ATTENTION-BASED MODEL

IDEA

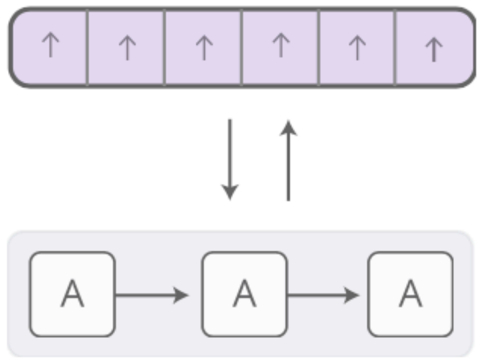
- Selectively concentrate on a small portion of the overall data
  - Longer memories
  - Local concise representation
  - Decoupling representation from a problem (different problem require different way to attend to the representation)
  - More computational efficiency.
  - Better interpretability for model

APPROACH

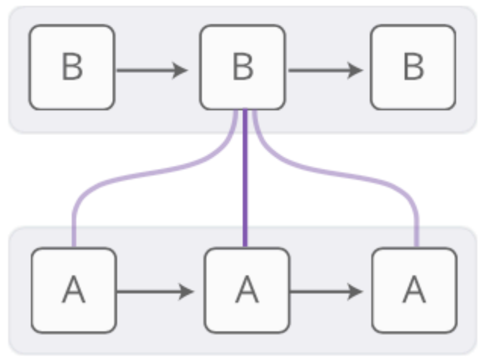
- RNN encoder-decoder attention
- Attentive Convolution
- Direction for implementation:
  - Implicit: integrating the attention mechanism into the network architecture
  - Explicit: using external model

RNN ENCODER-DECODER ATTENTION

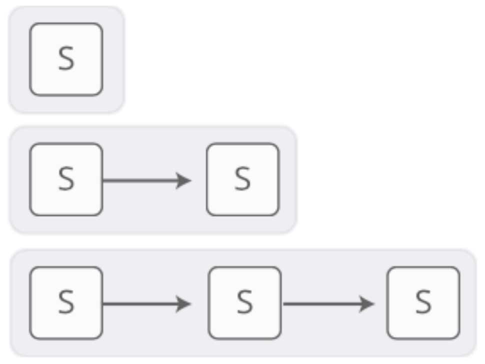
- How to improve RNN for seq2seq task



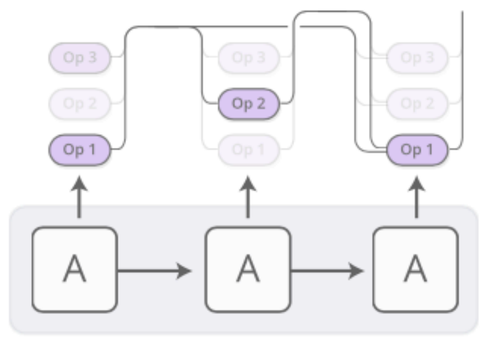
Neural Turing Machines have external memory that they can read and write to.



Attentional Interfaces allow RNNs to focus on parts of their input.

