets,
We spare with hours, but cut thy council I am great,
Murdered and by thy master's ready there
My power to give thee but so much as hell:
Some service in the noble bondman here,
Would show him to her wine.

KING LEAR:
O, if you were a feeble sight, the courtesy of your law,
Your sight and several breath, will wear the gods
With his heads, and my hands are wonder'd at the deeds,
So drop upon your lordship's head, and your opinion
Shall be against your honour.
```

Generating code with RNNs



```
static void do_command(struct seq_file *m, void *v)
{
  int column = 32 << (cmd[2] & 0x80);
  if (state)
    cmd = (int)(int_state ^ (in_8(&ch->ch_flags) & Cmd) ? 2 : 1);
  else
    seq = 1;
  for (i = 0; i < 16; i++) {
    if (k & (1 << 1))
      pipe = (in_use & UMXTHREAD_UNCCA) +
        ((count & 0x00000000fffffff8) & 0x000000f) << 8;
    if (count == 0)
      sub(pid, ppc_md.kexec_handle, 0x20000000);
    pipe_set_bytes(i, 0);
  }
  /* Free our user pages pointer to place camera if all dash */
  subsystem_info = &of_changes[PAGE_SIZE];
  rek_controls(offset, idx, &soffset);
  /* Now we want to deliberately put it to device */
  control_check_polarity(&context, val, 0);
  for (i = 0; i < COUNTER; i++)
    seq_puts(s, "policy ");
}
```

Andrej Karpathy

GRUs

Gated Recurrent Units (GRUs)

- More complex hidden unit computation in recurrence!
- Introduced by Cho et al. 2014
- Main ideas:
  - keep around memories to capture long distance dependencies
  - allow error messages to flow at different strengths depending on the inputs

Gated Recurrent Units (GRUs)

- Standard RNN computes hidden layer at next time step directly: $h_t = f\left(W^{(hh)}h_{t-1} + W^{(hx)}x_t\right)$
- GRU first computes an update **gate** (another layer) based on current input word vector and hidden state

$$z_t = \sigma\left(W^{(z)}x_t + U^{(z)}h_{t-1}\right)$$

- Compute reset gate similarly but with different weights

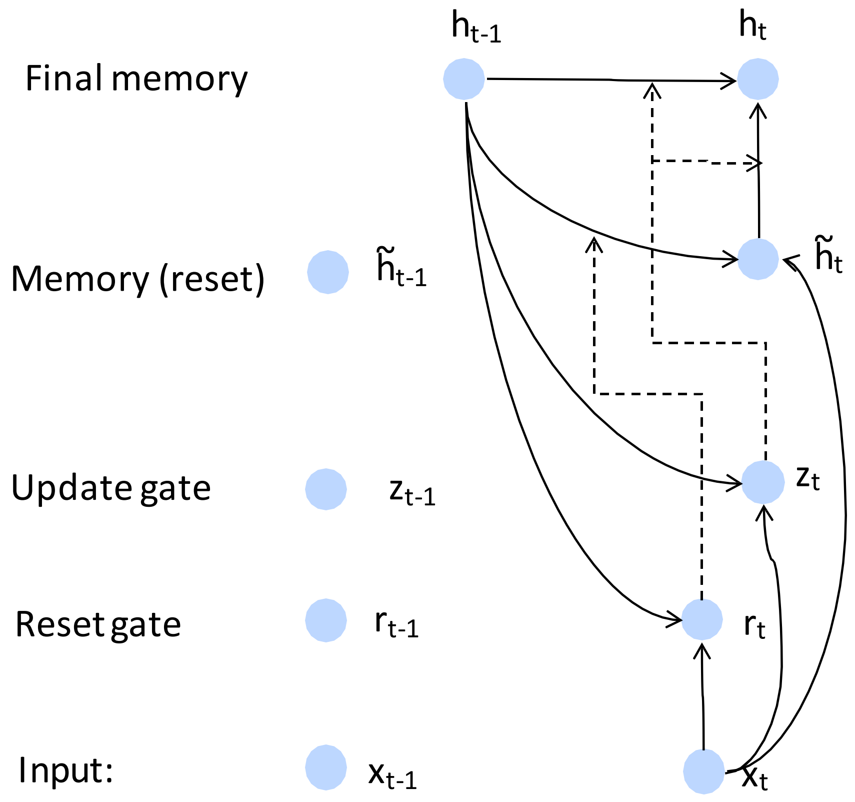
$$r_t = \sigma\left(W^{(r)}x_t + U^{(r)}h_{t-1}\right)$$

Richard Socher

Gated Recurrent Units (GRUs)

- Update gate $z_t = \sigma\left(W^{(z)}x_t + U^{(z)}h_{t-1}\right)$
- Reset gate $r_t = \sigma\left(W^{(r)}x_t + U^{(r)}h_{t-1}\right)$
- New memory content: $\tilde{h}_t = \tanh\left(Wx_t + r_t \circ Uh_{t-1}\right)$
  If reset gate unit is ~0, then this ignores previous memory and only stores the new word information
- Final memory at time step combines current and previous time steps: $h_t = z_t \circ h_{t-1} + (1 - z_t) \circ \tilde{h}_t$

Gated Recurrent Units (GRUs)



$$z_t = \sigma\left(W^{(z)}x_t + U^{(z)}h_{t-1}\right)$$
$$r_t = \sigma\left(W^{(r)}x_t + U^{(r)}h_{t-1}\right)$$
$$\tilde{h}_t = \tanh\left(Wx_t + r_t \circ Uh_{t-1}\right)$$
$$h_t = z_t \circ h_{t-1} + (1 - z_t) \circ \tilde{h}_t$$

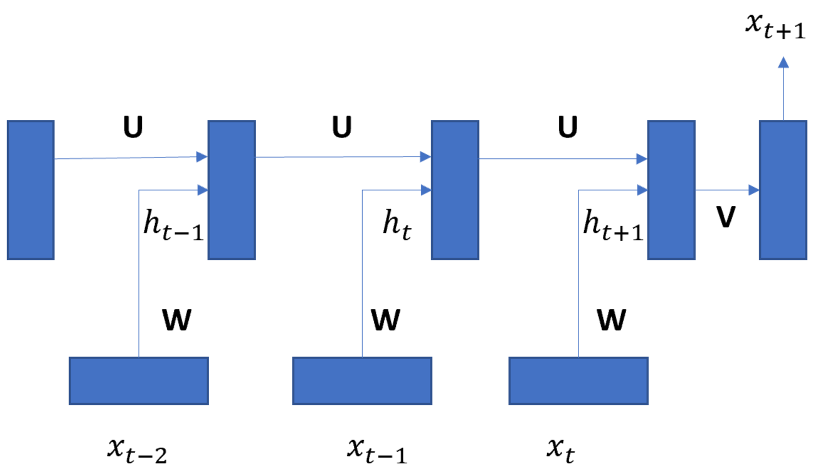
Gated Recurrent Units (GRUs)

$$z_t = \sigma\left(W^{(z)}x_t + U^{(z)}h_{t-1}\right)$$
$$r_t = \sigma\left(W^{(r)}x_t + U^{(r)}h_{t-1}\right)$$
$$\tilde{h}_t = \tanh\left(Wx_t + r_t \circ Uh_{t-1}\right)$$
$$h_t = z_t \circ h_{t-1} + (1 - z_t) \circ \tilde{h}_t$$

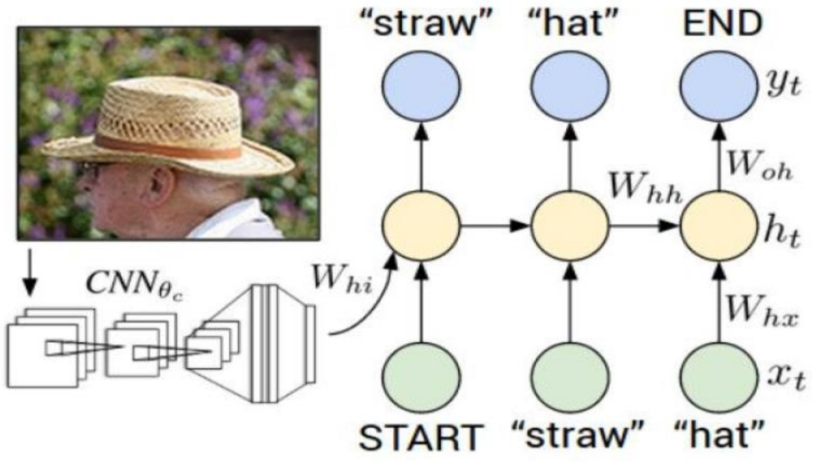
- If reset is close to 0, ignore previous hidden state: Allows model to drop information that is irrelevant in the future
- Update gate z controls how much of past state should matter now
  - If z close to 1, then we can copy information in that unit through many time steps! **Less vanishing gradient**!
- Units with short-term dependencies often have reset gates (r) very active; ones with long-term dependencies have active update gates (z)

Richard Socher

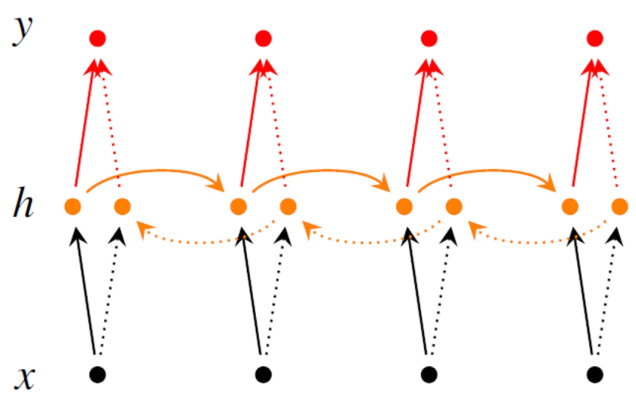
Take-aways



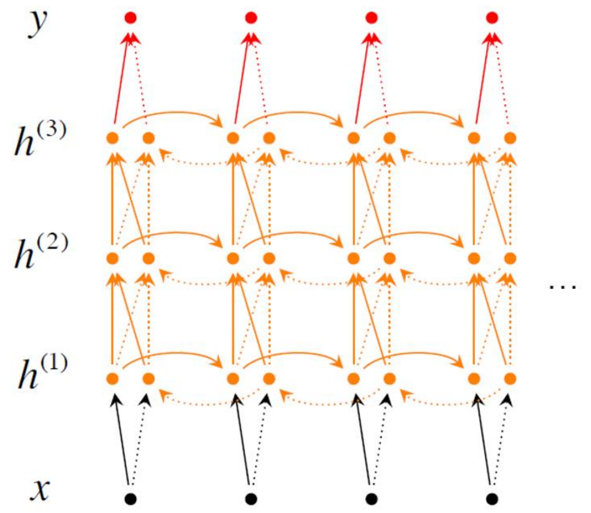
x_{t+1}
U
U
U
h_{t-1}
h_t
h_{t+1}
V
W
W
W
x_{t-2}
x_{t-1}
x_t



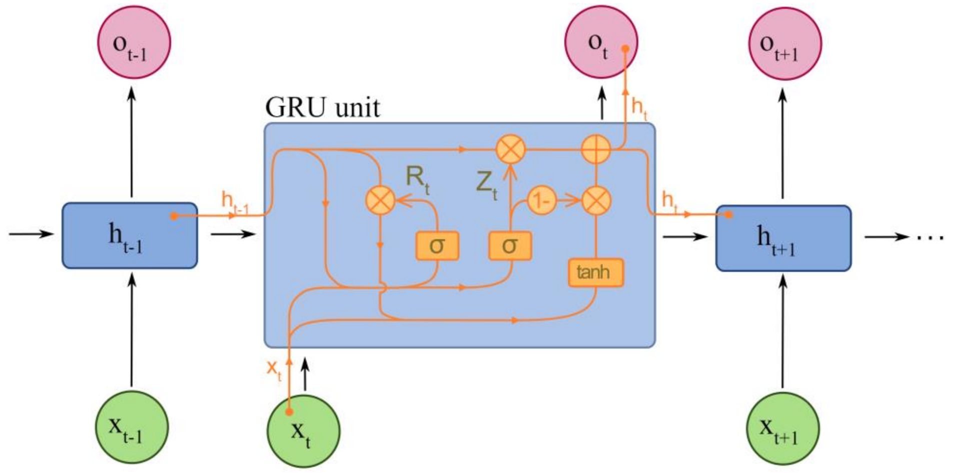
"straw"
"hat"
END
y_t
W_{oh}
W_{hh}
h_t
W_{hi}
CNN_{θ_c}
W_{hx}
x_t
START
"straw"
"hat"



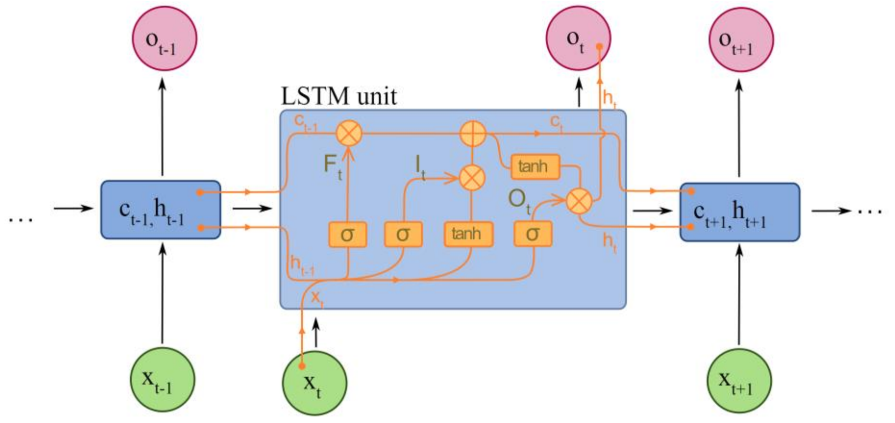
y
h
x



y
$h^{(3)}$
$h^{(2)}$
$h^{(1)}$
x
…



o_{t-1}
o_t
o_{t+1}
GRU unit
h_{t-1}
R_t
Z_t
1-
σ
σ
tanh
h_t
h_{t+1}
x_t
x_{t-1}
x_{t+1}
…



o_{t-1}
o_t
o_{t+1}
LSTM unit
c_{t-1}
c_t
F_t
I_t
O_t
tanh
σ
σ
tanh
σ
c_{t-1}, h_{t-1}
c_{t+1}, h_{t+1}
h_{t-1}
h_t
x_t
x_{t-1}
x_{t+1}
…

Encoder-decoder models, Attention models

Agenda

- Encoder-decoder models
- Attention based models

Agenda

- **Encoder-decoder models**
- Attention based models

Encoder-decoder models for machine translation

The encoder steps through the input time steps and encodes the entire sequence into a fixed length vector called a context vector.

