Structured Attention Networks

Yoon Kim* Carl Denton* Luong Hoang Alexander M. Rush

HarvardNLP

Pure Encoder-Decoder Network

Input (sentence, image, etc.)

⇓

Fixed-Size Encoder (MLP, RNN, CNN)

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

⇓

Decoder

Decoder(Encoder(input))

Pure Encoder-Decoder Network

Input (sentence, image, etc.)

⇓

Fixed-Size Encoder (MLP, RNN, CNN)

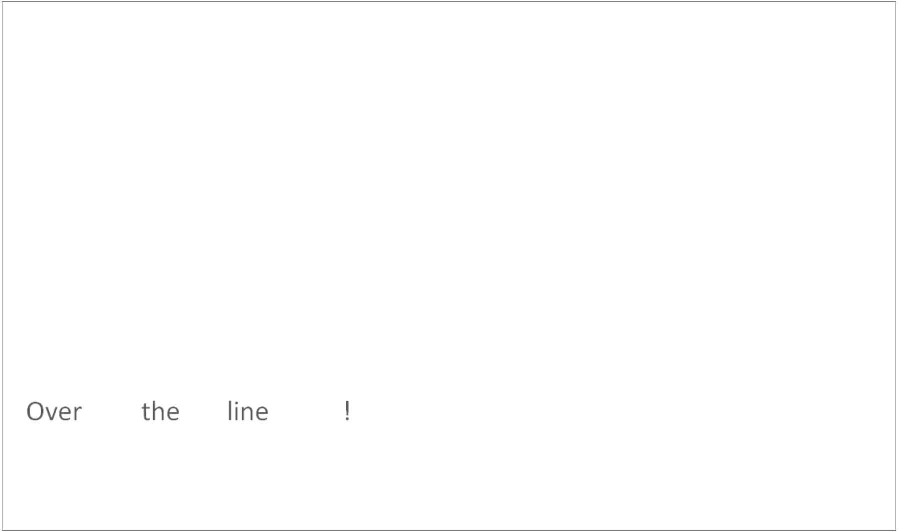
$\text{Encoder}(\text{input}) \in \mathbb{R}^D$

⇓

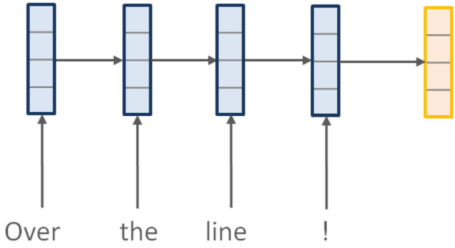
Decoder

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

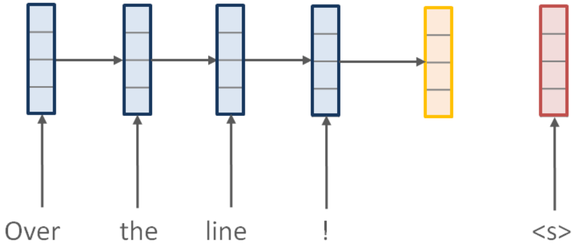
Example: Neural Machine Translation (Sutskever et al., 2014)



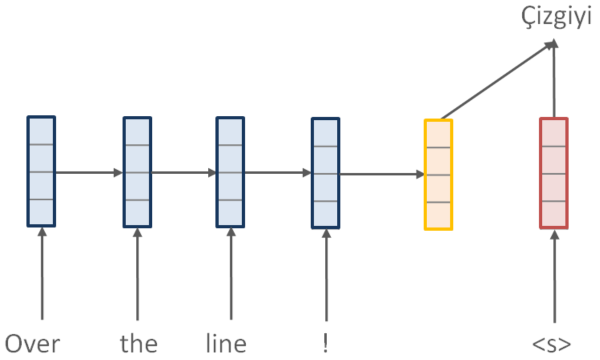
Example: Neural Machine Translation (Sutskever et al., 2014)



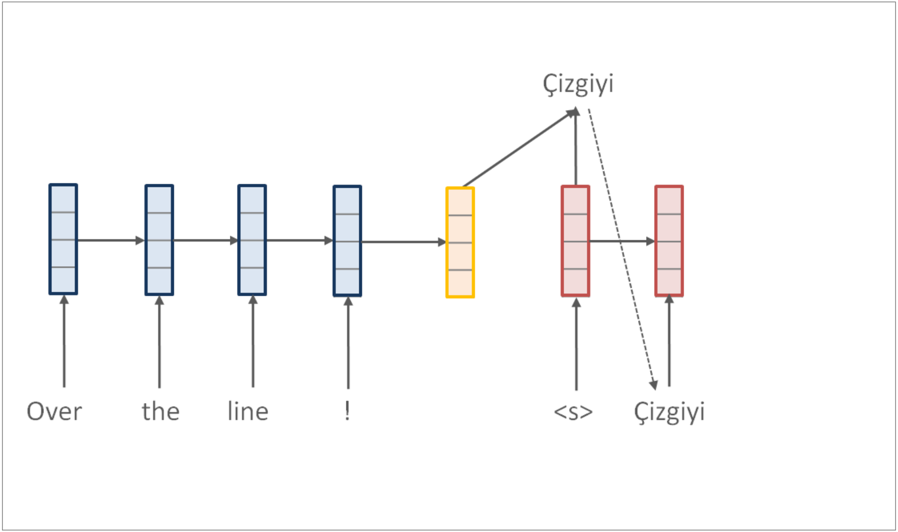
Example: Neural Machine Translation (Sutskever et al., 2014)



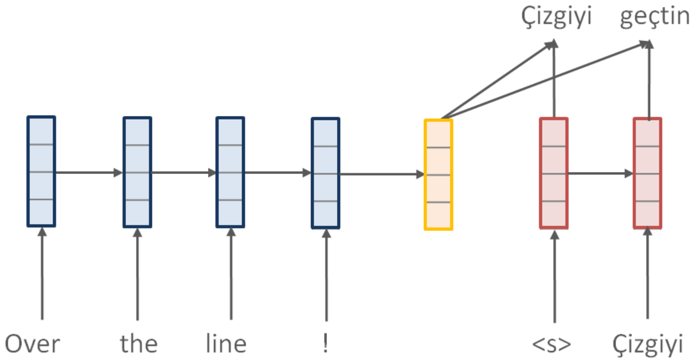
Example: Neural Machine Translation (Sutskever et al., 2014)



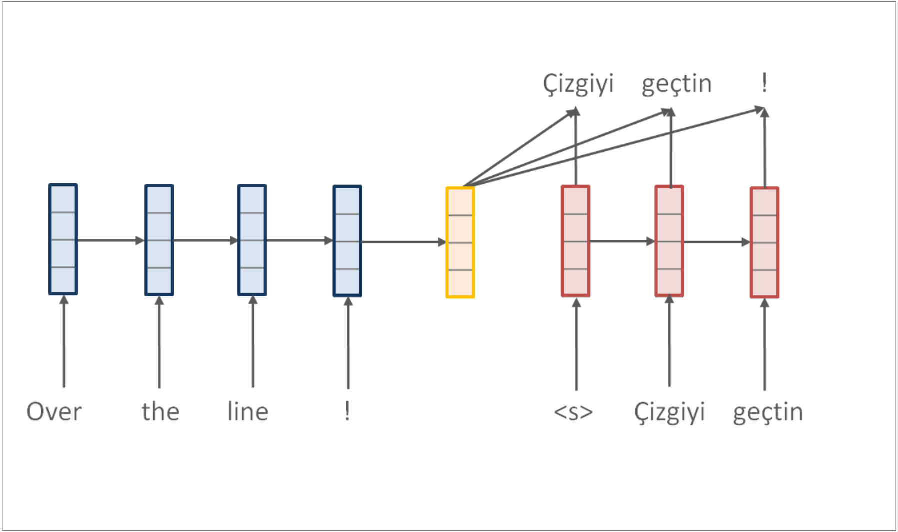
Example: Neural Machine Translation (Sutskever et al., 2014)



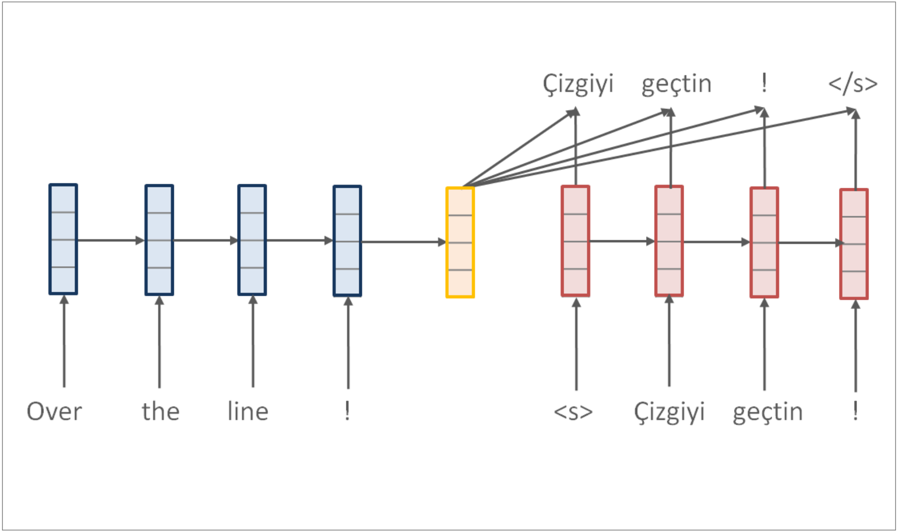
Example: Neural Machine Translation (Sutskever et al., 2014)



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



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



Encoder-Decoder with Attention

- 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...

Neural Attention

Input (sentence, image, etc.)

Memory-Bank Encoder (MLP, RNN, CNN)

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

Attention Distribution "where"

Context Vector "what"

Decoder

Neural Attention

Input (sentence, image, etc.)

⇓

Memory-Bank Encoder (MLP, RNN, CNN)

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

⇓

Attention Distribution "where"

Context Vector "what"

⇓

Decoder

Neural Attention

Input (sentence, image, etc.)

Memory-Bank Encoder (MLP, RNN, CNN)

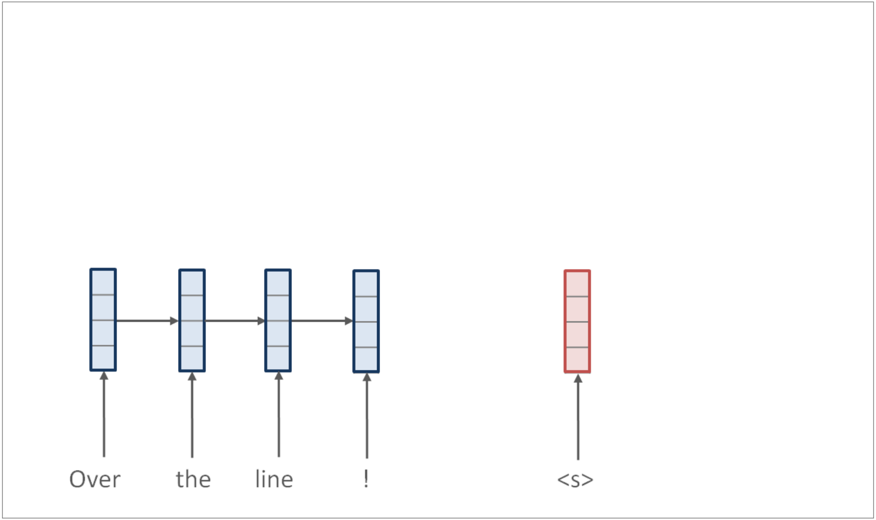
$$\text{Encoder}(\text{input}) = x_1, x_2, \ldots, x_T$$

Attention Distribution
"where"

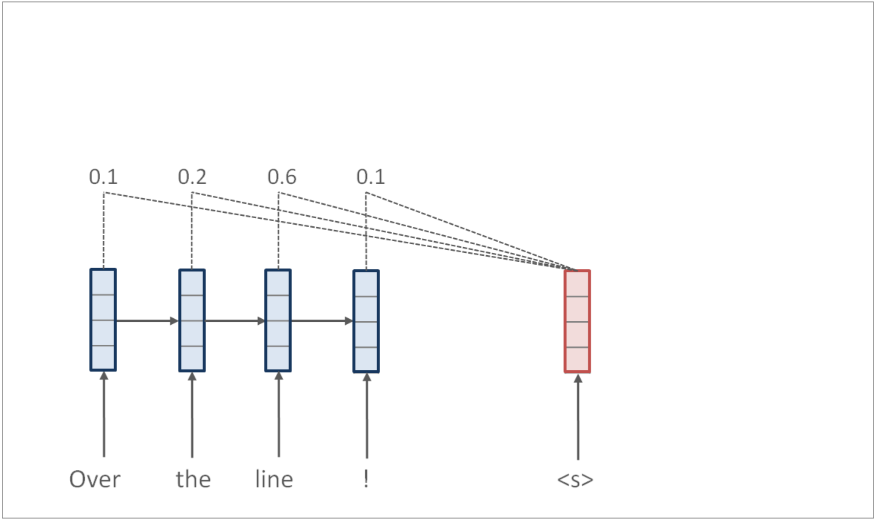
Context Vector
"what"

Decoder

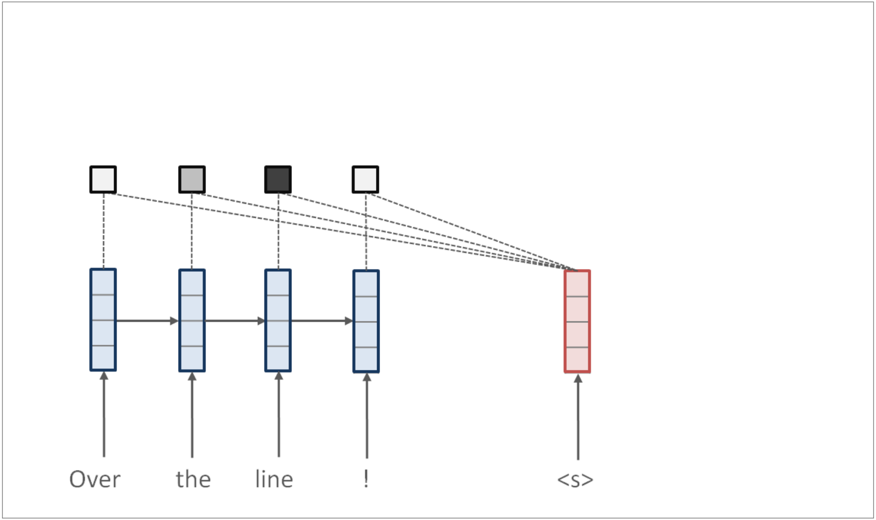
Attention-based Neural Machine Translation (Bahdanau et al., 2015)