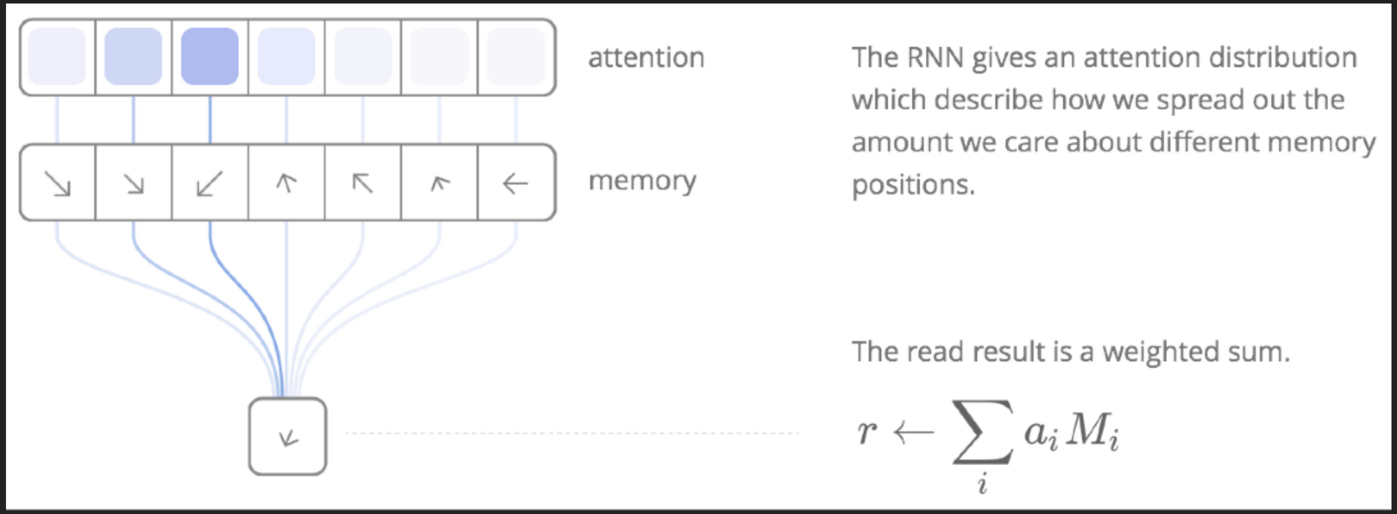


Adaptive Computation Time allows for varying amounts of computation per step.



Neural Programmers can call functions, building programs as they run.

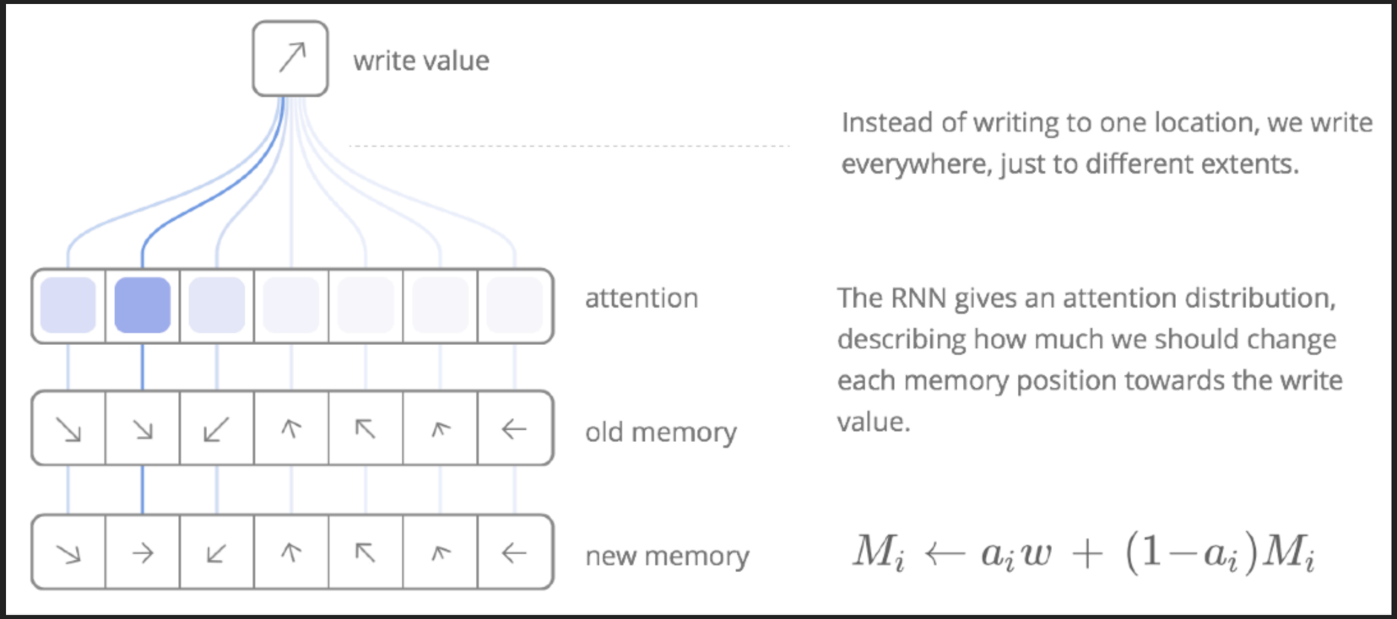
NEURAL TURING MACHINE



The RNN gives an attention distribution which describe how we spread out the amount we care about different memory positions.

