[SK Academy - 2019 AI Course]

Attention

Heung-Il Suk

hisuk@korea.ac.kr

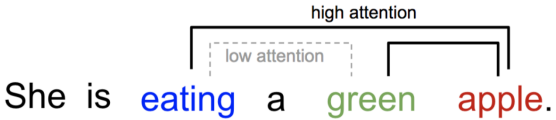
http://milab.korea.ac.kr

Department of Brain and Cognitive Engineering,
Korea University

Motivation

Visual attention to different regions of an image

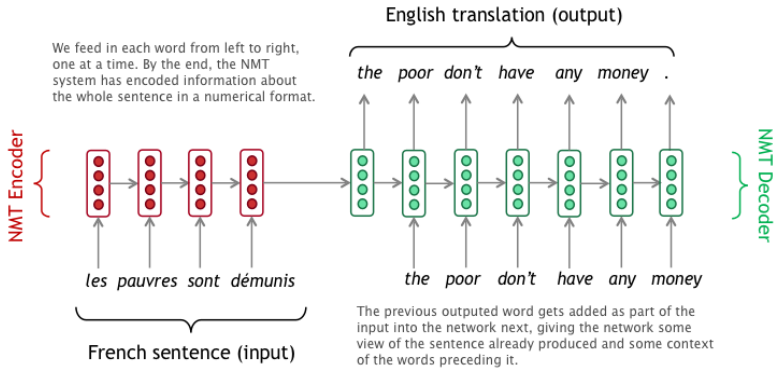
Correlating words in one sentence



Machine Intelligence Lab

Seq2Seq Model

- Aims to transform an input sequence (source) to a new one (target) and both sequences can be of arbitrary lengths
  - *e.g.*, machine translation between multiple languages in either text or audio, question-answer dialog generation, parsing sentences into grammar trees, *etc.*

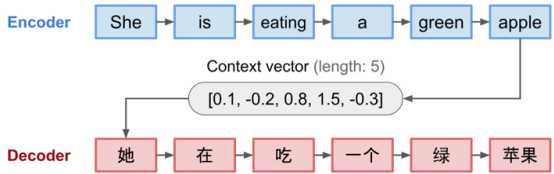


Encoder

- processes the input sequence and compresses the information into a *context vector* (also known as sentence embedding or "thought" vector) of a fixed length
- expected to be a good summary of the meaning of the whole source sequence

Decoder

- initialized with the context vector to emit the transformed output
- The early work only used the last state of the encoder network as the decoder initial state.



Fixed-length context vector: incapable of remembering long sentences

Attention as Rescue

Broadly interpreted as **a vector of importance weights**

- in order to predict or infer one element, such as a pixel in an image or a word in a sentence
- to create shortcuts between the context vector and the entire source input
  - estimate using the **attention vector** how strongly it is correlated with other elements
- weights of these shortcut connections are customizable for each output element
  - take the sum of their values weighted by the attention vector as the approximation of the target

Attention Interface

To be differentiable

- focus everywhere, just to different extents

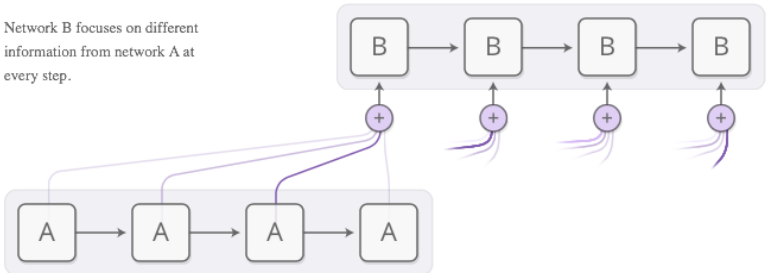
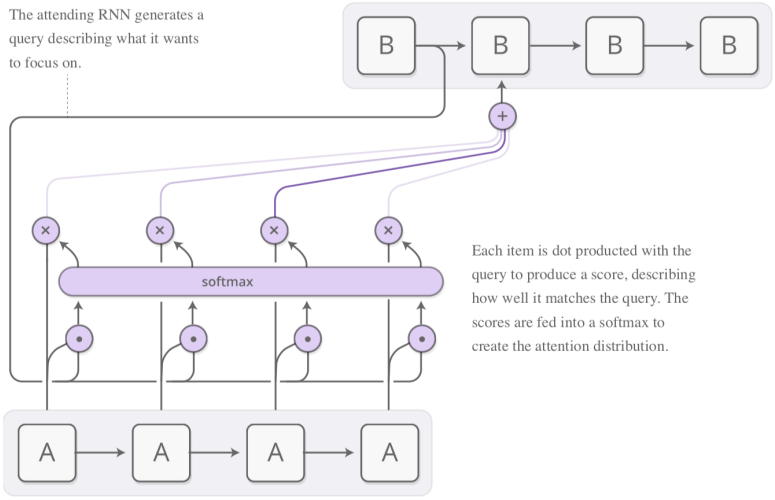