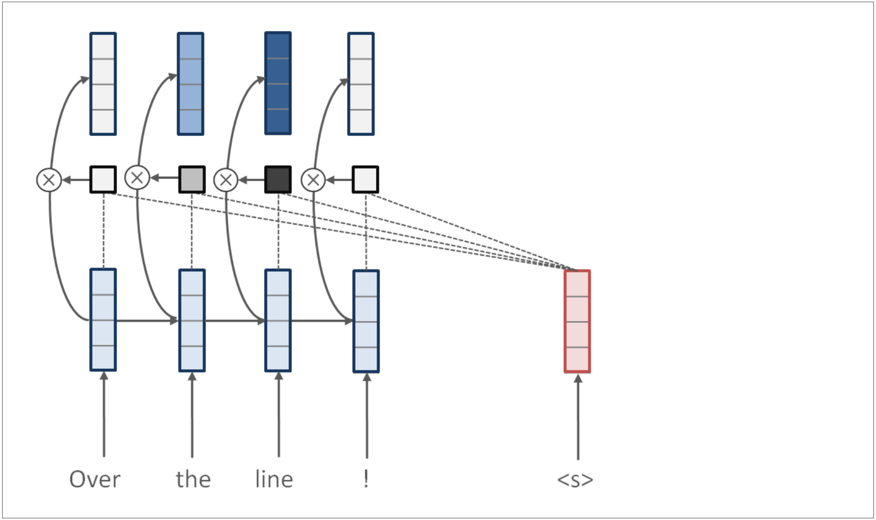
Attention-based Neural Machine Translation (Bahdanau et al., 2015)



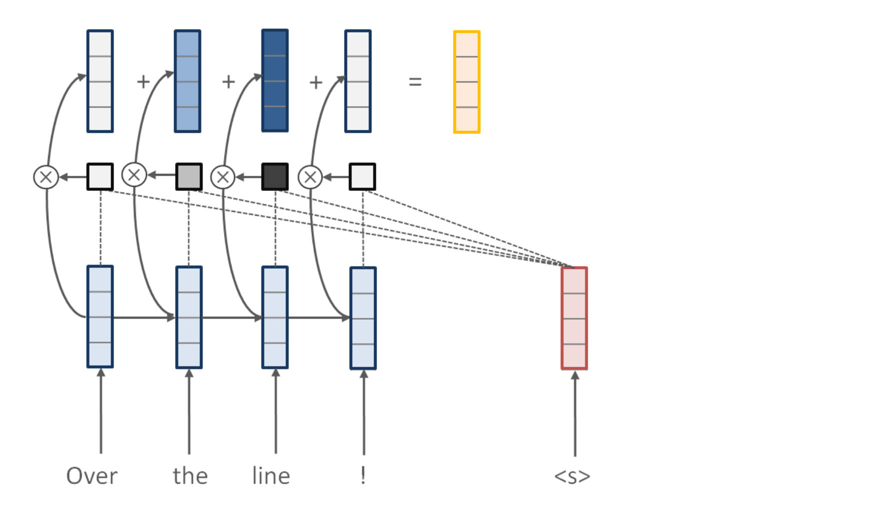
Attention-based Neural Machine Translation (Bahdanau et al., 2015)



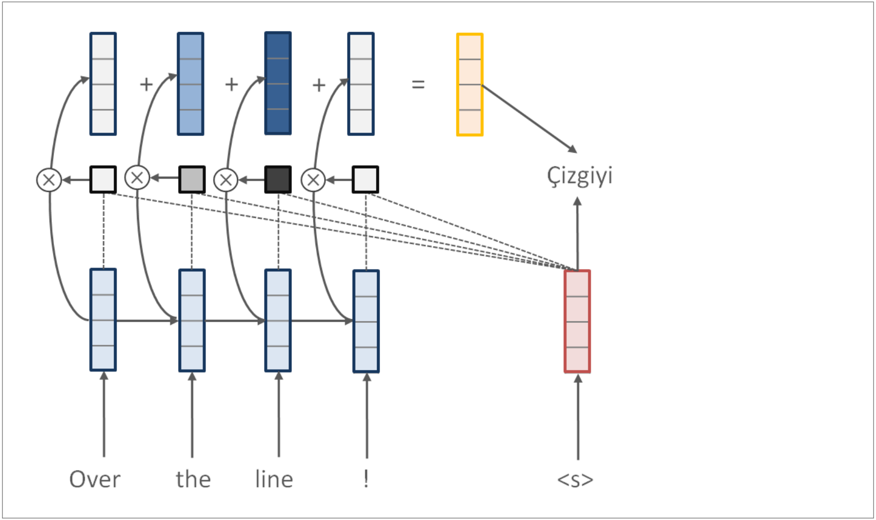
Attention-based Neural Machine Translation (Bahdanau et al., 2015)



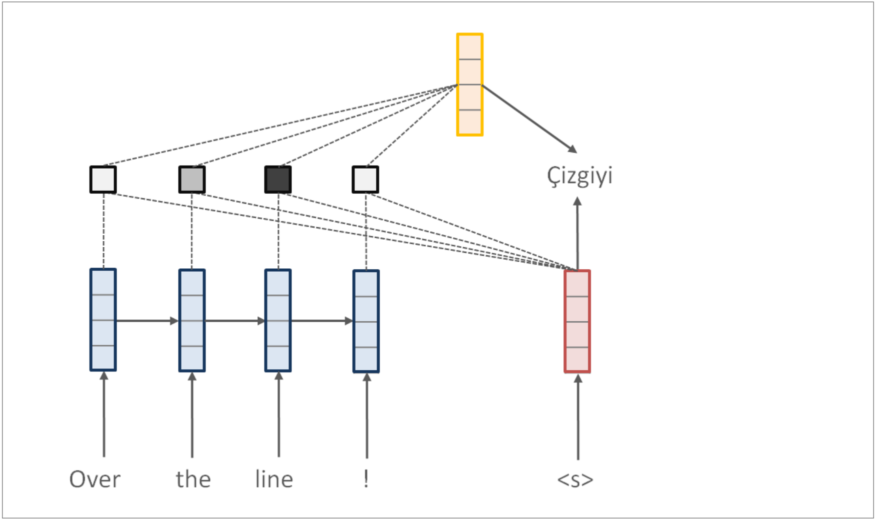
Attention-based Neural Machine Translation (Bahdanau et al., 2015)



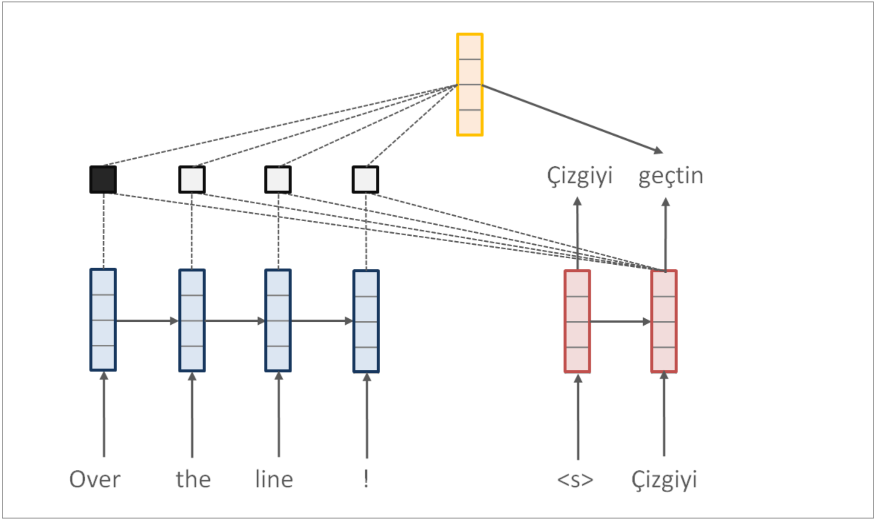
Attention-based Neural Machine Translation (Bahdanau et al., 2015)



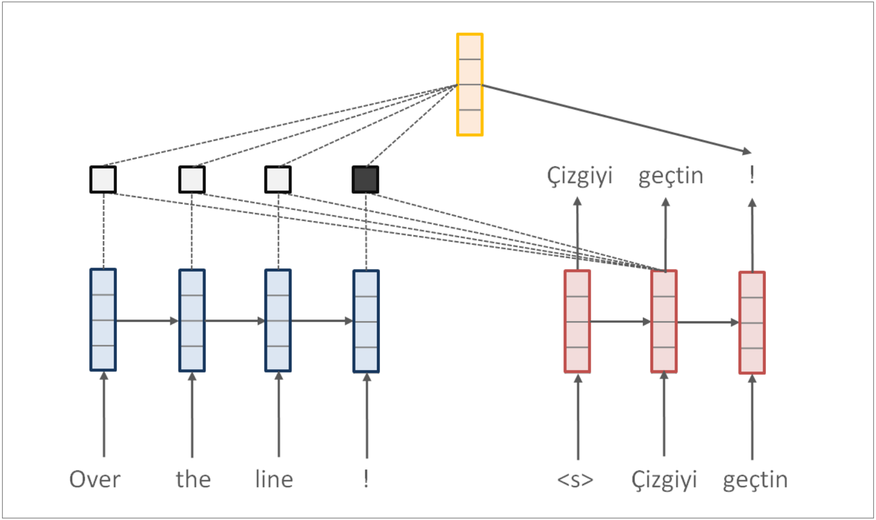
Attention-based Neural Machine Translation (Bahdanau et al., 2015)



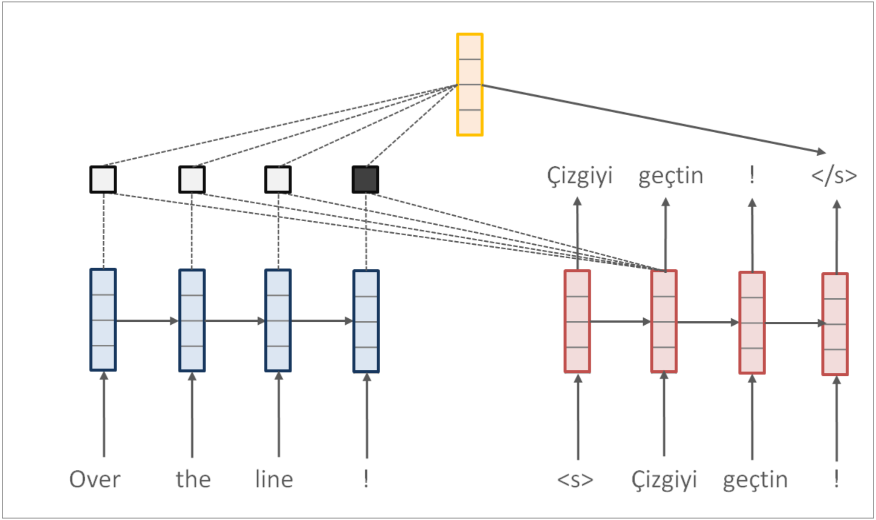
Attention-based Neural Machine Translation (Bahdanau et al., 2015)



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



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



Question Answering (Sukhbaatar et al., 2015)

Greg is a frog

Brian is a rhino

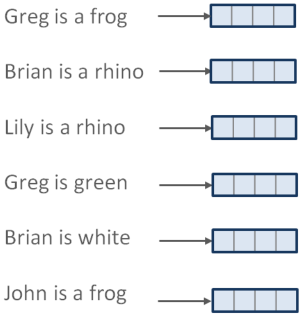
Lily is a rhino

Greg is green

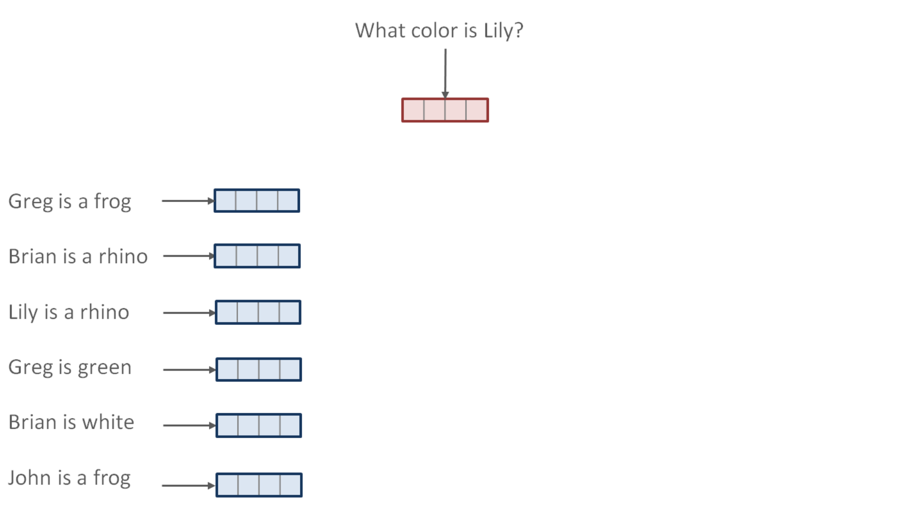
Brian is white

John is a frog

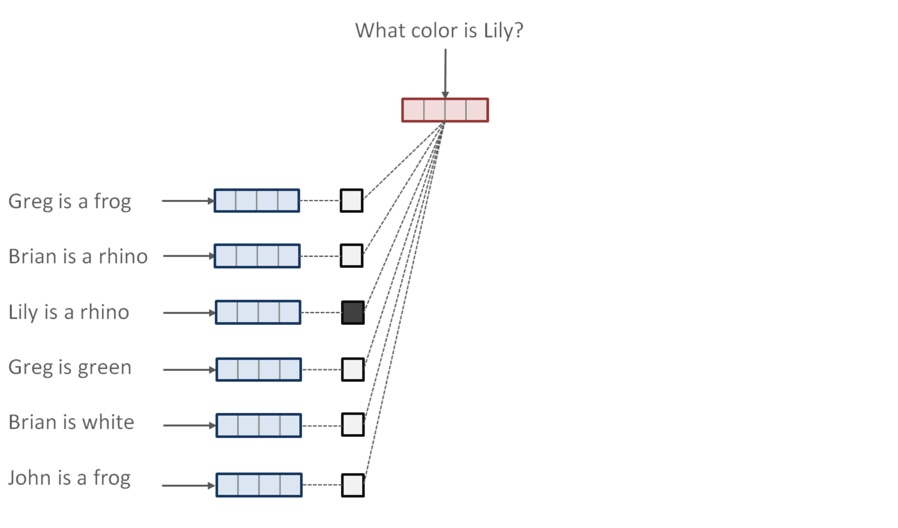
Question Answering (Sukhbaatar et al., 2015)