The read result is a weighted sum.

$$r \leftarrow \sum_i a_i M_i$$



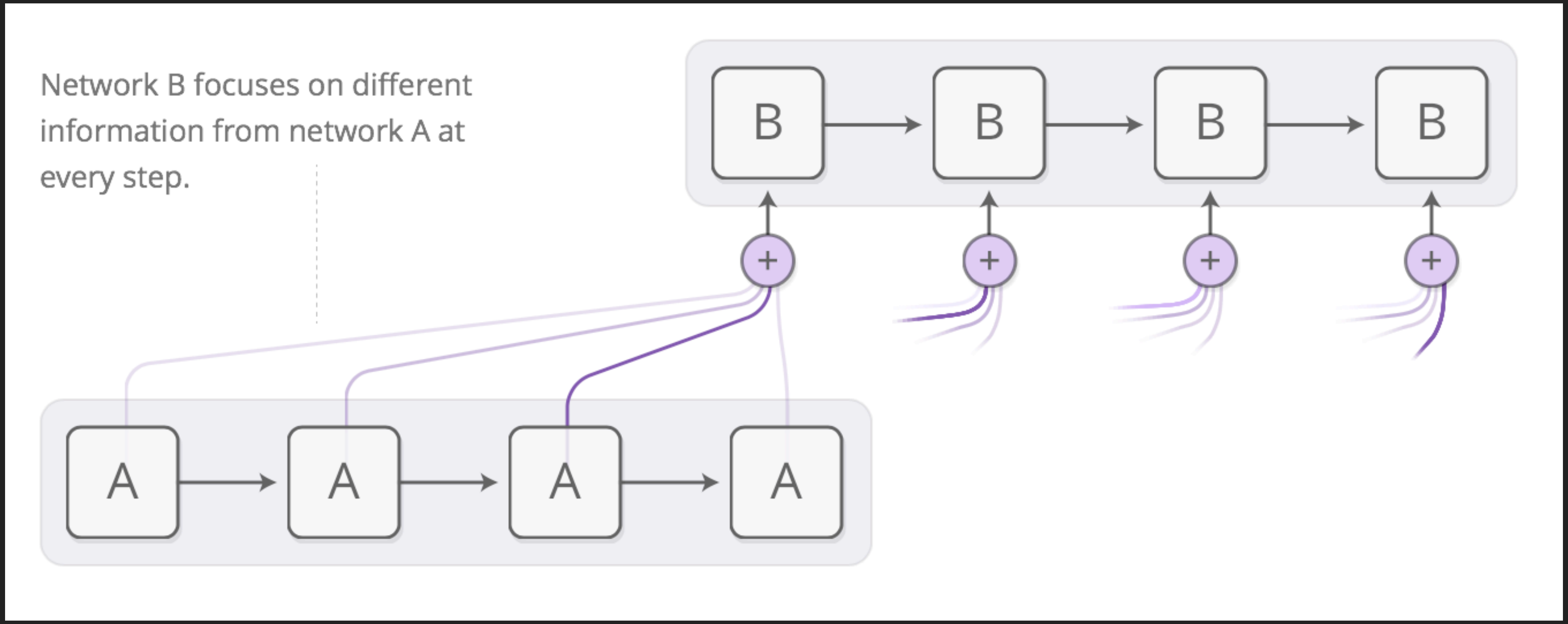
Instead of writing to one location, we write everywhere, just to different extents.

The RNN gives an attention distribution, describing how much we should change each memory position towards the write value.

$$M_i \leftarrow a_i w + (1 - a_i) M_i$$

- Inspired by Turing machine
  - Tape: memory external memory representation
  - Head: a controller select location of memory
  - Read network: learn how to represent the memory row for given task
  - Write network: encode states into memory representation.
- 2 types of attention:
  - Content-based attention: search through memory and focus on rows.
  - Location-based attention: precisely moving the relative location within row.

ENCODER-DECODER ATTENTION


Network B focuses on different information from network A at every step.
B
B
B
B
+
+
+
+
A
A
A
A

- An RNN can attend over the output of another RNN. At every time step, it focuses on different positions in the other RNN.
- Each item is dot-producted with the query to produce a score, describing how well it matches the query. The scores are fed into a softmax to create the attention distribution.

