Attention-Based Models

@Xiaohan

Not All Contexts Are Created Equally
Better Word Representations with Variable Attention

@CMU

Building it up from a simple start.
To improve Word Embedding.

What’s Word Embedding?

- Words are symbolic, hard to incorporate with the numerical machine learning.
- Some institutive methods are proposed.
  - One-Hoc…
  - Disadvantages: Non-informative.
- Representation Learning is involved to condense the word vector.
  - Word2Vec, Glove…
  - Disadvantages:
    - Treating all the contexts equally to involve much noise.
    - Example: We won the *game*, *Nicely* played.

Mathematically

- CBOW

$$p(\mathbf{v}_0 \mid \mathbf{w}_{[-b,b]-\{0\}}) = \frac{\exp \mathbf{v}_0^\top \mathbf{O}\mathbf{c}}{\sum_{\mathbf{v}\in V} \exp \mathbf{v}^\top \mathbf{O}\mathbf{c}}$$

$$\mathbf{c} = \frac{1}{2b} \sum_{i\in[-b,b]-\{0\}} \mathbf{w}_i$$

Treating Equally

- CBOW with Attention

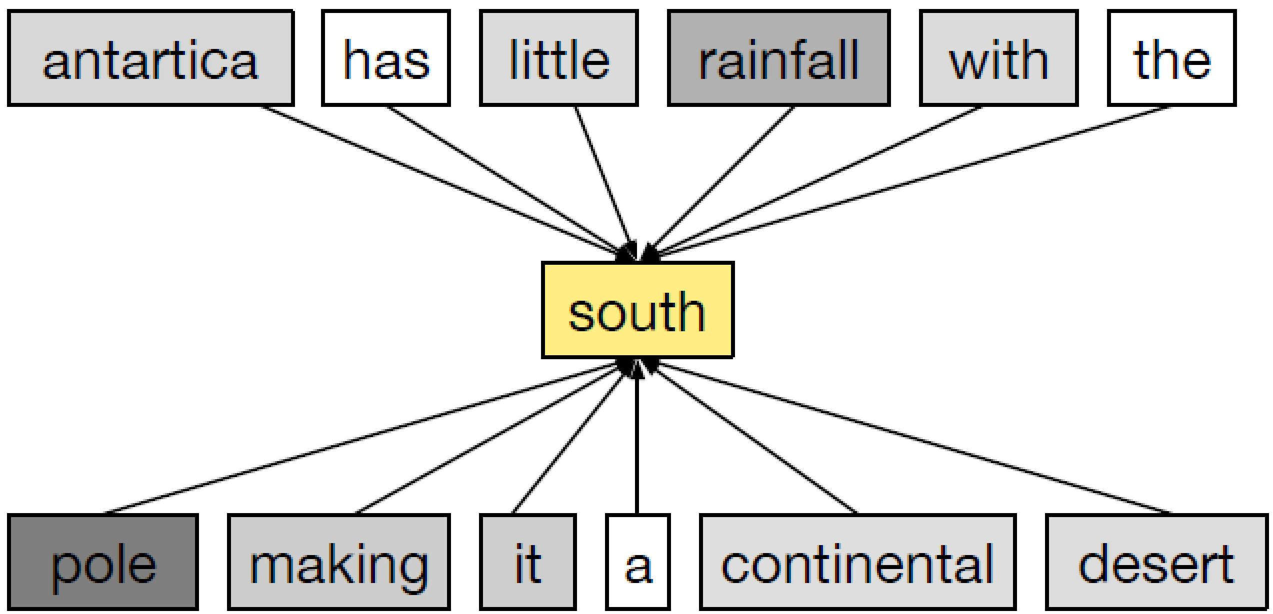
$$\mathbf{c} = \sum_{i\in[-b,b]-\{0\}} a_i(w_i)\mathbf{w}_i$$

Weighting is the core idea of Attention.

Weights are defined varying with applications.

$$a_i(w) = \frac{\exp k_{w,i} + s_i}{\sum_{j\in[-b,b]-\{0\}} \exp k_{w,j} + s_j}$$

Case Study



antartica
has
little
rainfall
with
the
south
pole
making
it
a
continental
desert

Experiments

	POS Induction	POS Tagging	Sentiment Analysis
CBOW	50.40	97.03	71.99
Skip-ngram	33.86	97.19	**72.10**
SSkip-ngram	47.64	**97.40**	69.96
CBOW Attention	**54.00**	97.39	71.39

- Attention aims at concerning the instinct structure. Here is the dependency syntax.
- Maybe, semantic analysis could be improved by future work.
  - Ship-Gram with Attention.

What's Attention-Based Models?

- Attention-based models are a mechanism to address the important components and avoid the noise.
- Usually, it could be represented as below, stated as **Attention Equation**.

$$\overrightarrow{c_i} = \boldsymbol{A}\,\overrightarrow{h_j}$$

- $\boldsymbol{A}$ is the **Alignment Matrix**, which is generated by empirical consideration.
- Noted as **ABM**.

What’s the advantages?

- Long-term sequence analysis.
    - Expanding the sibling window size or Enhancing with LSTM or GRU…
    - Long-term makes more noise, degrading the performance.
    - ABM is just born for this issue. **By learning to weight the importance and ignore the irrelevance, long-term analysis is enhanced.**
- **Three Key Points**
    - **Why Attention?**
    - **Where to put Attention?**
    - **How to weight Attention?**

Applications

Understanding Attention-Based Models by Applications.

- Machine Translation.
- Speech Recognition.
- Image Caption Generation.
- Abstractive Summarization.
- Reasoning about Entailment.
- Language Models.

Neural Machine Translation By Jointly Learning to Align and Translate

@Jacobs.University

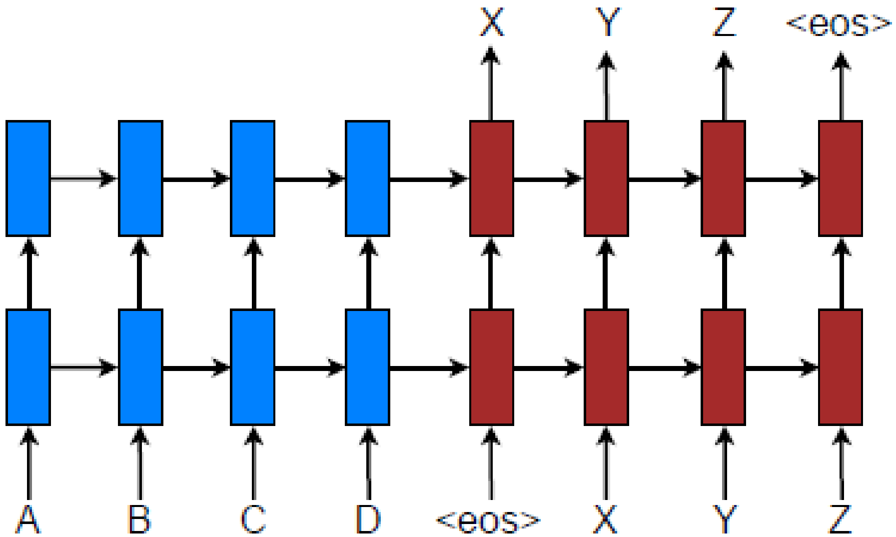
Effective Approaches to Attention-Based Neural Machine Translation

@Standford

Machine Translation

- Traditional Methods
  - Phrase-based models with rules, syntax and probabilistic modelling.
  - Main Concerns.
    - Alignment.
    - Generation.
    - …
- Neural Machine Translation
  - Automatically learning to align and translate.
  - Encoder-Decoder model.
    - Encode the source into a fixed-length vector.
    - Decode the target from the fixed-length vector.
  - Disadvantages: Very bad performance for long sentences.