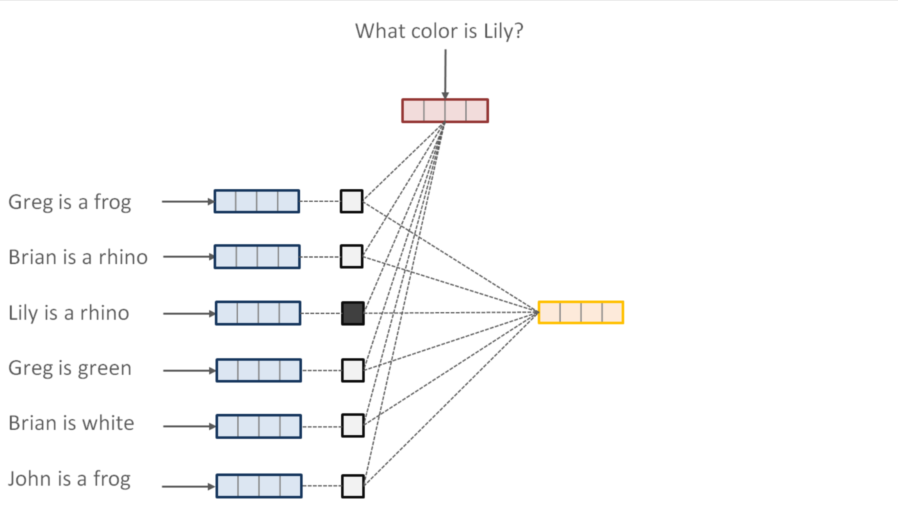
Question Answering (Sukhbaatar et al., 2015)



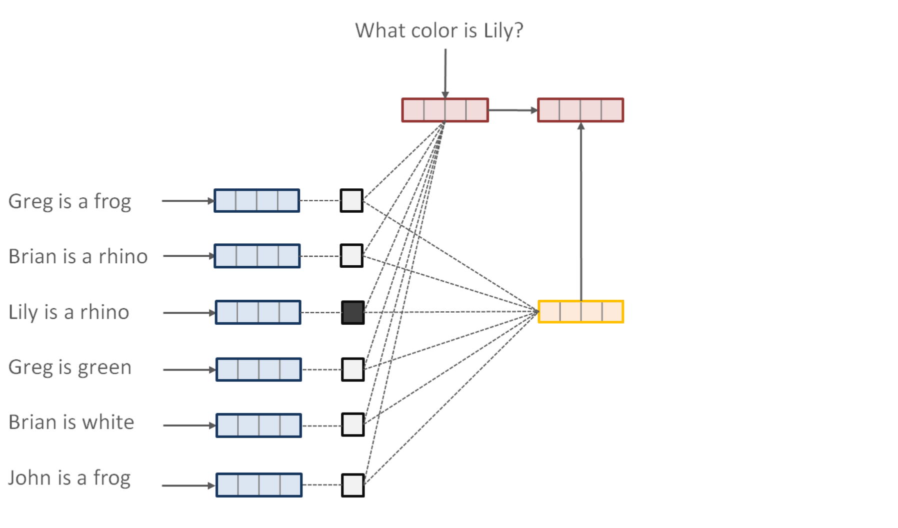
Question Answering (Sukhbaatar et al., 2015)



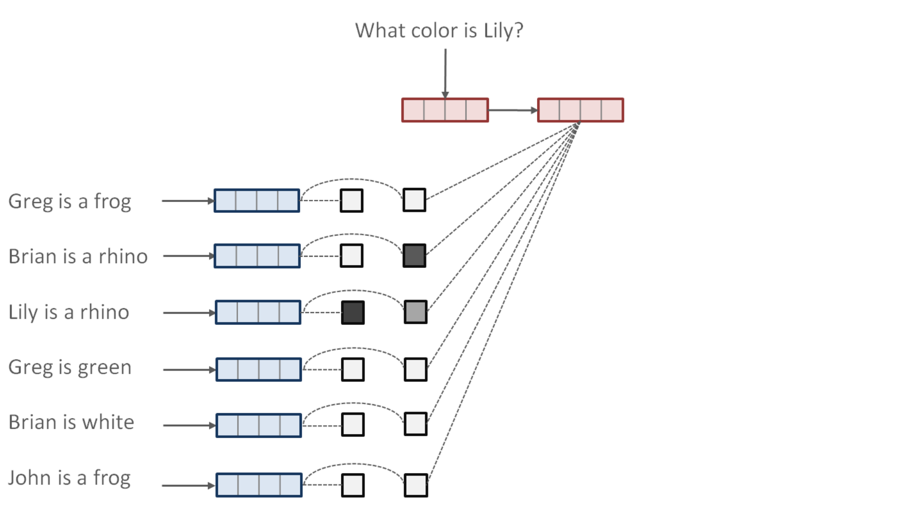
Question Answering (Sukhbaatar et al., 2015)



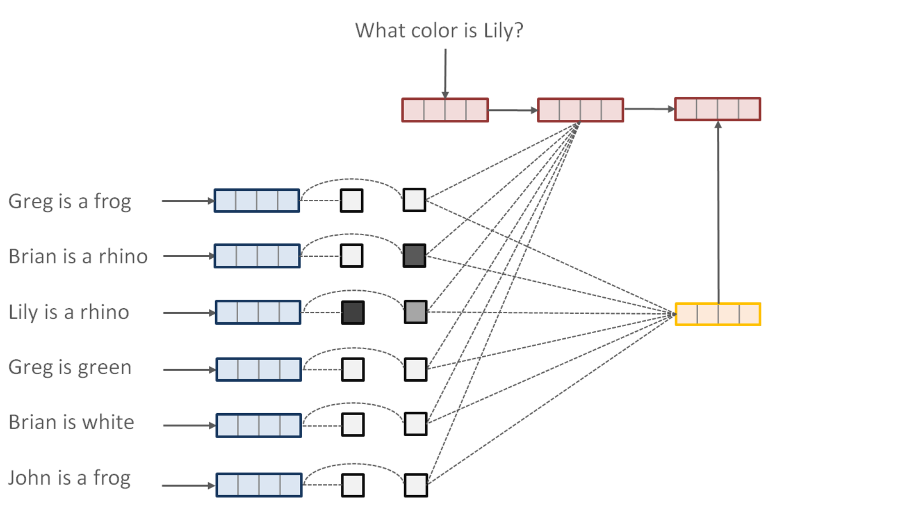
Question Answering (Sukhbaatar et al., 2015)



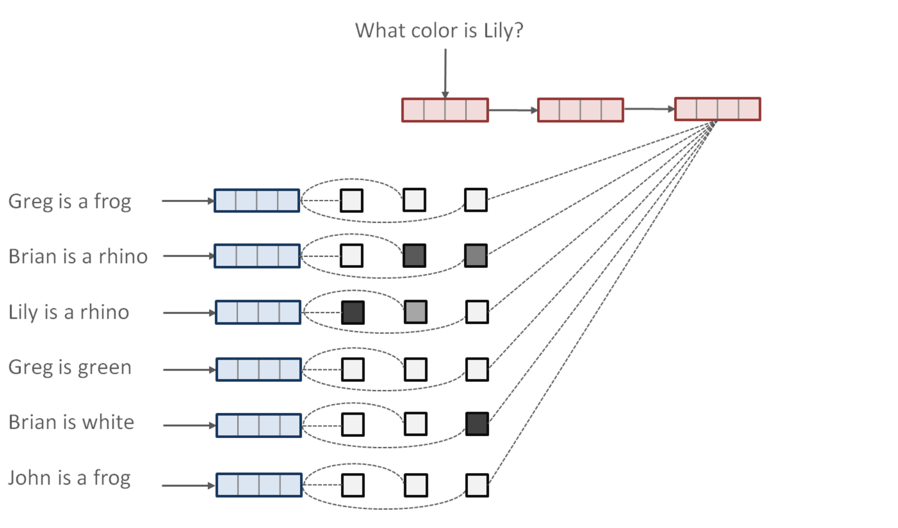
Question Answering (Sukhbaatar et al., 2015)



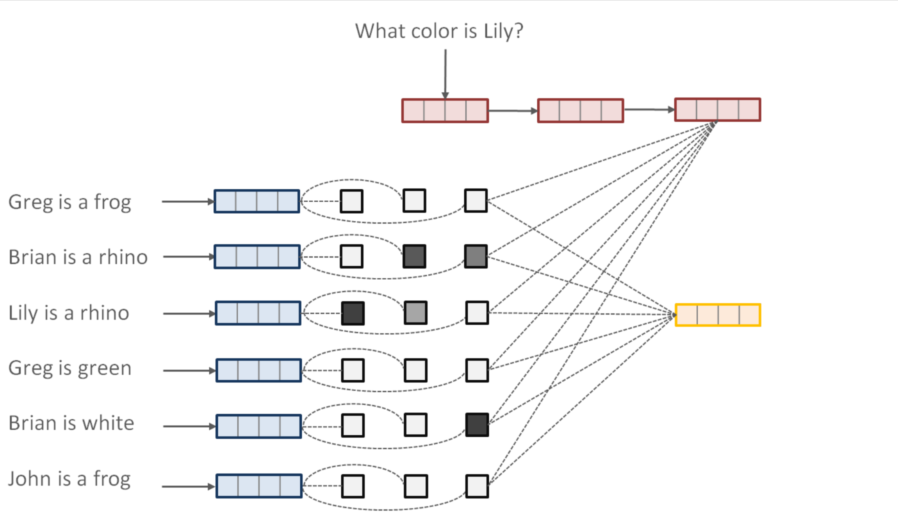
Question Answering (Sukhbaatar et al., 2015)



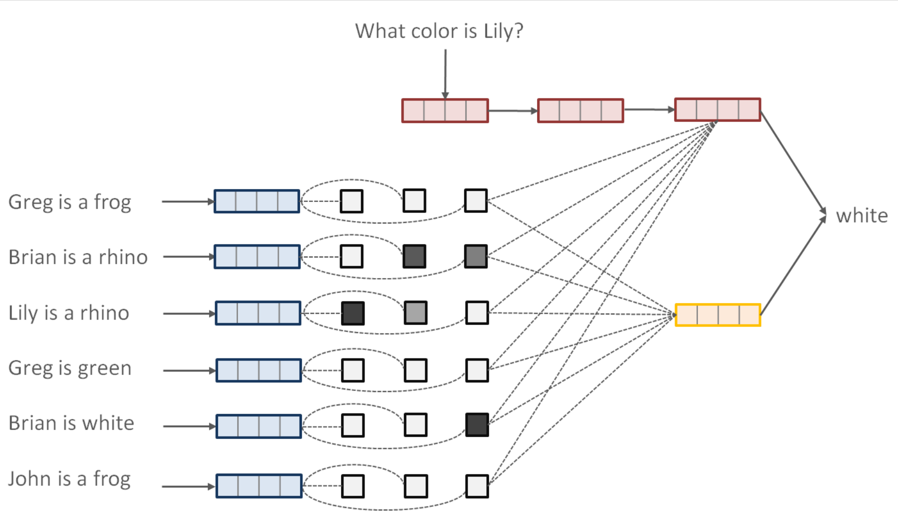
Question Answering (Sukhbaatar et al., 2015)



Question Answering (Sukhbaatar et al., 2015)



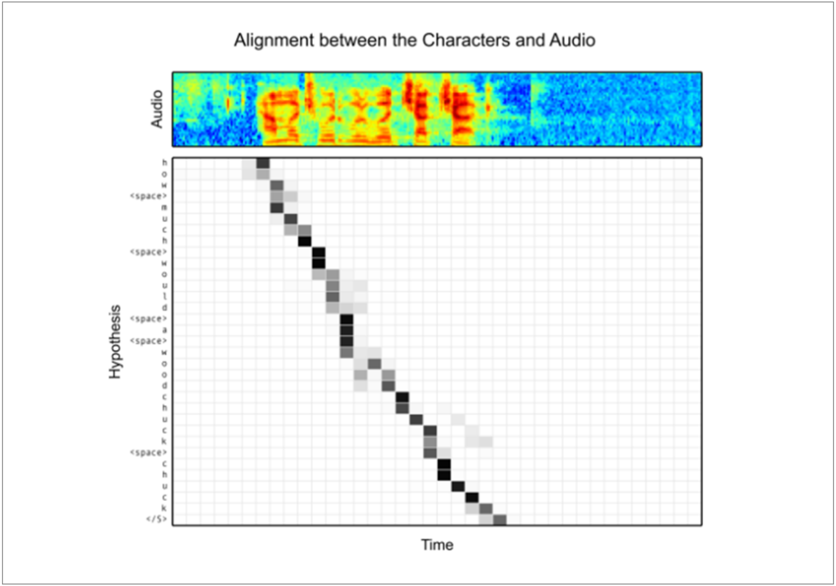
Question Answering (Sukhbaatar et al., 2015)



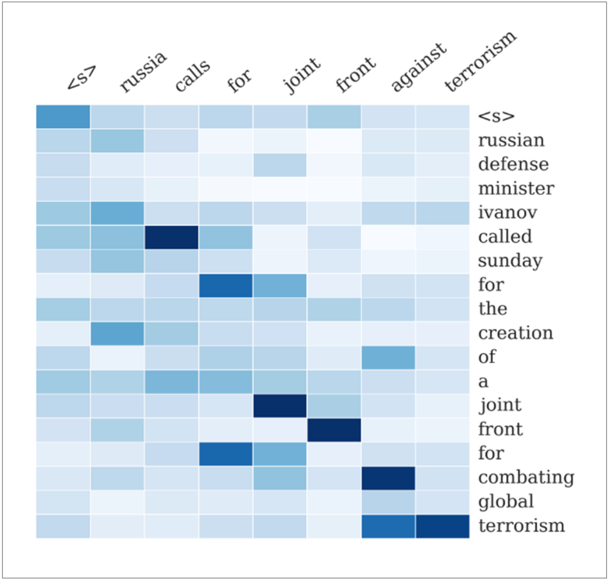
Other Applications: Image Captioning (Xu et al., 2015)

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

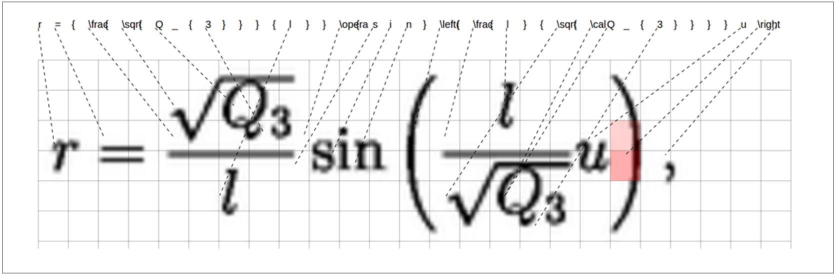
Other Applications: Speech Recognition (Chan et al., 2015)



Applications From HarvardNLP: Summarization (Rush et al., 2015)



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



Applications From HarvardNLP: OpenNMT

NMT

An open-source neural machine translation system.