ENCODER-DECODER ATTENTION

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

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \mathbf{g}_t \tag{2}$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t). \tag{3}$$

E - embedding matrix

y - captions

h - previous hidden state

z - context vector, a dynamic representation of the relevant part of the image input at time t

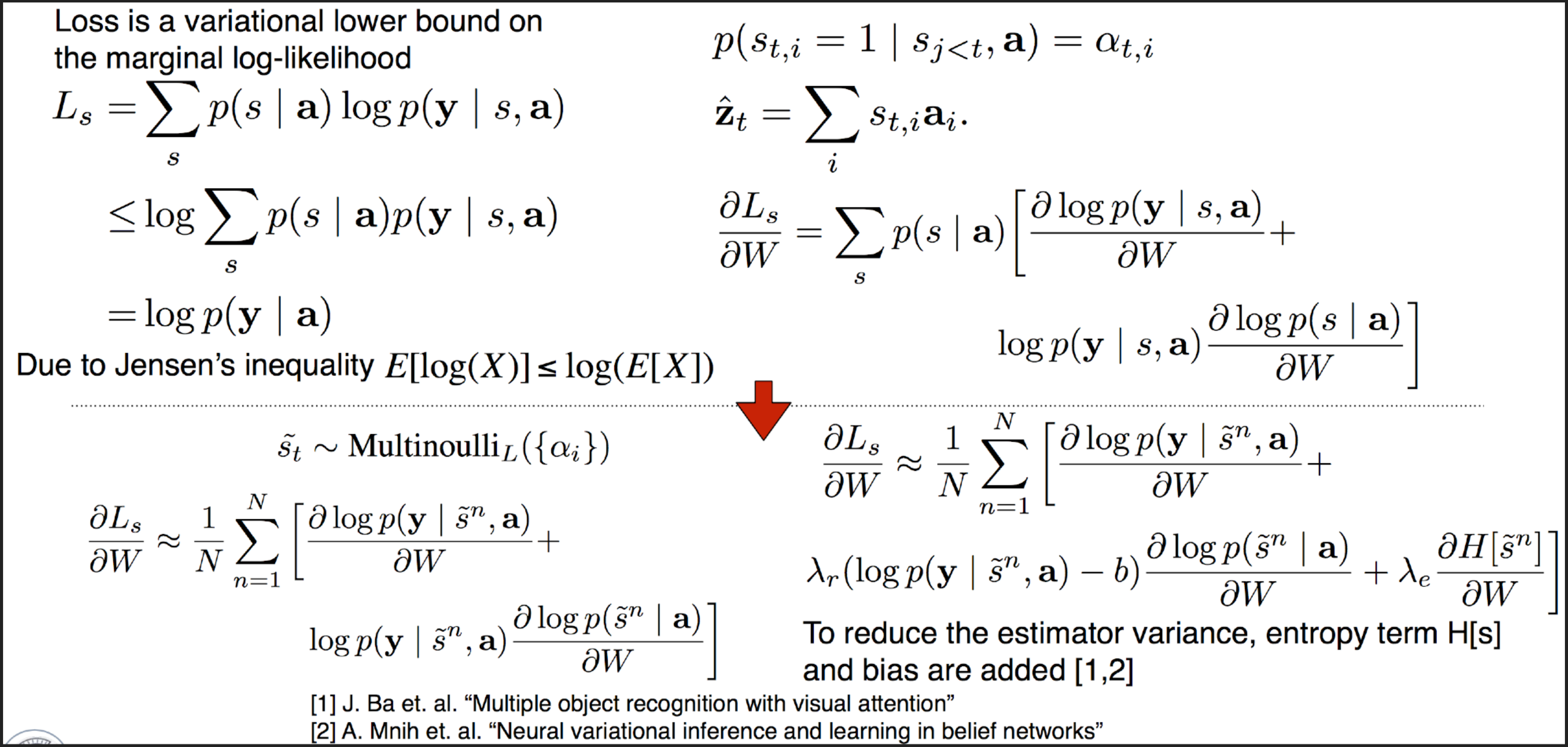
$$e_{ti} = f_{\text{att}}(\mathbf{a}_i, \mathbf{h}_{t-1})$$

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

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

ϕ is the ‘attention’ (‘focus’) function - ‘soft’ / ‘hard’

HARD ATTENTION



Loss is a variational lower bound on the marginal log-likelihood

$$L_s = \sum_s p(s \mid \mathbf{a}) \log p(\mathbf{y} \mid s, \mathbf{a})$$
$$\leq \log \sum_s p(s \mid \mathbf{a}) p(\mathbf{y} \mid s, \mathbf{a})$$
$$= \log p(\mathbf{y} \mid \mathbf{a})$$

