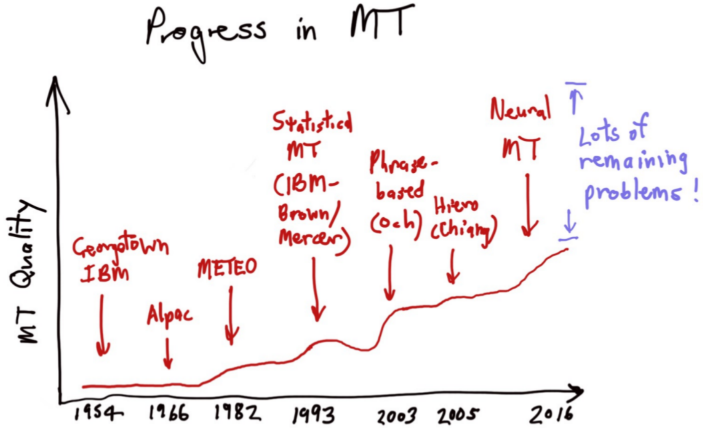
Neural Machine Translation

February 8, 2018

Machine Translation

- $40 billion industry
- Google: translates 100 billion words a day

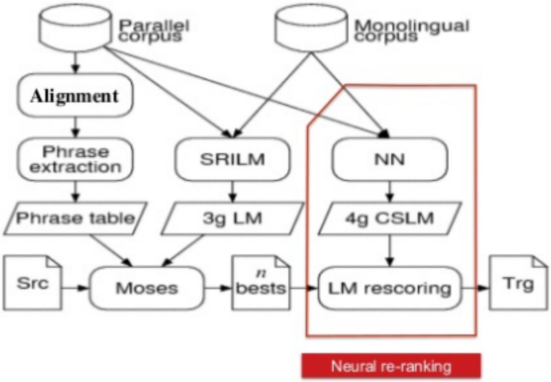
Machine Translation History



(from Manning/Luong/Cho)

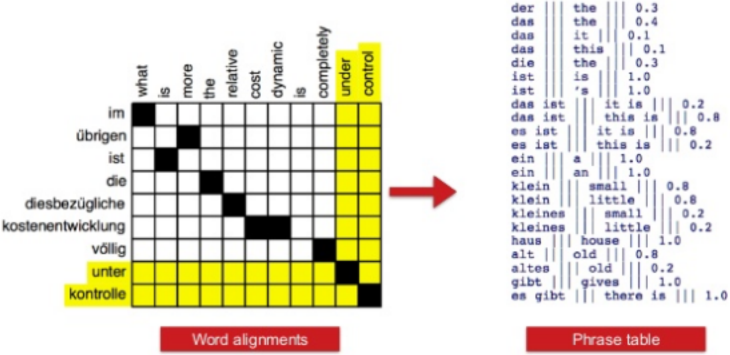
Phrase-based MT

Complex pipelines, all trained separately



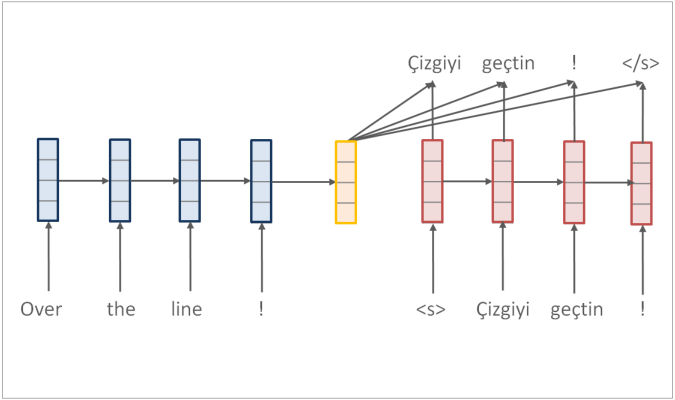
Phrase-based MT

Alignment Model

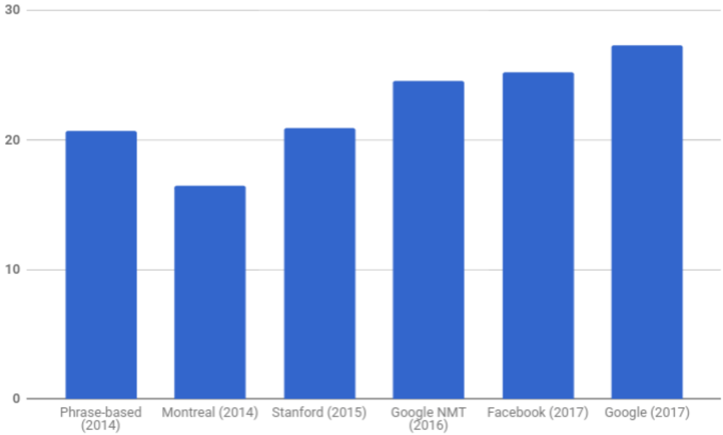


Neural Machine Translation

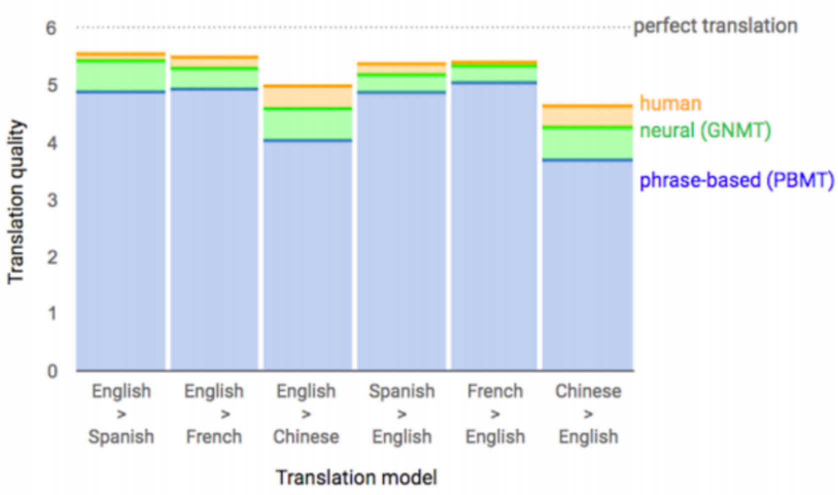
- No pipelines. Single model trained end-to-end with backprop.
- Essentially a conditional language model



Machine Translation Progress (English-German)



Machines vs. Humans



Pure Encoder-Decoder Network

Input (sentence, image, etc.)

Fixed-Size Encoder (MLP, RNN, CNN)

Encoder(input) $\in \mathbb{R}^D$

Decoder

Decoder(Encoder(input))

Pure Encoder-Decoder Network

Input (sentence, image, etc.)

Fixed-Size Encoder (MLP, RNN, CNN)

Encoder(input) $\in \mathbb{R}^D$

Decoder

Decoder(Encoder(input))

Pure Encoder-Decoder Network

Input (sentence, image, etc.)

Fixed-Size Encoder (MLP, RNN, CNN)

$\text{Encoder}(\text{input}) \in \mathbb{R}^D$

Decoder

$\text{Decoder}(\text{Encoder}(\text{input}))$

Example: Neural Machine Translation (Sutskever et al., 2014)

Source sentence: $\mathbf{x} = [x_1, \ldots, x_T]$

Target sentence: $\mathbf{y} = [y_1, \ldots, y_L]$

- $h_t = \mathrm{RNN}(x_t, h_{t-1})$ (Encoder RNN)
- $h_T =$ Last hidden state of RNN encoder (summary of source)
- $q_i = \mathrm{RNN}(y_l, q_{i-1})$ (Decoder RNN)
- $p(y_i | y_{<i}, \mathbf{x}) = \mathrm{softmax}(\mathrm{MLP}([q_i, h_T]))$
- Training: word-level maximum likelihood

$$\arg\max_{\theta} \sum_{i=1}^{L} \log p(y_i | y_{<i}, \mathbf{x})$$

Example: Neural Machine Translation (Sutskever et al., 2014)

Source sentence: $\mathbf{x} = [x_1, \ldots, x_T]$

Target sentence: $\mathbf{y} = [y_1, \ldots, y_L]$

- $h_t = \mathrm{RNN}(x_t, h_{t-1})$ (Encoder RNN)
- $h_T =$ Last hidden state of RNN encoder (summary of source)
- $q_i = \mathrm{RNN}(y_l, q_{i-1})$ (Decoder RNN)
- $p(y_i | y_{<i}, \mathbf{x}) = \mathrm{softmax}(\mathrm{MLP}([q_i, h_T]))$
- Training: word-level maximum likelihood

$$\arg\max_{\theta} \sum_{i=1}^{L} \log p(y_i | y_{<i}, \mathbf{x})$$

Example: Neural Machine Translation (Sutskever et al., 2014)

Source sentence: $\mathbf{x} = [x_1, \dots, x_T]$

Target sentence: $\mathbf{y} = [y_1, \dots, y_L]$

- $h_t = \mathrm{RNN}(x_t, h_{t-1})$ (Encoder RNN)
- $h_T =$ Last hidden state of RNN encoder (summary of source)
- $q_i = \mathrm{RNN}(y_l, q_{i-1})$ (Decoder RNN)
- $p(y_i | y_{<i}, \mathbf{x}) = \mathrm{softmax}(\mathrm{MLP}([q_i, h_T]))$
- Training: word-level maximum likelihood

$$\arg\max_{\theta} \sum_{i=1}^{L} \log p(y_i | y_{<i}, \mathbf{x})$$

Example: Neural Machine Translation (Sutskever et al., 2014)

Source sentence: $\mathbf{x} = [x_1, \ldots, x_T]$

Target sentence: $\mathbf{y} = [y_1, \ldots, y_L]$

- $h_t = \mathrm{RNN}(x_t, h_{t-1})$ (Encoder RNN)
- $h_T =$ Last hidden state of RNN encoder (summary of source)
- $q_i = \mathrm{RNN}(y_l, q_{i-1})$ (Decoder RNN)
- $p(y_i | y_{<i}, \mathbf{x}) = \mathrm{softmax}(\mathrm{MLP}([q_i, h_T]))$
- Training: word-level maximum likelihood

$$\arg\max_{\theta} \sum_{i=1}^{L} \log p(y_i | y_{<i}, \mathbf{x})$$

Example: Neural Machine Translation (Sutskever et al., 2014)

Source sentence: $\mathbf{x} = [x_1, \ldots, x_T]$

Target sentence: $\mathbf{y} = [y_1, \ldots, y_L]$

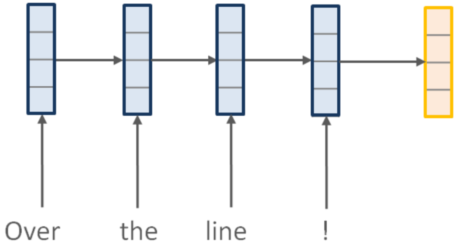
- $h_t = \mathrm{RNN}(x_t, h_{t-1})$ (Encoder RNN)
- $h_T =$ Last hidden state of RNN encoder (summary of source)
- $q_i = \mathrm{RNN}(y_l, q_{i-1})$ (Decoder RNN)
- $p(y_i | y_{<i}, \mathbf{x}) = \mathrm{softmax}(\mathrm{MLP}([q_i, h_T]))$
- Training: word-level maximum likelihood

$$\arg\max_{\theta} \sum_{i=1}^{L} \log p(y_i | y_{<i}, \mathbf{x})$$

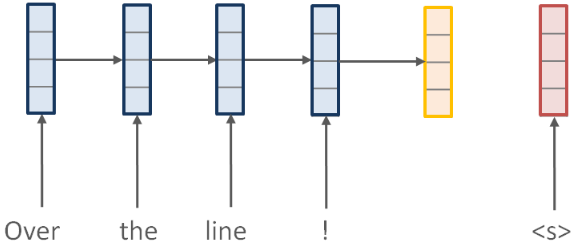
Example: Neural Machine Translation (Sutskever et al., 2014)

Over the line !