General RNN Encoder-Decoder



Source Encoding

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

$$c = q(\{h_1, \cdots, h_{T_x}\})$$

Target Decoding

$$s_i = f(s_{i-1}, y_{i-1}, c_i).$$

$$p(y_t \mid \{y_1, \cdots, y_{t-1}\}, c) = g(y_{t-1}, s_t, c),$$

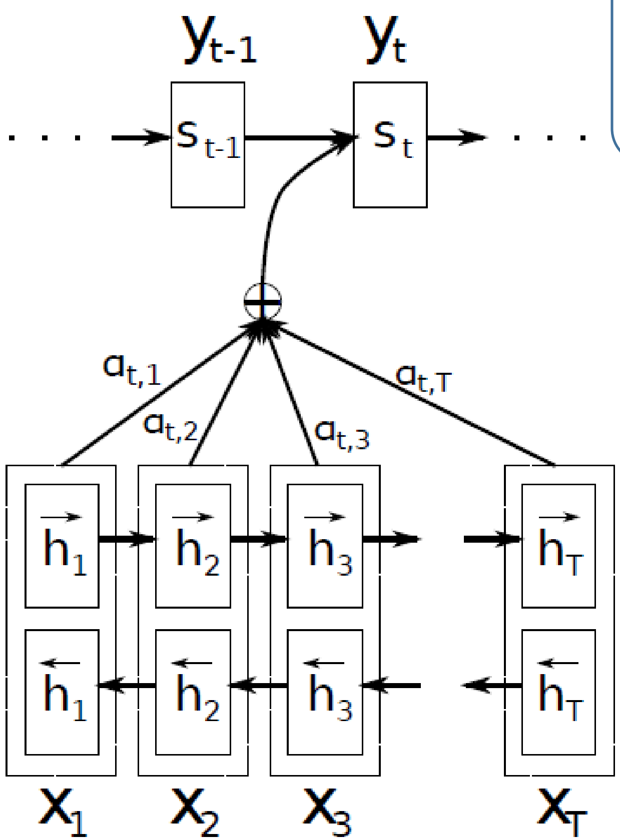
$$p(\mathbf{y}) = \prod_{t=1}^{T} p(y_t \mid \{y_1, \cdots, y_{t-1}\}, c),$$

Objective

Why Attention?
Where to put
our 'Attention'?

Attention-Based Enhancement

- Mathematically.