Due to Jensen's inequality $E[\log(X)] \leq \log(E[X])$

$$p(s_{t,i} = 1 \mid s_{j<t}, \mathbf{a}) = \alpha_{t,i}$$

$$\hat{\mathbf{z}}_t = \sum_i s_{t,i} \mathbf{a}_i.$$

$$\frac{\partial L_s}{\partial W} = \sum_s p(s \mid \mathbf{a}) \left[\frac{\partial \log p(\mathbf{y} \mid s, \mathbf{a})}{\partial W} + \log p(\mathbf{y} \mid s, \mathbf{a}) \frac{\partial \log p(s \mid \mathbf{a})}{\partial W} \right]$$

$$\tilde{s}_t \sim \text{Multinoulli}_L(\{\alpha_i\})$$

$$\frac{\partial L_s}{\partial W} \approx \frac{1}{N} \sum_{n=1}^{N} \left[\frac{\partial \log p(\mathbf{y} \mid \tilde{s}^n, \mathbf{a})}{\partial W} + \log p(\mathbf{y} \mid \tilde{s}^n, \mathbf{a}) \frac{\partial \log p(\tilde{s}^n \mid \mathbf{a})}{\partial W} \right]$$

$$\frac{\partial L_s}{\partial W} \approx \frac{1}{N} \sum_{n=1}^{N} \left[\frac{\partial \log p(\mathbf{y} \mid \tilde{s}^n, \mathbf{a})}{\partial W} + \lambda_r (\log p(\mathbf{y} \mid \tilde{s}^n, \mathbf{a}) - b) \frac{\partial \log p(\tilde{s}^n \mid \mathbf{a})}{\partial W} + \lambda_e \frac{\partial H[\tilde{s}^n]}{\partial W} \right]$$

To reduce the estimator variance, entropy term H[s] and bias are added [1,2]

[1] J. Ba et. al. "Multiple object recognition with visual attention"
[2] A. Mnih et. al. "Neural variational inference and learning in belief networks"

SOFT ATTENTION

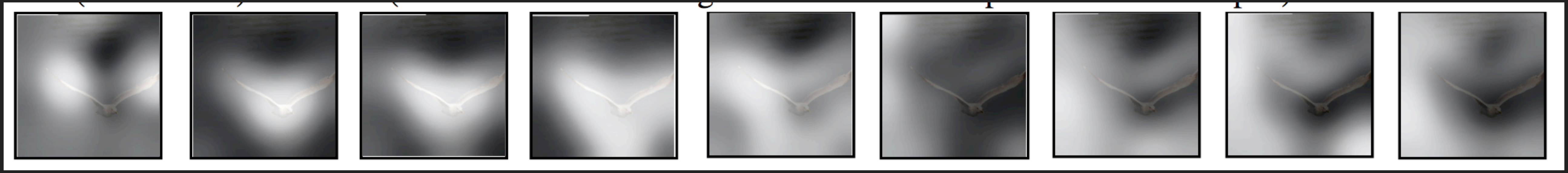
$$\hat{\mathbf{z}}_t = \sum_i s_{t,i}\mathbf{a}_i$$

Instead of making hard decisions, we take the expected context vector

$$\mathbb{E}_{p(s_t|a)}[\hat{\mathbf{z}}_t] = \sum_{i=1}^{L} \alpha_{t,i}\mathbf{a}_i$$

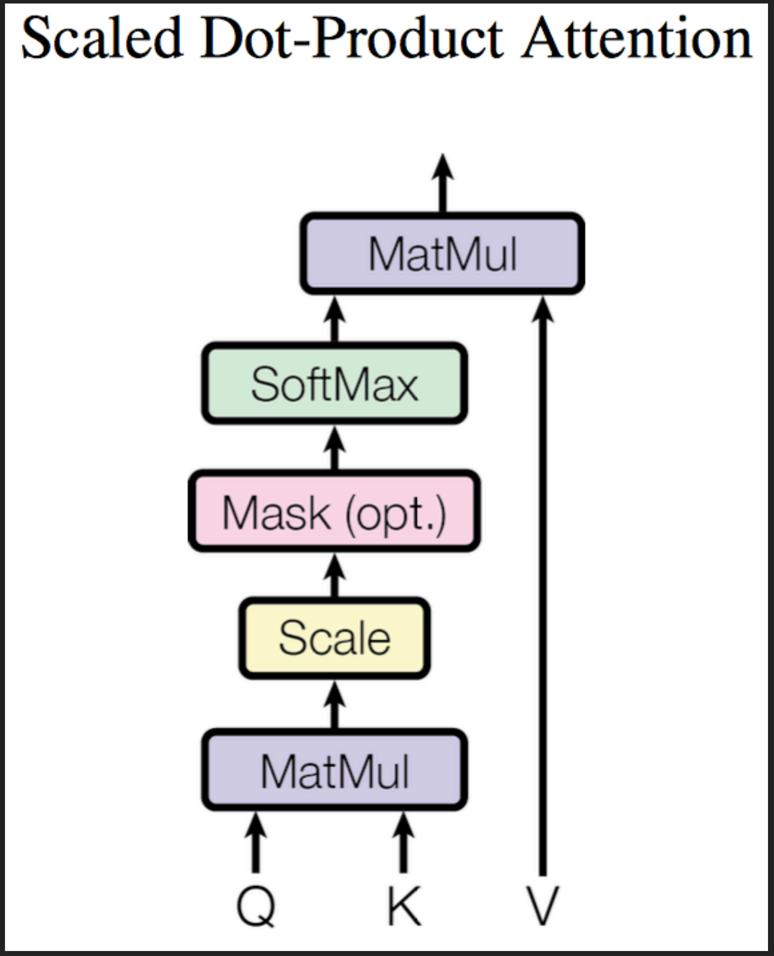
The whole model is smooth and differentiable under the deterministic attention; learning via a standard backprop.