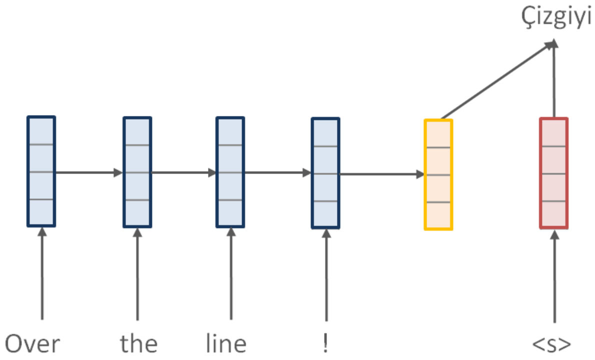
Example: Neural Machine Translation (Sutskever et al., 2014)



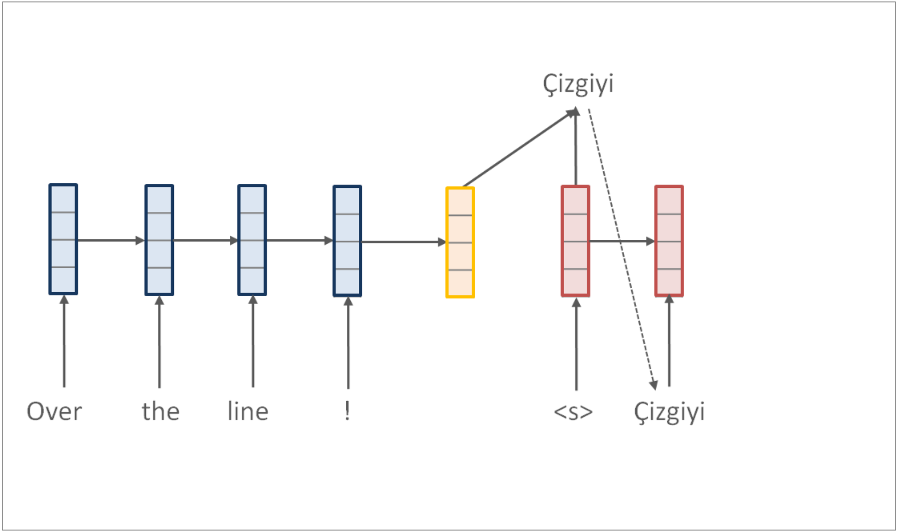
Example: Neural Machine Translation (Sutskever et al., 2014)



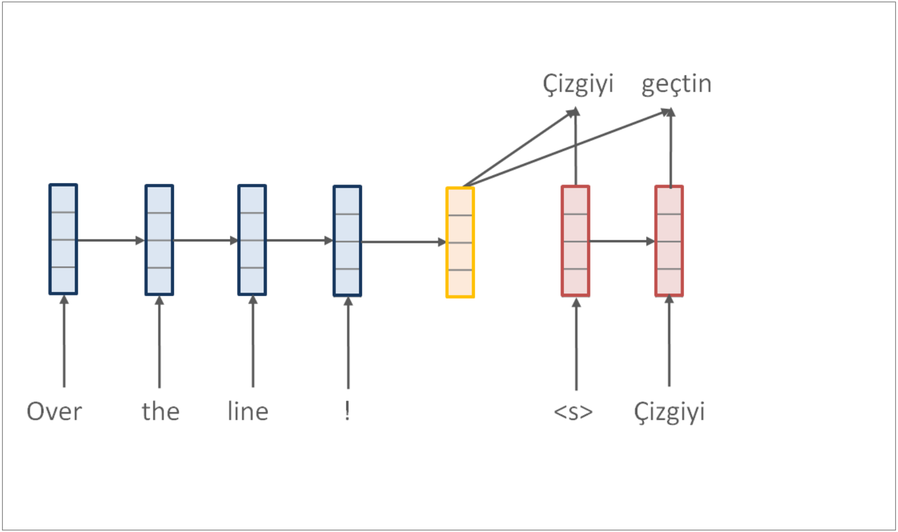
Example: Neural Machine Translation (Sutskever et al., 2014)



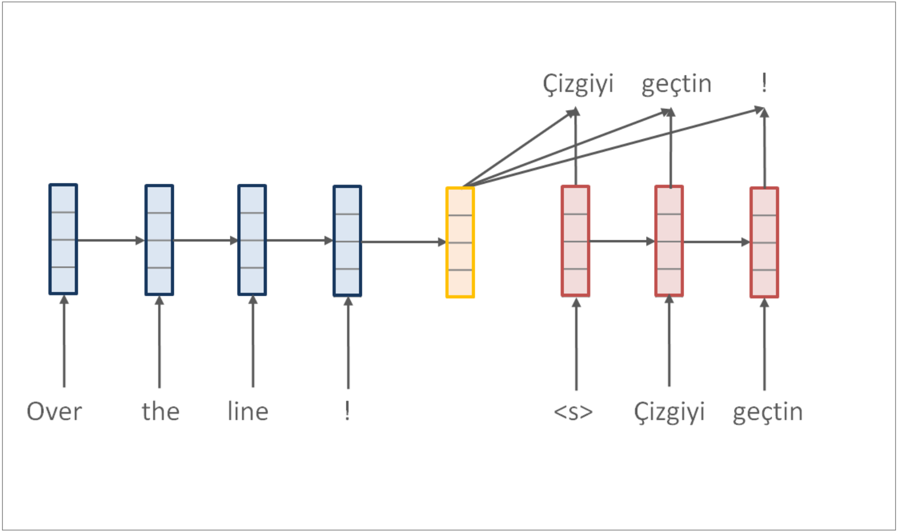
Example: Neural Machine Translation (Sutskever et al., 2014)



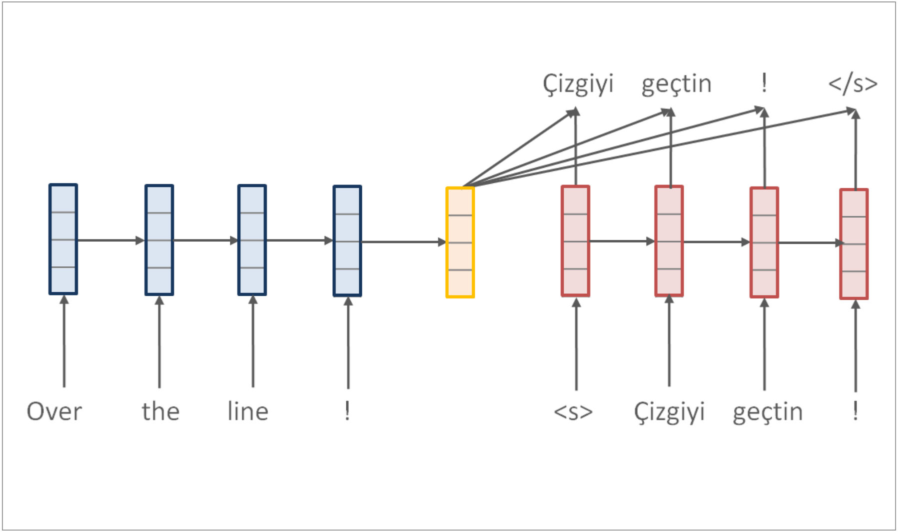
Example: Neural Machine Translation (Sutskever et al., 2014)



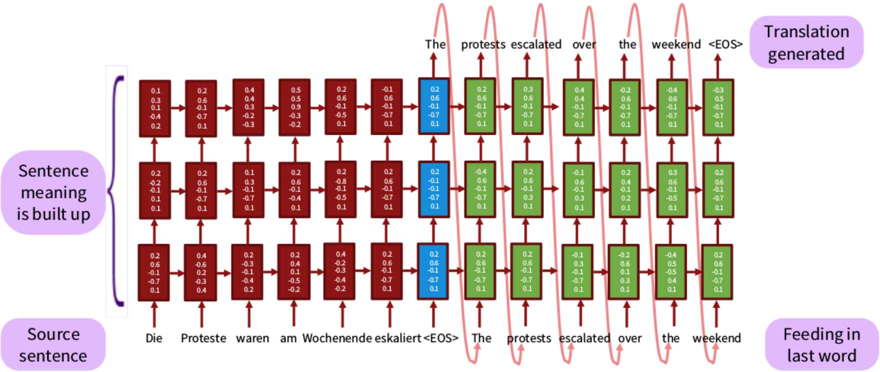
Example: Neural Machine Translation (Sutskever et al., 2014)



Example: Neural Machine Translation (Sutskever et al., 2014)



Example: Neural Machine Translation (Sutskever et al., 2014)



Communication Bottleneck

All input information communicated through fixed-size hidden vector.

$$\text{Encoder}(\text{input})$$

- Training: All gradients have to flow through single bottleneck.
- Test: All input encoded in single vector.

Neural Attention

Input (sentence, image, etc.)

⇓

Memory-Bank Encoder (MLP, RNN, CNN)

Encoder(input) $= x_1, x_2, \dots, x_T$

⇓

Attention Distribution "where" | Annotation Function "what"

⇓

Context Vector ("soft selection")

⇓

Decoder

Neural Attention

Input (sentence, image, etc.)

Memory-Bank Encoder (MLP, RNN, CNN)

$$\text{Encoder(input)} = x_1, x_2, \ldots, x_T$$

Attention Distribution
"where"

Annotation Function
"what"

Context Vector ("soft selection")

Decoder

Neural Attention

Input (sentence, image, etc.)