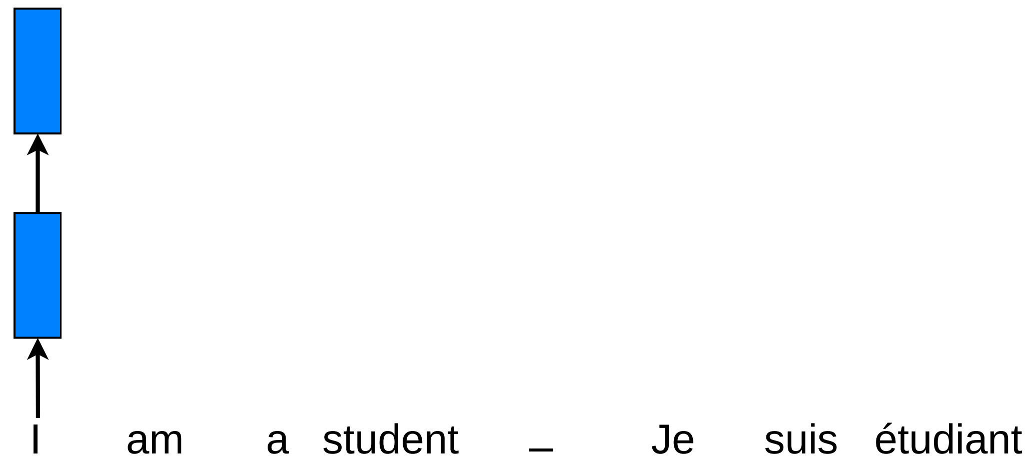
I am a student _ Je suis étudiant

- Big RNNs trained end-to-end.

Ilya Sutskever, et al. "Sequence to Sequence Learning with Neural Networks" using LSTMs.
Kyunghyun Cho, et al. "Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation".

Encoder-decoder models for machine translation

The encoder steps through the input time steps and encodes the entire sequence into a fixed length vector called a context vector.

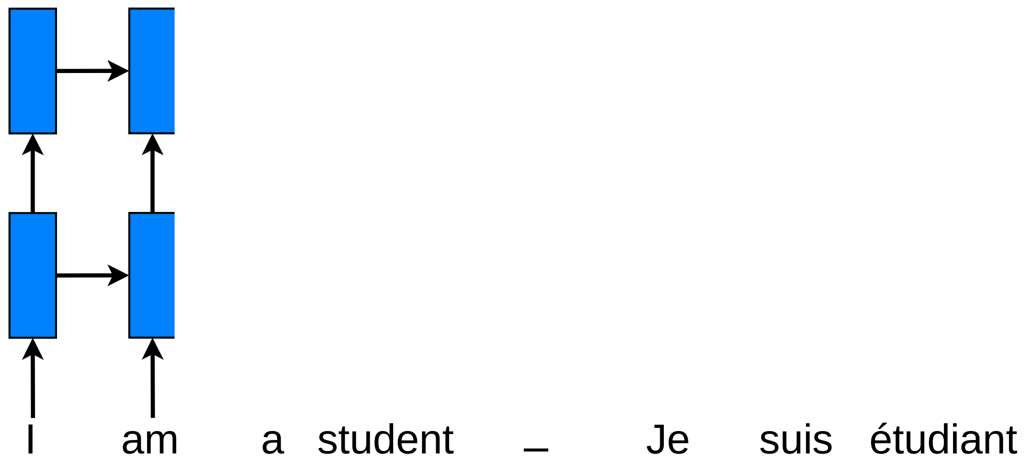


- Big RNNs trained end-to-end.

Ilya Sutskever, et al. "Sequence to Sequence Learning with Neural Networks" using LSTMs.
Kyunghyun Cho, et al. "Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation".

Encoder-decoder models for machine translation

The encoder steps through the input time steps and encodes the entire sequence into a fixed length vector called a context vector.

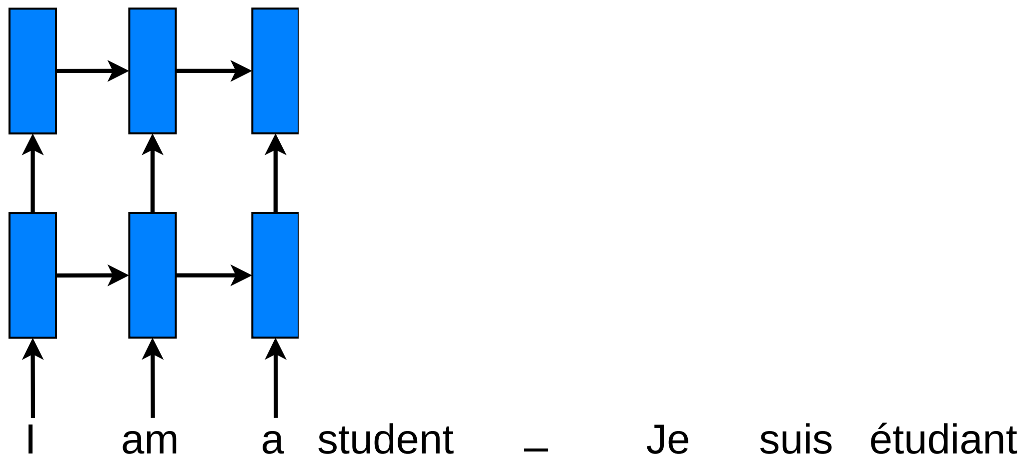


- Big RNNs trained end-to-end.

Ilya Sutskever, et al. "Sequence to Sequence Learning with Neural Networks" using LSTMs.
Kyunghyun Cho, et al. "Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation".

Encoder-decoder models for machine translation

The encoder steps through the input time steps and encodes the entire sequence into a fixed length vector called a context vector.

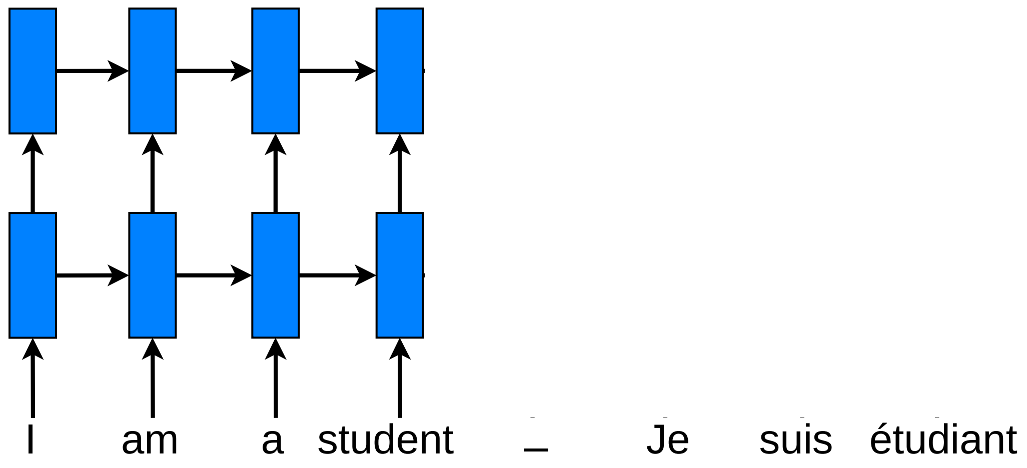


- Big RNNs trained end-to-end.

Ilya Sutskever, et al. "Sequence to Sequence Learning with Neural Networks" using LSTMs.
Kyunghyun Cho, et al. "Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation".

Encoder-decoder models for machine translation

The encoder steps through the input time steps and encodes the entire sequence into a fixed length vector called a context vector.

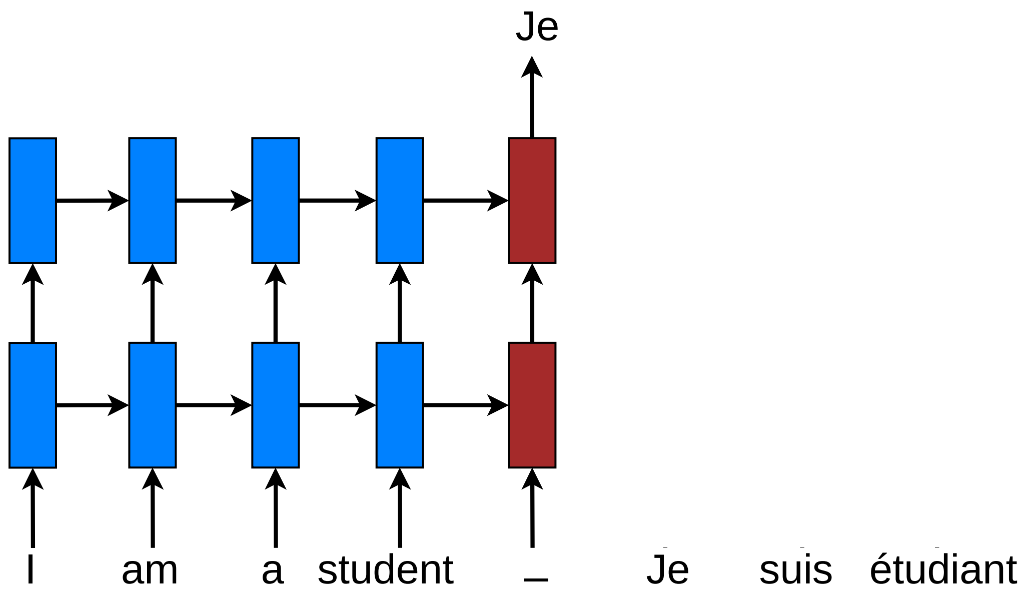


- Big RNNs trained end-to-end.