$$\phi\left(\{\mathbf{a}_i\},\{\alpha_i\}\right) = \sum_i^L \alpha_i\mathbf{a}_i$$

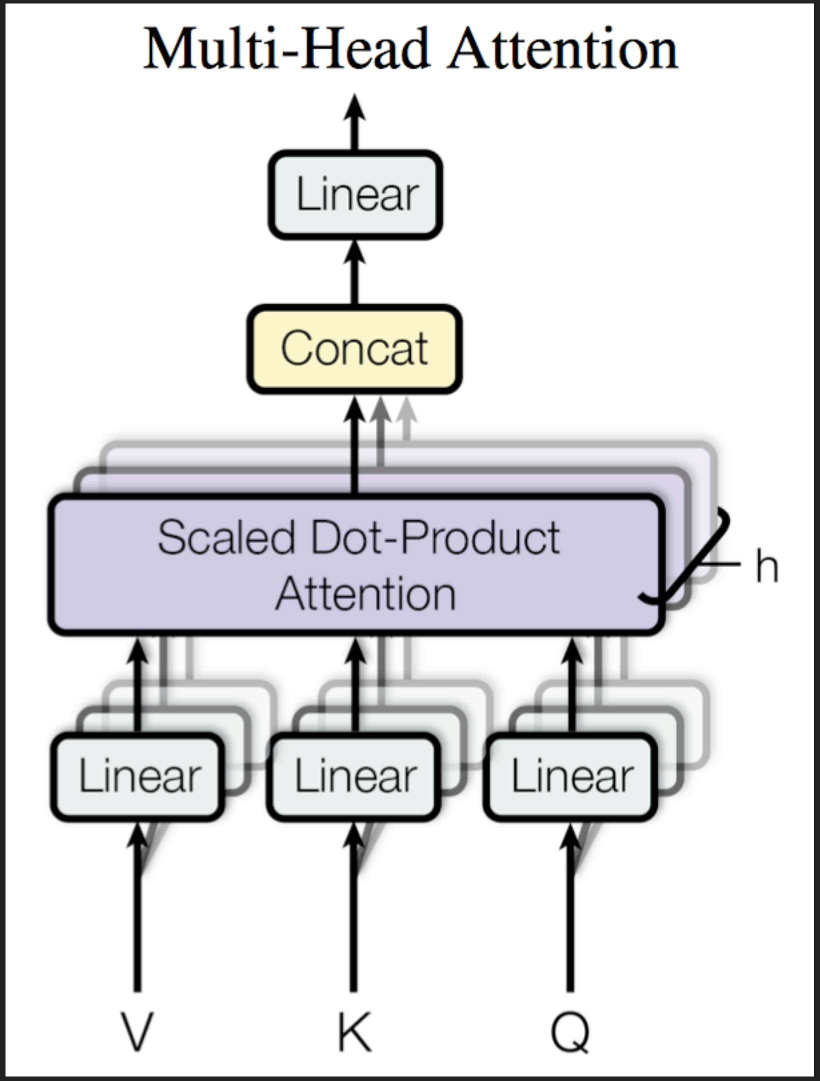


- Soft
  - Weighted average of whole input
  - Differentiable loss
  - Increased computational cost
- Hard
  - Sample parts of input
  - Policy gradient
  - Variational methods
  - Decreased computational cost