Memory-Bank Encoder (MLP, RNN, CNN)

Encoder(input) $= x_1, x_2, \ldots, x_T$

Attention Distribution "where"

Annotation Function "what"

Context Vector ("soft selection")

Decoder

Neural Attention

Input (sentence, image, etc.)

Memory-Bank Encoder (MLP, RNN, CNN)

Encoder(input) $= x_1, x_2, \ldots, x_T$

Attention Distribution — "where"

Annotation Function — "what"

Context Vector ("soft selection")

Decoder

Neural Attention

Input (sentence, image, etc.)

⬇

Memory-Bank Encoder (MLP, RNN, CNN)

Encoder(input) $= x_1, x_2, \ldots, x_T$

⬇

Attention Distribution   Annotation Function

"where"   "what"

⬇

Context Vector ("soft selection")

⬇

Decoder

Attention-based Neural Machine Translation (Bahdanau et al., 2015)

Source sentence: $\mathbf{x} = [x_1, \ldots, x_T]$

Target sentence: $\mathbf{y} = [y_1, \ldots, y_L]$

- $[h_1, \ldots, h_T] = \mathrm{RNN}(\mathbf{x})$ (Memory/Annotation Function)
- $q_i = \mathrm{RNN}(y_l, q_{i-1})$ (Decoder RNN)
- $\alpha_{i,t} = \frac{\exp(q_i^\top h_t)}{\sum_{j=1}^{T} \exp(q_i^\top h_j)}$ (Attention distribution)
- $c_i = \sum_{t=1}^{T} \alpha_{i,t} h_t$ (Context vector)
- $p(y_i | y_{<i}, \mathbf{x}) = \mathrm{softmax}(\mathrm{MLP}([q_i, c_i]))$

Attention-based Neural Machine Translation (Bahdanau et al., 2015)

Source sentence: $\mathbf{x} = [x_1, \ldots, x_T]$

Target sentence: $\mathbf{y} = [y_1, \ldots, y_L]$

- $[h_1, \ldots, h_T] = \mathrm{RNN}(\mathbf{x})$ (Memory/Annotation Function)
- $q_i = \mathrm{RNN}(y_l, q_{i-1})$ (Decoder RNN)
- $\alpha_{i,t} = \frac{\exp(q_i^\top h_t)}{\sum_{j=1}^{T} \exp(q_i^\top h_j)}$ (Attention distribution)
- $c_i = \sum_{t=1}^{T} \alpha_{i,t} h_t$ (Context vector)
- $p(y_i | y_{<i}, \mathbf{x}) = \mathrm{softmax}(\mathrm{MLP}([q_i, c_i]))$

Attention-based Neural Machine Translation (Bahdanau et al., 2015)

Source sentence: $\mathbf{x} = [x_1, \ldots, x_T]$

Target sentence: $\mathbf{y} = [y_1, \ldots, y_L]$

- $[h_1, \ldots, h_T] = \mathrm{RNN}(\mathbf{x})$ (Memory/Annotation Function)
- $q_i = \mathrm{RNN}(y_l, q_{i-1})$ (Decoder RNN)
- $\alpha_{i,t} = \frac{\exp(q_i^\top h_t)}{\sum_{j=1}^{T} \exp(q_i^\top h_j)}$ (Attention distribution)
- $c_i = \sum_{t=1}^{T} \alpha_{i,t} h_t$ (Context vector)
- $p(y_i | y_{<i}, \mathbf{x}) = \mathrm{softmax}(\mathrm{MLP}([q_i, c_i]))$

Attention-based Neural Machine Translation (Bahdanau et al., 2015)

Source sentence: $\mathbf{x} = [x_1, \ldots, x_T]$

Target sentence: $\mathbf{y} = [y_1, \ldots, y_L]$

- $[h_1, \ldots, h_T] = \mathrm{RNN}(\mathbf{x})$ (Memory/Annotation Function)
- $q_i = \mathrm{RNN}(y_l, q_{i-1})$ (Decoder RNN)
- $\alpha_{i,t} = \frac{\exp(q_i^\top h_t)}{\sum_{j=1}^{T} \exp(q_i^\top h_j)}$ (Attention distribution)
- $c_i = \sum_{t=1}^{T} \alpha_{i,t} h_t$ (Context vector)
- $p(y_i | y_{<i}, \mathbf{x}) = \mathrm{softmax}(\mathrm{MLP}([q_i, c_i]))$

Attention-based Neural Machine Translation (Bahdanau et al., 2015)

Source sentence: $\mathbf{x} = [x_1, \dots, x_T]$

Target sentence: $\mathbf{y} = [y_1, \dots, y_L]$

- $[h_1, \dots, h_T] = \mathrm{RNN}(\mathbf{x})$ (Memory/Annotation Function)
- $q_i = \mathrm{RNN}(y_l, q_{i-1})$ (Decoder RNN)
- $\alpha_{i,t} = \frac{\exp(q_i^\top h_t)}{\sum_{j=1}^{T} \exp(q_i^\top h_j)}$ (Attention distribution)
- $c_i = \sum_{t=1}^{T} \alpha_{i,t} h_t$ (Context vector)
- $p(y_i | y_{<i}, \mathbf{x}) = \mathrm{softmax}(\mathrm{MLP}([q_i, c_i]))$

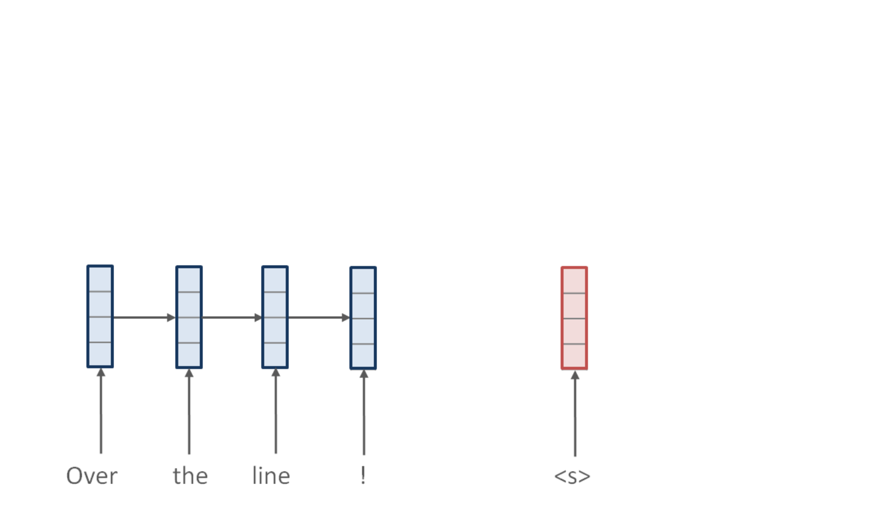
Attention-based Neural Machine Translation (Bahdanau et al., 2015)

Source sentence: $\mathbf{x} = [x_1, \ldots, x_T]$

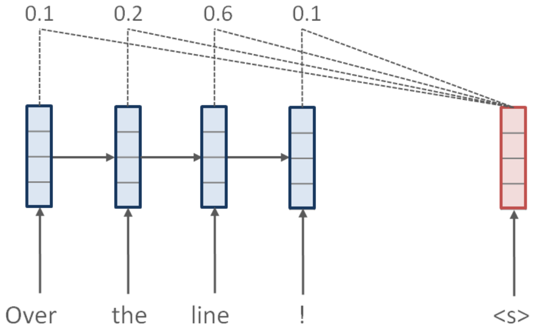
Target sentence: $\mathbf{y} = [y_1, \ldots, y_L]$

- $[h_1, \ldots, h_T] = \mathrm{RNN}(\mathbf{x})$ (Memory/Annotation Function)
- $q_i = \mathrm{RNN}(y_l, q_{i-1})$ (Decoder RNN)
- $\alpha_{i,t} = \frac{\exp(q_i^\top h_t)}{\sum_{j=1}^{T} \exp(q_i^\top h_j)}$ (Attention distribution)
- $c_i = \sum_{t=1}^{T} \alpha_{i,t} h_t$ (Context vector)
- $p(y_i | y_{<i}, \mathbf{x}) = \mathrm{softmax}(\mathrm{MLP}([q_i, c_i]))$

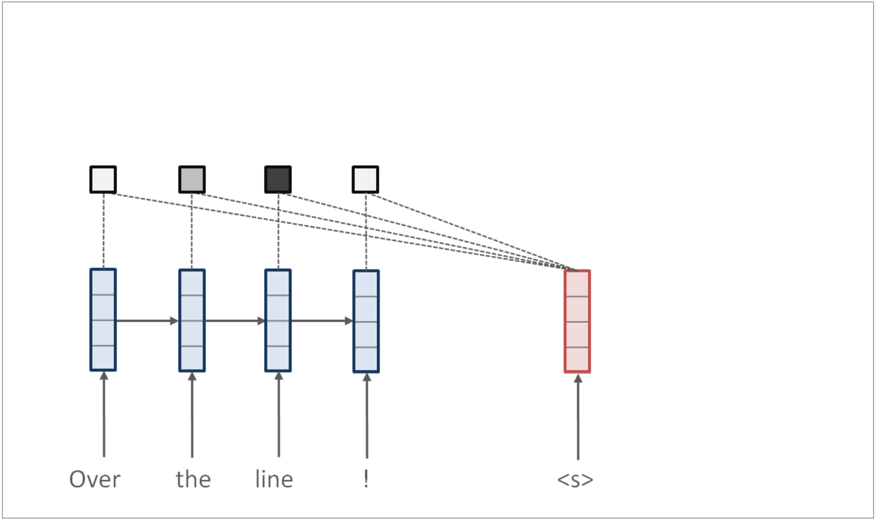
Attention-based Neural Machine Translation (Bahdanau et al., 2015)



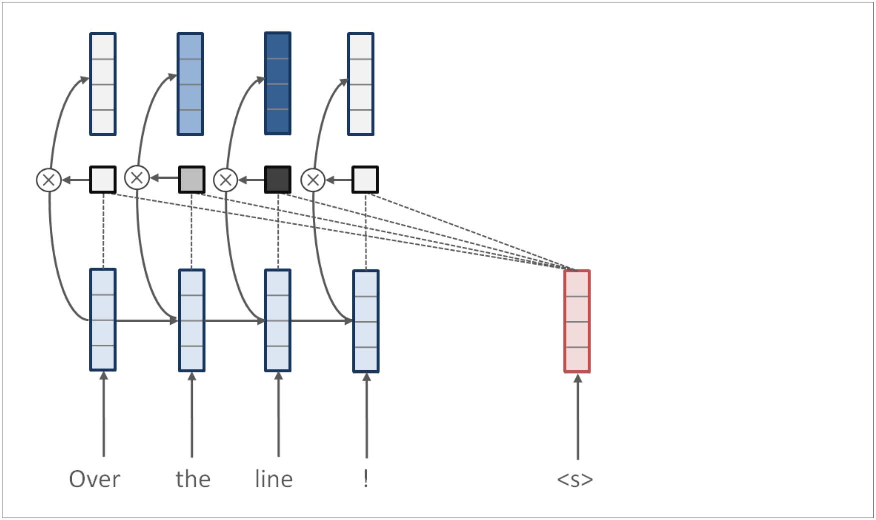
Attention-based Neural Machine Translation (Bahdanau et al., 2015)