Ilya Sutskever, et al. "Sequence to Sequence Learning with Neural Networks" using LSTMs.
Kyunghyun Cho, et al. "Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation".

Encoder-decoder models for machine translation

The encoder steps through the input time steps and encodes the entire sequence into a fixed length vector called a context vector.



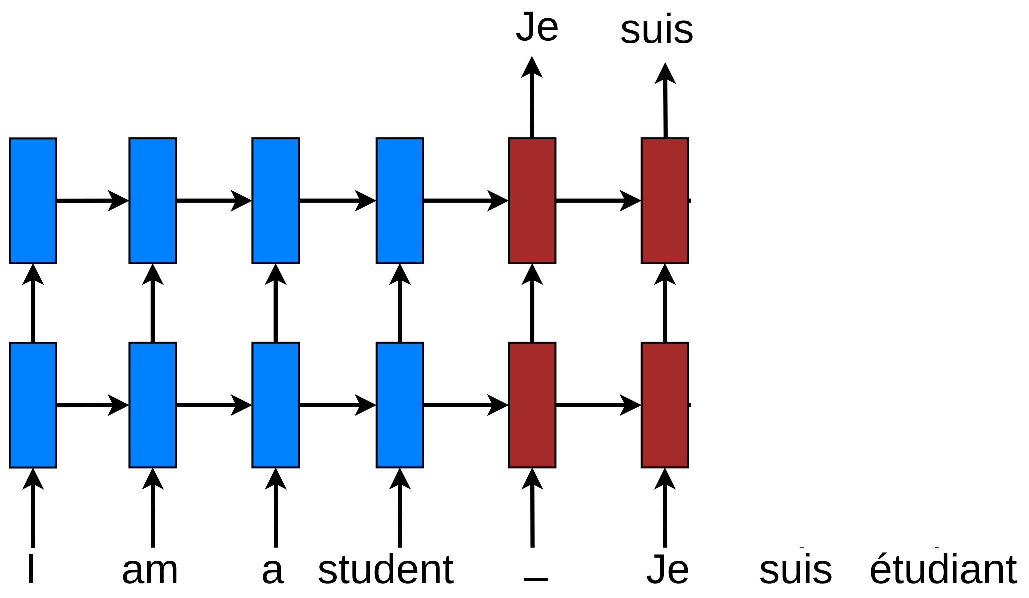
The decoder steps through the output time steps while reading from the context vector.

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

Ilya Sutskever, et al. “Sequence to Sequence Learning with Neural Networks” using LSTMs.
Kyunghyun Cho, et al. “Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation”.

Encoder-decoder models for machine translation

The encoder steps through the input time steps and encodes the entire sequence into a fixed length vector called a context vector.



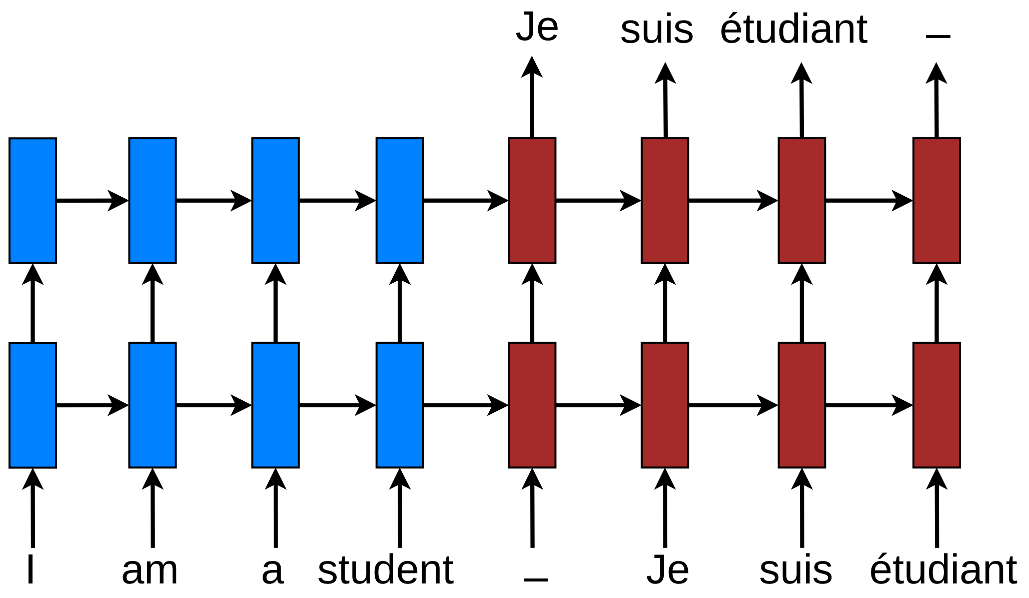
The decoder steps through the output time steps while reading from the context vector.

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

Ilya Sutskever, et al. “Sequence to Sequence Learning with Neural Networks” using LSTMs.
Kyunghyun Cho, et al. “Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation”.

Encoder-decoder models for machine translation

The encoder steps through the input time steps and encodes the entire sequence into a fixed length vector called a context vector.



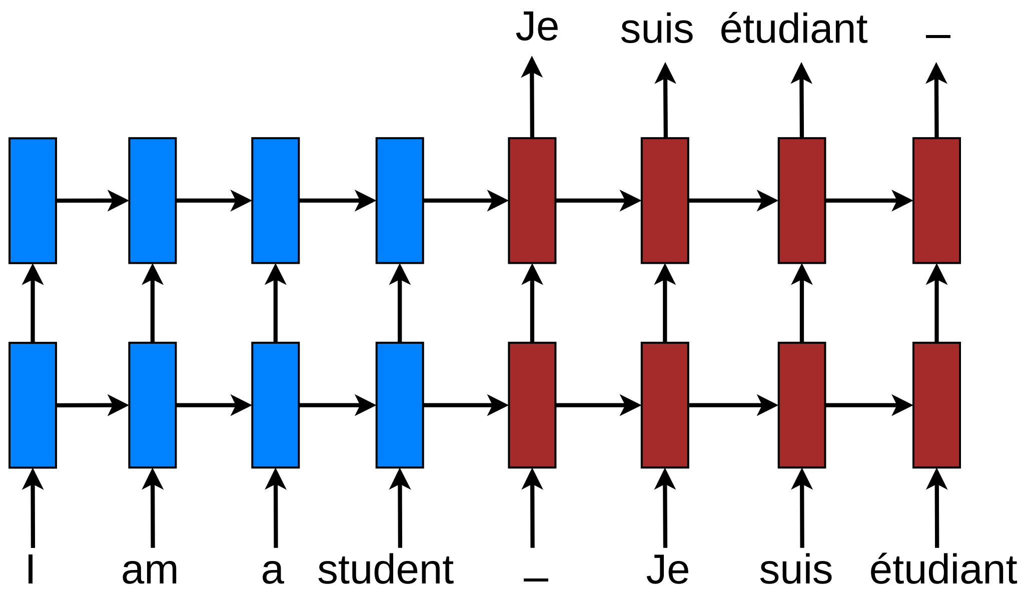
The decoder steps through the output time steps while reading from the context vector.

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

Ilya Sutskever, et al. "Sequence to Sequence Learning with Neural Networks" using LSTMs.
Kyunghyun Cho, et al. "Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation".

Encoder-decoder models for machine translation

The encoder steps through the input time steps and encodes the entire sequence into a fixed length vector called a context vector.



The decoder steps through the output time steps while reading from the context vector.

- Big RNNs trained end-to-end: encoder-decoder.
  - Generalize well to long sequences.
  - Small memory footprint.
  - Simple decoder.

Ilya Sutskever, et al. "Sequence to Sequence Learning with Neural Networks" using LSTMs.
Kyunghyun Cho, et al. "Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation".

Beam search for decoding

- We search for the most likely translation using a simple left-to-right beam search decoder which maintains a small number B of partial hypotheses, where a partial hypothesis is a prefix of some translation.
- At each timestep we extend each partial hypothesis in the beam with every possible word in the vocabulary.
- This greatly increases the number of the hypotheses so we discard all but the B most likely hypotheses according to the model's log probability.
- As soon as the "<EOS>" symbol is appended to a hypothesis, it is removed from the beam and is added to the set of complete hypotheses.

Ilya Sutskever, et al. "Sequence to Sequence Learning with Neural Networks" using LSTMs.
Kyunghyun Cho, et al. "Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation".

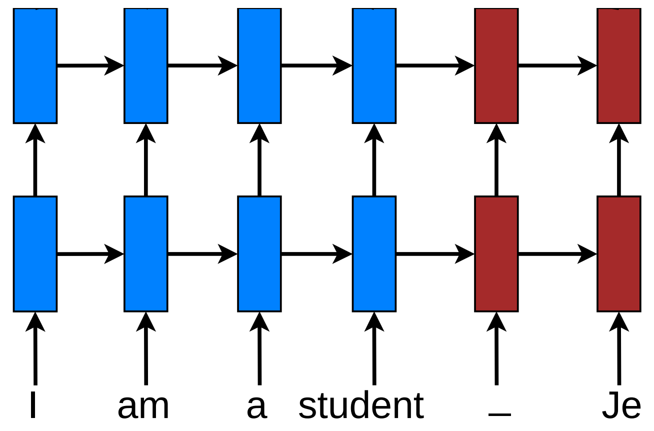
Agenda

- Encoder-decoder models
- **Attention based models**

Issue with encoder decoder models

- A neural network needs to be able to compress all the necessary information of a source sentence into a fixed-length vector.
- This may make it difficult for the neural network to cope with long sentences, especially those that are longer than the sentences in the training corpus.
- Solution:
  - Each time the proposed model generates a word in a translation, it (soft-)searches for a set of positions in a source sentence where the most relevant information is concentrated.
  - The model then predicts a target word based on the context vectors associated with these source positions and all the previous generated target words.

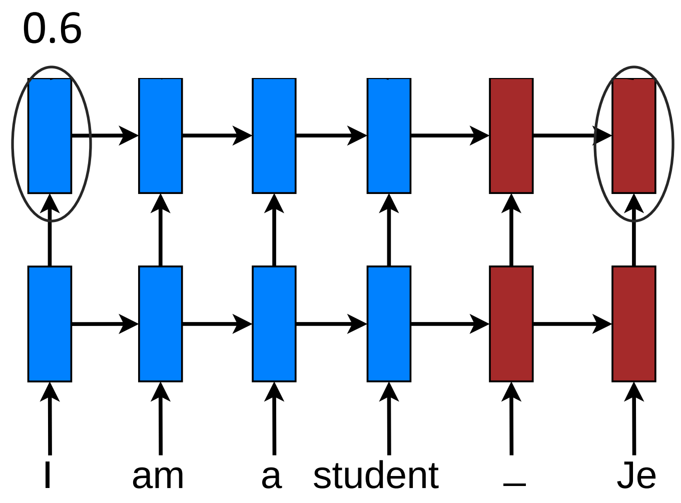
Attention Mechanism



- Maintain a memory of source hidden states

Bahdanau, Dzmitry, Kyung Hyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." In *3rd International Conference on Learning Representations, ICLR 2015*. 2015.

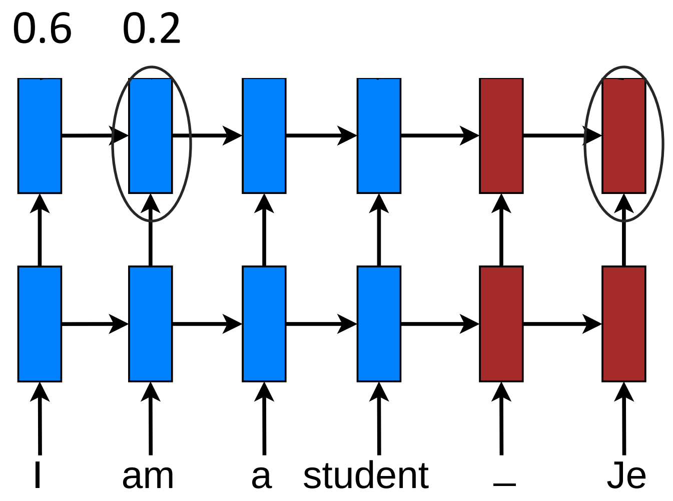
Attention Mechanism



- Maintain a memory of source hidden states
  - Compare target and source hidden states

Bahdanau, Dzmitry, Kyung Hyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." In *3rd International Conference on Learning Representations, ICLR 2015*. 2015.