$$s_i = f(s_{i-1}, y_{i-1}, c_i).$$

$$p(y_t \mid \{y_1, \cdots, y_{t-1}\}, c) = g(y_{t-1}, s_t, c),$$

Target Decoding

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

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

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

Attention-Based

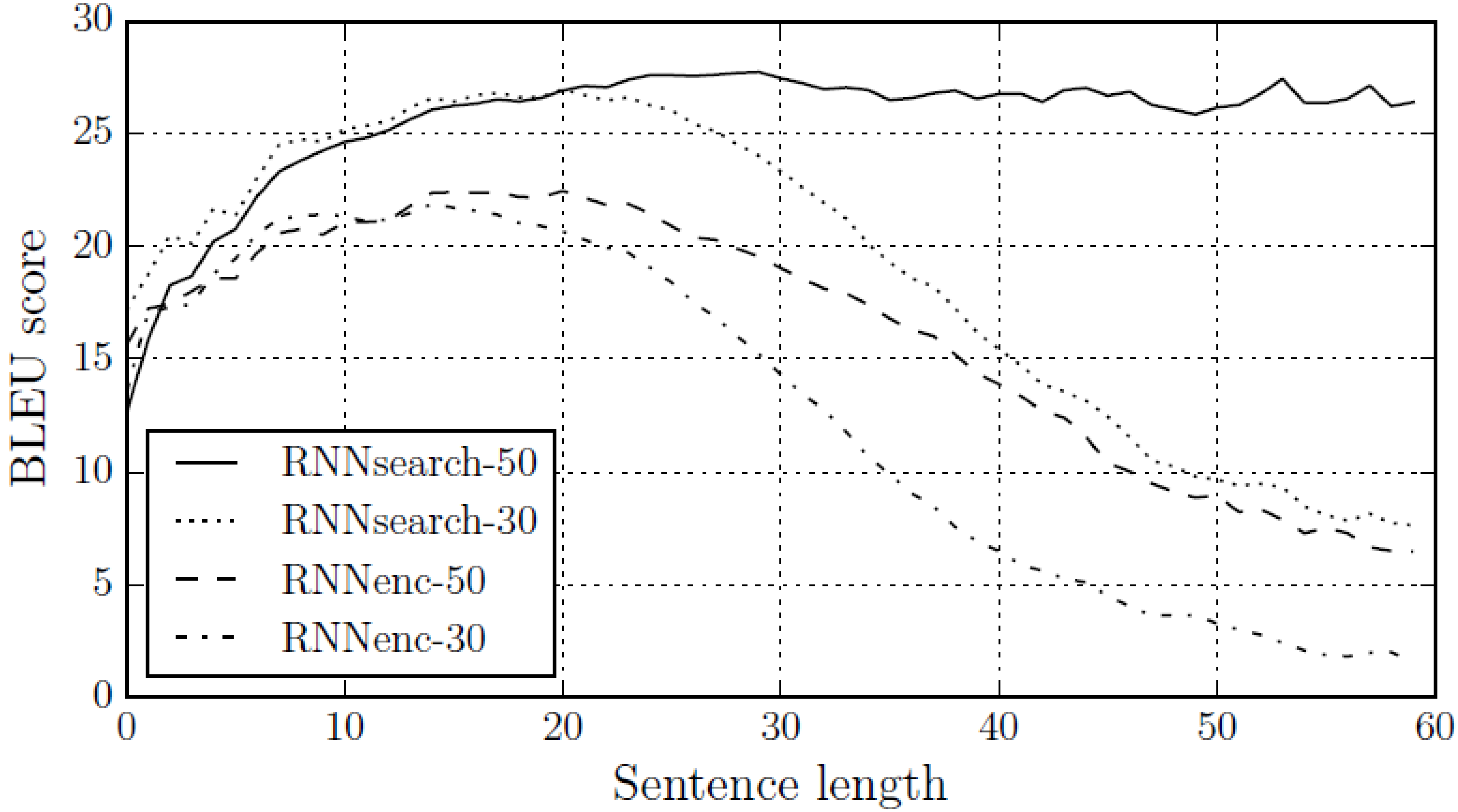
Probabilistic Perspective

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

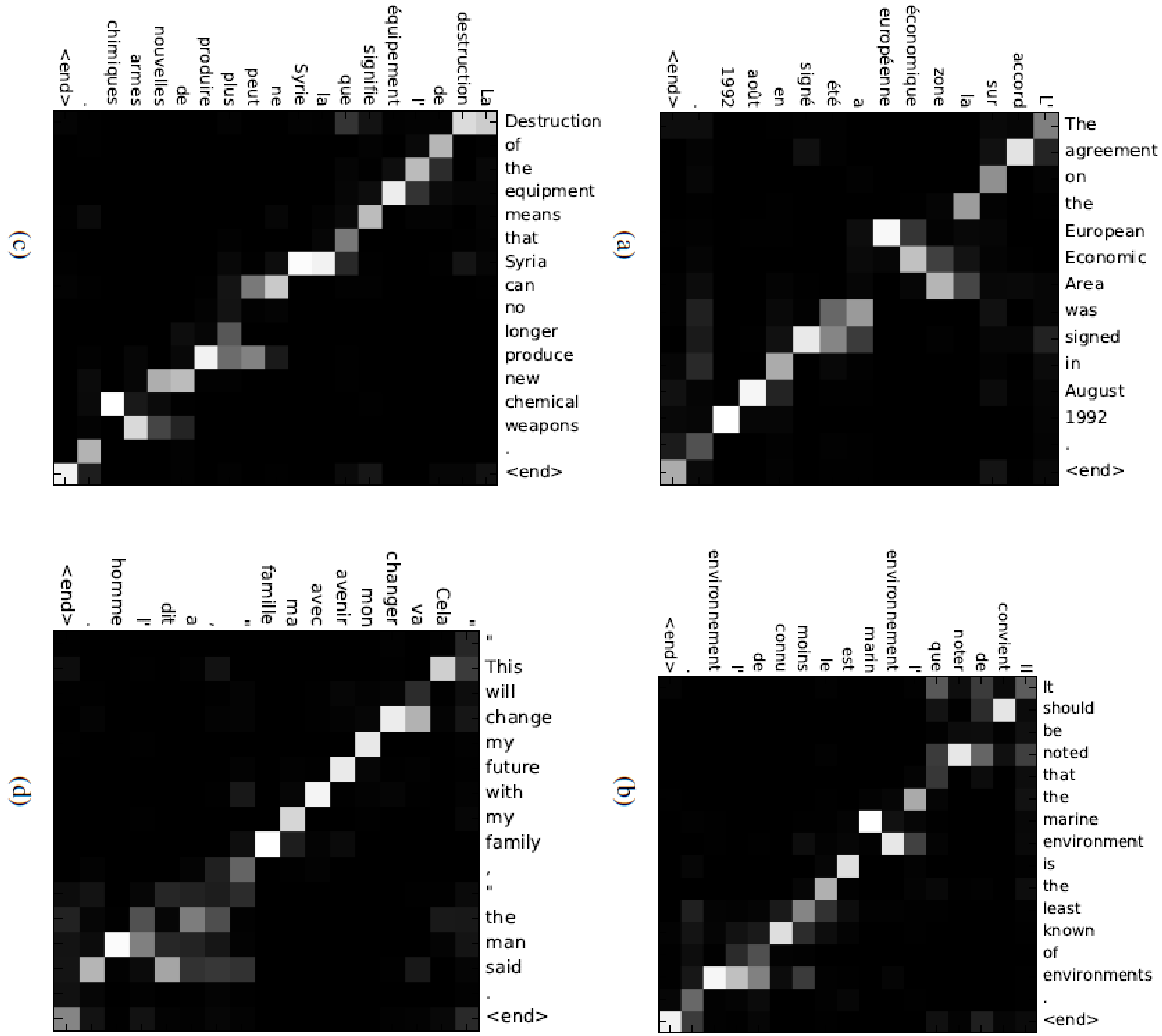
$$c = q(\{h_1, \cdots, h_{T_x}\})$$

Source Encoding

Experiments



Experiments



Are there more powerful weights?

- General Forms:

$$a_t(s) = \text{align}(h_t, \bar{h}_s) = \frac{\exp\left(\text{score}(h_t, \bar{h}_s)\right)}{\sum_{s'} \exp\left(\text{score}(h_t, \bar{h}_{s'})\right)}$$

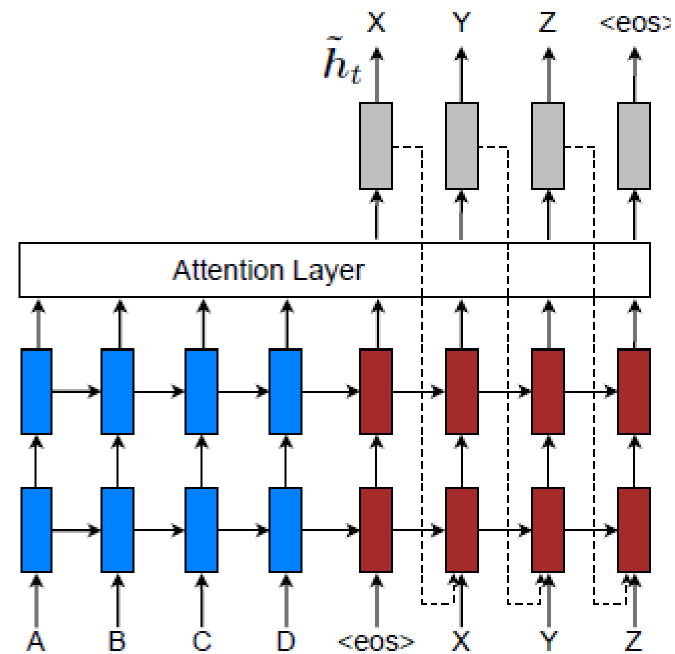
$$\text{score}(h_t, \bar{h}_s) = \begin{cases} h_t^\top \bar{h}_s & \textit{dot} \\ h_t^\top W_a \bar{h}_s & \textit{general} \\ v_a^\top \tanh\left(W_a[h_t; \bar{h}_s]\right) & \textit{concat} \end{cases}$$

- To consider the position prior:

$$p_t = S \cdot \text{sigmoid}(v_p^\top \tanh(W_p h_t)),$$

$$a_t(s) = \text{align}(h_t, \bar{h}_s) \exp\left(-\frac{(s - p_t)^2}{2\sigma^2}\right)$$

- To consider the input-feeding:



Experiments

System	Ppl	BLEU
Winning WMT'14 system – *phrase-based + large LM* (Buck et al., 2014)		20.7
Existing NMT systems		
RNNsearch (Jean et al., 2015)		16.5
RNNsearch + unk replace (Jean et al., 2015)		19.0
RNNsearch + unk replace + large vocab + *ensemble* 8 models (Jean et al., 2015)		**21.6**
Our NMT systems		
Base	10.6	11.3
Base + reverse	9.9	12.6 (+*1.3*)
Base + reverse + dropout	8.1	14.0 (+*1.4*)
Base + reverse + dropout + global attention (*location*)	7.3	16.8 (+*2.8*)
Base + reverse + dropout + global attention (*location*) + feed input	6.4	18.1 (+*1.3*)
Base + reverse + dropout + local-p attention (*general*) + feed input	5.9	19.0 (+*0.9*)
Base + reverse + dropout + local-p attention (*general*) + feed input + unk replace		20.9 (+*1.9*)
Ensemble 8 models + unk replace		**23.0** (+*2.1*)