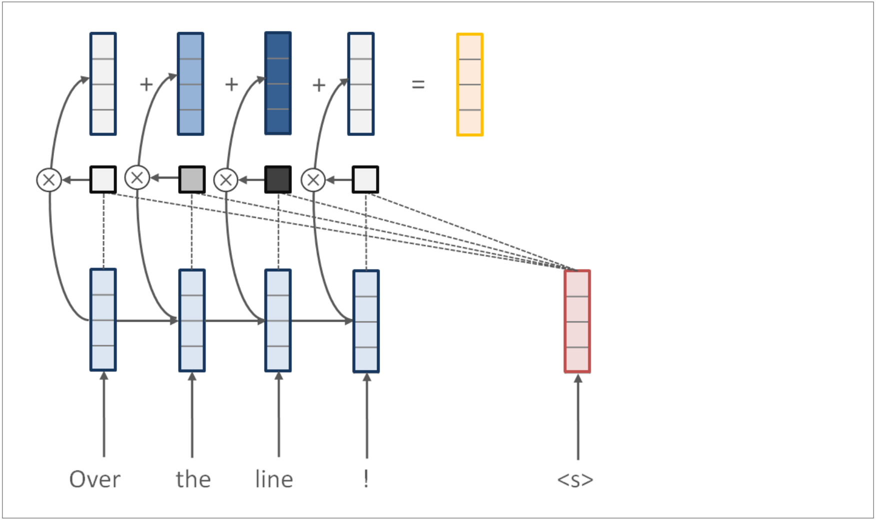
Attention-based Neural Machine Translation (Bahdanau et al., 2015)



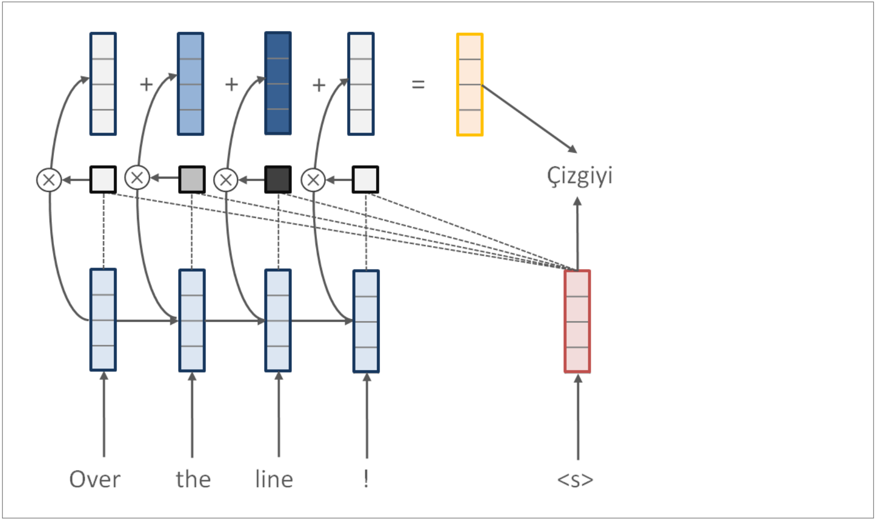
Attention-based Neural Machine Translation (Bahdanau et al., 2015)



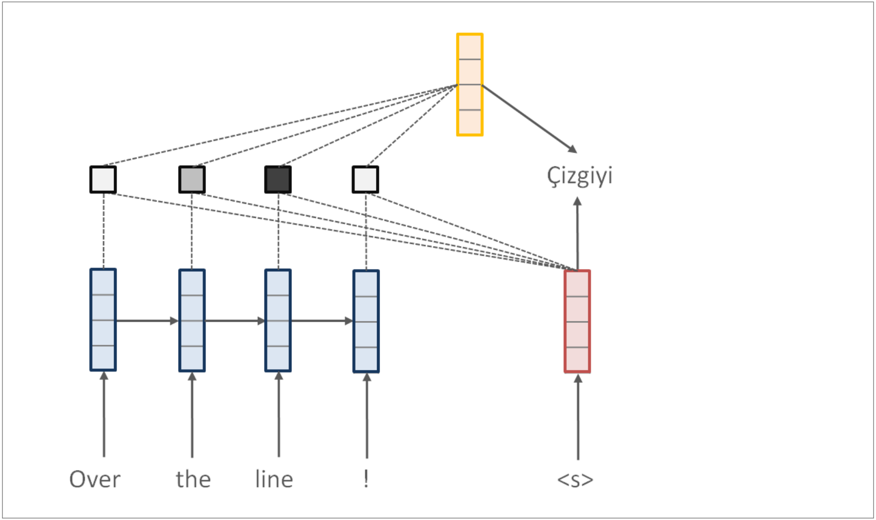
Attention-based Neural Machine Translation (Bahdanau et al., 2015)



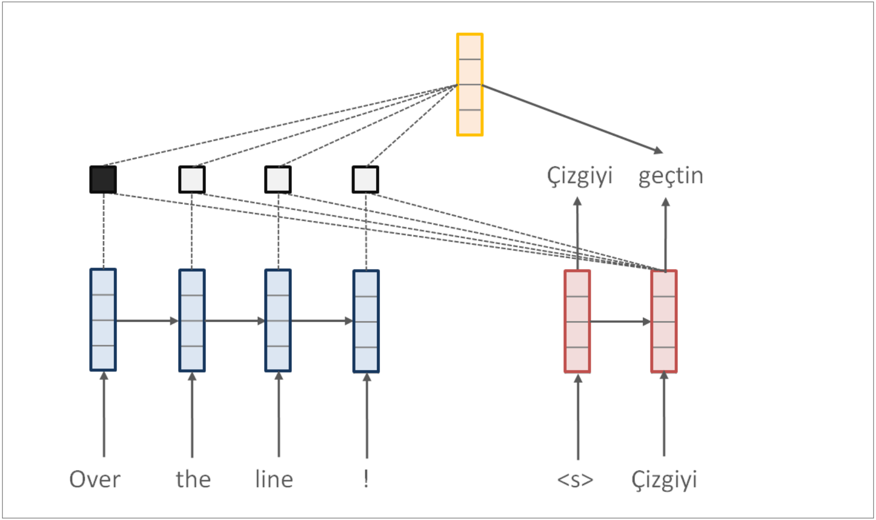
Attention-based Neural Machine Translation (Bahdanau et al., 2015)



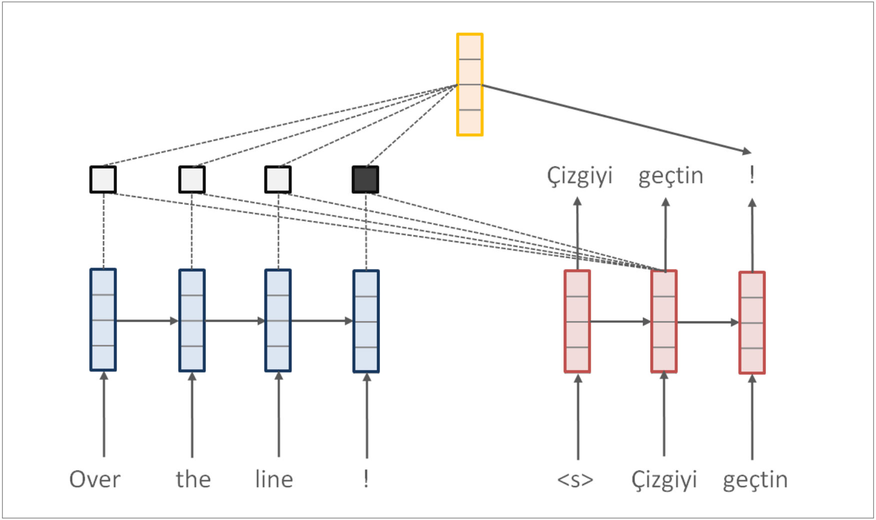
Attention-based Neural Machine Translation (Bahdanau et al., 2015)



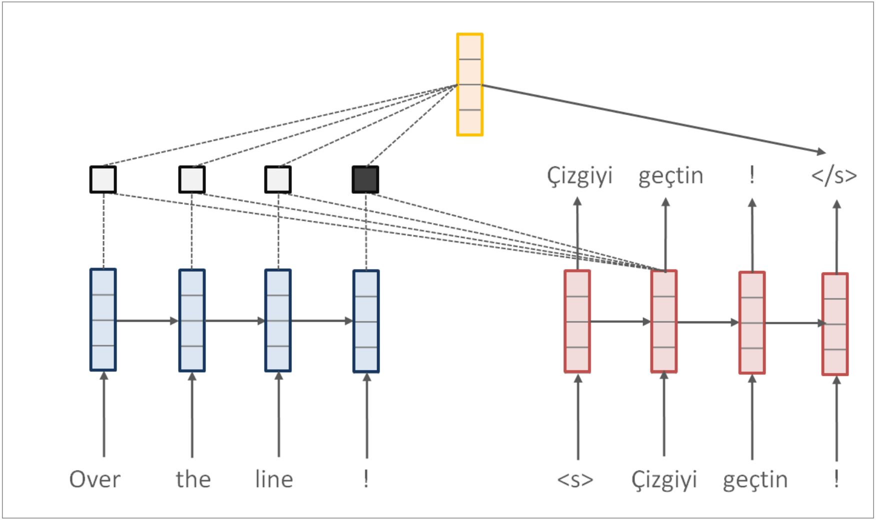
Attention-based Neural Machine Translation (Bahdanau et al., 2015)



Attention-based Neural Machine Translation (Bahdanau et al., 2015)



Attention-based Neural Machine Translation (Bahdanau et al., 2015)



Attention Networks: Notation

$x_1, \dots, x_T$	Memory bank
q	Query
z	Memory selection ("where")
$p(z = i \mid x, q; \theta)$	Attention distribution
$f(x, z)$	Annotation function ("what")
$c = \mathbb{E}_z[f(x, z)]$	Context vector ("soft selection")

End-to-End Requirements:

1. Need to compute attention distribution $p(z = i \mid x, q; \theta)$
2. Need to backpropagate to learn parameters θ

Attention Networks: Notation

$x_1, \dots, x_T$	Memory bank
q	Query
z	Memory selection ("where")
$p(z = i \mid x, q; \theta)$	Attention distribution
$f(x, z)$	Annotation function ("what")
$c = \mathbb{E}_z[f(x, z)]$	Context vector ("soft selection")

End-to-End Requirements:

1. Need to compute attention distribution $p(z = i \mid x, q; \theta)$
2. Need to backpropagate to learn parameters θ

Attention Networks: Machine Translation

$x_1, \ldots, x_T$	Memory bank	Source RNN hidden states
q	Query	Decoder hidden state
z	Memory selection	Source position $\{1, \ldots, T\}$
$p(z = i \mid x, q; \theta)$	Attention distribution	$\mathrm{softmax}(x_i^\top q)$
$f(x, z)$	Annotation function	Memory at time z, i.e. x_z
$c = \mathbb{E}_z[f(x, z)]$	Context vector	$\sum_{i=1}^T p(z = i \mid x, q) x_i$

End-to-End Requirements:

1. Need to compute attention $p(z = i \mid x, q; \theta)$
   $\Longrightarrow$ softmax function
2. Need to backpropagate to learn parameters θ
   $\Longrightarrow$ Backprop through softmax function

Attention Networks: Machine Translation

$x_1, \ldots, x_T$	Memory bank	Source RNN hidden states
q	Query	Decoder hidden state
z	Memory selection	Source position $\{1, \ldots, T\}$
$p(z = i \mid x, q; \theta)$	Attention distribution	$\mathrm{softmax}(x_i^\top q)$
$f(x, z)$	Annotation function	Memory at time z, i.e. x_z
$c = \mathbb{E}_z[f(x, z)]$	Context vector	$\sum_{i=1}^{T} p(z = i \mid x, q) x_i$

End-to-End Requirements:

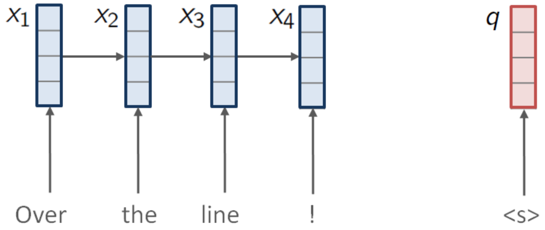
1. Need to compute attention $p(z = i \mid x, q; \theta)$

   $\implies$ softmax function

2. Need to backpropagate to learn parameters θ

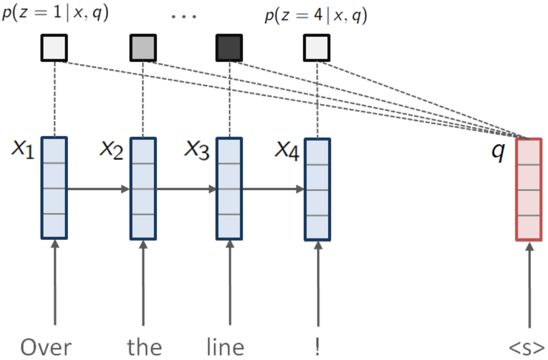
   $\implies$ Backprop through softmax function

Attention Networks: Machine Translation

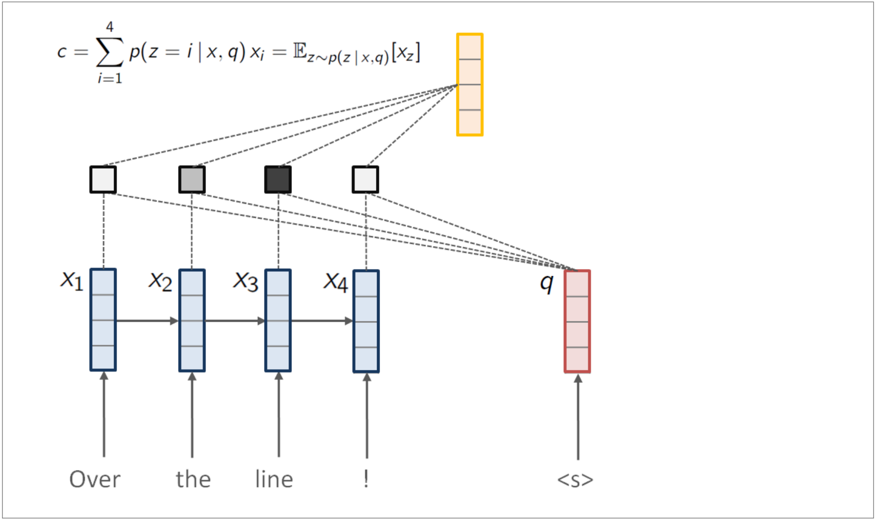


Attention Networks: Machine Translation

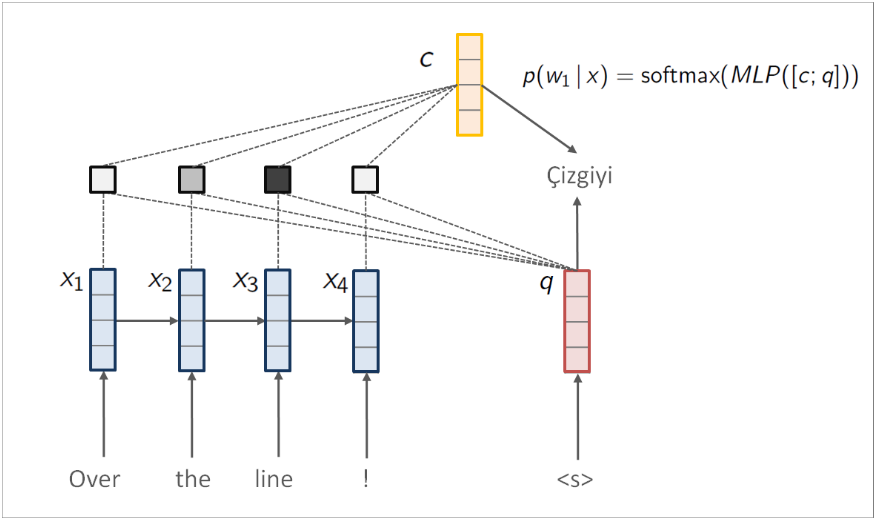
$$p(z = i \mid x, q) = \mathrm{softmax}(x_i^\top q) = \frac{\exp(x_i^\top q)}{\sum_{k=1}^{4} \exp(x_k^\top q)}$$



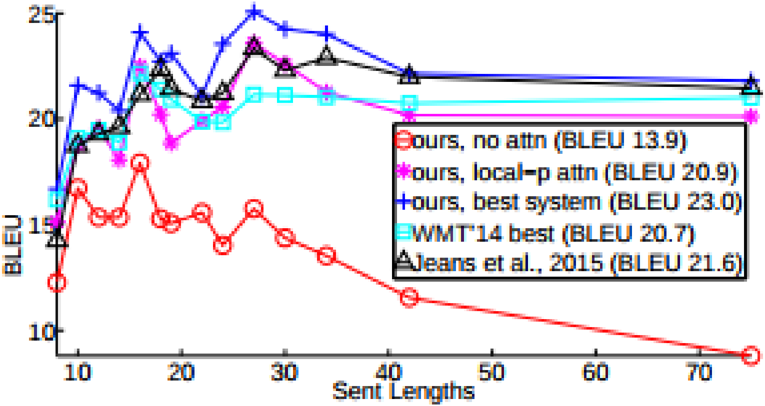
Attention Networks: Machine Translation



Attention Networks: Machine Translation



Performance vs Length

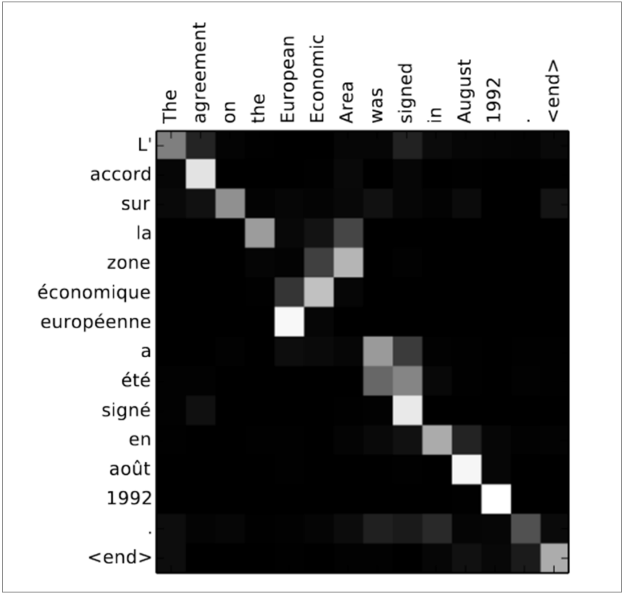


Practical Issues

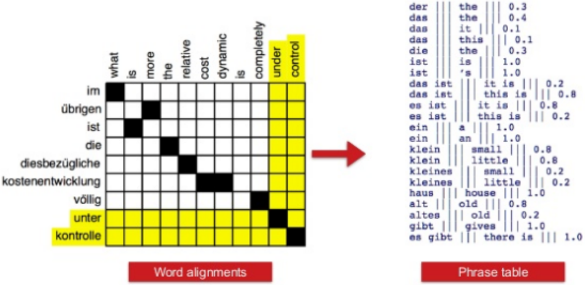
Some issues you will explore in the homework...

- How to set up data for batch training? (source/target sentences have varying lengths)
- What type of encoder/decoder architecture (GRU/LSTM/CNN)?
- How to find $\arg\max_{\mathbf{y} \in \mathcal{Y}} p(\mathbf{y} | \mathbf{x})$?
- How to deal with unknown tokens?