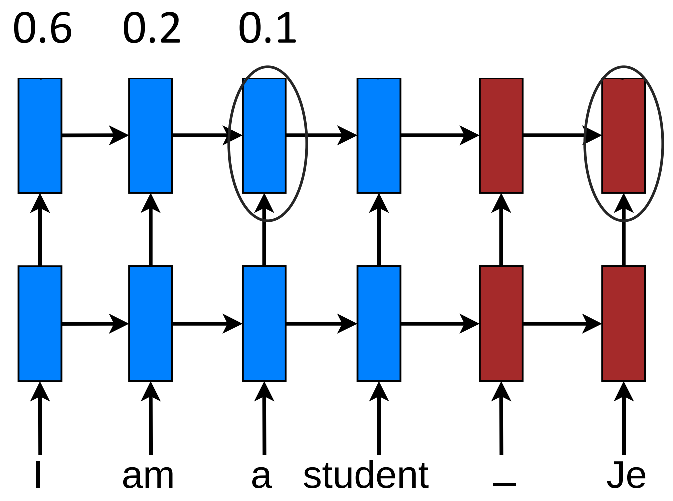
Attention Mechanism



- Maintain a memory of source hidden states
  - Compare target and source hidden states

Bahdanau, Dzmitry, Kyung Hyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." In *3rd International Conference on Learning Representations, ICLR 2015*. 2015.

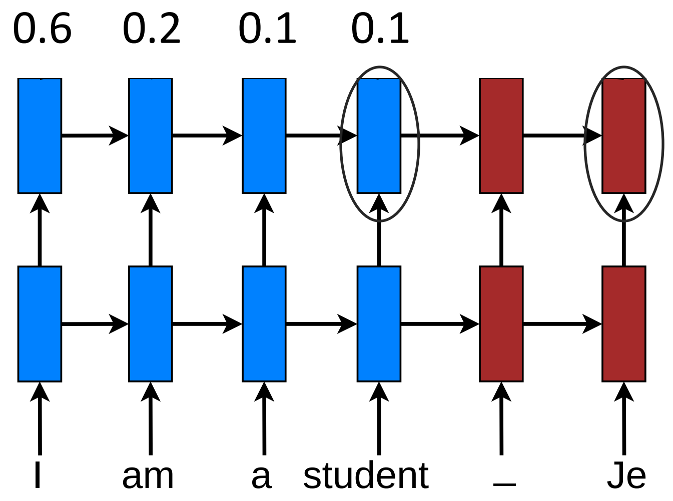
Attention Mechanism



- Maintain a memory of source hidden states
  - Compare target and source hidden states

Bahdanau, Dzmitry, Kyung Hyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." In *3rd International Conference on Learning Representations, ICLR 2015*. 2015.

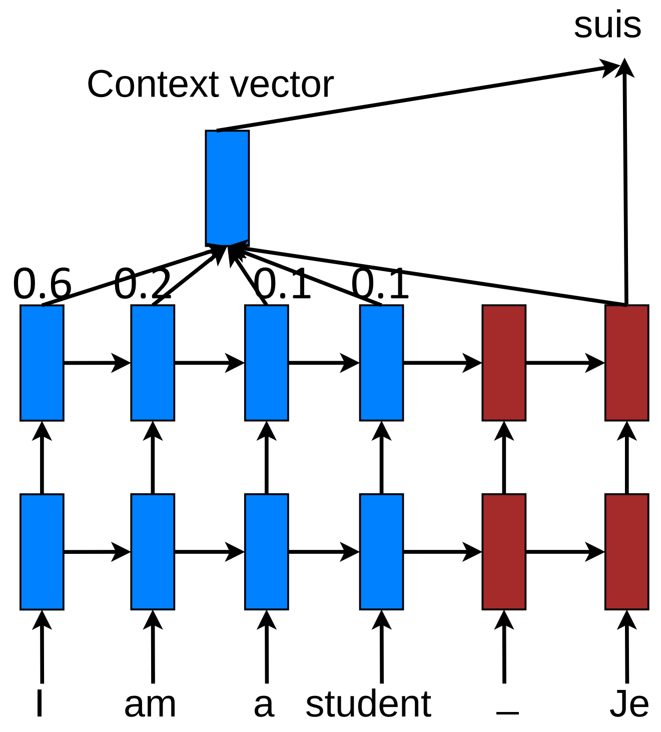
Attention Mechanism



- Maintain a memory of source hidden states
  - Compare target and source hidden states

Bahdanau, Dzmitry, Kyung Hyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." In *3rd International Conference on Learning Representations, ICLR 2015*. 2015.

Attention Mechanism



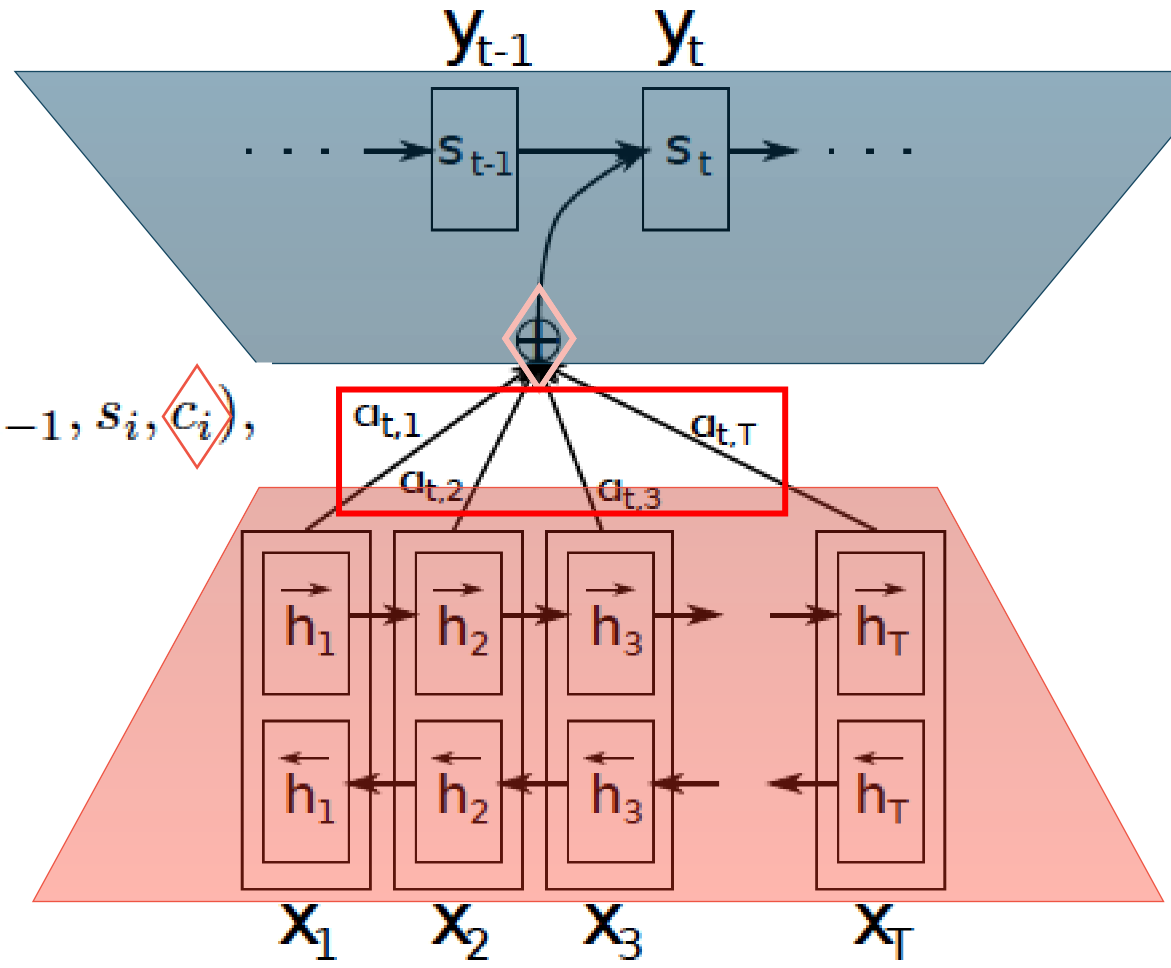
- Maintain a memory of source hidden states
  - Able to translate long sentences.

Bahdanau, Dzmitry, Kyung Hyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." In *3rd International Conference on Learning Representations, ICLR 2015*. 2015.

Jointly Learning to Align and Translate

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

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



Bahdanau, Dzmitry, Kyung Hyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." In *3rd International Conference on Learning Representations, ICLR 2015*. 2015.

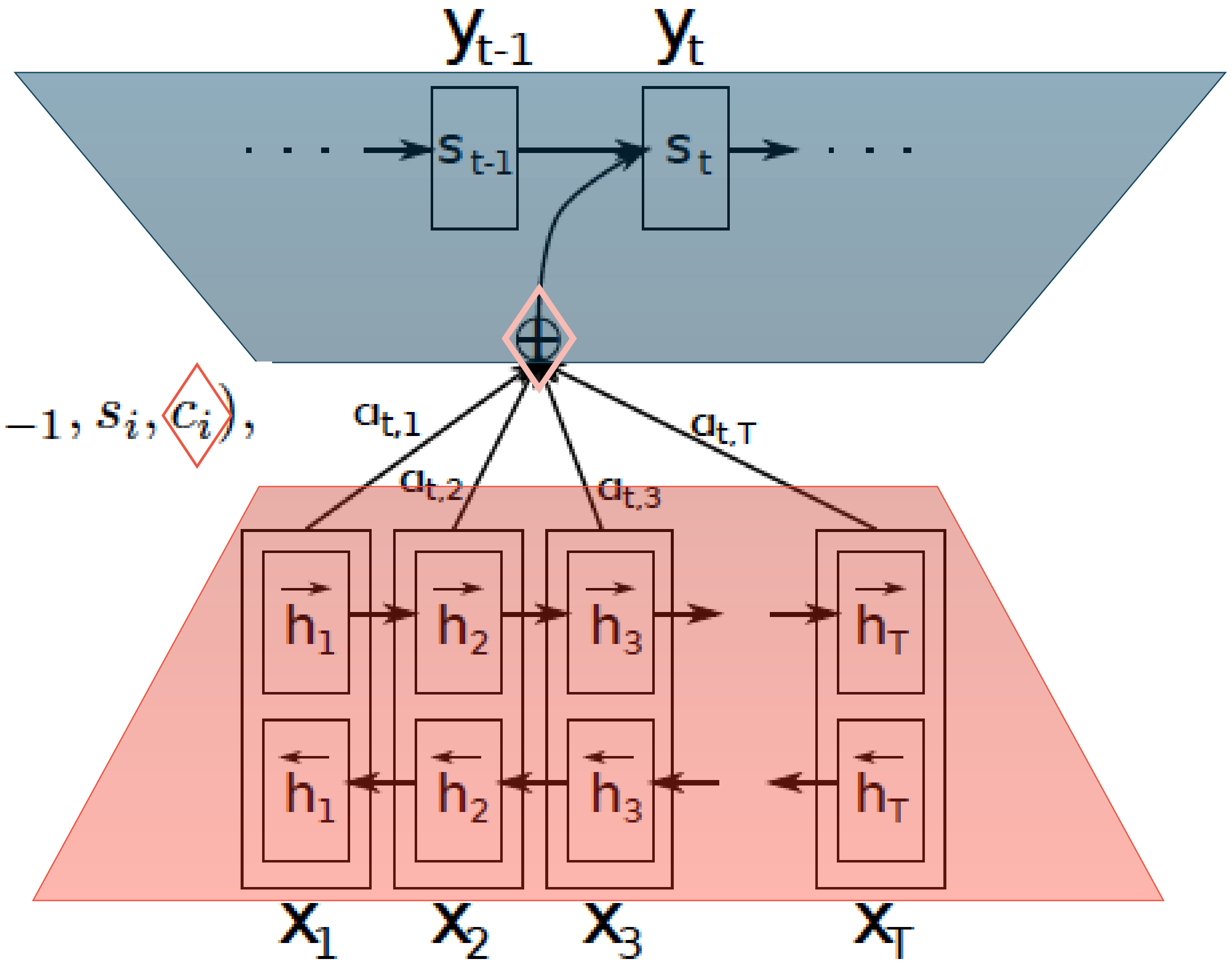
Jointly Learning to Align and Translate

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

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

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

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



Bahdanau, Dzmitry, Kyung Hyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." In *3rd International Conference on Learning Representations, ICLR 2015*. 2015.