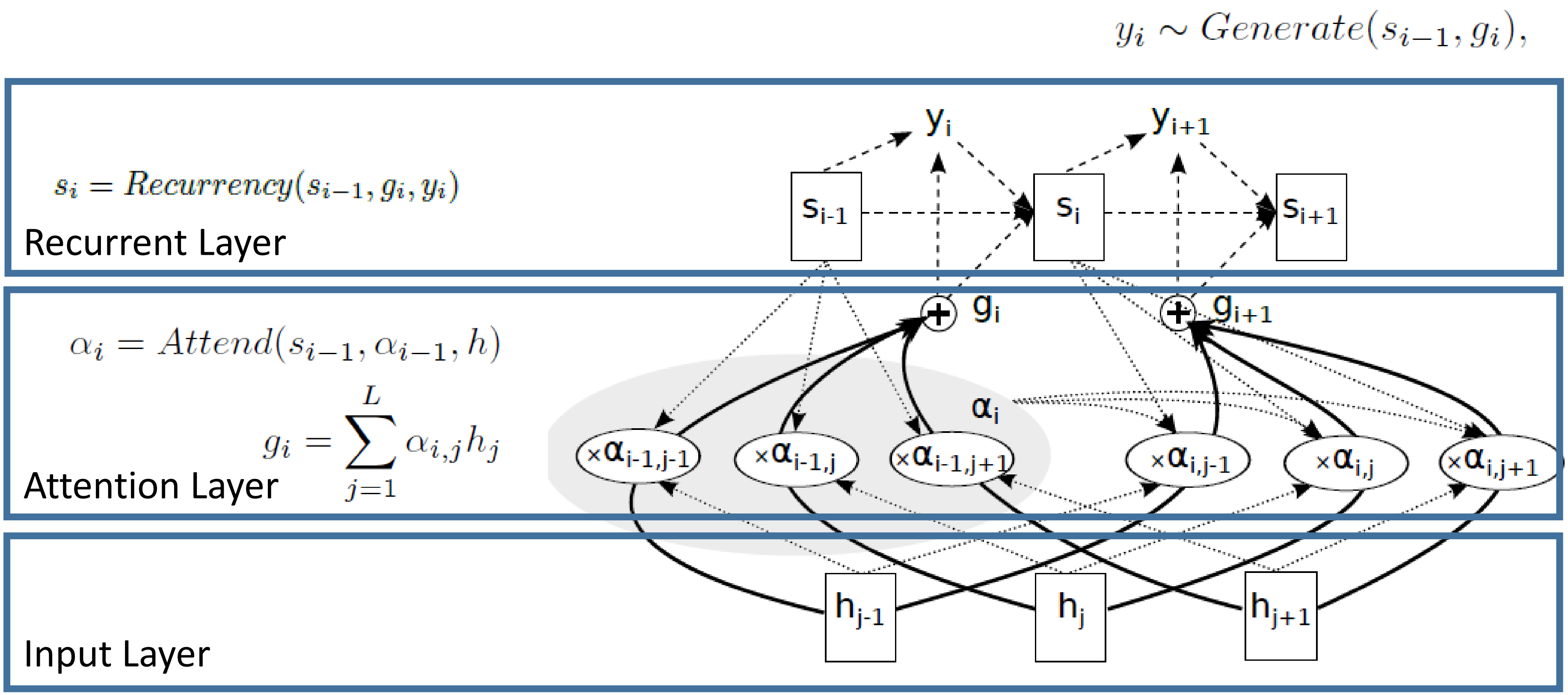
Conclusions For Three Key Points

- Why Attention?
  - Long-term de-noising.
- Where to put Attention?
  - To add an attention layer between source encoding and target decoding.
- How to weight Attention?
  - To consider Position Prior.
  - To consider Input-Feeding.

Attention-Based Models for Speech Recognition

@Universite de Montreal

Attention-Based Recurrent Model for Speech Generation (ARSG)



Weighting Attention?

$$\alpha_{i,j} = \exp(e_{i,j}) \Bigg/ \sum_{j=1}^{L} \exp(e_{i,j})$$

MLP for the weights

$$e_{i,j} = w^\top \tanh(W s_{i-1} + V h_j + U f_{i,j} + b)$$

$$f_i = F * \alpha_{i-1}.$$

Two Tricks:

- Sharpening:

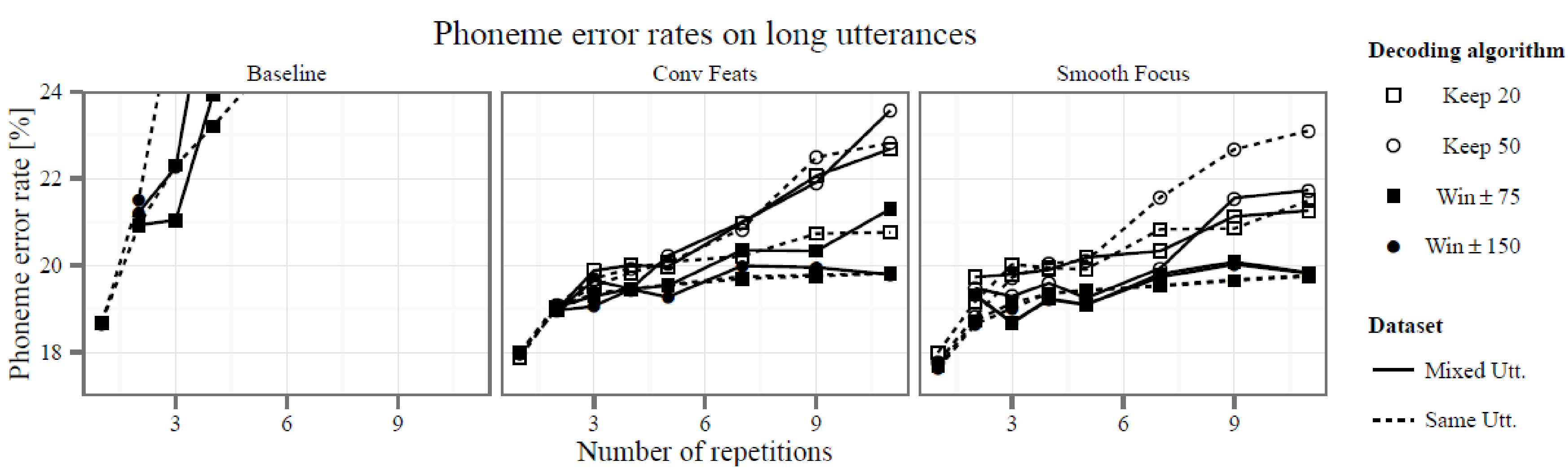
$$a_{i,j} = \exp(\beta e_{i,j}) \Bigg/ \sum_{j=1}^{L} \exp(\beta e_{i,j}) \ ,$$

- Smoothing:

$$a_{i,j} = \sigma(e_{i,j}) \Bigg/ \sum_{j=1}^{L} \sigma(e_{i,j}) \ .$$

Experiments

Model	Dev	Test
Baseline Model	15.9%	18.7%
Baseline + Conv. Features	16.1%	18.0%
Baseline + Conv. Features + Smooth Focus	15.8%	**17.6%**
RNN Transducer [16]	N/A	17.7%
HMM over Time and Frequency Convolutional Net [25]	13.9%	16.7%