Attention Visualization



Alignment Model as a "Hidden Layer"



Question Answering (Sukhbaatar et al., 2015)

Greg is a frog

Brian is a rhino

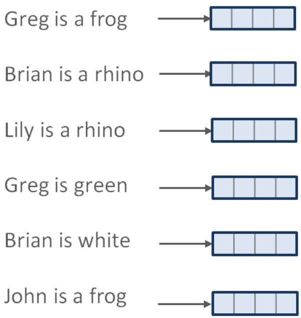
Lily is a rhino

Greg is green

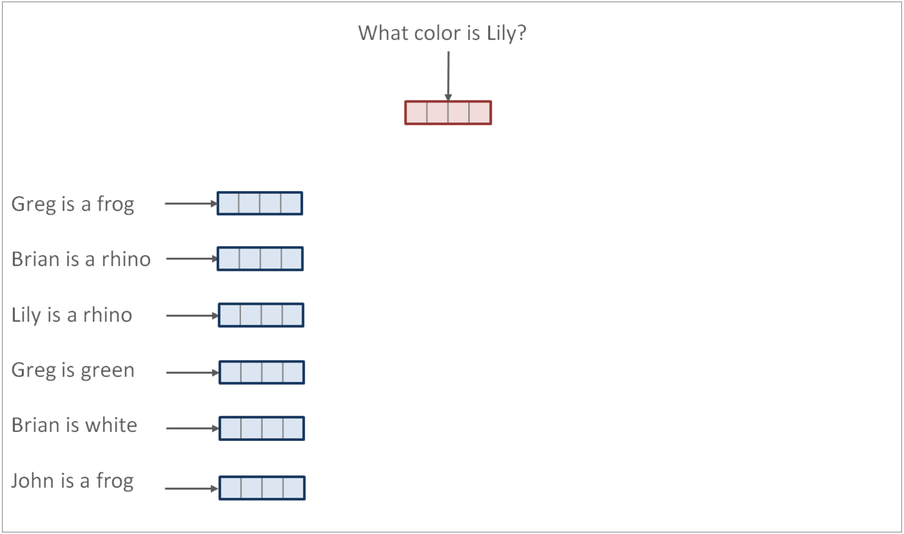
Brian is white

John is a frog

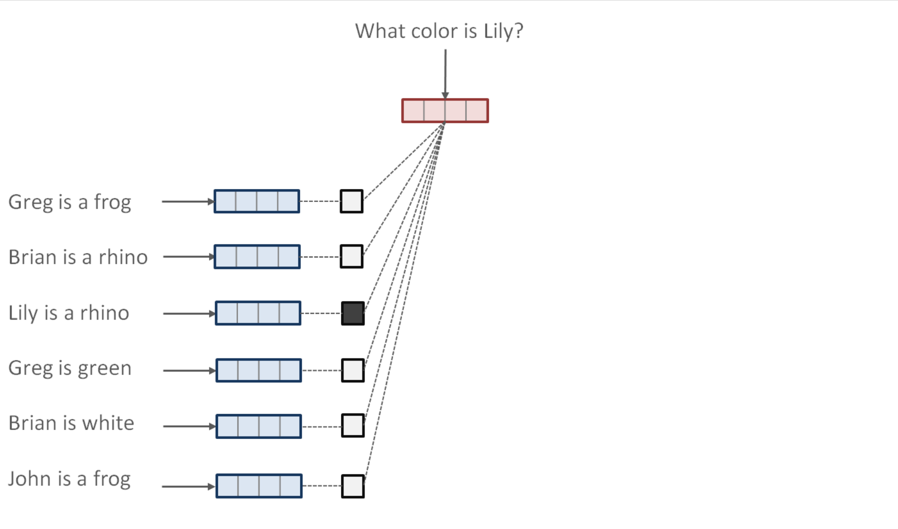
Question Answering (Sukhbaatar et al., 2015)



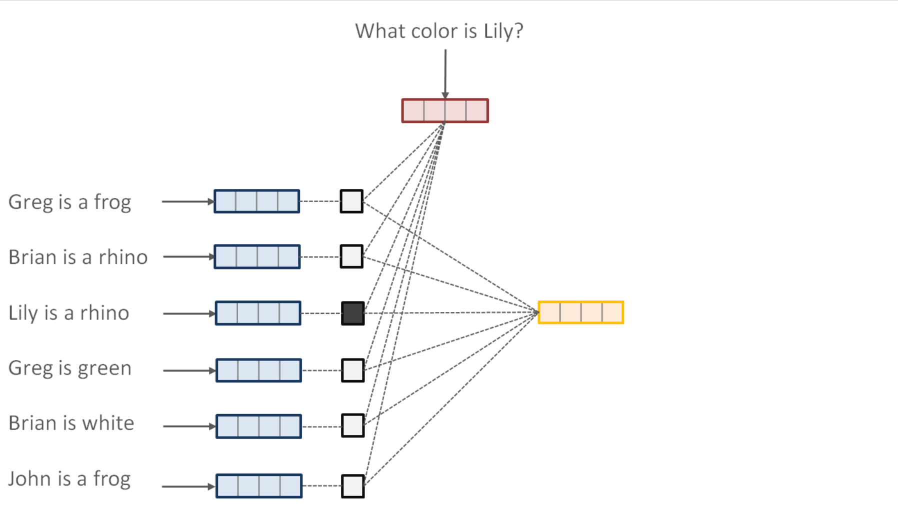
Question Answering (Sukhbaatar et al., 2015)



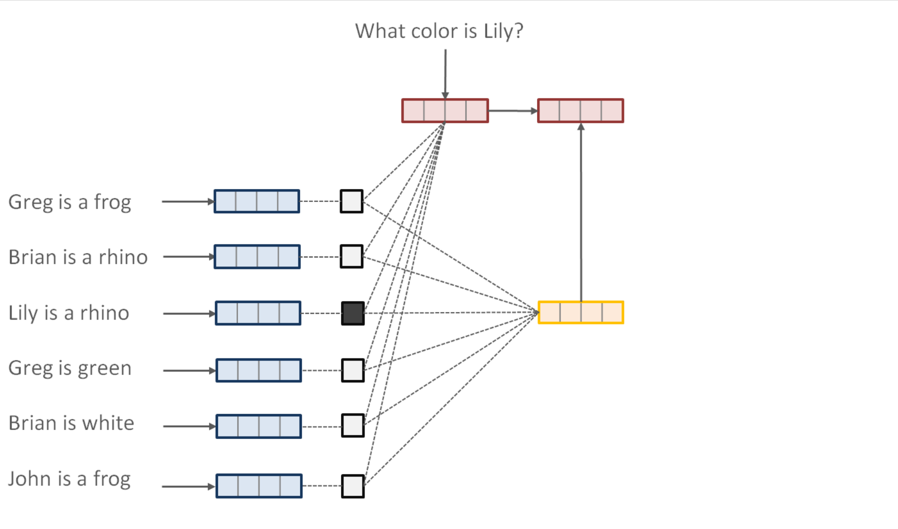
Question Answering (Sukhbaatar et al., 2015)



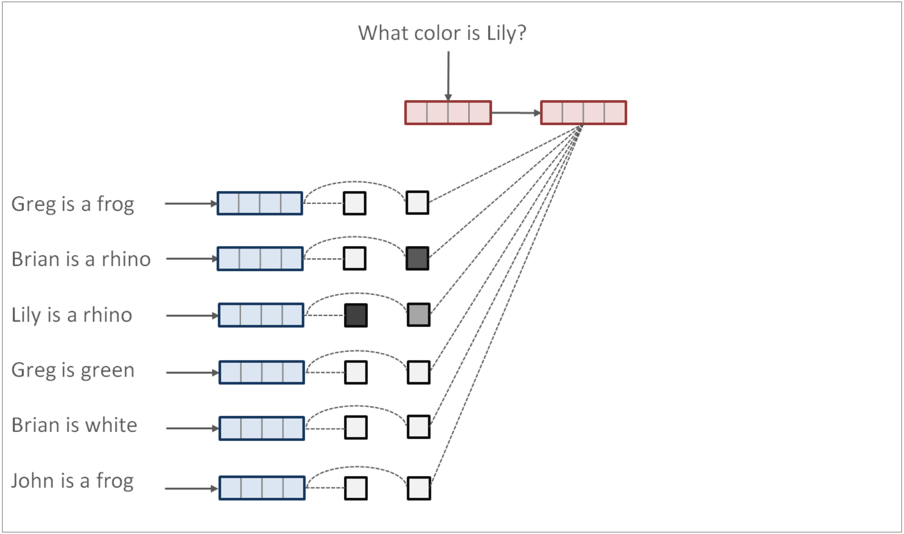
Question Answering (Sukhbaatar et al., 2015)



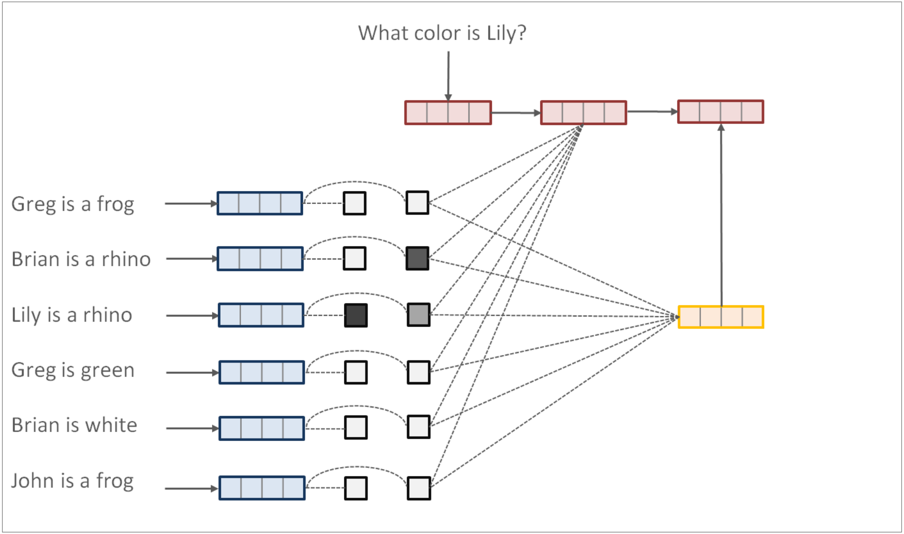
Question Answering (Sukhbaatar et al., 2015)



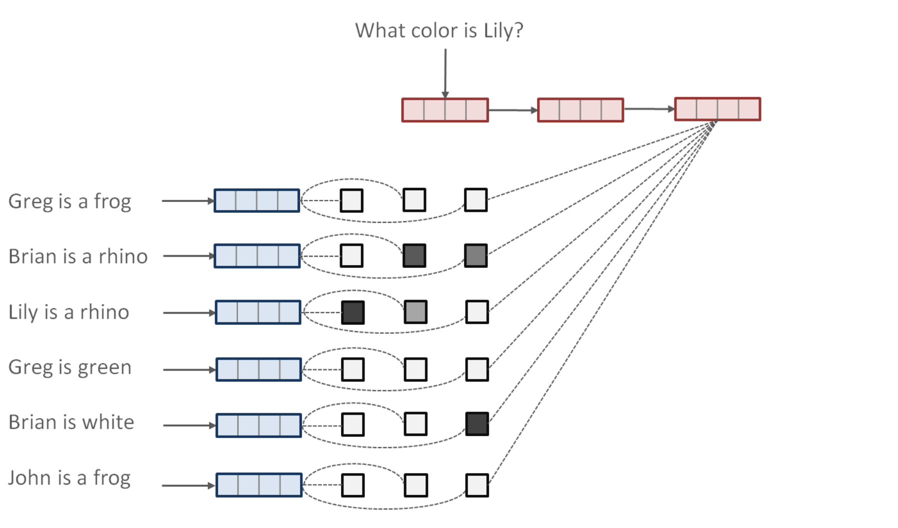
Question Answering (Sukhbaatar et al., 2015)