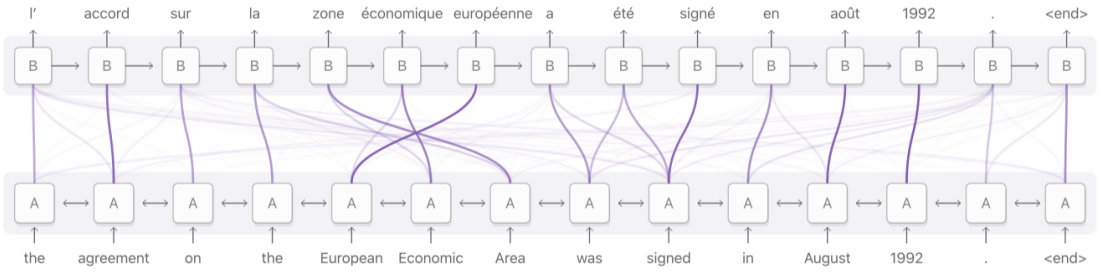
Image source: https://distill.pub/2016/augmented-rnns/

Machine Intelligence Lab

Content-based attention



Machine translation [Bahdanau *et al.*, 2015]



Voice recognition [Chen *et al.*, 2015]

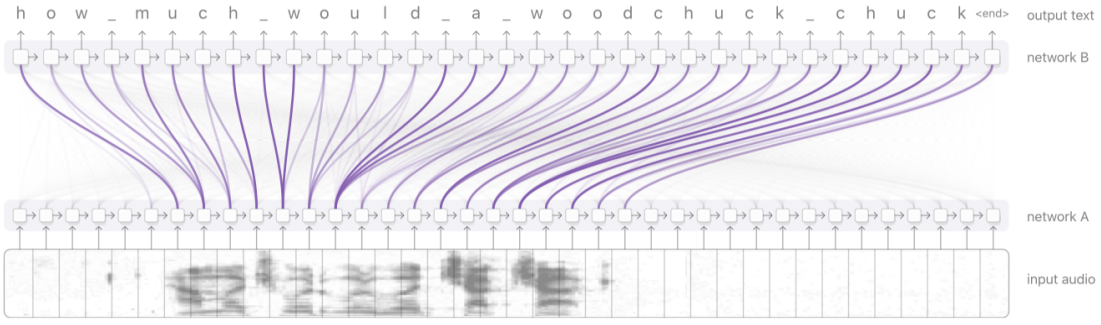


Image captioning [Xu *et al.*, 2016]

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.

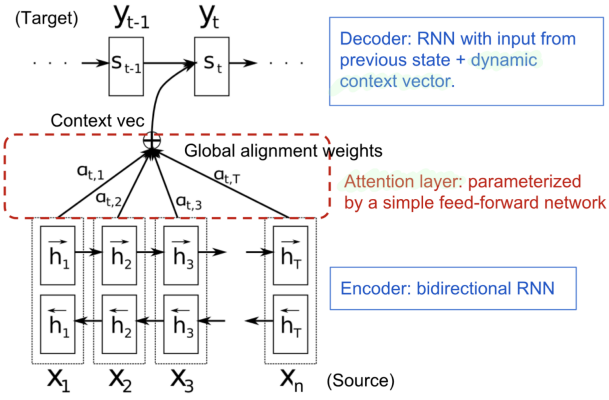
A group of people sitting on a boat in the water.