English Français 简体中文 한국어
日本語 Русский العربية

Home

Quickstart [Lua]

Quickstart [Python]

Advanced guide

Models and Recipes

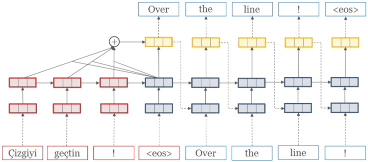
FAQ

About

Documentation

Home

OpenNMT is a industrial-strength, open-source (MIT) neural machine translation system utilizing the Torch/PyTorch mathematical toolkit.



OpenNMT is used as provided in production by major translation providers. The system is designed to be simple to use and easy to extend, while maintaining efficiency and state-of-the-art translation accuracy.

1. Deep Neural Networks for Text Processing and Generation
2. **Attention Networks**
3. Structured Attention Networks
   - Computational Challenges
   - Structured Attention In Practice
4. Conclusion and Future Work

Attention Networks: Notation

$x_1, \ldots, x_T$	Memory bank
q	Query
z	Memory selection (random variable)
$p(z \mid x, q; \theta)$	Attention distribution ("where")
$f(x, z)$	Annotation function ("what")
$c = \mathbb{E}_{z \mid x,q}[f(x, z)]$	Context Vector

End-to-End Requirements:

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

Attention Networks: Notation

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

End-to-End Requirements:

1. Need to compute attention $p(z = i \mid x, q; \theta)$
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
$c = \mathbb{E}[f(x, z)]$	Context Vector	

End-to-End Requirements:

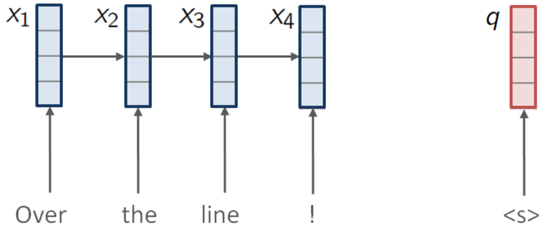
1. Need to compute attention $p(z = i \mid x, q; \theta)$

   $\implies$ softmax function

2. Need to backpropagate to learn parameters θ

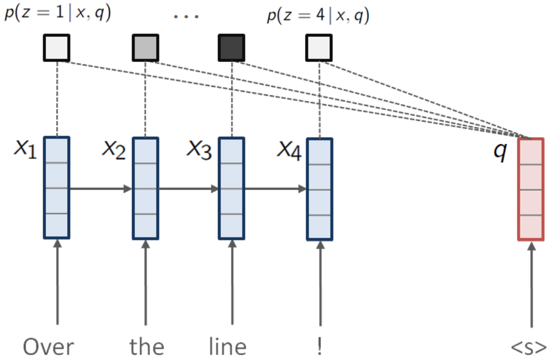
   $\implies$ Backprop through softmax function

Attention Networks: Machine Translation



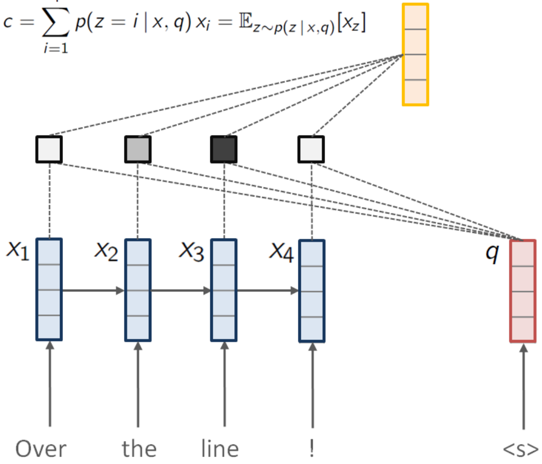
Attention Networks: Machine Translation

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

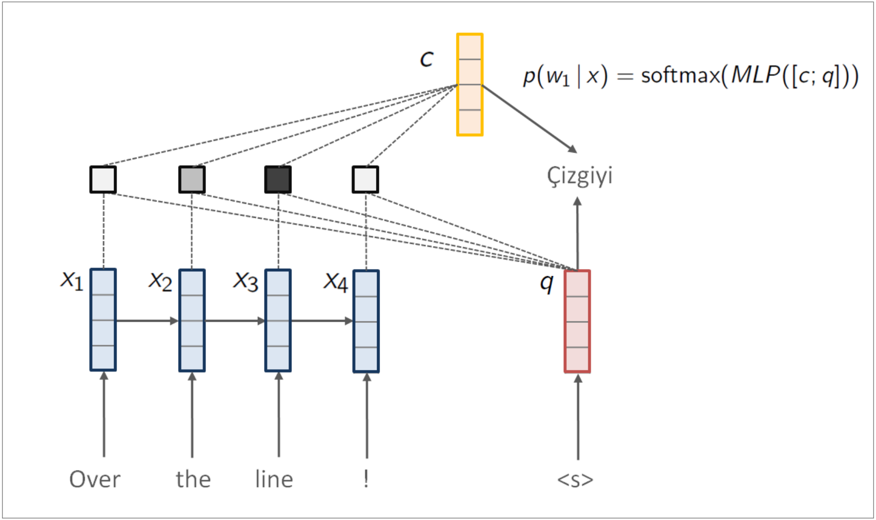


Attention Networks: Machine Translation

$$c = \sum_{i=1}^{4} p(z = i \mid x, q)\, x_i = \mathbb{E}_{z \sim p(z \mid x, q)}[x_z]$$



Attention Networks: Machine Translation



Structured Attention Networks

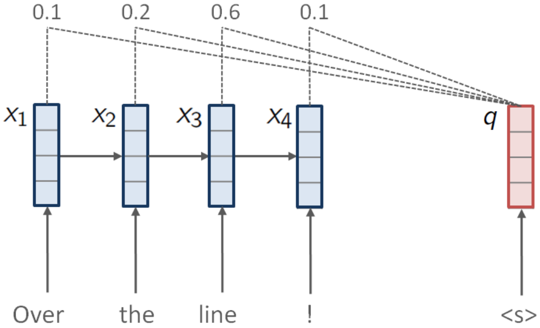
- Replace simple attention with distribution over a combinatorial set of structures
- Attention distribution represented with graphical model over multiple latent variables
- Compute attention using embedded inference .

New Model

$p(z \mid x, q; \theta)$ Attention distribution over structures z

Structured Attention Networks for Neural Machine Translation

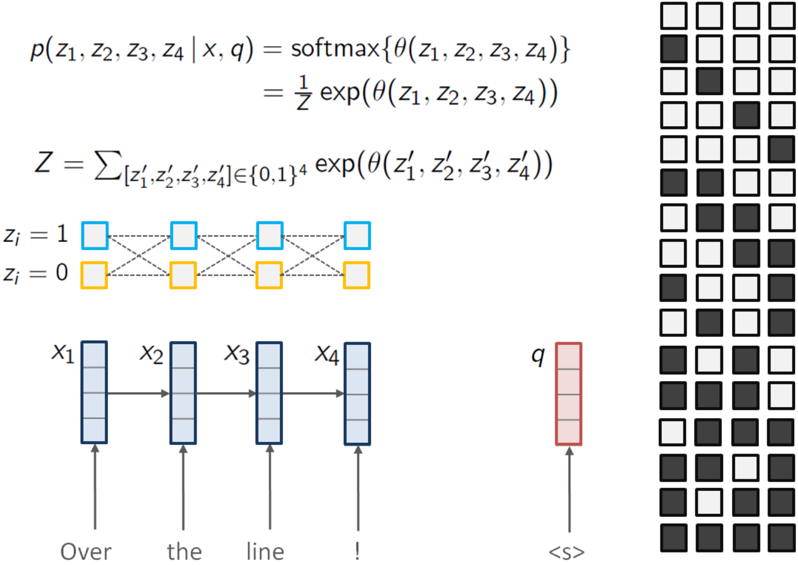
$\sum_{i=1}^{4} p(z = i \mid x, q) = 1$



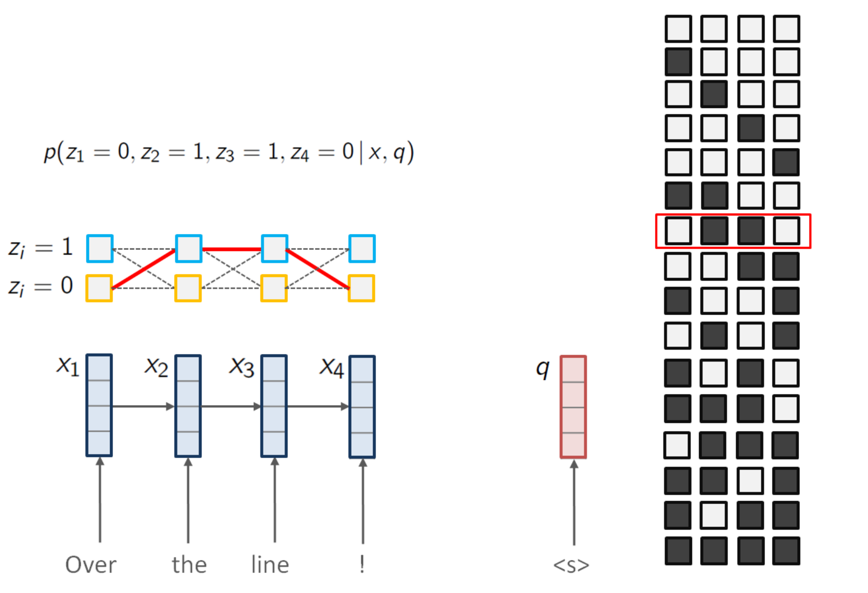
Structured Attention Networks

$$p(z_1, z_2, z_3, z_4 \mid x, q) = \text{softmax}\{\theta(z_1, z_2, z_3, z_4)\}$$
$$= \frac{1}{Z}\exp(\theta(z_1, z_2, z_3, z_4))$$

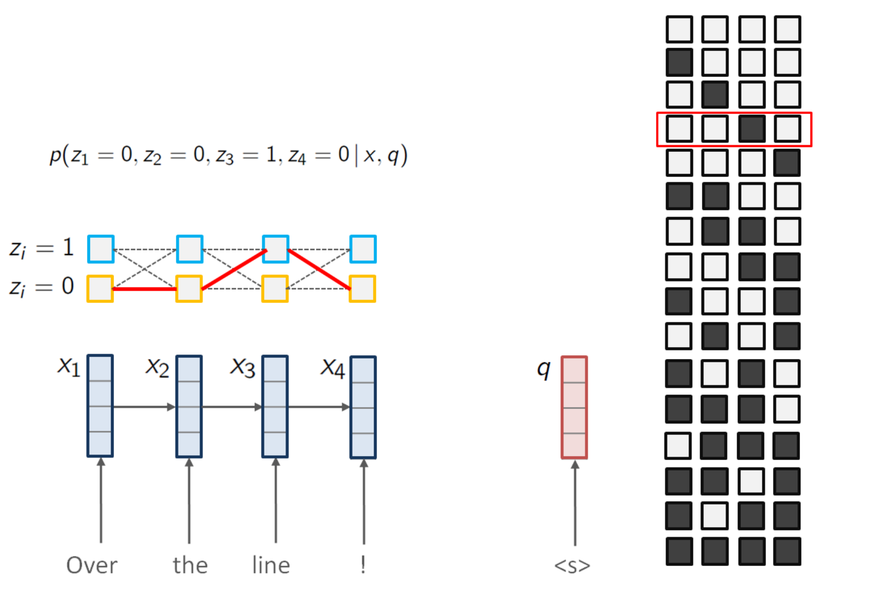
$$Z = \sum_{[z'_1, z'_2, z'_3, z'_4] \in \{0,1\}^4} \exp(\theta(z'_1, z'_2, z'_3, z'_4))$$



Structured Attention Networks for Neural Machine Translation

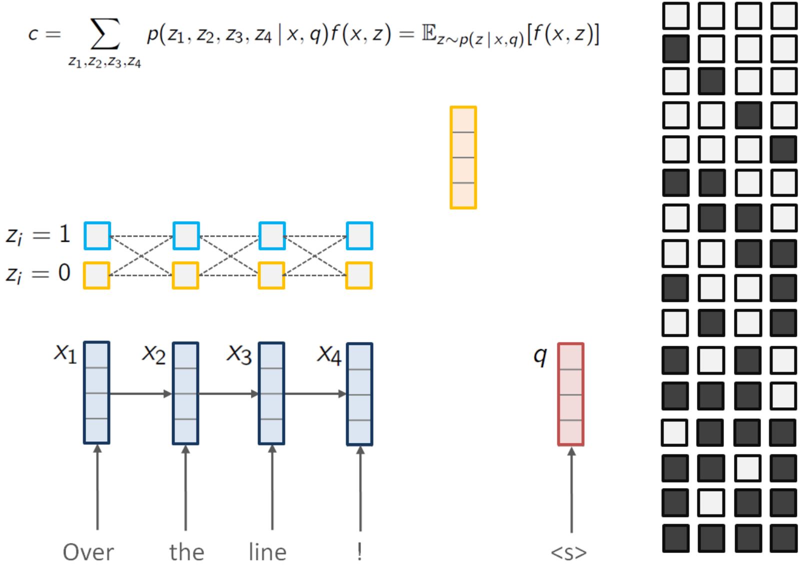


Structured Attention Networks for Neural Machine Translation



Structured Attention Networks for Neural Machine Translation

$$c = \sum_{z_1, z_2, z_3, z_4} p(z_1, z_2, z_3, z_4 \mid x, q) f(x, z) = \mathbb{E}_{z \sim p(z \mid x, q)}[f(x, z)]$$