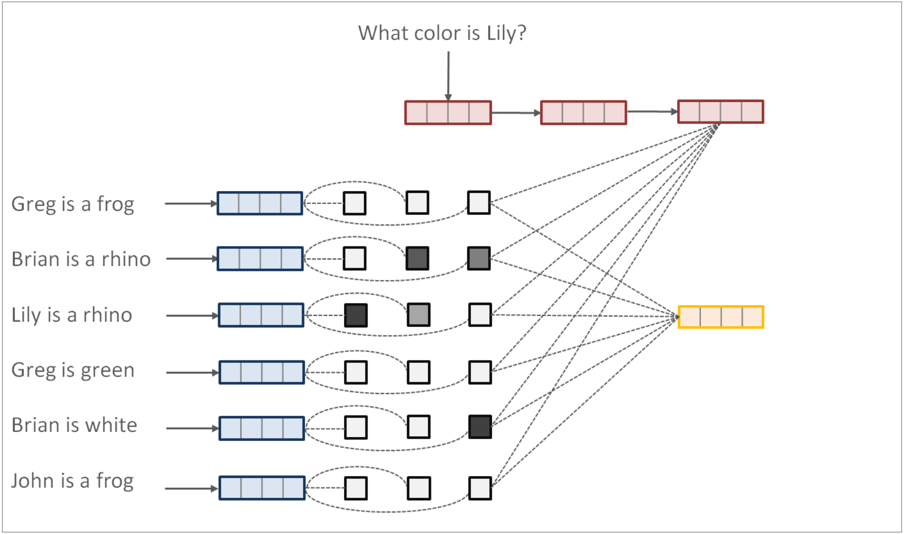
Question Answering (Sukhbaatar et al., 2015)



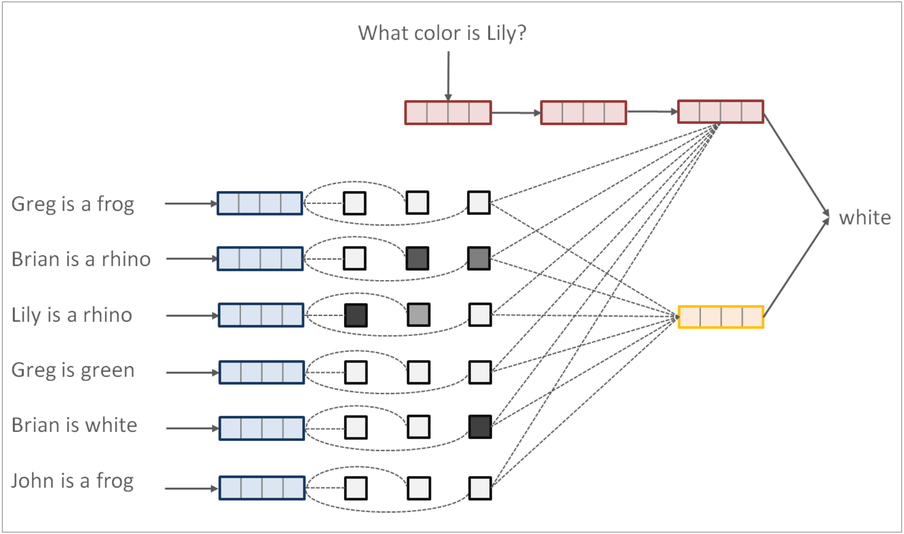
Question Answering (Sukhbaatar et al., 2015)



Question Answering (Sukhbaatar et al., 2015)



Question Answering (Sukhbaatar et al., 2015)



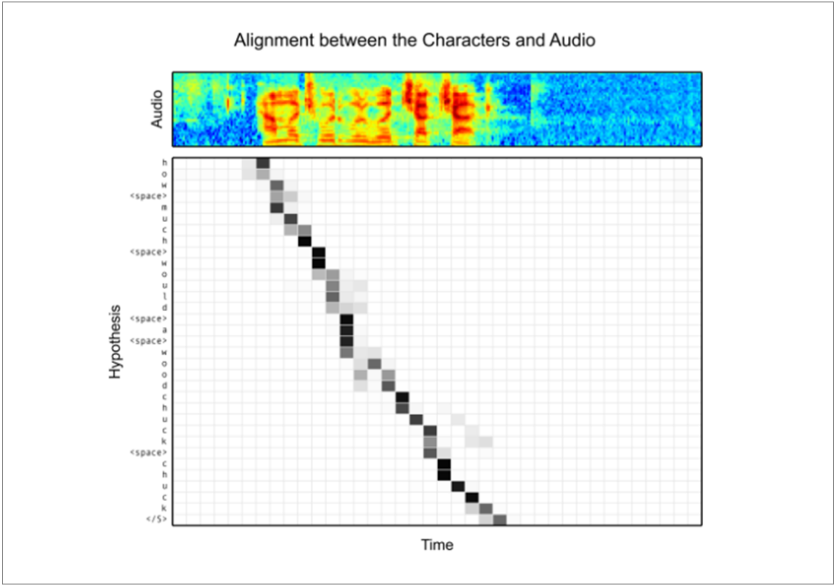
Other Applications of Attention Networks

- Machine Translation (Bahdanau et al., 2015; Luong et al., 2015)
- Question Answering (Hermann et al., 2015; Sukhbaatar et al., 2015)
- Natural Language Inference (Rocktäschel et al., 2016; Parikh et al., 2016)
- Algorithm Learning (Graves et al., 2014, 2016; Vinyals et al., 2015a)
- Parsing (Vinyals et al., 2015b)
- Speech Recognition (Chorowski et al., 2015; Chan et al., 2015)
- Summarization (Rush et al., 2015)
- Caption Generation (Xu et al., 2015)
- and more...

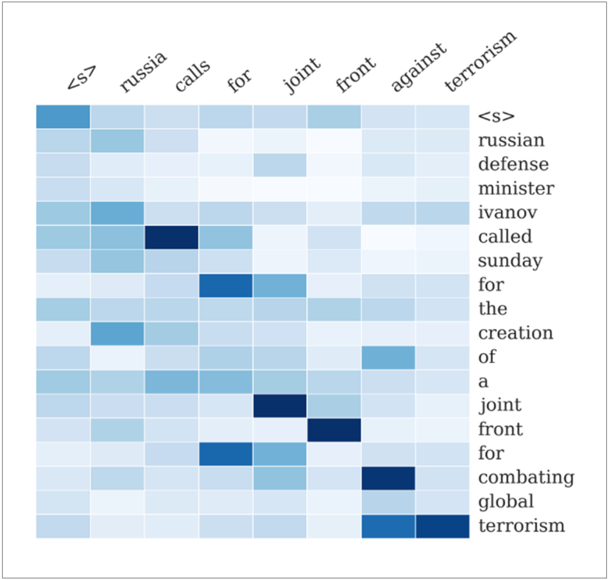
Other Applications: Image Captioning (Xu et al., 2015)

(b) A woman is throwing a frisbee in a park.

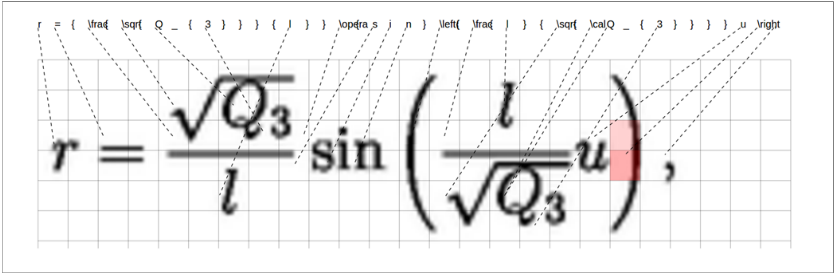
Other Applications: Speech Recognition (Chan et al., 2015)



Other Applications: Summarization (Rush et al., 2015)



Other Applications: Image-to-Latex (Deng et al., 2016)



Parameterizations of the Attention Distribution

s_{ij} = score of target word i attending to source word j

$\alpha_{ij} = \frac{\exp s_{ij}}{\sum_{t \in T} \exp s_{it}}$

- $s_{ij} = q_i^\top h_j$ (Rush et al. 2015)
- $s_{ij} = w^\top \tanh(W q_i + U h_j)$ (Bahdanau et al. 2015)
- $s_{ij} = (q_i W)^\top h_j$ (Luong et al. 2015)
- $s_{ij} = \mathrm{ForwardBackward}([s_{i1}, \ldots, s_{iT}])$ (Me et al. 2017)

Parameterizations of the Attention Distribution

s_{ij} = score of target word i attending to source word j

$\alpha_{ij} = \frac{\exp s_{ij}}{\sum_{t \in T} \exp s_{it}}$

- $s_{ij} = q_i^\top h_j$ (Rush et al. 2015)
- $s_{ij} = w^\top \tanh(W q_i + U h_j)$ (Bahdanau et al. 2015)
- $s_{ij} = (q_i W)^\top h_j$ (Luong et al. 2015)
- $s_{ij} = \mathrm{ForwardBackward}([s_{i1}, \ldots, s_{iT}])$ (Me et al. 2017)

Parameterizations of the Attention Distribution

s_{ij} = score of target word i attending to source word j

$\alpha_{ij} = \frac{\exp s_{ij}}{\sum_{t \in T} \exp s_{it}}$

- $s_{ij} = q_i^\top h_j$ (Rush et al. 2015)
- $s_{ij} = w^\top \tanh(Wq_i + Uh_j)$ (Bahdanau et al. 2015)
- $s_{ij} = (q_i W)^\top h_j$ (Luong et al. 2015)
- $s_{ij} = \mathrm{ForwardBackward}([s_{i1}, \dots, s_{iT}])$ (Me et al. 2017)

Parameterizations of the Attention Distribution

s_{ij} = score of target word i attending to source word j

$\alpha_{ij} = \frac{\exp s_{ij}}{\sum_{t \in T} \exp s_{it}}$

- $s_{ij} = q_i^\top h_j$ (Rush et al. 2015)
- $s_{ij} = w^\top \tanh(W q_i + U h_j)$ (Bahdanau et al. 2015)
- $s_{ij} = (q_i W)^\top h_j$ (Luong et al. 2015)
- $s_{ij} = \mathrm{ForwardBackward}([s_{i1}, \ldots, s_{iT}])$ (Me et al. 2017)

Parameterizations of the Attention Distribution

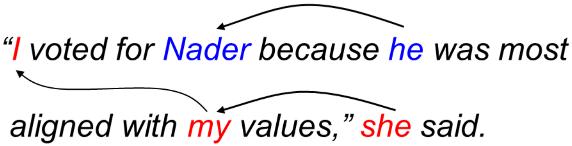
s_{ij} = score of target word i attending to source word j

$\alpha_{ij} = \frac{\exp s_{ij}}{\sum_{t \in T} \exp s_{it}}$

- $s_{ij} = q_i^\top h_j$ (Rush et al. 2015)
- $s_{ij} = w^\top \tanh(W q_i + U h_j)$ (Bahdanau et al. 2015)
- $s_{ij} = (q_i W)^\top h_j$ (Luong et al. 2015)
- $s_{ij} = \mathrm{ForwardBackward}([s_{i1}, \ldots, s_{iT}])$ (Me et al. 2017)

Self-Attention

- Ideally, encoder RNN state h_t should capture all information about x_t including the global sentence context (if using bidirectional RNN).
- In practice, this is hard to, and h_t only captures local information.
- In language, there are often long-range dependencies.