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

Machine Intelligence Lab

Neural Machine Translation [Bahdanau *et al.*, 2015]

- The **alignment** between the source and target is learned and controlled by the context vector.
- The context vector consumes three pieces of information
  - encoder hidden states
  - decoder hidden states
  - **alignment** between source and target



- source sequence $\mathbf{x} = [x_1, x_2, \ldots, x_n]$, target sequence $\mathbf{y} = [y_1, y_2, \ldots, y_m]$
- encoder state: simple concatenation of two represents the encoder state

$$\mathbf{h}_i = \left[\overrightarrow{\mathbf{h}_i}^\top; \overleftarrow{\mathbf{h}_i}^\top\right]^\top, \quad i = 1, \ldots, n$$

- context vector $\mathbf{c}_t$

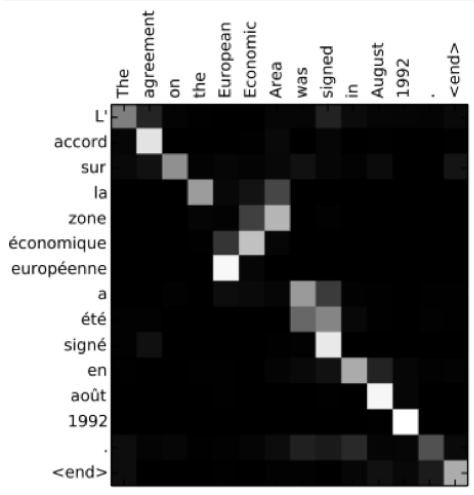
$$\begin{aligned}
\mathbf{c}_t &= \sum_{i=1}^{n} \alpha_{t,i} \mathbf{h}_i \\
\alpha_{t,i} &= \text{align}\left(y_t, x_i\right) \quad \text{(how well two words } y_t \text{ and } x_i \text{ are aligned)} \\
&= \text{softmax}\left(\text{score}\left(\mathbf{s}_{t-1}, \mathbf{h}_i\right)\right) = \frac{\exp\left[\text{score}\left(\mathbf{s}_{t-1}, \mathbf{h}_i\right)\right]}{\sum_{j=1}^{n} \exp\left[\text{score}\left(\mathbf{s}_{t-1}, \mathbf{h}_j\right)\right]}
\end{aligned}$$

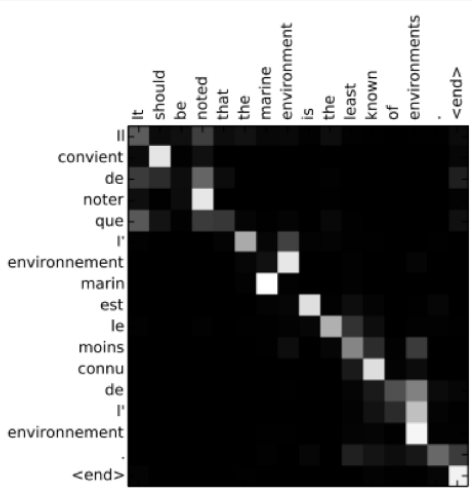
- decoder state

$$\mathbf{s}_t = f\left(\mathbf{s}_{t-1}, y_{t-1}, \mathbf{c}_t\right), \quad t = 1, \ldots, m$$

Name	Alignment score function	Citation
Content-base attention	$\text{score}(\boldsymbol{s}_t, \boldsymbol{h}_i) = \text{cosine}[\boldsymbol{s}_t, \boldsymbol{h}_i]$	Graves2014
Additive(*)	$\text{score}(\boldsymbol{s}_t, \boldsymbol{h}_i) = \mathbf{v}_a^\top \tanh(\mathbf{W}_a[\boldsymbol{s}_t; \boldsymbol{h}_i])$	Bahdanau2015
Location-Base	$\alpha_{t,i} = \text{softmax}(\mathbf{W}_a \boldsymbol{s}_t)$ Note: This simplifies the softmax alignment to only depend on the target position.	Luong2015
General	$\text{score}(\boldsymbol{s}_t, \boldsymbol{h}_i) = \boldsymbol{s}_t^\top \mathbf{W}_a \boldsymbol{h}_i$ where $\mathbf{W}_a$ is a trainable weight matrix in the attention layer.	Luong2015
Dot-Product	$\text{score}(\boldsymbol{s}_t, \boldsymbol{h}_i) = \boldsymbol{s}_t^\top \boldsymbol{h}_i$	Luong2015
Scaled Dot-Product(^)	$\text{score}(\boldsymbol{s}_t, \boldsymbol{h}_i) = \frac{\boldsymbol{s}_t^\top \boldsymbol{h}_i}{\sqrt{n}}$ Note: very similar to the dot-product attention except for a scaling factor; where n is the dimension of the source hidden state.	Vaswani2017