SOME IMPLEMENTATION OF SOFT ATTENTION

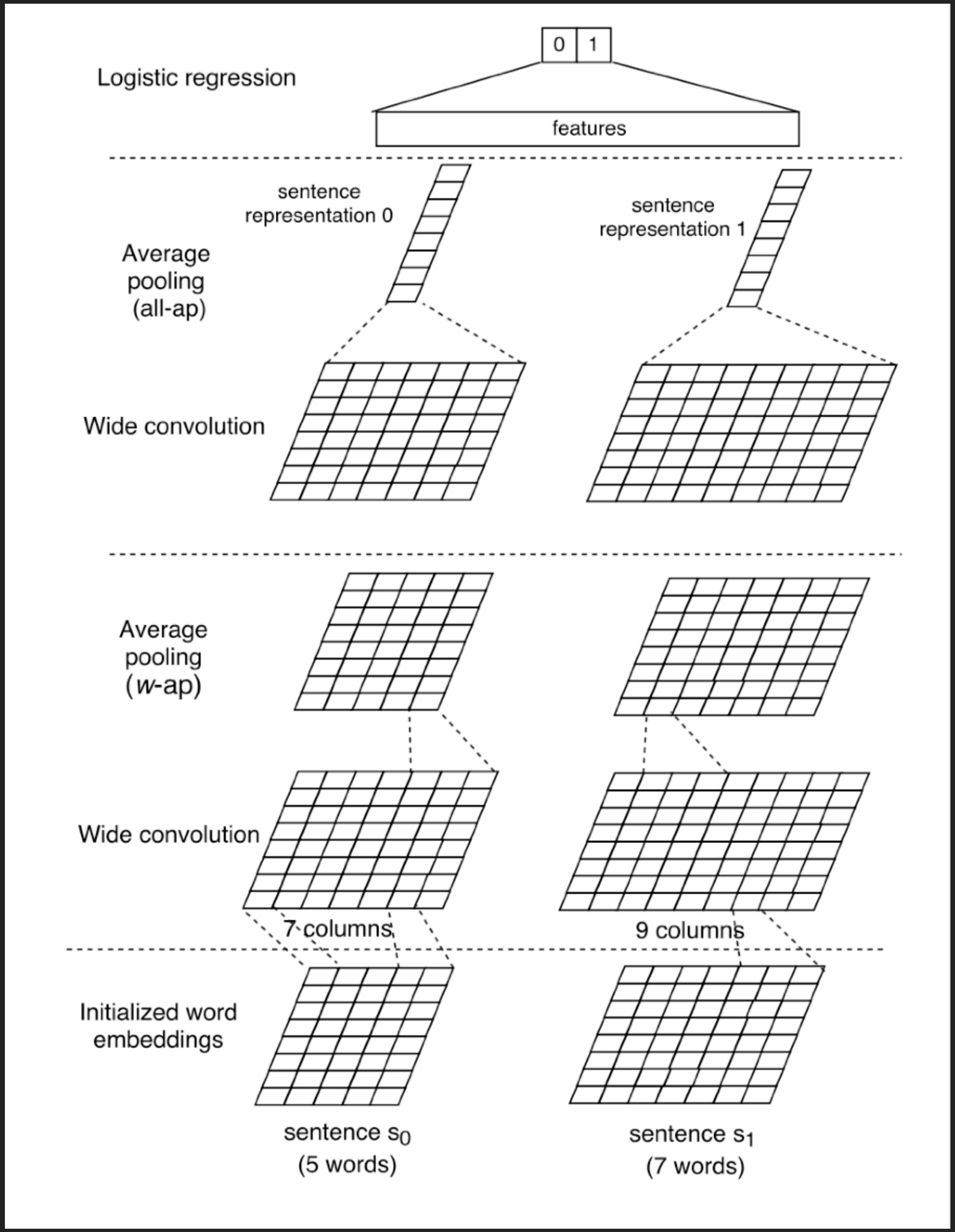


$$\mathrm{Attention}(Q,K,V) = \mathrm{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$