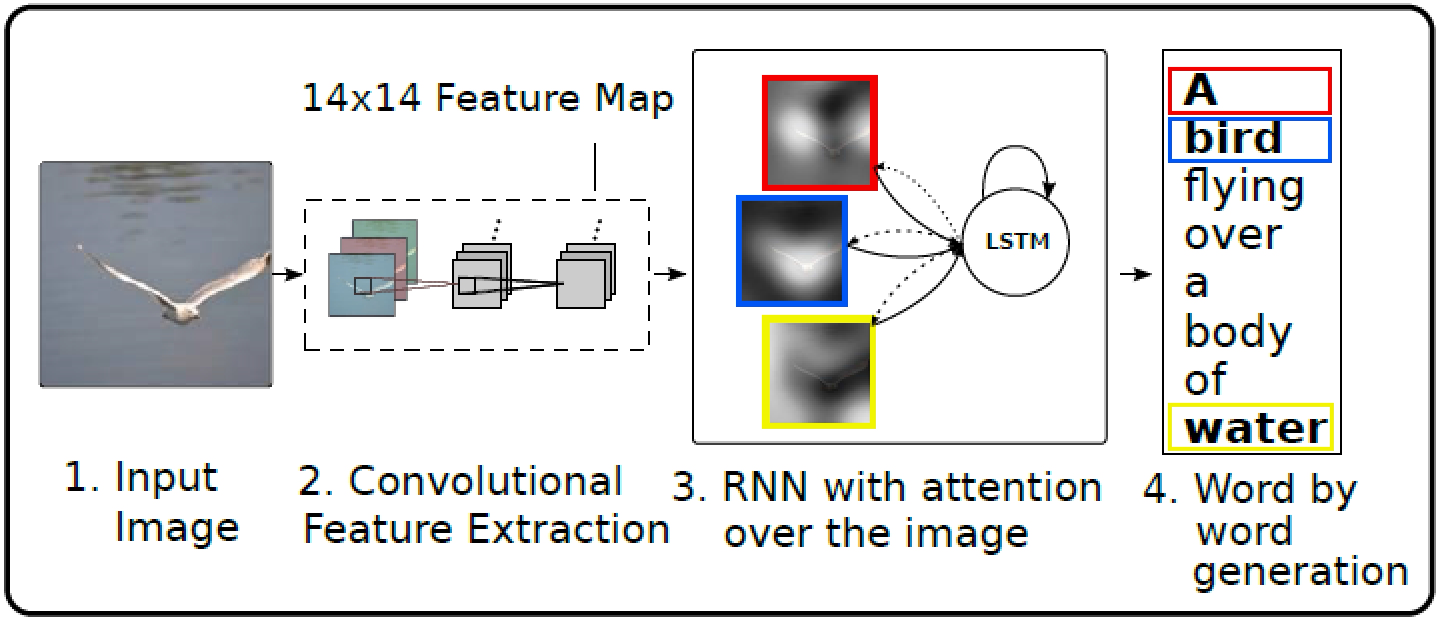
Conclusions For Three Key Points

- Why Attention?
  - Long-term de-noising.
- Where to put Attention?
  - To add an attention layer between input layer and hidden state recurrent layer.
- How to weight Attention?
  - To consider an MLP
  - To consider the smoothing trick.

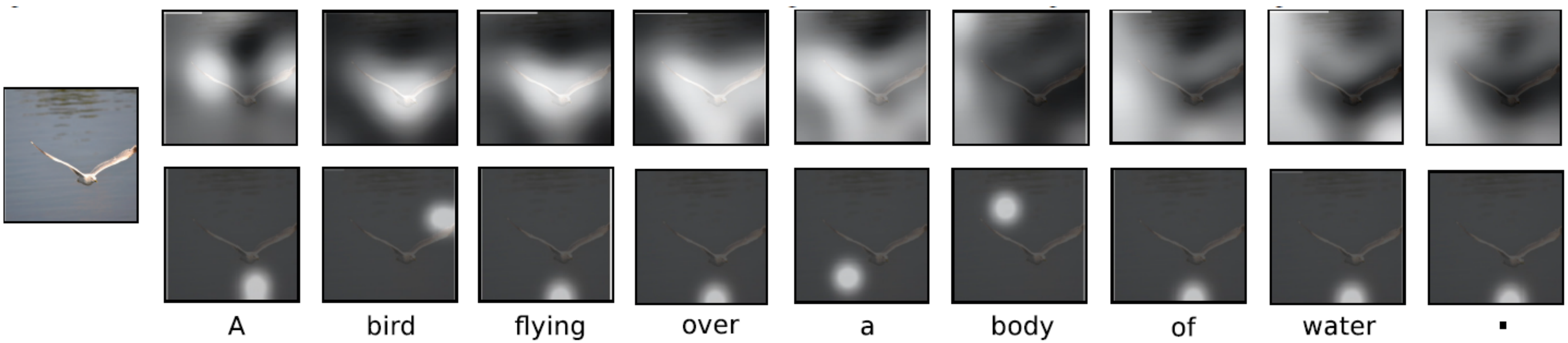
Show, Attend and Tell: Neural Image Caption Generation with Visual Attention

@Toronto University

Architecture



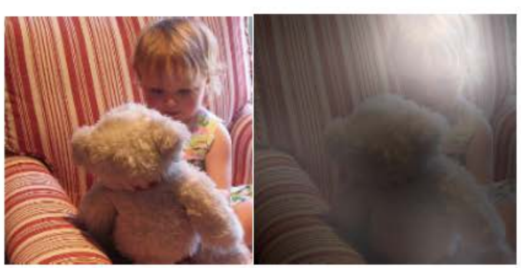
Visualizing the process



A woman is throwing a frisbee in a park.

A dog is standing on a hardwood floor.

A stop sign is on a road with a mountain in the background.



A little girl sitting on a bed with a teddy bear.

A group of people sitting on a boat in the water.

A giraffe standing in a forest with trees in the background.

LSTM with Attention

- Noted we have already extracted a vector sequence.

LSTM

$$\begin{pmatrix} \mathbf{i}_t \\ \mathbf{f}_t \\ \mathbf{o}_t \\ \mathbf{g}_t \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} T_{D+m+n,n} \begin{pmatrix} \mathbf{E}\mathbf{y}_{t-1} \\ \mathbf{h}_{t-1} \\ \hat{\mathbf{z}}_t \end{pmatrix}$$

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \mathbf{g}_t$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t).$$

$$e_{ti} = f_{\text{att}}(\mathbf{a}_i, \mathbf{h}_{t-1})$$

$$\alpha_{ti} = \frac{\exp(e_{ti})}{\sum_{k=1}^{L} \exp(e_{tk})}.$$

$$\hat{\mathbf{z}}_t = \phi\left(\{\mathbf{a}_i\}, \{\alpha_i\}\right),$$

Attention to select input.

Experiments

		BLEU				
Dataset	Model	B-1	B-2	B-3	B-4	METEOR
Flickr8k	Google NIC(Vinyals et al., 2014)$^{\dagger\Sigma}$	63	41	27	—	—
	Log Bilinear (Kiros et al., 2014a)$^{\circ}$	65.6	42.4	27.7	17.7	17.31
	Soft-Attention	**67**	44.8	29.9	19.5	18.93
	Hard-Attention	**67**	**45.7**	**31.4**	**21.3**	**20.30**
Flickr30k	Google NIC$^{\dagger\circ\Sigma}$	66.3	42.3	27.7	18.3	—
	Log Bilinear	60.0	38	25.4	17.1	16.88
	Soft-Attention	66.7	43.4	28.8	19.1	**18.49**
	Hard-Attention	**66.9**	**43.9**	**29.6**	**19.9**	18.46
COCO	CMU/MS Research (Chen & Zitnick, 2014)^{a}	—	—	—	—	20.41
	MS Research (Fang et al., 2014)$^{\dagger a}$	—	—	—	—	20.71
	BRNN (Karpathy & Li, 2014)$^{\circ}$	64.2	45.1	30.4	20.3	—
	Google NIC$^{\dagger\circ\Sigma}$	66.6	46.1	32.9	24.6	—
	Log Bilinear$^{\circ}$	70.8	48.9	34.4	24.3	20.03
	Soft-Attention	70.7	49.2	34.4	24.3	**23.90**
	Hard-Attention	**71.8**	**50.4**	**35.7**	**25.0**	23.04

Conclusions For Three Key Points

- Why Attention?
  - To select the supposed input.
- Where to put Attention?
  - To add an attention layer between input layer and LSTM layer.
- How to weight Attention?
  - Nothing interesting.