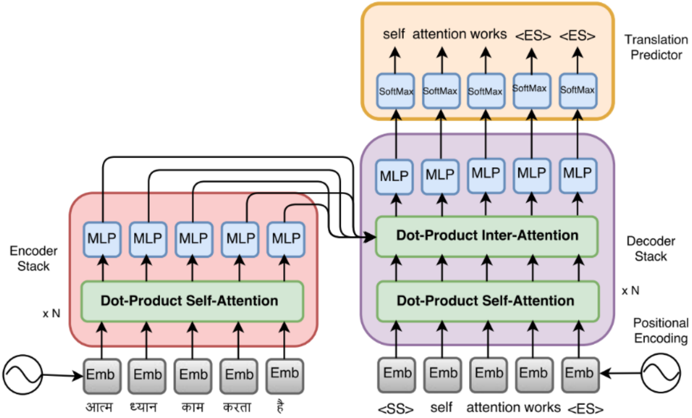
Self-Attention

- Idea: Perform **self**-attention over the sentence
- $\beta_{ij} = \mathrm{softmax}(h_i^\top h_j)$ where h_i, h_j are source hidden states
- For each word x_i, get its "soft-match":

$$\tilde{x}_i = \sum_{i=1}^{T} \beta_{ij} x_j$$

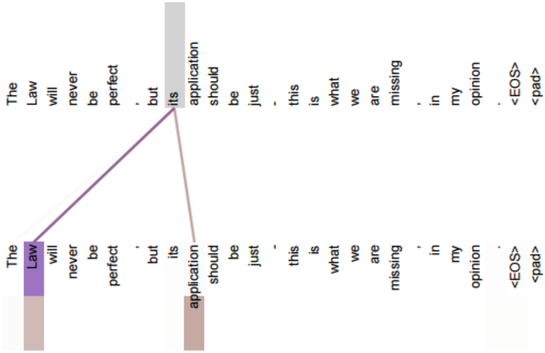
- Use $\tilde{x}_i$ along with x_i as input into encoder

Self-Attention



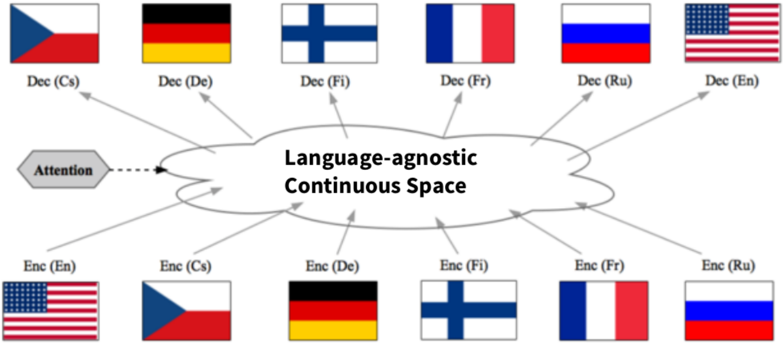
(Vaswani et al. 2017)

Self-Attention



(Vaswani et al. 2017)

Multilingual Translation

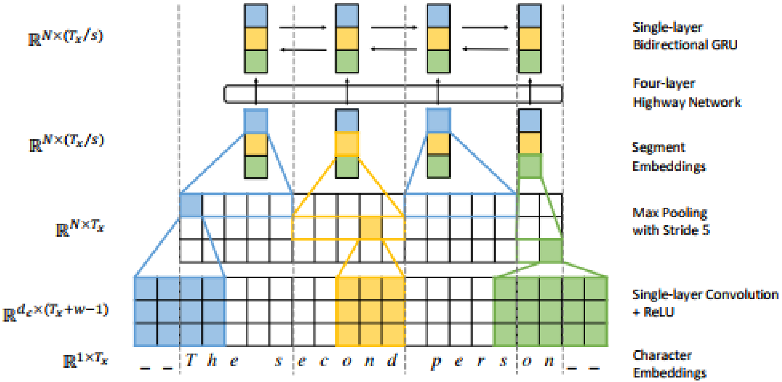


Low Resource Translation

Language	Train size	Test size	SBMT BLEU	NMT BLEU
Hausa	1.0m	11.3K	23.7	16.8
Turkish	1.4m	11.6K	20.4	11.4
Uzbek	1.8m	11.5K	17.9	10.7
Urdu	0.2m	11.4K	17.9	5.2

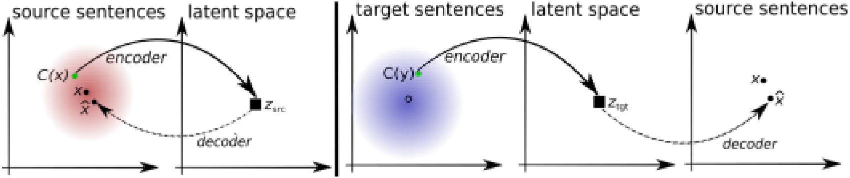
(Zoph et al. 2016)

Subword Machine Translation



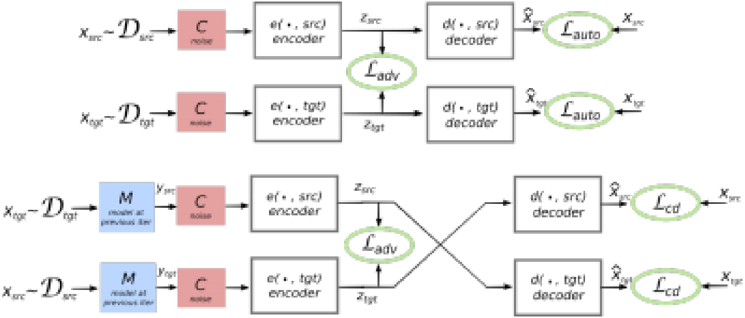
(Lee et al. 2016)

Unsupervised Machine Translation



(Lample et al. 2018)

Unsupervised Machine Translation



(Lample et al. 2018)

Unsupervised Machine Translation

		FR-EN	EN-FR	DE-EN	EN-DE
Unsupervised	1. Baseline (emb. nearest neighbor)	9.98	6.25	7.07	4.39
	2. Proposed (denoising)	7.28	5.33	3.64	2.40
	3. Proposed (+ backtranslation)	15.56	15.13	10.21	6.55
	4. Proposed (+ BPE)	15.56	14.36	10.16	6.89
Semi-supervised	5. Proposed (full) + 10k parallel	18.57	17.34	11.47	7.86
	6. Proposed (full) + 100k parallel	21.81	21.74	15.24	10.95
Supervised	7. Comparable NMT (10k parallel)	1.88	1.66	1.33	0.82
	8. Comparable NMT (100k parallel)	10.40	9.19	8.11	5.29
	9. Comparable NMT (full parallel)	20.48	19.89	15.04	11.05
	10. GNMT (Wu et al., 2016)	-	38.95	-	24.61

(Lample et al. 2018)

References I

Bahdanau, D., Cho, K., and Bengio, Y. (2015). Neural Machine Translation by Jointly Learning to Align and Translate. In *Proceedings of ICLR*.

Chan, W., Jaitly, N., Le, Q., and Vinyals, O. (2015). Listen, Attend and Spell. *arXiv:1508.01211*.

Chorowski, J., Bahdanau, D., Serdyuk, D., Cho, K., and Bengio, Y. (2015). Attention-Based Models for Speech Recognition. In *Proceedings of NIPS*.

Deng, Y., Kanervisto, A., and Rush, A. M. (2016). What You Get Is What You See: A Visual Markup Decompiler. *arXiv:1609.04938*.

Graves, A., Wayne, G., and Danihelka, I. (2014). Neural Turing Machines. *arXiv:1410.5401*.

References II

Graves, A., Wayne, G., Reynolds, M., Harley, T., Danihelka, I., Grabska-Barwinska, A., Colmenarejo, S. G., Grefenstette, E., Ramalho, T., Agapiou, J., Badia, A. P., Hermann, K. M., Zwols, Y., Ostrovski, G., Cain, A., King, H., Summerfield, C., Blunsom, P., Kavukcuoglu, K., and Hassabis, D. (2016). Hybrid Computing Using a Neural Network with Dynamic External Memory. *Nature*.

Hermann, K. M., Kocisky, T., Grefenstette, E., Espeholt, L., Kay, W., Suleyman, M., and Blunsom, P. (2015). Teaching Machines to Read and Comprehend. In *Proceedings of NIPS*.

Luong, M.-T., Pham, H., and Manning, C. D. (2015). Effective Approaches to Attention-based Neural Machine Translation. In *Proceedings of EMNLP*.

Parikh, A. P., Tackstrom, O., Das, D., and Uszkoreit, J. (2016). A Decomposable Attention Model for Natural Language Inference. In *Proceedings of EMNLP*.

References III

Rocktäschel, T., Grefenstette, E., Hermann, K. M., Kocisky, T., and Blunsom, P. (2016). Reasoning about Entailment with Neural Attention. In *Proceedings of ICLR*.

Rush, A. M., Chopra, S., and Weston, J. (2015). A Neural Attention Model for Abstractive Sentence Summarization. In *Proceedings of EMNLP*.

Sukhbaatar, S., Szlam, A., Weston, J., and Fergus, R. (2015). End-To-End Memory Networks. In *Proceedings of NIPS*.

Sutskever, I., Vinyals, O., and Le, Q. (2014). Sequence to Sequence Learning with Neural Networks. In *Proceedings of NIPS*.

Vinyals, O., Fortunato, M., and Jaitly, N. (2015a). Pointer Networks. In *Proceedings of NIPS*.

Vinyals, O., Kaiser, L., Koo, T., Petrov, S., Sutskever, I., and Hinton, G. (2015b). Grammar as a Foreign Language. In *Proceedings of NIPS*.

References IV

Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhutdinov, R., Zemel, R., and Bengio, Y. (2015). Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. In *Proceedings of ICML*.