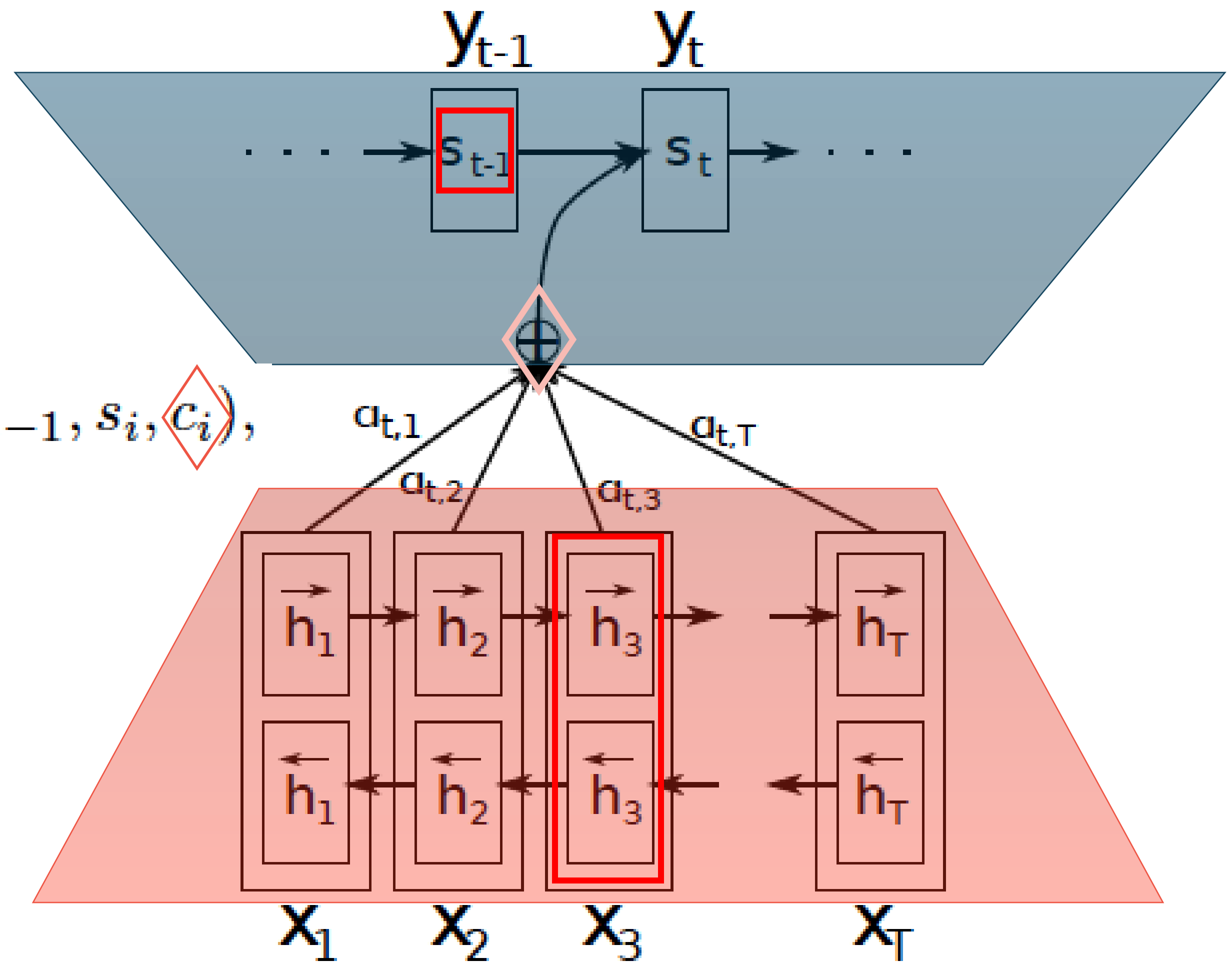
Jointly Learning to Align and Translate

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

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

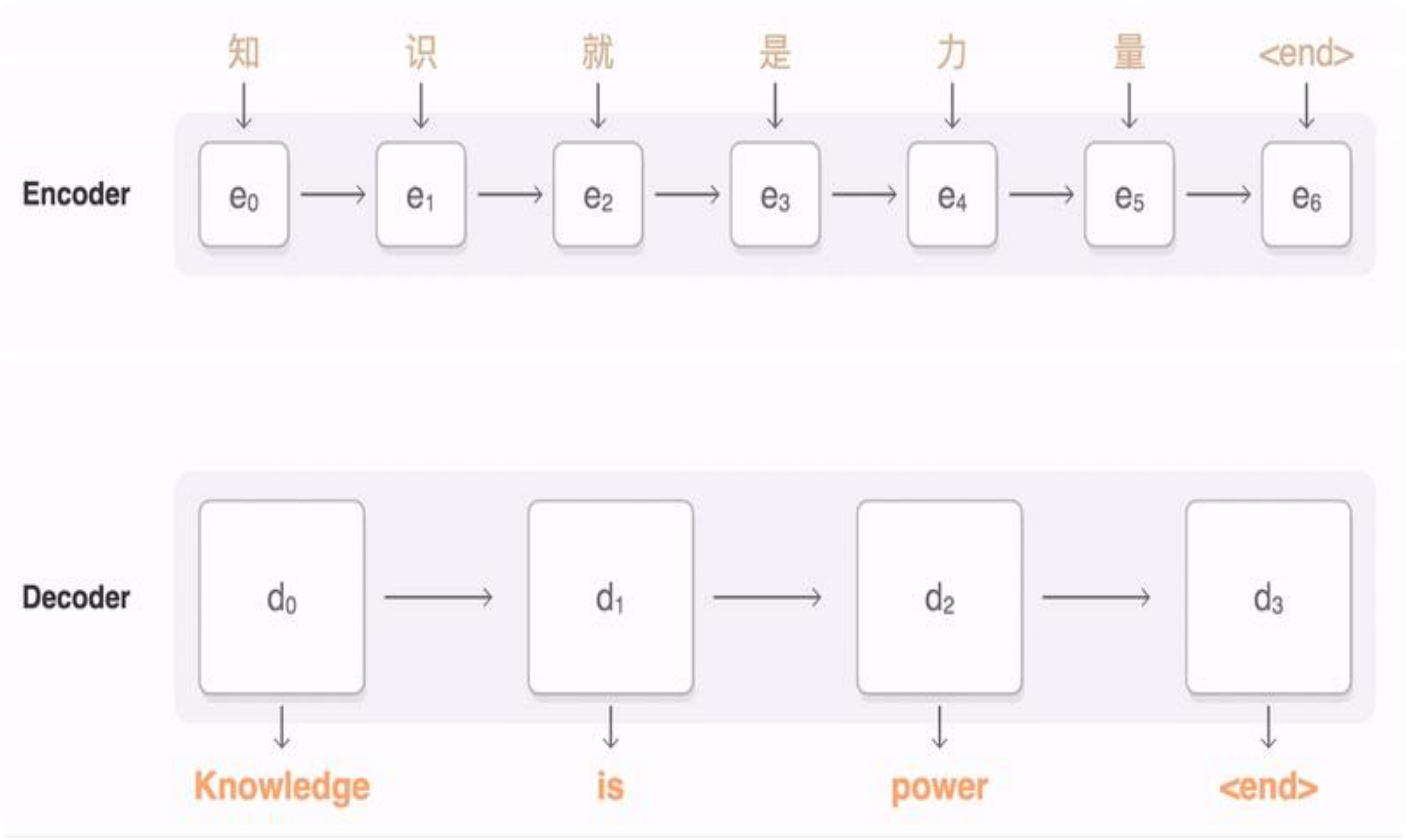
$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x}\exp(e_{ik})},$$

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



Bahdanau, Dzmitry, Kyung Hyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." In *3rd International Conference on Learning Representations, ICLR 2015*. 2015.

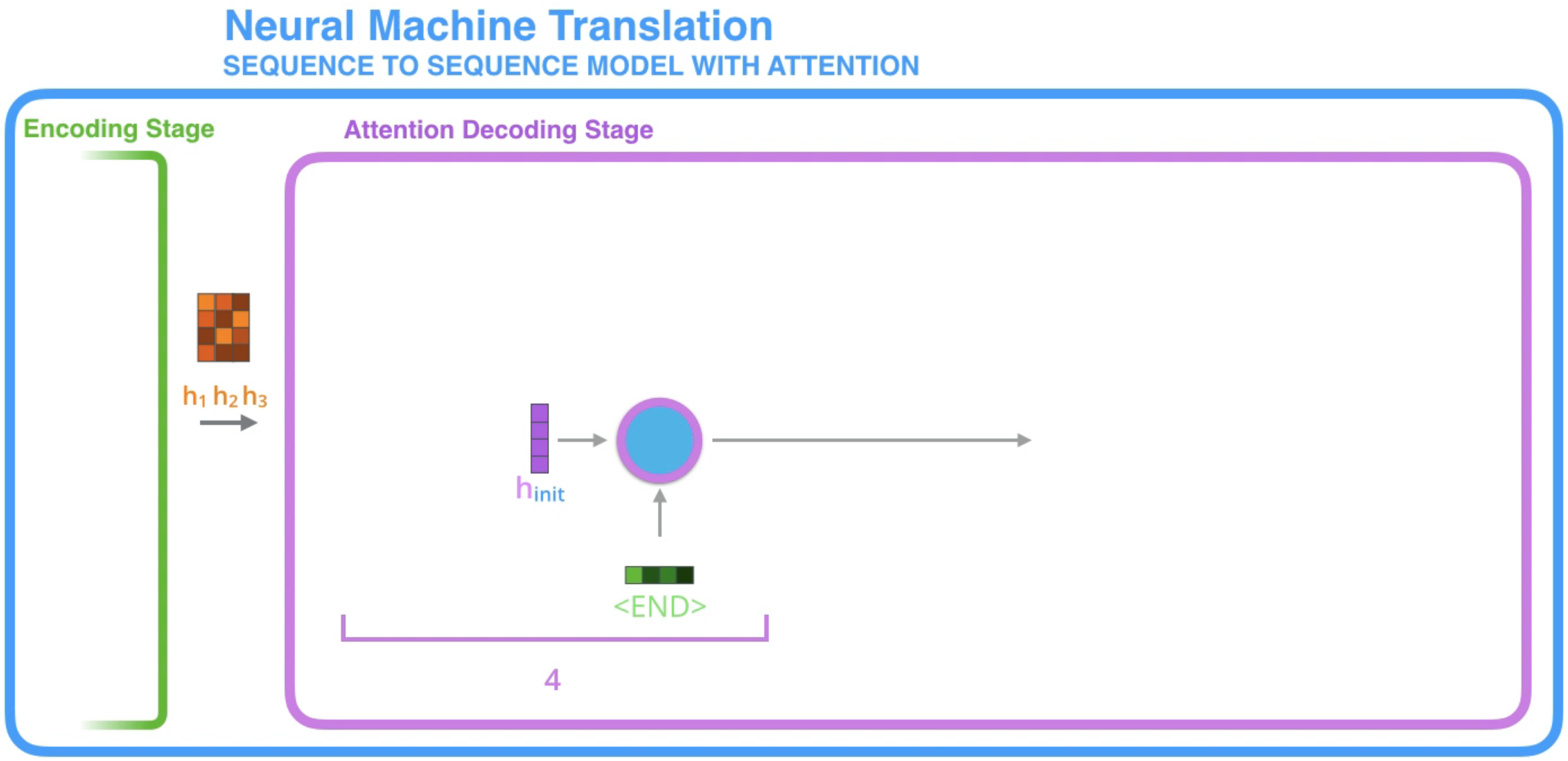
Attention Network



Attention Mechanism

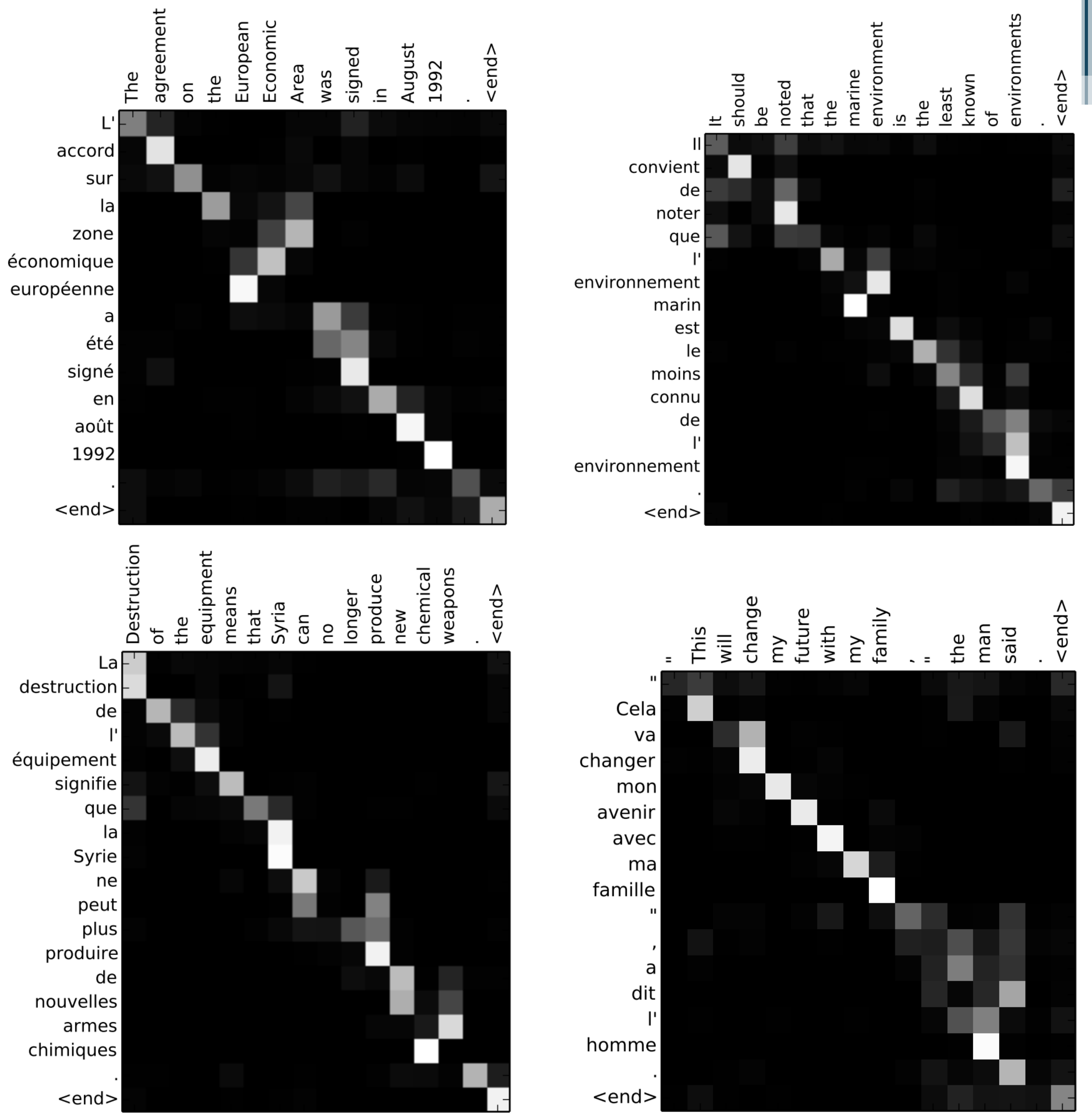
Attention at time step 4

Attention Mechanism


Neural Machine Translation
SEQUENCE TO SEQUENCE MODEL WITH ATTENTION
Encoding Stage
Attention Decoding Stage
h1 h2 h3
hinit
<END>
4

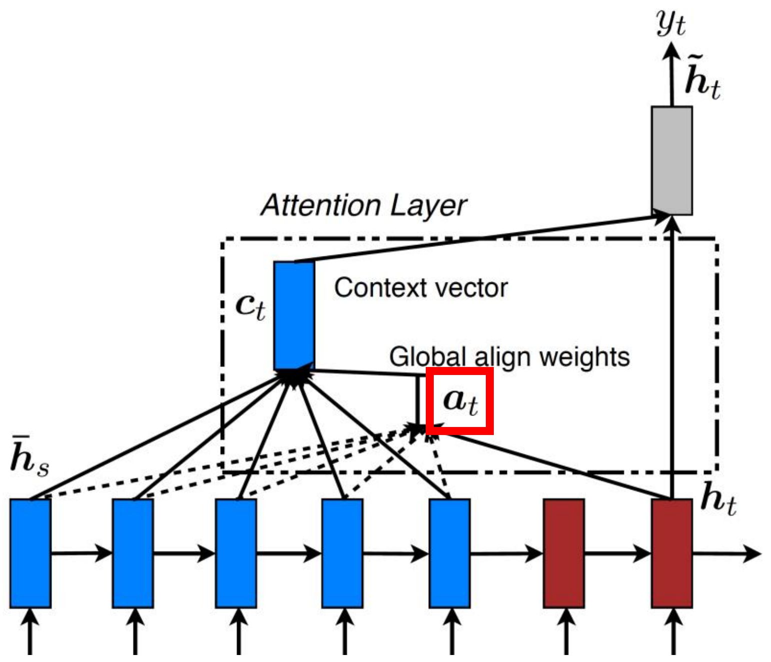
Alignments (α_{ij})

- Alignment of words between English and French is largely monotonic.
- Adjectives and nouns are typically ordered differently between French and English.
  - [European Economic Area] to [zone´economique europ´een].
- Multi-word examples
  - [The equipment] to [l’ equipement]