A Neural Attention Model for Abstractive Sentence Summarization

@Facebook AI Research

Neural Language Model Based

- Totally, we would like an MAP estimator.

$$\log p(\mathbf{y}|\mathbf{x};\theta) \approx \sum_{i=0}^{N-1} \log p(\mathbf{y}_{i+1}|\mathbf{x},\mathbf{y}_{\mathrm{c}};\theta),$$

- The terms are based on a language model that is the neural language model.

$$\begin{aligned} p(\mathbf{y}_{i+1}|\mathbf{y}_{\mathrm{c}},\mathbf{x};\theta) &\propto \exp(\mathbf{V}\mathbf{h} + \mathbf{W}\mathrm{enc}(\mathbf{x},\mathbf{y}_{\mathrm{c}})), \\ \tilde{\mathbf{y}}_{\mathrm{c}} &= [\mathbf{E}\mathbf{y}_{i-C+1},\ldots,\mathbf{E}\mathbf{y}_i], \\ \mathbf{h} &= \tanh(\mathbf{U}\tilde{\mathbf{y}}_{\mathrm{c}}). \end{aligned}$$

- What we focus is the encoder.

Encoders

- Bag-of-Word Encoder:

$$
\begin{aligned}
\mathrm{enc}_1(\mathbf{x}, \mathbf{y}_{\mathrm{c}}) &= \mathbf{p}^{\top}\tilde{\mathbf{x}}, \\
\mathbf{p} &= [1/M, \ldots, 1/M], \\
\tilde{\mathbf{x}} &= [\mathbf{F}\mathbf{x}_1, \ldots, \mathbf{F}\mathbf{x}_M].
\end{aligned}
$$

Treating each word equally

- Attention-Based Encoder:

Familiar Attention Trick

$$
\begin{aligned}
\mathrm{enc}_3(\mathbf{x}, \mathbf{y}_{\mathrm{c}}) &= \mathbf{p}^{\top}\bar{\mathbf{x}}, \\
\mathbf{p} &\propto \exp(\tilde{\mathbf{x}}\mathbf{P}\tilde{\mathbf{y}}'_{\mathrm{c}}), \\
\tilde{\mathbf{x}} &= [\mathbf{F}\mathbf{x}_1, \ldots, \mathbf{F}\mathbf{x}_M], \\
\tilde{\mathbf{y}}'_{\mathrm{c}} &= [\mathbf{G}\mathbf{y}_{i-C+1}, \ldots, \mathbf{G}\mathbf{y}_i], \\
\forall i \quad \bar{\mathbf{x}}_i &= \sum_{q=i-Q}^{i+Q} \tilde{\mathbf{x}}_i / Q.
\end{aligned}
$$

Training

- It’s a direct HMM model with multiple time-delay circles.
  - Viterbi Decoding.
  - Beam-Search Viterbi Decoding.

$$\log p(\mathbf{y}|\mathbf{x};\theta) \approx \sum_{i=0}^{N-1} \log p(\mathbf{y}_{i+1}|\mathbf{x},\mathbf{y}_{\mathrm{c}};\theta),$$

Experiments

Model	DUC-2004 ROUGE-1	ROUGE-2	ROUGE-L	Gigaword ROUGE-1	ROUGE-2	ROUGE-L	Ext. %
IR	11.06	1.67	9.67	16.91	5.55	15.58	29.2
PREFIX	22.43	6.49	19.65	23.14	8.25	21.73	100
COMPRESS	19.77	4.02	17.30	19.63	5.13	18.28	100
W&L	22	6	17	-	-	-	-
TOPIARY	25.12	6.46	20.12	-	-	-	-
MOSES+	26.50	8.13	22.85	28.77	12.10	26.44	70.5
ABS	26.55	7.06	22.05	30.88	12.22	27.77	85.4
ABS+	28.18	8.49	23.81	31.00	12.65	28.34	91.5
REFERENCE	29.21	8.38	24.46	-	-	-	45.6

Conclusions For Three Key Points

- Why Attention?
  - To focus on important parts.
- Where to put Attention?
  - To add an attention layer for word weighting.
- How to weight Attention?
  - Nothing interesting.

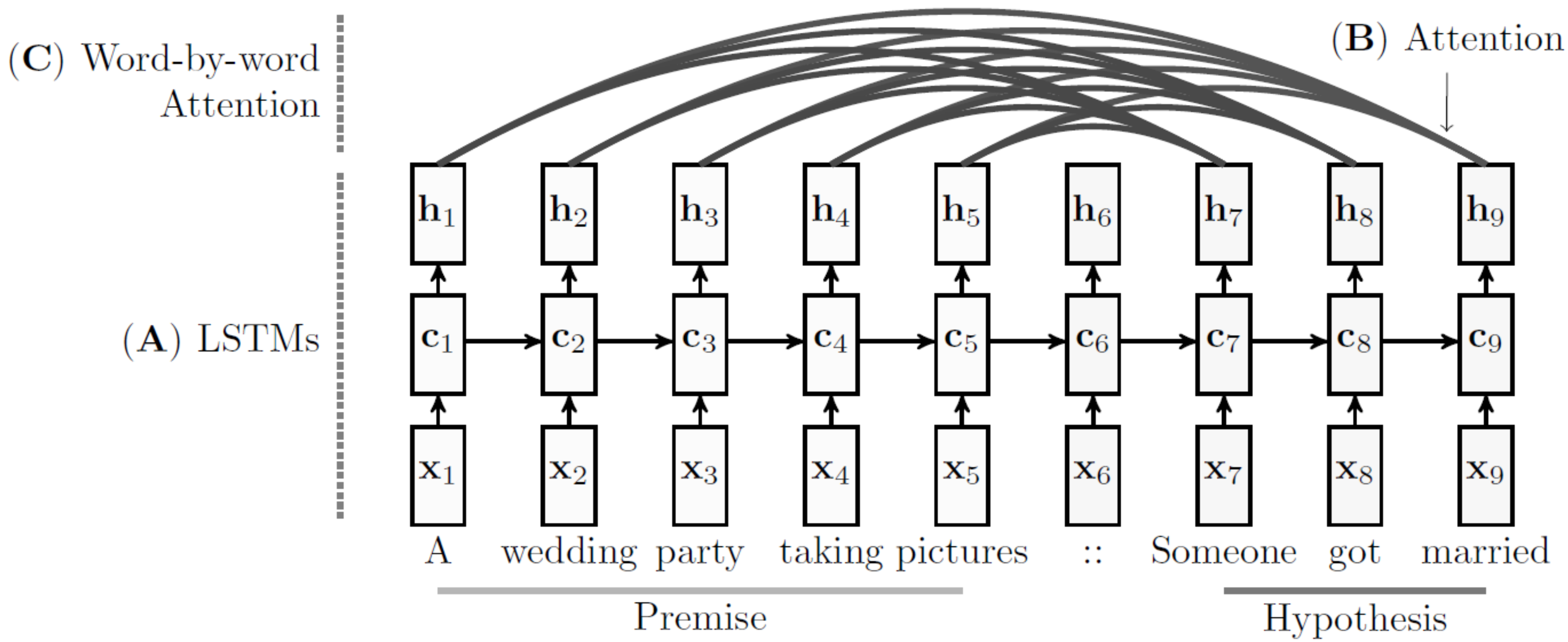
Reasoning about Entailment with Neural Attention

@Google Deep Mind

Problem

- Automatically recognizing entailment relations between pairs of natural language sentences.
  - A **wedding** party taking pictures
  - Someone got **married**.