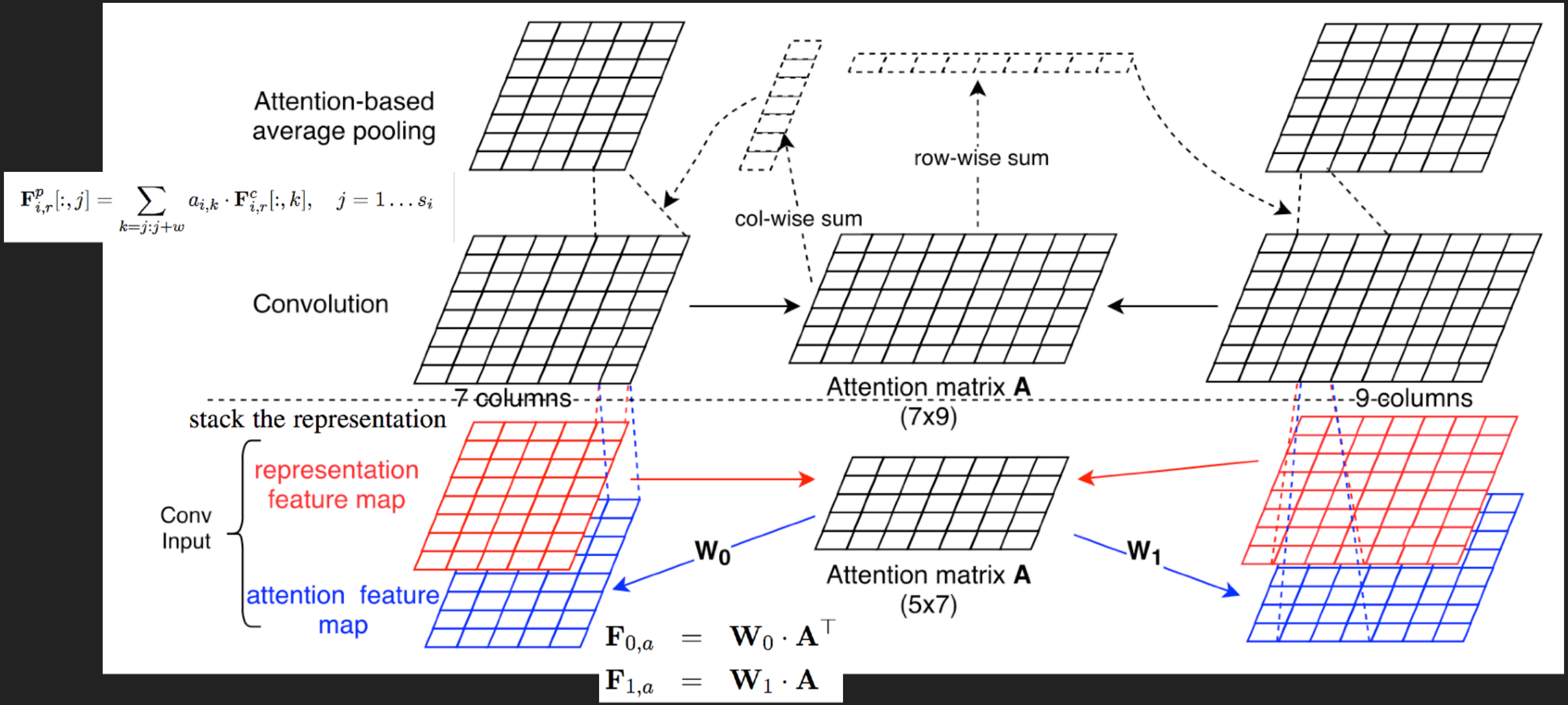
$$\begin{aligned}\mathrm{MultiHead}(Q,K,V) &= \mathrm{Concat}(\mathrm{head_1},\dots,\mathrm{head_h})W^O\\ \text{where } \mathrm{head_i} &= \mathrm{Attention}(QW_i^Q, KW_i^K, VW_i^V)\end{aligned}$$

ATTENTIVE CONVOLUTION



- Modeling sentence pair
- 2 weight-sharing CNN (left and right)

ATTENTIVE CONVOLUTION



$$\mathbf{F}^p_{i,r}[:,j] = \sum_{k=j:j+w} a_{i,k} \cdot \mathbf{F}^c_{i,r}[:,k], \quad j = 1 \dots s_i$$

$$\begin{aligned} \mathbf{F}_{0,a} &= \mathbf{W}_0 \cdot \mathbf{A}^\top \\ \mathbf{F}_{1,a} &= \mathbf{W}_1 \cdot \mathbf{A} \end{aligned}$$

- Bottom: directly on the *input* representation with the aim of improving the features computed by convolution. Matrix **A** mapping (coordinating) the representation of left and right feature map (row for left, column for right)
- Top: computes attention weights on the *output* of convolution with the aim of re-weighting this convolution output . Matrix **A** compares all unit in left and right