Bahdanau, Dzmitry, Kyung Hyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." In *3rd International Conference on Learning Representations, ICLR 2015*. 2015.

Global Attention

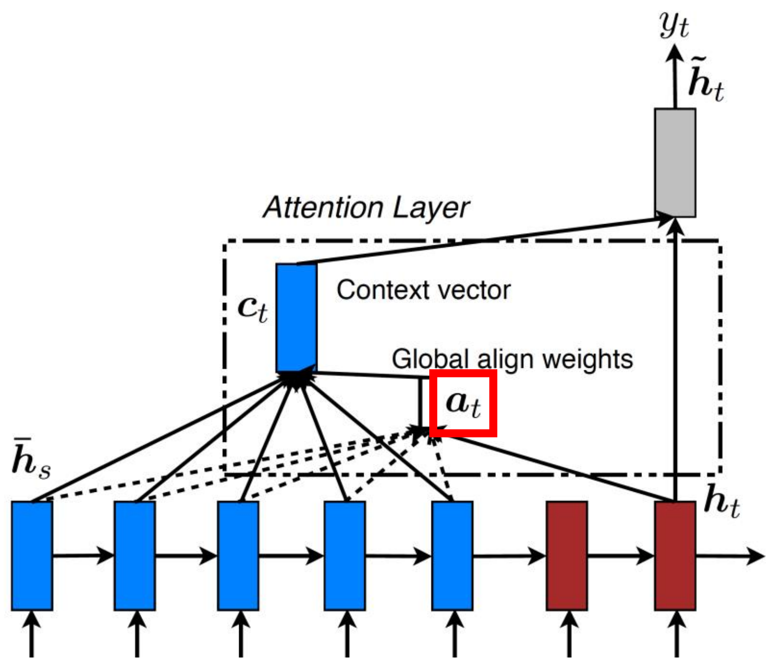


- Alignment weight vector:

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

Effective Approaches to Attention-based Neural Machine Translation, 2015. Luong15_emnlp

Global Attention



Global attention has a drawback that it has to attend to all words on the source side for each target word, which is expensive

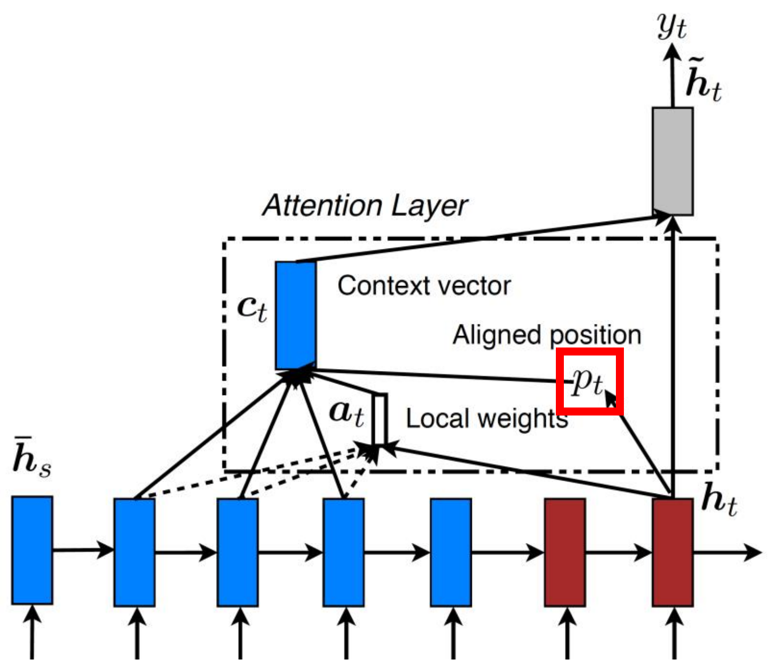
- Alignment weight vector:

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

(Bahdanau et al., 15)

Effective Approaches to Attention-based Neural Machine Translation, 2015. Luong15_emnlp

Local Attention



aligned positions?

- Monotonic alignment (local-m): set p_t = t
- Predictive alignment (local-p):

$$p_t = S \cdot \text{sigmoid}(\boldsymbol{v}_p^\top \tanh(\boldsymbol{W_p h}_t)), \qquad (9)$$

$\boldsymbol{W_p}$ and $\boldsymbol{v}_p$ are the model parameters which will be learned to predict positions. S is the source sentence length. As a result of sigmoid, $p_t \in [0, S]$.

- p_t defines a focused w $[p_t - D, p_t + D]$.