Architectures



$$\mathbf{M}_t = \tanh(\mathbf{W}^y\mathbf{Y} + (\mathbf{W}^h\mathbf{h}_t + \mathbf{W}^r\mathbf{r}_{t-1}) \otimes \mathbf{e}_L)$$

$$\alpha_t = \text{softmax}(\mathbf{w}^T\mathbf{M}_t)$$

$$\mathbf{r}_t = \mathbf{Y}\alpha_t^T + \tanh(\mathbf{W}^t\mathbf{r}_{t-1})$$

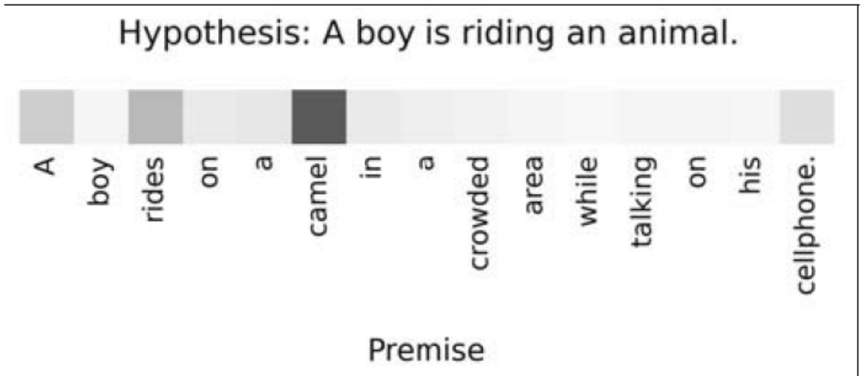
For Classification

$$\mathbf{h}^* = \tanh(\mathbf{W}^p\mathbf{r}_L + \mathbf{W}^x\mathbf{h}_N)$$

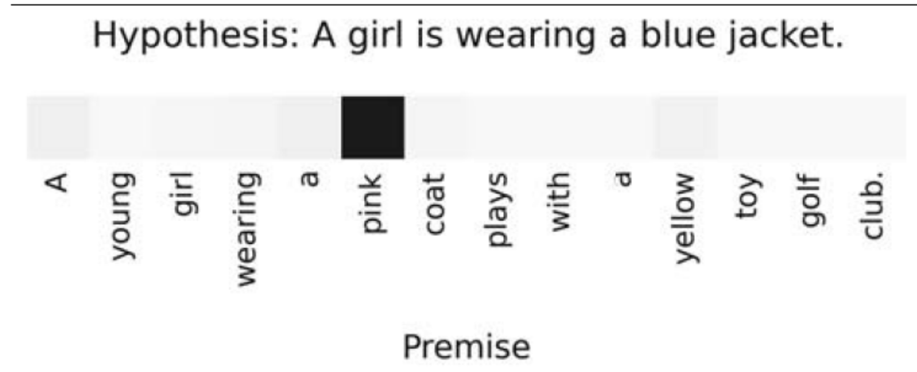
Experiments

Model	k	$\|\theta\|_{\text{W+M}}$	$\|\theta\|_{\text{M}}$	Train	Dev	Test
LSTM [Bowman et al., 2015]	100	$\approx$ 10M	221k	84.4	-	77.6
Classifier [Bowman et al., 2015]	-	-	-	99.7	-	78.2
LSTM shared	100	3.8M	111k	83.7	81.9	80.9
LSTM shared	159	3.9M	252k	84.4	83.0	81.4
LSTMs	116	3.9M	252k	83.5	82.1	80.9
Attention	100	3.9M	242k	85.4	83.2	82.3
Attention two-way	100	3.9M	242k	86.5	83.0	82.4
Word-by-word attention	100	3.9M	252k	85.3	**83.7**	**83.5**
Word-by-word attention two-way	100	3.9M	252k	86.6	83.6	83.2

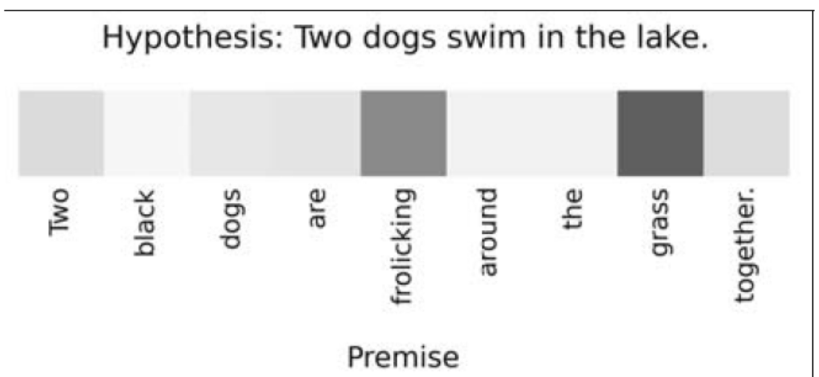
Experiments



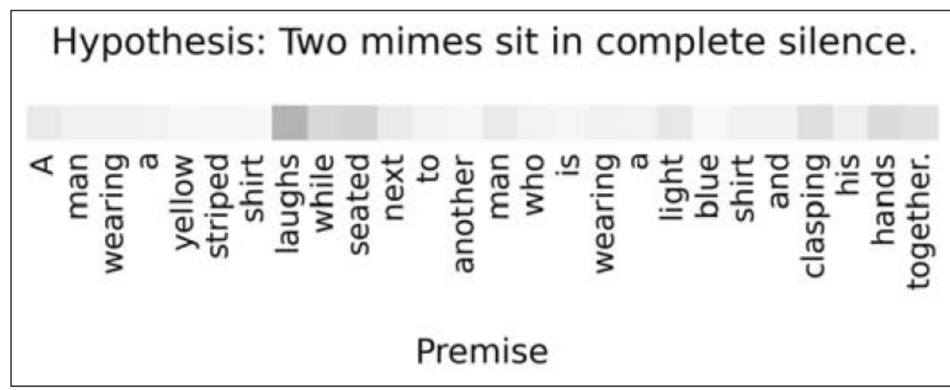
(a)



(b)



(c)



(d)

Conclusions For Three Key Points

- Why Attention?
  - To select important words.
- Where to put Attention?
  - Along with all the LSTMs.
- How to weight Attention?
  - Nothing interesting.

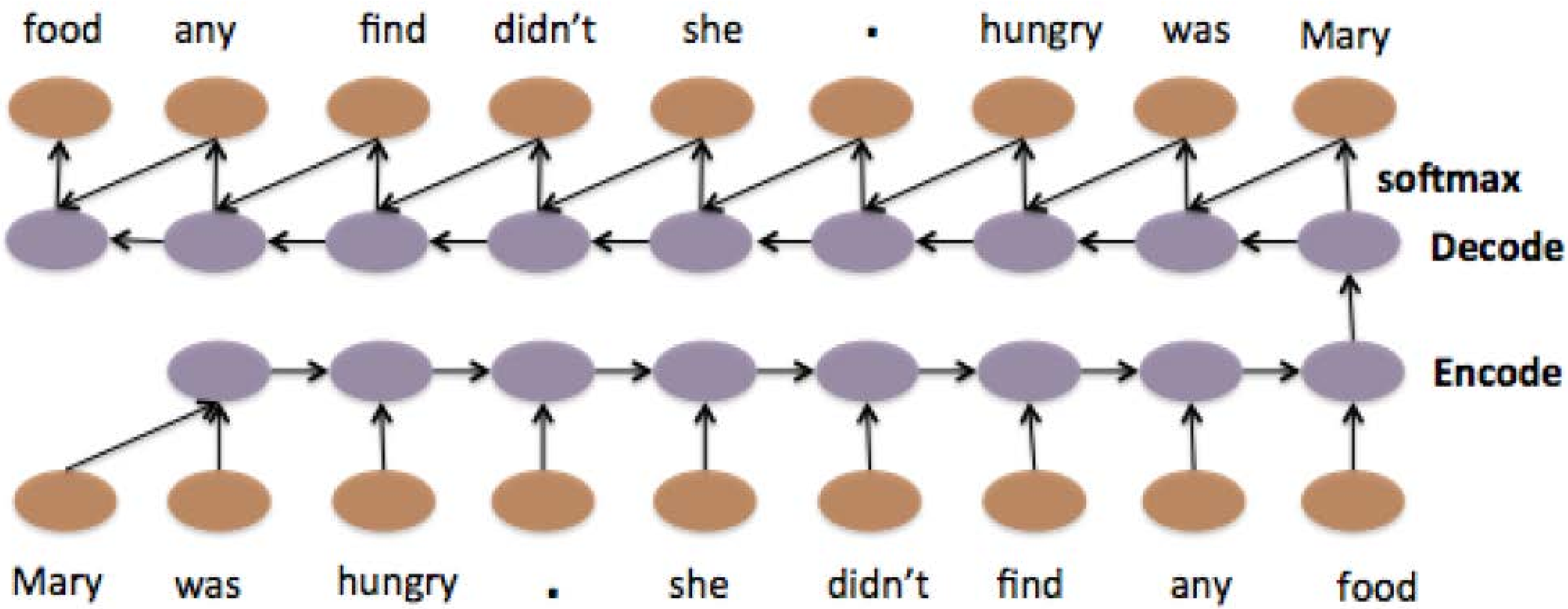
A Hierarchical Neural Auto-Encoder for Paragraphs and Documents