Motivation: Structured Output Prediction

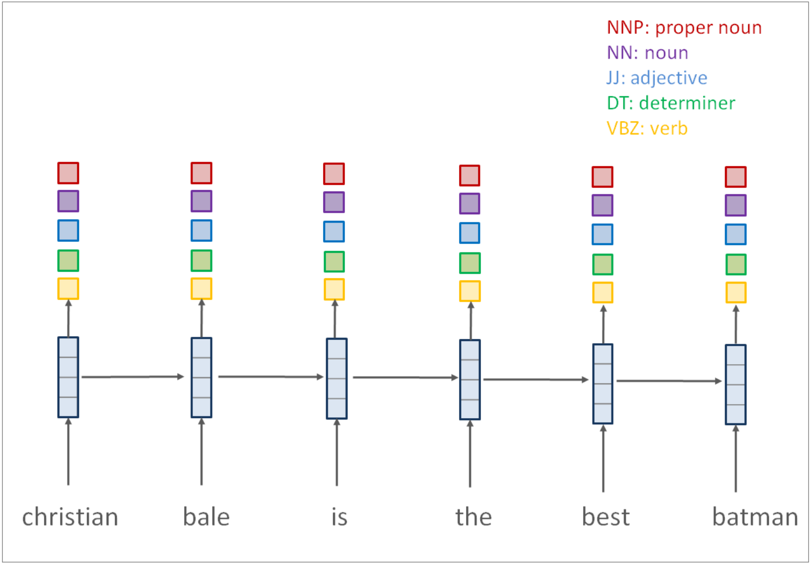
Modeling the structured **output** (i.e. graphical model on top of a neural net) has improved performance (LeCun et al., 1998; Lafferty et al., 2001; Collobert et al., 2011)

- Given a sequence $x = x_1, \dots, x_T$
- Factored potentials $\theta_{i,i+1}(z_i, z_{i+1}; x)$

$$p(z \mid x; \theta) = \operatorname{softmax}\left(\sum_{i=1}^{T-1} \theta_{i,i+1}(z_i, z_{i+1}; x)\right)$$

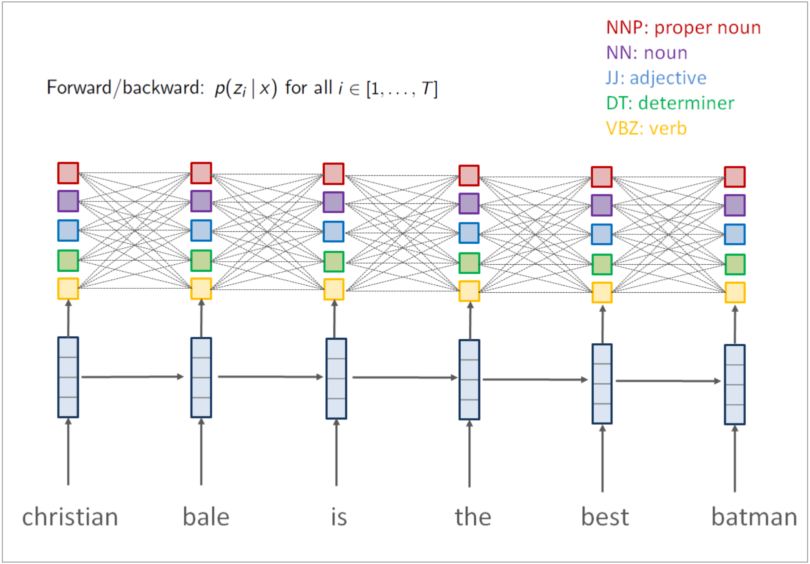
$$= \frac{1}{Z} \exp\left(\sum_{i=1}^{T-1} \theta_{i,i+1}(z_i, z_{i+1}; x)\right)$$

Neural CRF for Sequence Tagging (Collobert et al., 2011)



Factored potentials θ come from neural network.

Inference in Linear-Chain CRF



Fast algorithms for computing $p(z_i|x)$

1. Deep Neural Networks for Text Processing and Generation
2. Attention Networks
3. Structured Attention Networks
    - Computational Challenges
    - Structured Attention In Practice
4. Conclusion and Future Work

Structured Attention Networks: Notation

$x_1, \dots, x_T$	Memory bank	-
q	Query	-
$z_1, \dots, z_T$	Memory selection	Selection over structures
$p(z_i \mid x, q; \theta)$	Attention distribution	Marginal distributions
$f(x, z)$	Annotation function	Neural representation

Challenge: End-to-End Training

Requirements:

1. Compute attention distribution (marginals) $p(z_i \mid x, q; \theta)$

   $\Longrightarrow$ Forward-backward algorithm

2. Gradients wrt attention distribution parameters θ.

   $\Longrightarrow$ Backpropagation through forward-backward algorithm

Challenge: End-to-End Training

Requirements:

1. Compute attention distribution (marginals) $p(z_i \mid x, q; \theta)$

   $\implies$ Forward-backward algorithm

2. Gradients wrt attention distribution parameters θ.

   $\implies$ Backpropagation through forward-backward algorithm

Challenge: End-to-End Training

Requirements:

1. Compute attention distribution (marginals) $p(z_i \mid x, q; \theta)$

   $\implies$ Forward-backward algorithm

2. Gradients wrt attention distribution parameters θ.

   $\implies$ Backpropagation through forward-backward algorithm

Review: Forward-Backward Algorithm in Practice

θ: input potentials (e.g. from NN)

α, β: dynamic programming tables

procedure $\textsc{StructAttention}(\theta)$

Forward

for $i = 1, \ldots, n; z_i$ **do**

$\alpha[i, z_i] \leftarrow \sum_{z_{i-1}} \alpha[i-1, z_{i-1}] \times \exp(\theta_{i-1,i}(z_{i-1}, z_i))$

Backward

for $i = n, \ldots, 1; z_i$ **do**

$\beta[i, z_i] \leftarrow \sum_{z_{i+1}1} \beta[i+1, z_{i+1}] \times \exp(\theta_{i,i+1}(z_i, z_{i+1}))$

Forward-Backward Algorithm (Log-Space Semiring Trick)

θ: input potentials (e.g. from MLP or parameters)

$x \oplus y = \log(\exp(x) + \exp(y))$

$x \otimes y = x + y$

procedure $\textsc{StructAttention}(\theta)$

Forward

for $i = 1, \ldots, n; z_i$ **do**

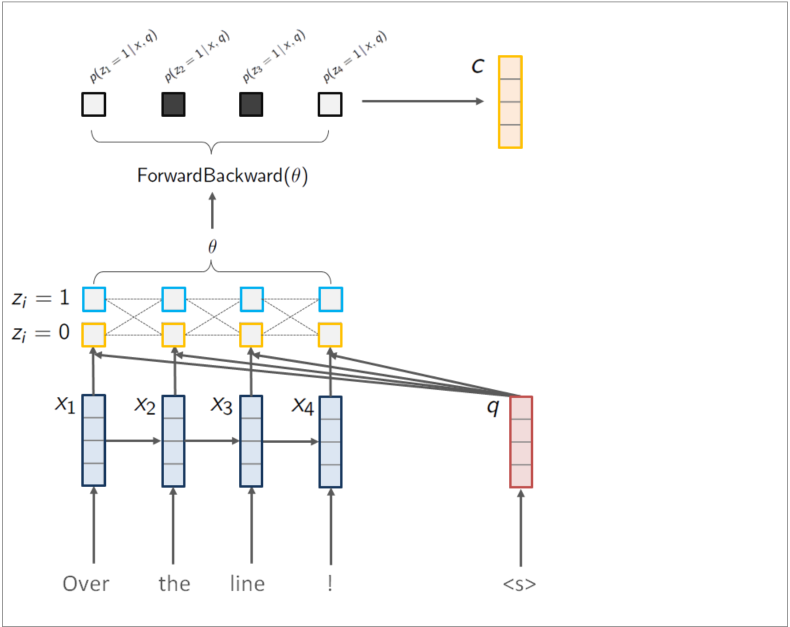
$\alpha[i, z_i] \leftarrow \bigoplus_{z_{i-1}} \alpha[i-1, y] \otimes \theta_{i-1,i}(z_{i-1}, z_i)$

Backward

for $i = n, \ldots, 1; z_i$ **do**

$\beta[i, z_i] \leftarrow \bigoplus_{z_{i+1}} \beta[i+1, z_{i+1}] \otimes \theta_{i,i+1}(z_i, z_{i+1})$

Structured Attention Networks for Neural Machine Translation



Backpropagating through Forward-Backward

$\nabla_p^{\mathcal{L}}$: Gradient of arbitrary loss $\mathcal{L}$ with respect to marginals p

procedure $\textsc{BackpropStructAtten}(\theta, p, \nabla_\alpha^{\mathcal{L}}, \nabla_\beta^{\mathcal{L}})$

Backprop Backward

for $i = n, \dots 1; z_i$ **do**

$$\hat{\beta}[i, z_i] \leftarrow \nabla_\alpha^{\mathcal{L}}[i, z_i] \oplus \bigoplus_{z_{i+1}} \theta_{i,i+1}(z_i, z_{i+1}) \otimes \hat{\beta}[i+1, z_{i+1}]$$

Backprop Forward

for $i = 1, \dots, n; z_i$ **do**

$$\hat{\alpha}[i, z_i] \leftarrow \nabla_\beta^{\mathcal{L}}[i, z_i] \oplus \bigoplus_{z_{i-1}} \theta_{i-1,i}(z_{i-1}, z_i) \otimes \hat{\alpha}[i-1, z_{i-1}]$$

Potential Gradients

for $i = 1, \dots, n; z_i, z_{i+1}$ **do**

$$\nabla_{\theta_{i-1,i}(z_i, z_{i+1})}^{\mathcal{L}} \leftarrow \mathrm{signexp}(\hat{\alpha}[i, z_i] \otimes \beta[i+1, z_{i+1}] \oplus \alpha[i, z_i] \otimes \hat{\beta}[i+1, z_{i+1}] \oplus \alpha[i, z_i] \otimes \beta[i+1, z_{i+1}] \otimes -A)$$

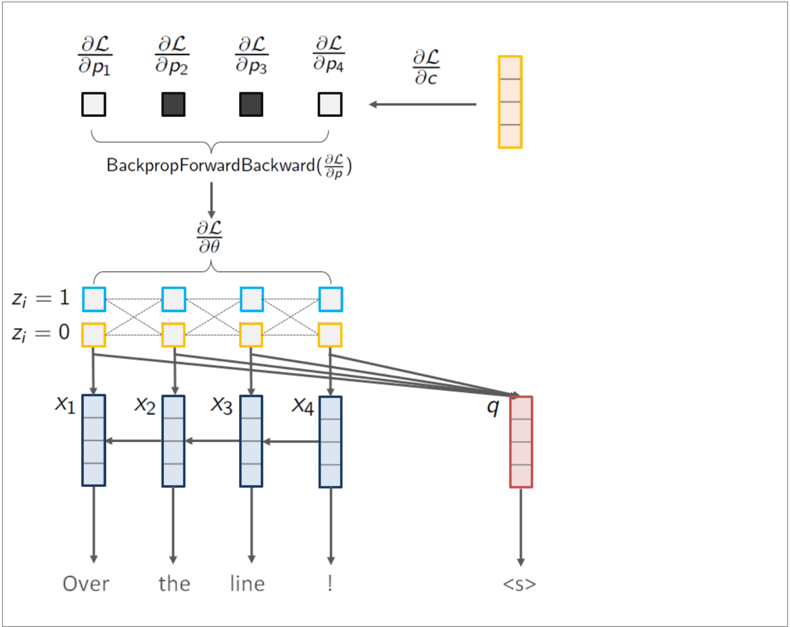
Interesting Issue: Negative Gradients Through Attention

- $\nabla_p^{\mathcal{L}}$: Gradient could be negative, but working in log-space!
- Signed Log-space semifield Trick (Li and Eisner, 2009)
- Use tuples (l_a, s_a) where $l_a = \log|a|$ and $s_a = \text{sign}(a)$

		$\oplus$	
s_a	s_b	l_{a+b}	s_{a+b}
$+$	$+$	$l_a + \log(1+d)$	$+$
$+$	$-$	$l_a + \log(1-d)$	$+$
$-$	$+$	$l_a + \log(1-d)$	$-$
$-$	$-$	$l_a + \log(1+d)$	$-$

(Similar rules for $\otimes$)

Structured Attention Networks for Neural Machine Translation



Implementation

(http://github.com/harvardnlp/struct-attn))

- General-purpose structured attention unit.
- All dynamic programming is GPU optimized for speed.
- Additionally supports pairwise potentials and marginals.

NLP Experiments

- Machine Translation
- Question Answering
- Natural Language Inference

Segmental-Attention for Neural Machine Translation

- Use segmentation CRF for attention, i.e. binary vectors of length n
- $p(z_1, \dots, z_T \mid x, q)$ parameterized with a linear-chain CRF.
- Neural "phrase-based" translation.