Effective Approaches to Attention-based Neural Machine Translation, 2015. Luong15_emnlp

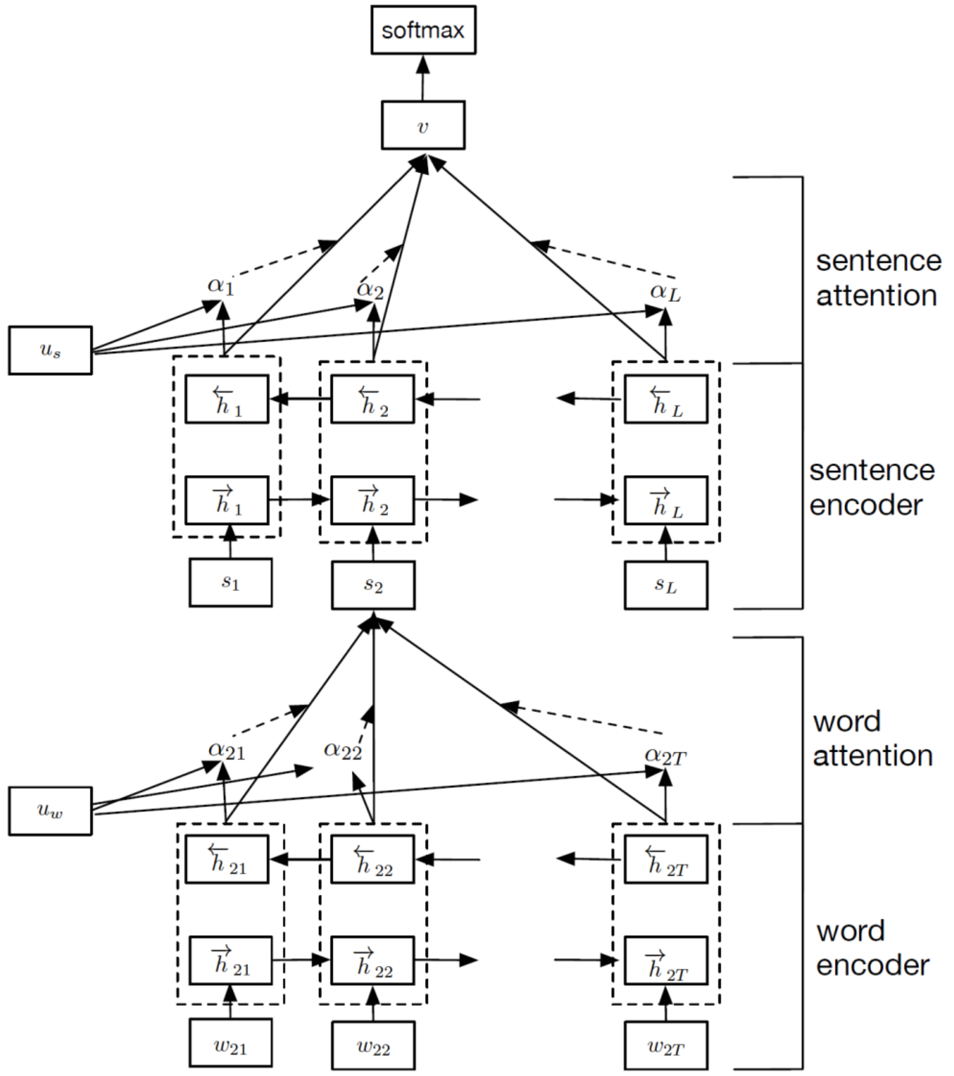
Hierarchical Attention Network (HAN) architecture

- Assume a doc has L sentences s_i and each sentence contains T_i words w_{it}.
- Aim is to learn a representation for the document.
- Context vector u_w is randomly initialized and jointly learned during the training process.
- Sentence attention uses a sentence level context vector u_s.

$$u_{it} = \tanh(W_w h_{it} + b_w)$$

$$\alpha_{it} = \frac{\exp(u_{it}^\top u_w)}{\sum_t \exp(u_{it}^\top u_w)}$$

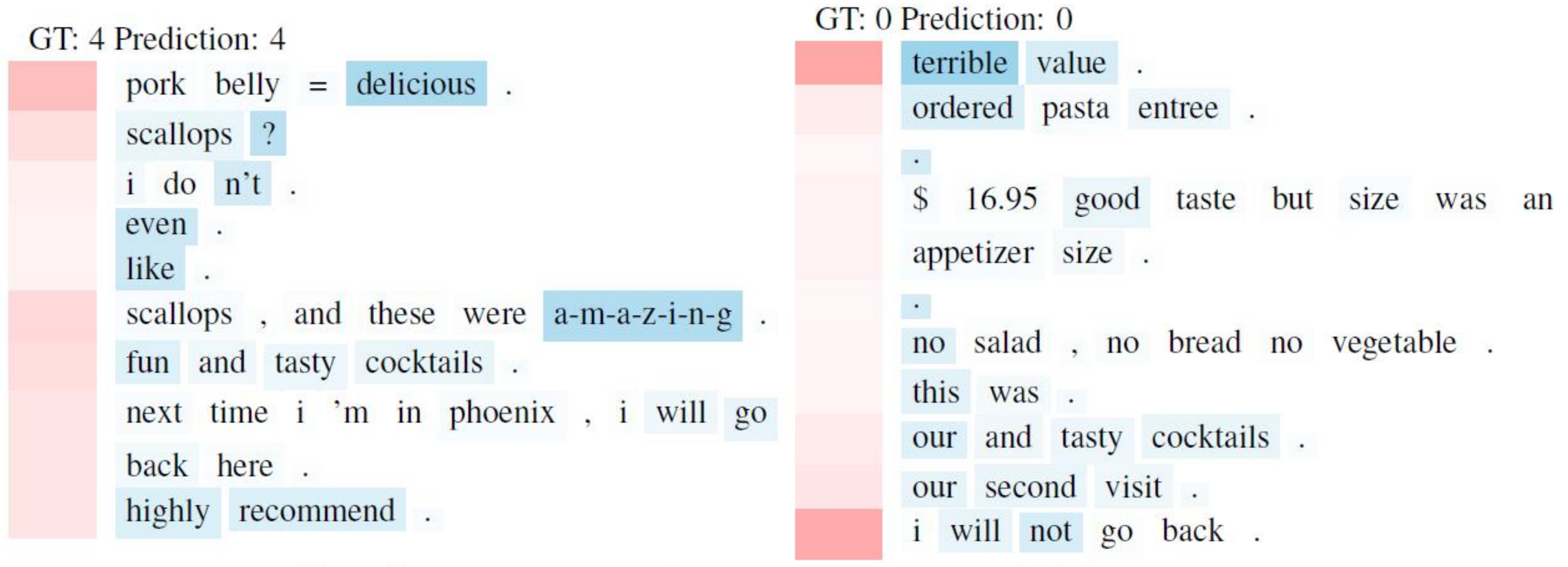
$$s_i = \sum_t \alpha_{it} h_{it}.$$



Yang, Zichao, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. "Hierarchical attention networks for document classification." NAACL, pp. 1480-1489. 2016.

Hierarchical Attention Network (HAN)

- Two levels of attention
  - one at word level and other at sentence level
- Figure 5 shows that our model can select the words carrying strong sentiment like delicious, amazing, terrible and their corresponding sentences.
- Sentences containing many words like cocktails, pasta, entree are disregarded. Note that our model can not only select words carrying strong sentiment, it can also deal with complex across-sentence context.
- For example, there are sentences like i don't even like scallops in the first document of Fig. 5, if looking purely at the single sentence, we may think this is negative comment.
- However, our model looks at the context of this sentence and figures out this is a positive review and chooses to ignore this sentence.
- For the left document in Figure 6 with label 1, which denotes Science and Mathematics, our model accurately localizes the words zebra, strips, camouflage, predator and their corresponding sentences. For the right document with label 4, which denotes Computers and Internet, our model focuses on web, searches, browsers and their corresponding sentences.



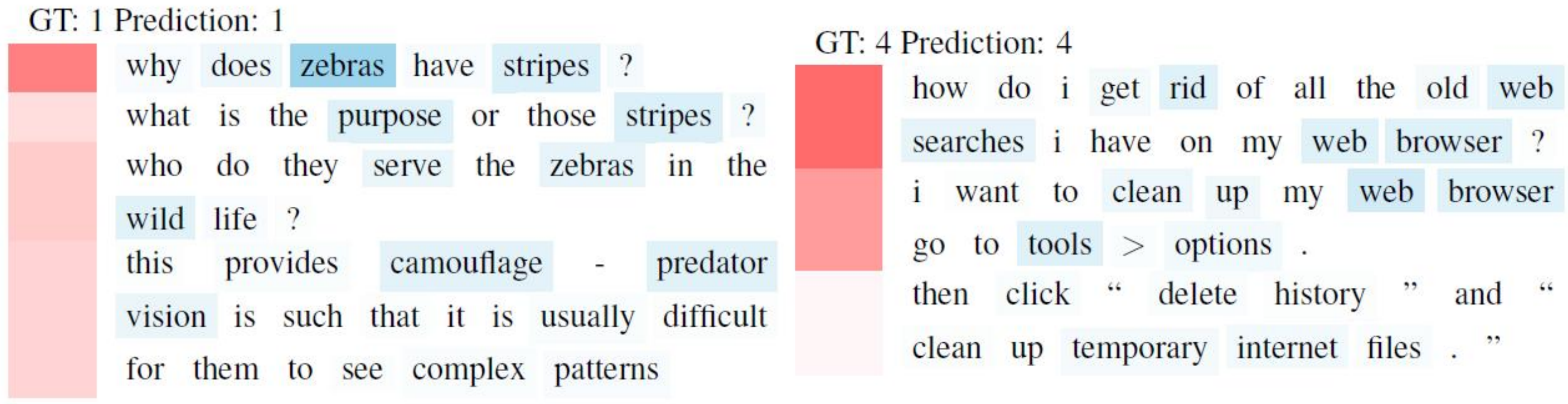
GT: 4 Prediction: 4

pork belly = delicious .
scallops ?
i do n't .
even .
like .
scallops , and these were a-m-a-z-i-n-g .
fun and tasty cocktails .
next time i 'm in phoenix , i will go
back here .
highly recommend .

GT: 0 Prediction: 0

terrible value .
ordered pasta entree .
.
$ 16.95 good taste but size was an
appetizer size .
.
no salad , no bread no vegetable .
this was .
our and tasty cocktails .
our second visit .
i will not go back .

Figure 5: Documents from Yelp 2013. Label 4 means star 5, label 0 means star 1.



GT: 1 Prediction: 1

why does zebras have stripes ?
what is the purpose or those stripes ?
who do they serve the zebras in the
wild life ?
this provides camouflage - predator
vision is such that it is usually difficult
for them to see complex patterns

GT: 4 Prediction: 4

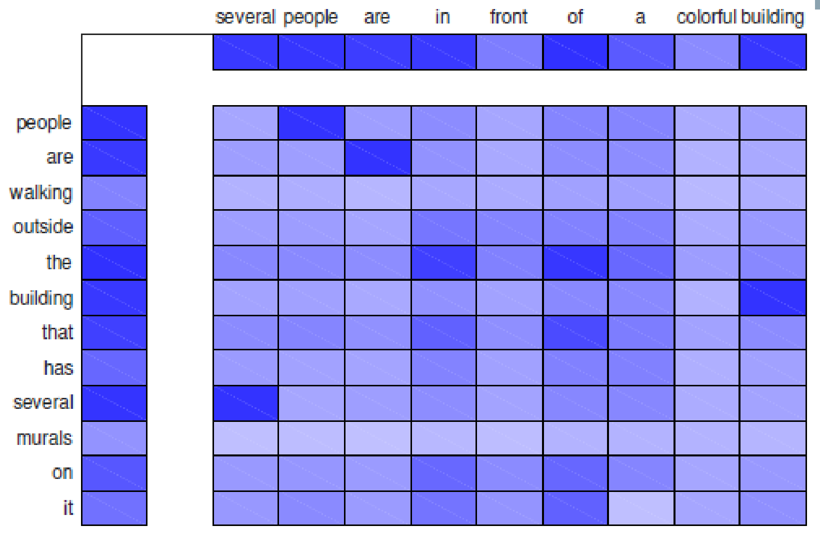
how do i get rid of all the old web
searches i have on my web browser ?
i want to clean up my web browser
go to tools > options .
then click " delete history " and "
clean up temporary internet files . "

Figure 6: Documents from Yahoo Answers. Label 1 denotes Science and Mathematics and label 4 denotes Computers and Internet.

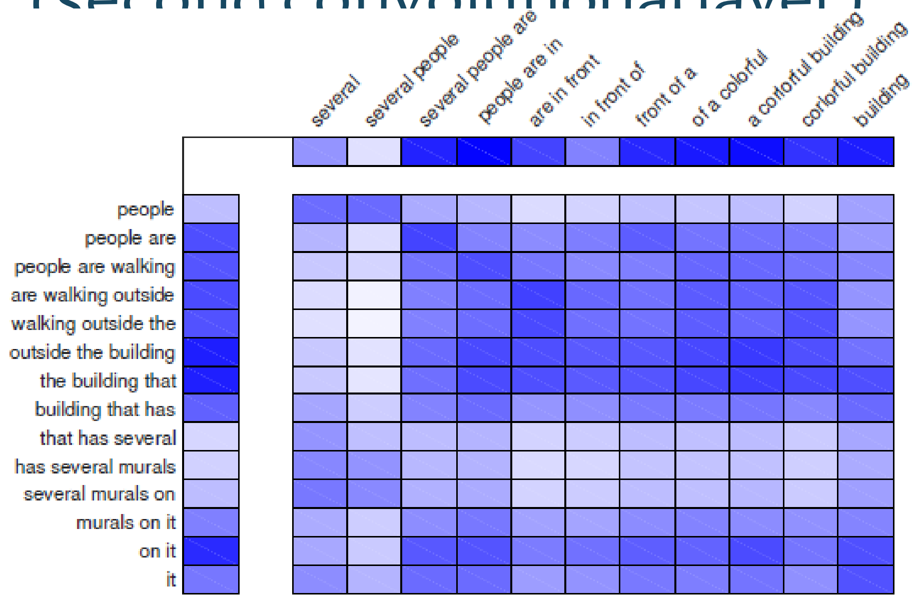
Yang, Zichao, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. "Hierarchical attention networks for document classification." NAACL, pp. 1480-1489. 2016.