@Standford

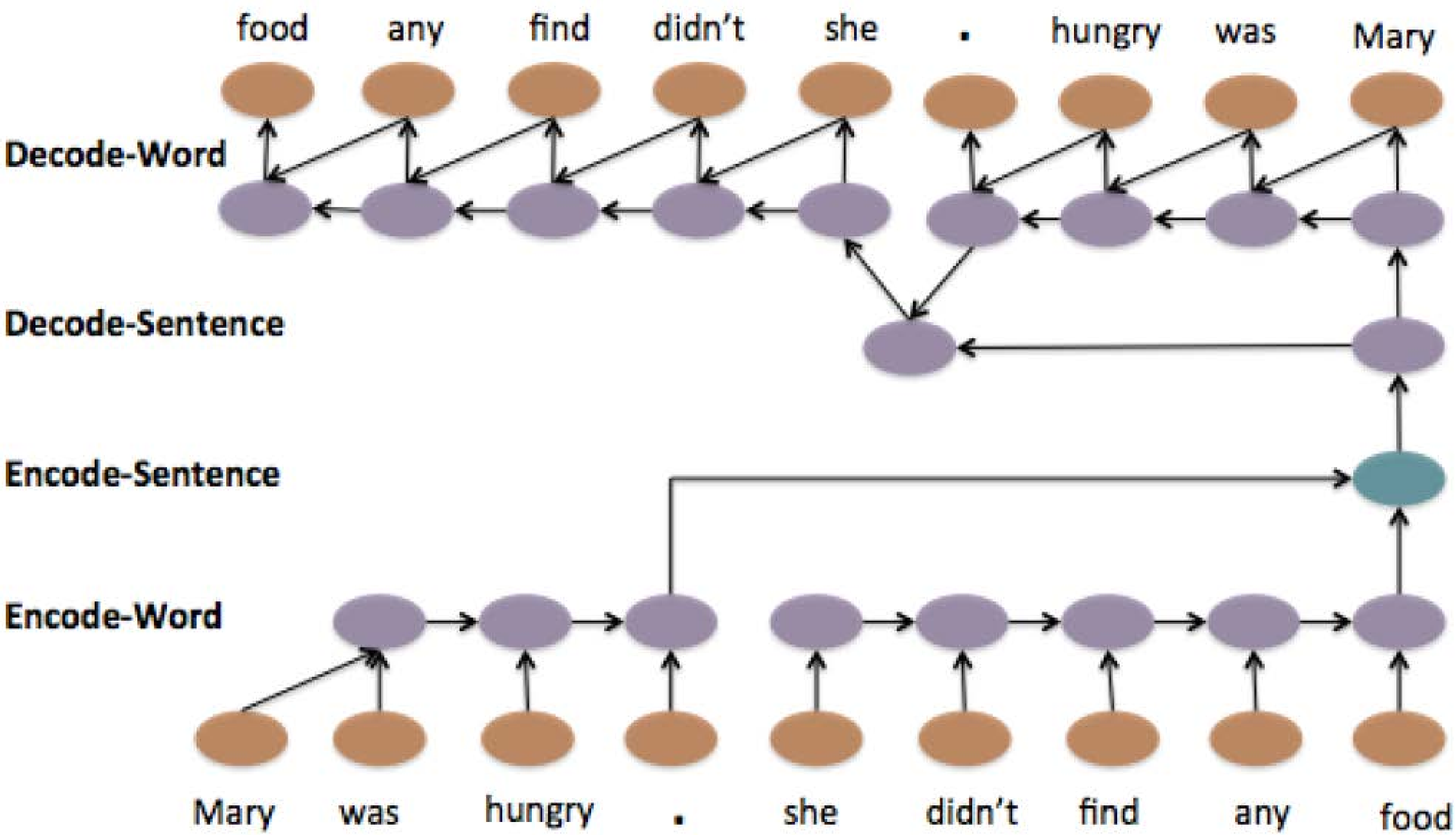
Encoding Source Language with Convolutional Neural Network for Machine Translation

@Chinese Academy of Sciences

Paragraph Auto-Encoder
Standard LSTM



Paragraph Auto-Encoder Hierarchical LSTM



LSTM with Attention

$$\begin{bmatrix} i_t \\ f_t \\ o_t \\ l_t \end{bmatrix} = \begin{bmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{bmatrix} W \cdot \begin{bmatrix} h^s_{t-1}(\text{dec}) \\ e^s_t \\ m_t \end{bmatrix}$$

Attention Term

$$c_t = f_t \cdot c_{t-1} + i_t \cdot l_t$$

$$h^s_t = o_t \cdot c_t$$

$$m_t = \sum_{i \in [1, N_D]} a_i h^s_i(\text{enc})$$

Attention Equation

Experiments

Model	Dataset	BLEU	ROUGE-1	ROUGE-2	Coherence(L)
Standard	Hotel Review	0.241	0.571	0.302	1.92
Hierarchical	Hotel Review	0.267	0.590	0.330	1.71
Hierarchical+Attention	Hotel Review	0.285	0.624	0.355	1.57
Standard	Wikipedia	0.178	0.502	0.228	2.75
Hierarchical	Wikipedia	0.202	0.529	0.250	2.30
Hierarchical+Attention	Wikipedia	0.220	0.544	0.291	2.04

Conclusions For Three Key Points

- Why Attention?
  - To select important words.
- Where to put Attention?
  - Along with all the LSTMs.
- How to weight Attention?
  - Nothing interesting.

An Overview of Attention-Based Models

@Xiaohan

Why Attention?

- Long-term de-noising.
  - Machine Translation.
  - Speech Recognition.
- To select effective input.
  - Image Caption Generator.
- To focus important parts.
  - Word Embedding.
  - Abstractive Summarization.
  - Reasoning about Entailment.
  - Language Model.

Where to put attention?

- To add an attention layer between source encoding and target decoding.
  - Machine Translation.
- To add an attention layer between input layer and hidden state recurrent layer.
  - Speech Recognition.
  - Language Models.
  - Reasoning about Entailment.
  - Image Caption Generator.
- To add an attention layer for word weighting.
  - Word Embedding.
  - Abstractive Summarization.

How to weight Attention?

- To consider Position Prior.
- To consider Input-Feeding.
- To consider an MLP
- To consider the smoothing trick.

Prospective

- I am supposed to share my options about the improvements of Attention-Based Models.
- But, time limits the reports.

Thanks.

- Q & A.