Unary potentials (Encoder RNN):

$$\theta_i(k) = \begin{cases} x_i W q, & k = 1 \\ 0, & k = 0 \end{cases}$$

Pairwise potentials (Simple Parameters):

4 additional binary parameters (i.e., $b_{0,0}, b_{0,1}, b_{1,0}, b_{1,1}$)

Neural Machine Translation Experiments

Data:

- Japanese $\rightarrow$ English (from WAT 2015)
- Traditionally, word segmentation as a preprocessing step
- Use structured attention learn an implicit segmentation model

Experiments:

- Japanese characters $\rightarrow$ English words
- Japanese words $\rightarrow$ English words

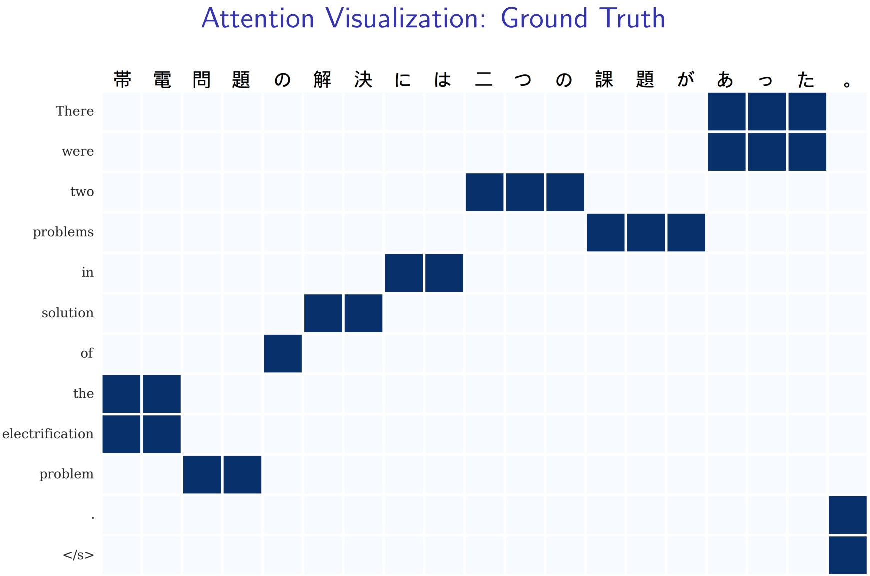
Neural Machine Translation Experiments

	Simple	Sigmoid	Structured
CHAR → WORD	12.6	13.1	14.6
WORD → WORD	14.1	13.8	14.3

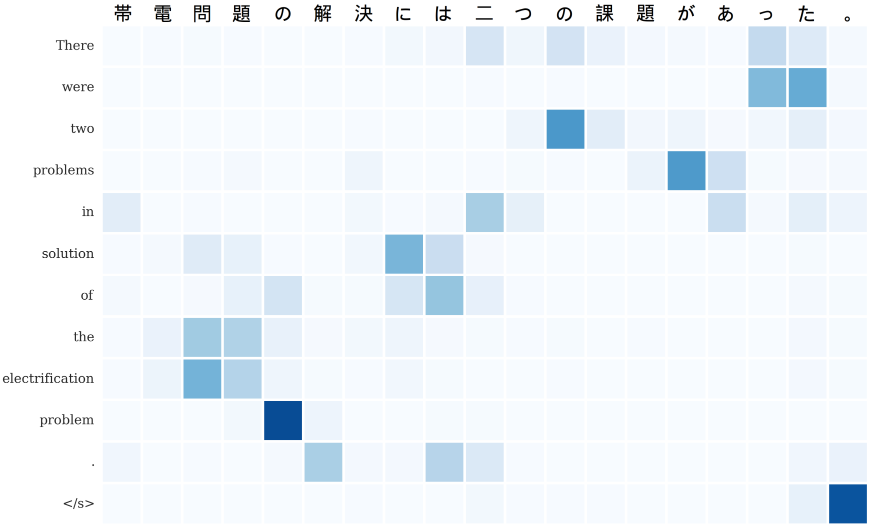
BLEU scores on test set (higher is better).

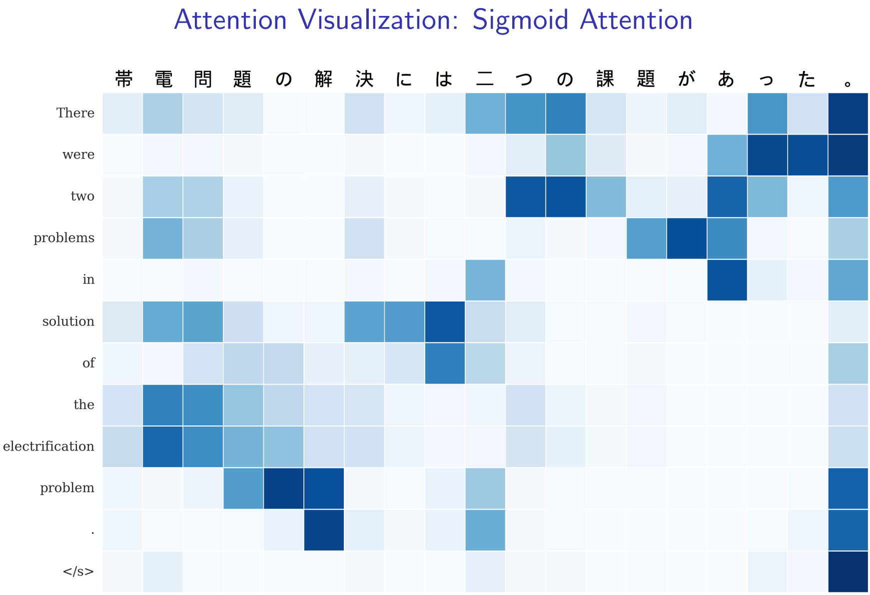
Models:

- Simple softmax attention
- Sigmoid attention
- Structured attention

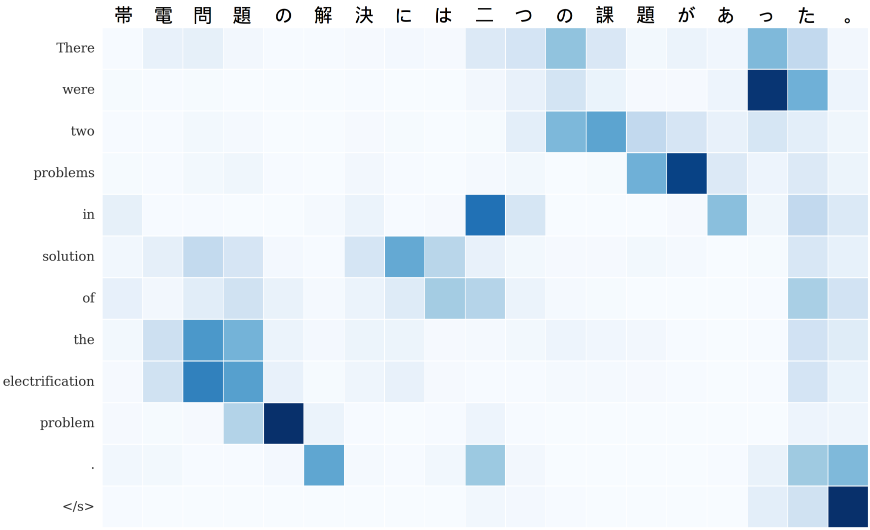

Attention Visualization: Ground Truth
帯 電 問 題 の 解 決 に は 二 つ の 課 題 が あ っ た 。
There
were
two
problems
in
solution
of
the
electrification
problem
.
</s>

Attention Visualization: Simple Attention



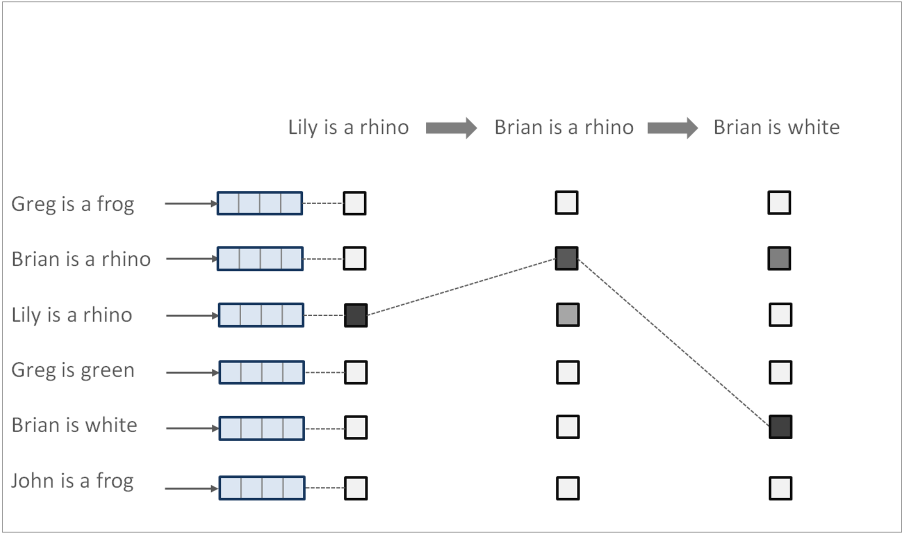

Attention Visualization: Sigmoid Attention
帯
電
問
題
の
解
決
に
は
二
つ
の
課
題
が
あ
っ
た
。
There
were
two
problems
in
solution
of
the
electrification
problem
.
</s>

Attention Visualization: Structured Attention



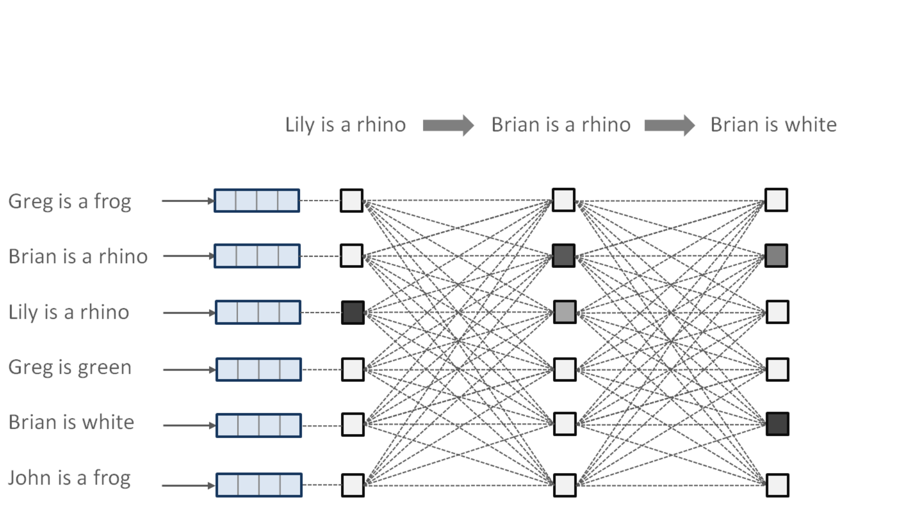
Simple Non-Factoid Question Answering

Simple attention: Greedy soft-selection of K supporting facts



Structured Attention Networks for Question Answering

Structured attention: Consider all possible sequences

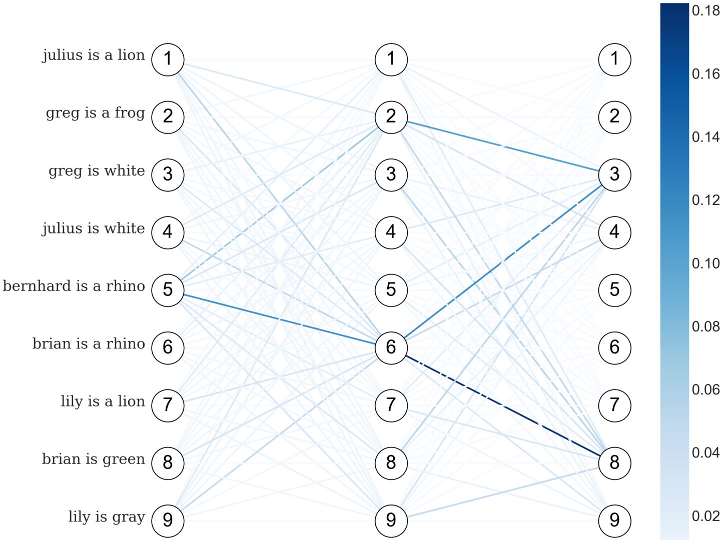


Structured Attention Networks for Question Answering

baBi tasks (Weston et al., 2015): 1k questions per task

		Simple		Structured	
Task	K	Ans %	Fact %	Ans %	Fact %
Task 02	2	87.3	46.8	84.7	81.8
Task 03	3	52.6	1.4	40.5	0.1
Task 11	2	97.8	38.2	97.7	80.8
Task 13	2	95.6	14.8	97.0	36.4
Task 14	2	99.9	77.6	99.7	98.2
Task 15	2	100.0	59.3	100.0	89.5
Task 16	3	97.1	91.0	97.9	85.6
Task 17	2	61.1	23.9	60.6	49.6
Task 18	2	86.4	3.3	92.2	3.9
Task 19	2	21.3	10.2	24.4	11.5
Average	–	81.4	39.6	81.0	53.7

Visualization of Structured Attention



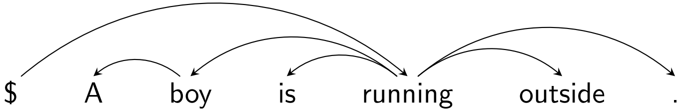
Question: what color is bernhard? green

Correct Facts: 5, 6, 8

Natural Language Inference

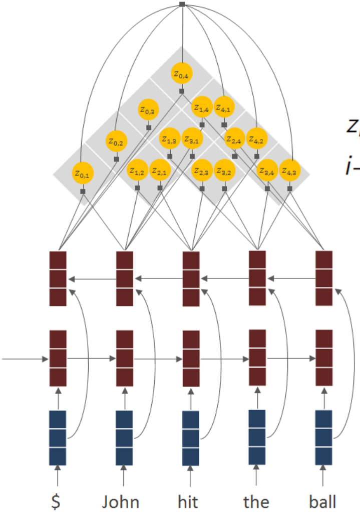
Given a premise (P) and a hypothesis (H), predict the relationship: Entailment (E), Contradiction (C), Neutral (N)

P	The boy is running through a grassy area.	
H	The boy is in his room.	C
	A boy is running outside.	E
	The boy is in a park.	N



Many existing models run parsing as a preprocessing step and attend over parse trees.