Attention visualization for TE

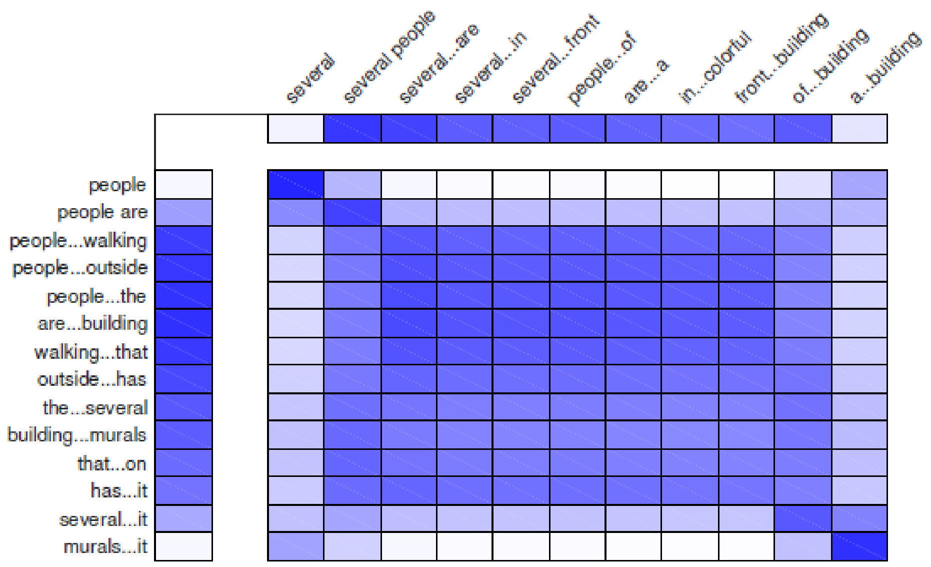
- Visualization of the attention matrices for one TE sentence pair for blocks b1 (unigrams), b2 (first convolutional layer) and b3 (second convolutional layer)



unigrams, b1.



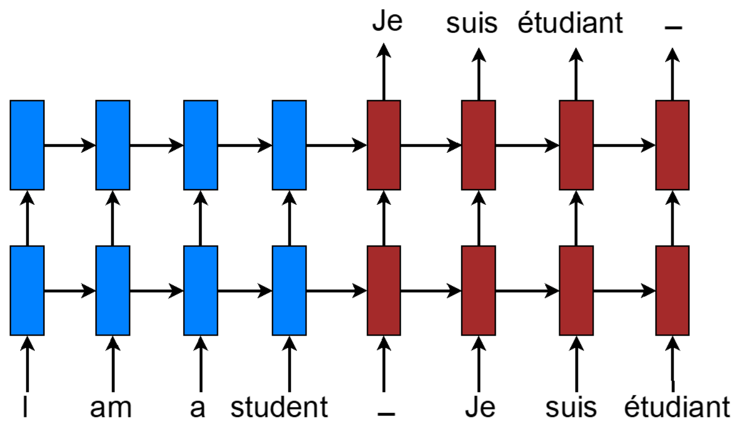
conv1, b2.

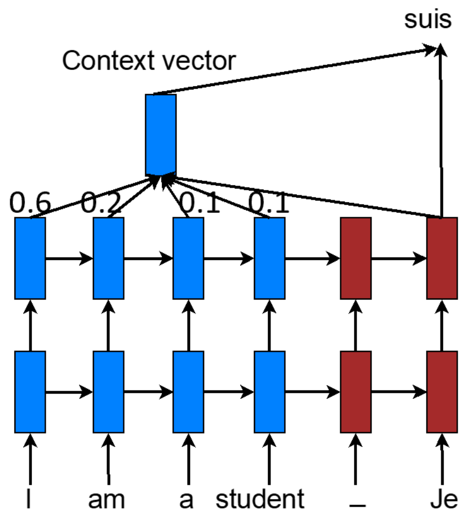


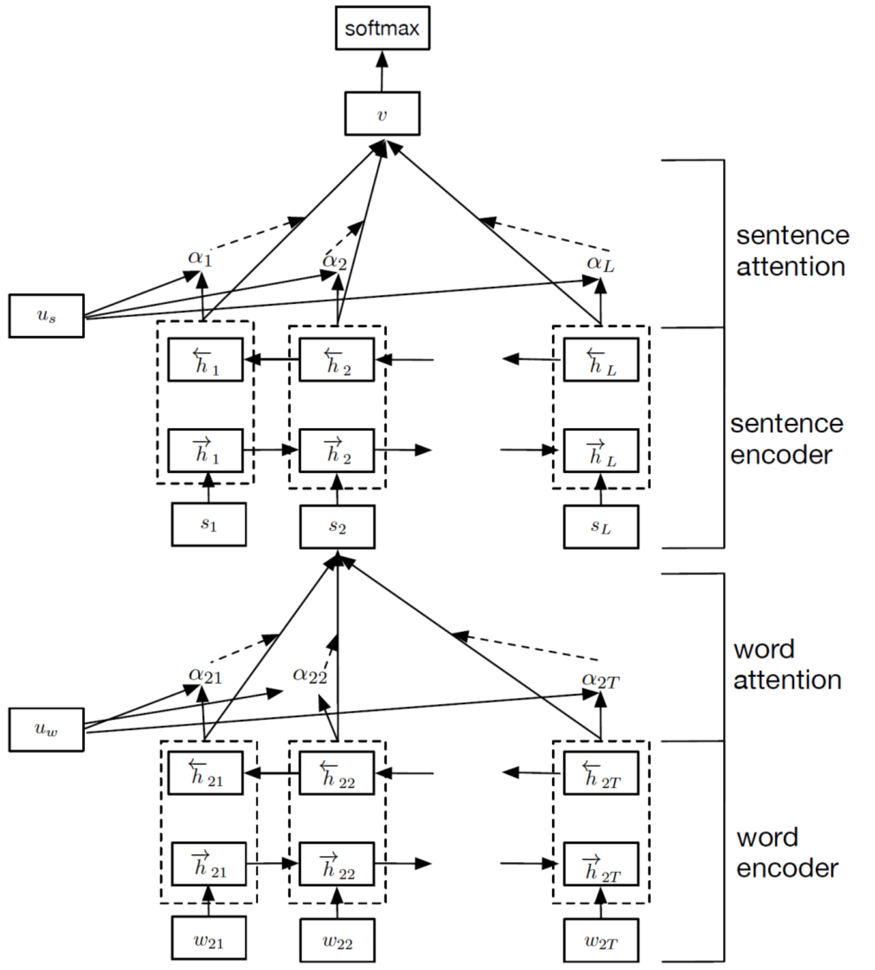
conv2, b3.

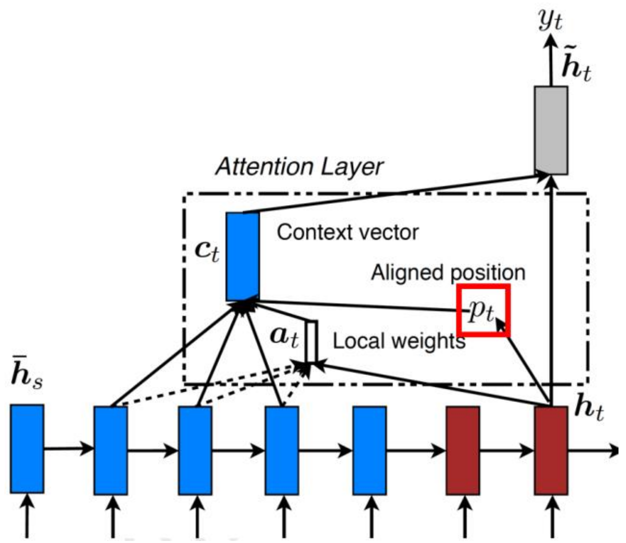
ABCNN: Attention-Based Convolutional Neural Network for Modeling Sentence Pairs. yin16_tacl

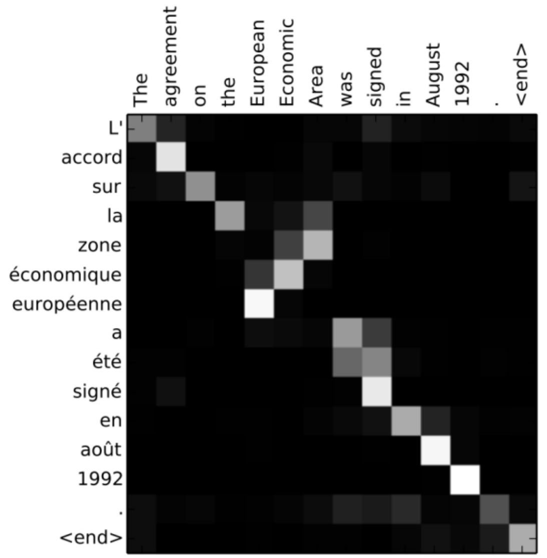
Take-aways


Je
suis
étudiant
_
I
am
a
student
_
Je
suis
étudiant


suis
Context vector
0.6
0.2
0.1
0.1
I
am
a
student
_
Je


softmax
v
α_1
α_2
α_L
u_s
$\overleftarrow{h}_1$
$\overleftarrow{h}_2$
$\overleftarrow{h}_L$
$\overrightarrow{h}_1$
$\overrightarrow{h}_2$
$\overrightarrow{h}_L$
s_1
s_2
s_L
sentence attention
sentence encoder
α_{21}
α_{22}
α_{2T}
u_w
$\overleftarrow{h}_{21}$
$\overleftarrow{h}_{22}$
$\overleftarrow{h}_{2T}$
$\overrightarrow{h}_{21}$
$\overrightarrow{h}_{22}$
$\overrightarrow{h}_{2T}$
w_{21}
w_{22}
w_{2T}
word attention
word encoder


y_t
$\tilde{h}_t$
Attention Layer
c_t
Context vector
Aligned position
p_t
a_t
Local weights
$\bar{h}_s$
h_t


The
agreement
on
the
European
Economic
Area
was
signed
in
August
1992
.
<end>
L'
accord
sur
la
zone
économique
européenne
a
été
signé
en
août
1992
.
<end>

INSOFE's Vision

The BEST GLOBAL DESTINATION for individuals and organizations to learn and adopt disruptive technologies for solving business and society's challenges.

GLASGOW
CLEVELAND
OTTAWA
RENNES
ROPAR
MUMBAI
HYDERABAD
BENGALURU

20+
PhDs

30+
Doctoral students

75+
Patents

300+
Publications

10000+
Data Science Alumni

INSOFE - HYDERABAD
2nd Floor, Jyothi Imperial, Vamsiram Builders
Janardana Hills, Gachibowli
Hyderabad – 500032
+91 93199 77257

INSOFE - BENGALURU
Floors 1-3, L77, 15th Cross Road
Sector 6, HSR Layout
Bengaluru - 560102
+91 93199 77267

INSOFE - MUMBAI
4th Floor - A Wing, Spaces - Kanaki
Andheri-Kurla Road, Chakala
Andheri East, Mumbai - 400093
+91 93199 77269

Website: www.insofe.edu.in

Email: info@insofe.edu.in

Follow us on Social Media:

/insofeglobal/ /school/insofe/ /INSOFEedu /insofe_global/ /InsofeVideos

This presentation may contain references to findings of various reports available in the public domain. INSOFE makes no representation as to their accuracy or that the organization subscribes to those findings.