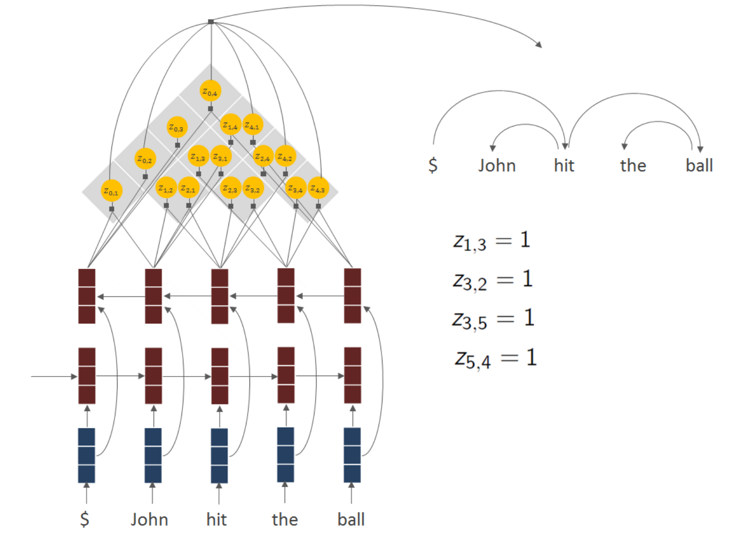
Neural CRF Parsing (Durrett and Klein, 2015; Kipperwasser and Goldberg, 2016)



$z_{ij} = 1 \implies$

i-th word is parent of j-th word

Neural CRF Parsing (Durrett and Klein, 2015; Kipperwasser and Goldberg, 2016)



$$z_{1,3} = 1$$

$$z_{3,2} = 1$$

$$z_{3,5} = 1$$

$$z_{5,4} = 1$$

Syntactic Attention Network

1. Attention distribution (probability of a parse tree)

   $\Longrightarrow$ Inside/outside algorithm

2. Gradients wrt attention distribution parameters: $\frac{\partial \mathcal{L}}{\partial \theta}$

   $\Longrightarrow$ Backpropagation through inside/outside algorithm

Forward/backward pass on inside-outside version of Eisner's algorithm (Eisner, 1996) takes $O(T^3)$ time.

Syntactic Attention Network

1. Attention distribution (probability of a parse tree)
   $\Longrightarrow$ Inside/outside algorithm
2. Gradients wrt attention distribution parameters: $\frac{\partial \mathcal{L}}{\partial \theta}$
   $\Longrightarrow$ Backpropagation through inside/outside algorithm

Forward/backward pass on inside-outside version of Eisner's algorithm (Eisner, 1996) takes $O(T^3)$ time.

Syntactic Attention Network

1. Attention distribution (probability of a parse tree)
   $\Longrightarrow$ Inside/outside algorithm
2. Gradients wrt attention distribution parameters: $\frac{\partial \mathcal{L}}{\partial \theta}$
   $\Longrightarrow$ Backpropagation through inside/outside algorithm

Forward/backward pass on inside-outside version of Eisner's algorithm (Eisner, 1996) takes $O(T^3)$ time.

Syntactic Attention Network

1. Attention distribution (probability of a parse tree)
   $\Longrightarrow$ Inside/outside algorithm
2. Gradients wrt attention distribution parameters: $\frac{\partial \mathcal{L}}{\partial \theta}$
   $\Longrightarrow$ Backpropagation through inside/outside algorithm

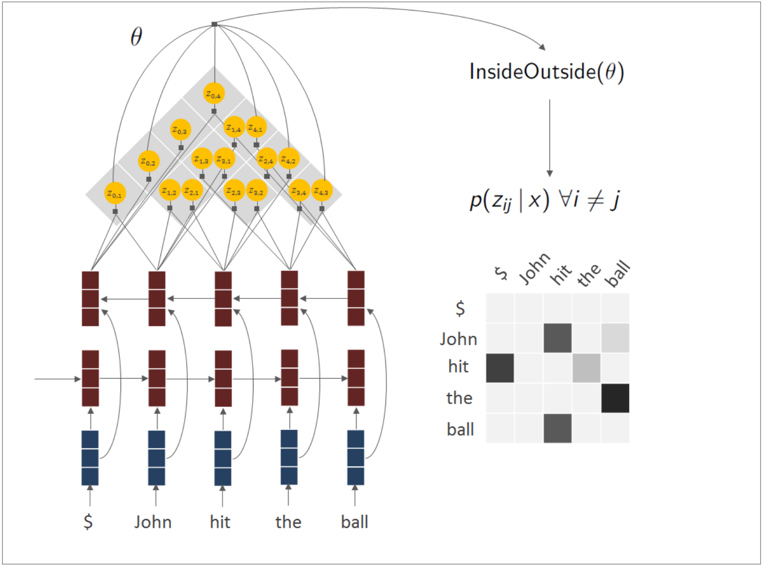
Forward/backward pass on inside-outside version of Eisner's algorithm (Eisner, 1996) takes $O(T^3)$ time.

Backpropagation through Inside-Outside Algorithm

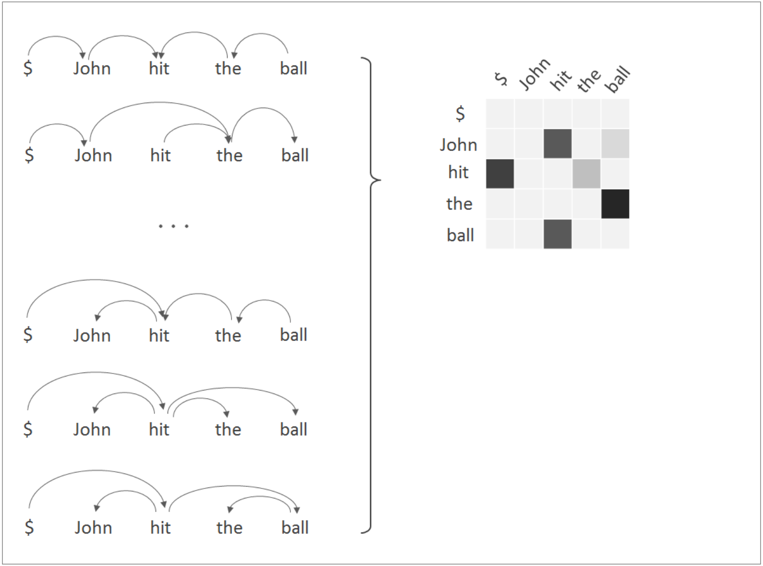
procedure InsideOutside(θ)
$\alpha, \beta \leftarrow -\infty$ ▷ Initialize log of inside (α), outside (β) tables
for $i = 1, \dots, n$ **do**
$\alpha[i, i, L, 1] \leftarrow 0$
$\alpha[i, i, R, 1] \leftarrow 0$
$\beta[1, n, R, 1] \leftarrow 0$
for $k = 1, \dots, n$ **do** ▷ Inside step
for $s = 1, \dots, n - k$ **do**
$t \leftarrow s + k$
$\alpha[s, t, R, 0] \leftarrow \bigoplus_{u \in [s, t-1]} \alpha[s, u, R, 1] \otimes \alpha[u + 1, t, L, 1] \otimes \theta_{st}$
$\alpha[s, t, L, 0] \leftarrow \bigoplus_{u \in [s, t-1]} \alpha[s, u, R, 1] \otimes \alpha[u + 1, t, L, 1] \otimes \theta_{ts}$
$\alpha[s, t, R, 1] \leftarrow \bigoplus_{u \in [s+1, t]} \alpha[s, u, R, 0] \otimes \alpha[u, t, R, 1]$
$\alpha[s, t, L, 1] \leftarrow \bigoplus_{u \in [s, t-1]} \alpha[s, u, L, 1] \otimes \alpha[u, t, L, 0]$
for $k = n, \dots, 1$ **do** ▷ Outside step
for $s = 1, \dots, n - k$ **do**
$t \leftarrow s + k$
for $u = s + 1, \dots, t$ **do**
$\beta[s, u, R, 0] \leftarrow_{\oplus} \beta[s, t, R, 1] \otimes \alpha[u, t, R, 1]$
$\beta[u, t, R, 1] \leftarrow_{\oplus} \beta[s, t, R, 1] \otimes \alpha[s, u, R, 0]$
if $s > 1$ **then**
for $u = s, \dots, t - 1$ **do**
$\beta[s, u, L, 1] \leftarrow_{\oplus} \beta[s, t, L, 1] \otimes \alpha[u, t, L, 0]$
$\beta[u, t, L, 0] \leftarrow_{\oplus} \beta[s, t, L, 1] \otimes \alpha[s, u, L, 1]$
for $u = s, \dots, t - 1$ **do**
$\beta[s, u, R, 1] \leftarrow_{\oplus} \beta[s, t, R, 0] \otimes \alpha[u + 1, t, L, 1] \otimes \theta_{st}$
$\beta[u + 1, t, L, 1] \leftarrow_{\oplus} \beta[s, t, R, 0] \otimes \alpha[s, u, R, 1] \otimes \theta_{st}$
if $s > 1$ **then**
for $u = s, \dots, t - 1$ **do**
$\beta[s, u, R, 1] \leftarrow_{\oplus} \beta[s, t, L, 0] \otimes \alpha[u + 1, t, L, 1] \otimes \theta_{ts}$
$\beta[u + 1, t, L, 1] \leftarrow_{\oplus} \beta[s, t, L, 0] \otimes \alpha[s, u, R, 1] \otimes \theta_{ts}$
$A \leftarrow \alpha[1, n, R, 1]$ ▷ Log partition
for $s = 1, \dots, n - 1$ **do** ▷ Compute marginals. Note that $p[s, t] = p(z_{st} = 1 \mid x)$
for $t = s + 1, \dots, n$ **do**
$p[s, t] \leftarrow \exp(\alpha[s, t, R, 0] \otimes \beta[s, t, R, 0] \otimes -A)$
if $s > 1$ **then**
$p[t, s] \leftarrow \exp(\alpha[s, t, L, 0] \otimes \beta[s, t, L, 0] \otimes -A)$
return p

procedure BackpropInsideOutside($\theta, p, \nabla^{\mathcal{L}}_{p}$)
for $s, t = 1, \dots, n; s \neq t$ **do** ▷ Backpropagation uses the identity $\nabla^{\mathcal{L}}_{\theta} = (p \odot \nabla^{\mathcal{L}}_{p}) \nabla^{\log p}_{\theta}$
$\delta[s, t] \leftarrow \log p[s, t] \otimes \log \nabla^{\mathcal{L}}_{p}[s, t]$ ▷ $\delta = \log(p \odot \nabla^{\mathcal{L}}_{p})$
$\nabla^{\mathcal{L}}_{\alpha}, \nabla^{\mathcal{L}}_{\beta}, \log \nabla^{\mathcal{L}}_{\theta} \leftarrow -\infty$ ▷ Initialize inside ($\nabla^{\mathcal{L}}_{\alpha}$), outside ($\nabla^{\mathcal{L}}_{\beta}$) gradients, and log of $\nabla^{\mathcal{L}}_{\theta}$
for $s = 1, \dots, n - 1$ **do** ▷ Backpropagate δ to $\nabla^{\mathcal{L}}_{\alpha}$ and $\nabla^{\mathcal{L}}_{\beta}$
for $t = s + 1, \dots, n$ **do**
$\nabla^{\mathcal{L}}_{\alpha}[s, t, R, 0], \nabla^{\mathcal{L}}_{\beta}[s, t, R, 0] \leftarrow \delta[s, t]$
$\nabla^{\mathcal{L}}_{\alpha}[1, n, R, 1] \leftarrow_{\oplus} -\delta[s, t]$
if $s > 1$ **then**
$\nabla^{\mathcal{L}}_{\alpha}[s, t, L, 0], \nabla^{\mathcal{L}}_{\beta}[s, t, L, 0] \leftarrow \delta[t, s]$
$\nabla^{\mathcal{L}}_{\alpha}[1, n, R, 1] \leftarrow_{\oplus} -\delta[s, t]$
for $k = 1, \dots, n$ **do** ▷ Backpropagate through outside step
for $s = 1, \dots, n - k$ **do**
$t \leftarrow s + k$
$\nu \leftarrow \nabla^{\mathcal{L}}_{\beta}[s, t, R, 0] \otimes \beta[s, t, R, 0]$ ▷ ν, γ are temporary values
for $u = t, \dots, n$ **do**
$\nabla^{\mathcal{L}}_{\beta}[s, u, R, 1], \nabla^{\mathcal{L}}_{\alpha}[t, u, R, 1] \leftarrow_{\oplus} \nu \otimes \beta[s, u, R, 1] \otimes \alpha[t, u, R, 1]$
if $s > 1$ **then**
$\nu \leftarrow \nabla^{\mathcal{L}}_{\beta}[s, t, L, 0] \otimes \beta[s, t, L, 0]$
for $u = 1, \dots, s$ **do**
$\nabla^{\mathcal{L}}_{\beta}[u, t, L, 1], \nabla^{\mathcal{L}}_{\alpha}[u, s, L, 1] \leftarrow_{\oplus} \nu \otimes \beta[u, t, L, 1] \otimes \alpha[u, s, L, 1]$
$\nu \leftarrow \nabla^{\mathcal{L}}_{\beta}[s, t, L, 1] \otimes \beta[s, t, L, 1]$
for $u = t, \dots, n$ **do**
$\nabla^{\mathcal{L}}_{\beta}[s, u, L, 1], \nabla^{\mathcal{L}}_{\alpha}[t, u, L, 0] \leftarrow_{\oplus} \nu \otimes \beta[s, u, L, 1] \otimes \alpha[t, u, L, 1]$
for $u = 1, \dots, s - 1$ **do**
$\gamma \leftarrow \beta[u, t, R, 0] \otimes \alpha[u, s - 1, R, 1] \otimes \theta_{ut}$
$\nabla^{\mathcal{L}}_{\beta}[u, t, R, 0], \nabla^{\mathcal{L}}_{\alpha}[u, s - 1, R, 1], \log \nabla^{\mathcal{L}}_{\theta}[u, t] \leftarrow_{\oplus} \nu \otimes \gamma$
$\gamma \leftarrow \beta[u, t, L, 0] \otimes \alpha[u, s - 1, R, 1] \otimes \theta_{tu}$
$\nabla^{\mathcal{L}}_{\beta}[u, t, L, 0], \nabla^{\mathcal{L}}_{\alpha}[u, s - 1, R, 1], \log \nabla^{\mathcal{L}}_{\theta}[t, u] \leftarrow_{\oplus} \nu \otimes \gamma$
$\nu \leftarrow \nabla^{\mathcal{L}}_{\beta}[s, t, R, 1] \otimes \beta[s, t, R, 1]$
for $u = 1, \dots, s$ **do**
$\nabla^{\mathcal{L}}_{\beta}[u, t, R, 1], \nabla^{\mathcal{L}}_{\alpha}[u, s, R, 0] \leftarrow_{\oplus} \nu \otimes \beta[u, t, R, 1] \otimes \alpha[u, s, R, 0]$
for $u = t + 1, \dots, n$ **do**
$\gamma \leftarrow \beta[s, u, R, 0] \otimes \alpha[t + 1, u, L, 1] \otimes \theta_{su}$
$\nabla^{\mathcal{L}}_{\beta}[s, u, R, 0], \nabla^{\mathcal{L}}_{\alpha}[t + 1, u, L, 1], \log \nabla^{\mathcal{L}}_{\theta}[s, u] \leftarrow_{\oplus} \nu \otimes \gamma$
$\gamma \leftarrow \beta[s, u, L, 0] \otimes \alpha[t + 1, u, L, 1] \otimes \theta_{us}$
$\nabla^{\mathcal{L}}_{\beta}[s, u, L, 0], \nabla^{\mathcal{L}}_{\alpha}[t + 1, u, L, 1], \log \nabla^{\mathcal{L}}_{\theta}[u, s] \leftarrow_{\oplus} \nu \otimes \gamma$
for $k = n, \dots, 1$ **do** ▷ Backpropagate through inside step
for $s = 1, \dots, n - k$ **do**
$t \leftarrow s + k$
$\nu \leftarrow \nabla^{\mathcal{L}}_{\alpha}[s, t, R, 1] \otimes \alpha[s, t, R, 1]$
for $u = s + 1, \dots, t$ **do**
$\nabla^{\mathcal{L}}_{\alpha}[u, t, R, 0], \nabla^{\mathcal{L}}_{\alpha}[u, t, R, 1] \leftarrow_{\oplus} \nu \otimes \alpha[s, u, R, 0] \otimes \alpha[u, t, R, 1]$
if $s > 1$ **then**
$\nu \leftarrow \nabla^{\mathcal{L}}_{\alpha}[s, t, L, 1] \otimes \alpha[s, t, L, 1]$
for $u = s, \dots, t - 1$ **do**
$\nabla^{\mathcal{L}}_{\alpha}[s, u, L, 1], \nabla^{\mathcal{L}}_{\alpha}[u, t, L, 0] \leftarrow_{\oplus} \nu \otimes \alpha[s, u, L, 1] \otimes \alpha[u, t, L, 0]$
$\nu \leftarrow \nabla^{\mathcal{L}}_{\alpha}[s, t, L, 0] \otimes \alpha[s, t, L, 0]$
for $u = s, \dots, t - 1$ **do**
$\gamma \leftarrow \alpha[s, u, R, 1] \otimes \alpha[u + 1, t, L, 1] \otimes \theta_{ts}$
$\nabla^{\mathcal{L}}_{\alpha}[s, u, R, 1], \nabla^{\mathcal{L}}_{\alpha}[u + 1, t, L, 1], \log \nabla^{\mathcal{L}}_{\theta}[t, s] \leftarrow_{\oplus} \nu \otimes \gamma$
$\nu \leftarrow \nabla^{\mathcal{L}}_{\alpha}[s, t, R, 0] \otimes \alpha[s, t, R, 0]$
for $u = s, \dots, t - 1$ **do**
$\gamma \leftarrow \alpha[s, u, R, 1] \otimes \alpha[u + 1, t, L, 1] \otimes \theta_{st}$
$\nabla^{\mathcal{L}}_{\alpha}[s, u, R, 1], \nabla^{\mathcal{L}}_{\alpha}[u + 1, t, L, 1], \log \nabla^{\mathcal{L}}_{\theta}[s, t] \leftarrow_{\oplus} \nu \otimes \gamma$
return signexp $\log \nabla^{\mathcal{L}}_{\theta}$ ▷ Exponentiate log gradient, multiply by sign, and return $\nabla^{\mathcal{L}}_{\theta}$

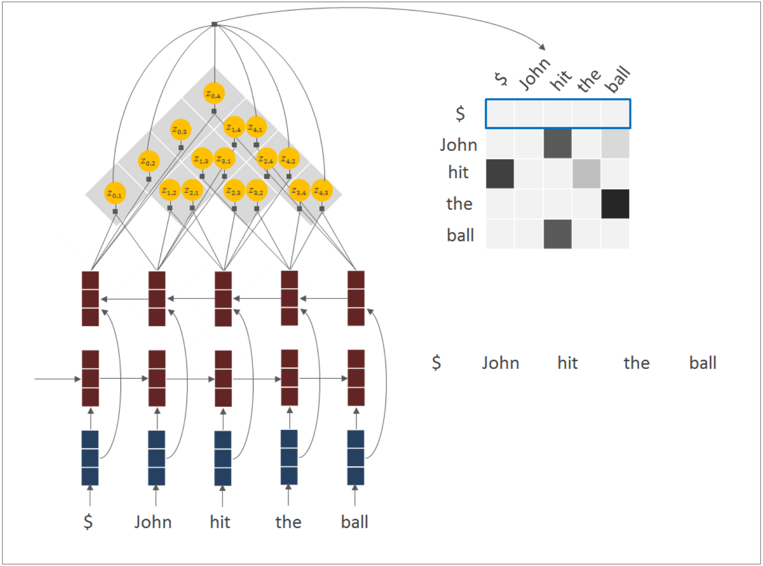
Structured Attention Networks with a Parser ("Syntactic Attention")



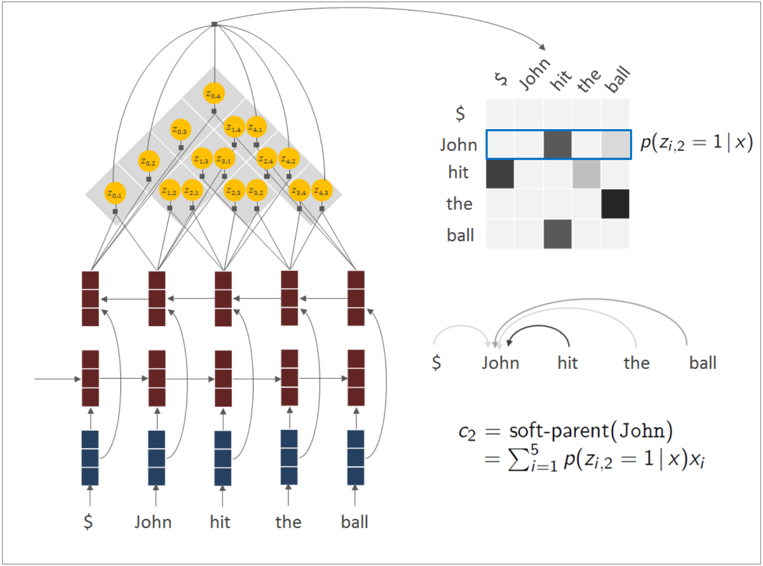
Structured Attention Networks with a Parser ("Syntactic Attention")



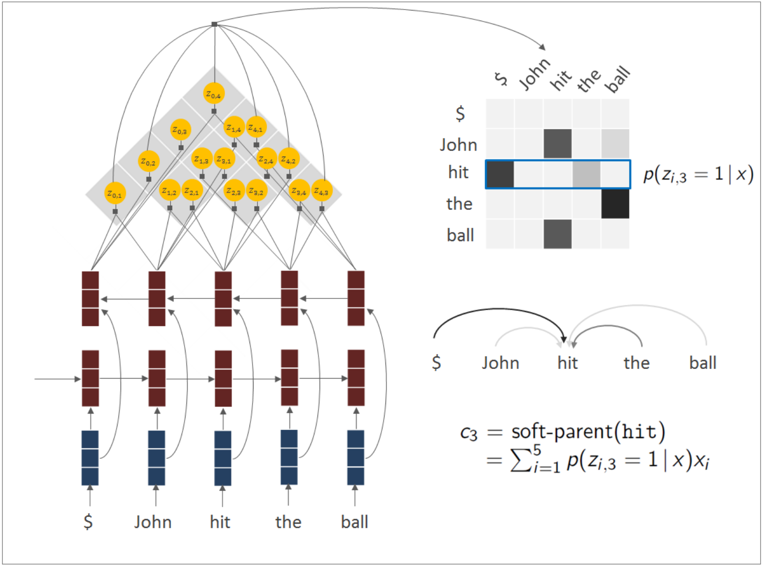
Structured Attention Networks with a Parser ("Syntactic Attention")



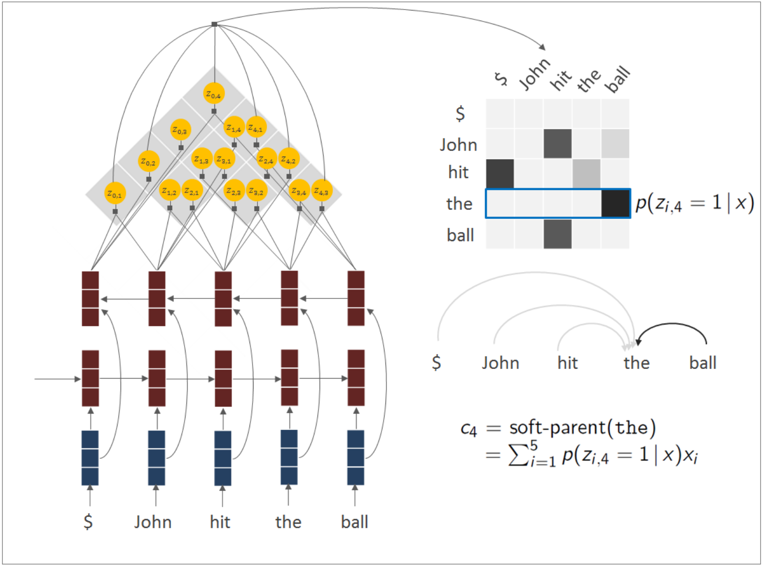
Structured Attention Networks with a Parser ("Syntactic Attention")



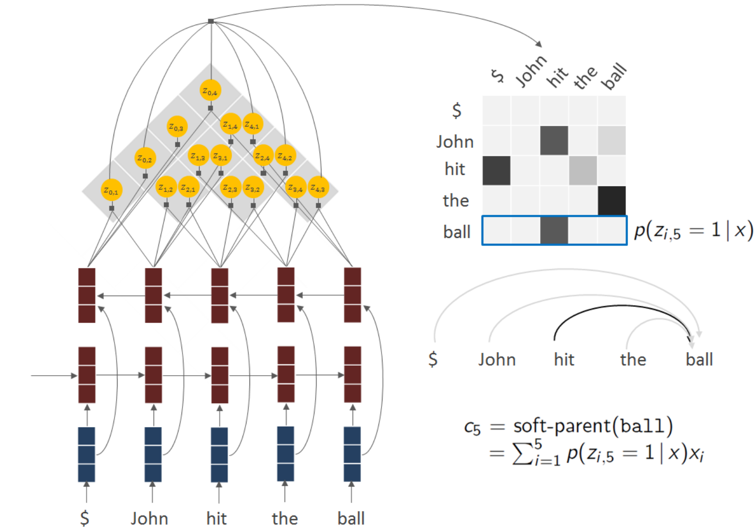
Structured Attention Networks with a Parser ("Syntactic Attention")



Structured Attention Networks with a Parser ("Syntactic Attention")



Structured Attention Networks with a Parser ("Syntactic Attention")



$$c_5 = \text{soft-parent}(\texttt{ball})$$
$$= \sum_{i=1}^{5} p(z_{i,5} = 1 \mid x) x_i$$

Structured Attention Networks for Natural Language Inference

Dataset: Stanford Natural Language Inference (Bowman et al., 2015)

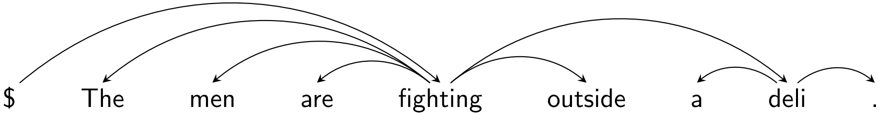
Model	Accuracy %
No Attention	85.8
Hard parent	86.1
Simple Attention	86.2
Structured Attention	86.8

- No attention: word embeddings only
- "Hard" parent from a pipelined dependency parser
- Simple attention (simple softmax instead of syntanctic attention)
- Structured attention (soft parents from syntactic attention)

Structured Attention Networks for Natural Language Inference

Run Viterbi algorithm on the parsing layer to get the MAP parse:

$$\hat{z} = \arg\max_{z} p(z \mid x, q)$$



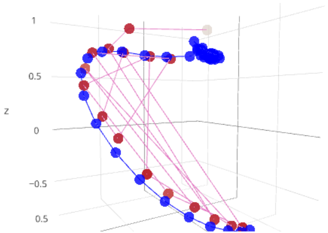
Structured Attention Networks

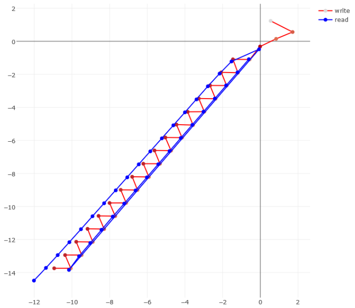
- Generalize attention to incorporate latent structure
- Exact inference through dynamic programming
- Training remains end-to-end.

Future work

- Approximate differentiable inference in neural networks
- Incorporate other probabilistic models into deep learning.
- Compare further to methods using EM or hard structures.

Other Work: Lie-Access Neural Memory (Yang and Rush, 2017)





References I

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

Bowman, S. R., Manning, C. D., and Potts, C. (2015). Tree-Structured Composition in Neural Networks without Tree-Structured Architectures. In *Proceedings of the NIPS workshop on Cognitive Computation: Integrating Neural and Symbolic Approaches*.

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*.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P. (2011). Natural Language Processing (almost) from Scratch. *Journal of Machine Learning Research*, 12:2493–2537.

References II

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

Durrett, G. and Klein, D. (2015). Neural CRF Parsing. In *Proceedings of ACL*.

Eisner, J. M. (1996). Three New Probabilistic Models for Dependency Parsing: An Exploration. In *Proceedings of ACL*.

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

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*.

References III

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*.

Kipperwasser, E. and Goldberg, Y. (2016). Simple and Accurate Dependency Parsing using Bidirectional LSTM Feature Representations. In *TACL*.

Lafferty, J., McCallum, A., and Pereira, F. (2001). Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proceedings of ICML*.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based Learning Applied to Document Recognition. In *Proceedings of IEEE*.

Li, Z. and Eisner, J. (2009). First- and Second-Order Expectation Semirings with Applications to Minimum-Risk Training on Translation Forests. In *Proceedings of EMNLP 2009*.

References IV

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*.

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*.

References V

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*.

Weston, J., Bordes, A., Chopra, S., Rush, A. M., van Merriënboer, B., Joulin, A., and Mikolov, T. (2015). Towards Ai-complete Question Answering: A Set of Prerequisite Toy Tasks. *arXiv preprint arXiv:1502.05698*.

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*.