Matrix of alignment scores: explicitly showing the correlation between source and target words





Machine Intelligence Lab

Categories of Attention Mechanism

Name	Definition	Citation
Self-Attention(&)	Relating different positions of the same input sequence. Theoretically the self-attention can adopt any score functions above, but just replace the target sequence with the same input sequence.	Cheng2016
Global/Soft	Attending to the entire input state space.	Xu2015
Local/Hard	Attending to the part of input state space; i.e. a patch of the input image.	Xu2015; Luong2015

Self-Attention

- Relating different positions of a single sequence in order to compute a representation of the same sequence
- Useful in machine reading, abstractive summarization, or image description generation

The FBI is chasing a criminal on the run .
The FBI is chasing a criminal on the run .
The FBI is chasing a criminal on the run .
The FBI is chasing a criminal on the run .
The FBI is chasing a criminal on the run .
The FBI is chasing a criminal on the run .
The FBI is chasing a criminal on the run .
The FBI is chasing a criminal on the run .
The FBI is chasing a criminal on the run .
The FBI is chasing a criminal on the run .

Machine Intelligence Lab

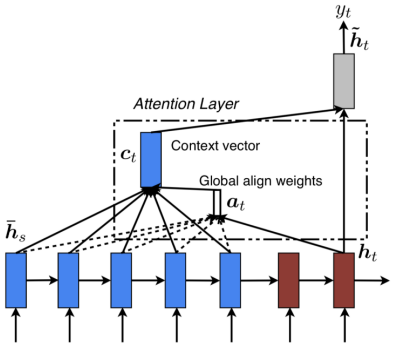
Soft vs. Hard Attention

- (Deterministic) Soft Attention: the alignment weights are learned and placed "softly" over all patches in the source image [Bahdanau et al., 2015]
  - Pro: **smooth and differentiable** model
  - Con: expensive when the source input is large

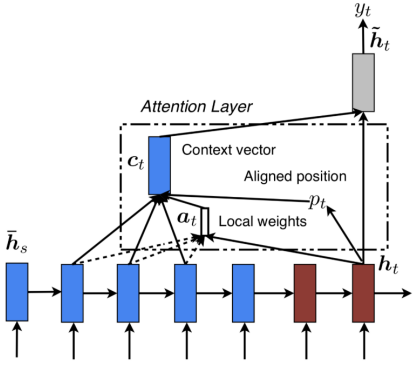
- (Stochastic) Hard Attention: only selects one patch of the image to attend to at a time
  - Pro: **less calculation at the inference time**
  - Con: non-differentiable model and requires more complicated techniques such as variance reduction or reinforcement learning to train [Luong, et al., 2015]

Global vs. Local Attention

- Global attention: similar to the soft attention
- Local attention: an interesting blend between hard and soft
  - an improvement over the hard attention to make it differentiable
  - the model first predicts a single aligned position for the current target word and a window centered around the source position is then used to compute a context vector

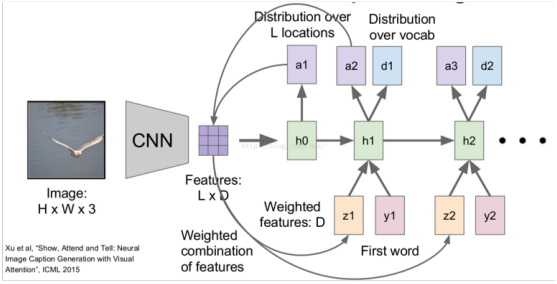


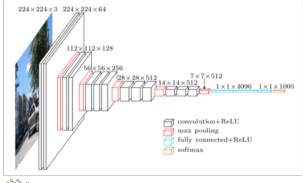
Global Attention Model



Local Attention Model

Neural Image Caption Generation [Xu *et al.*, 2015]





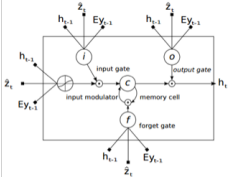


Image features

- extracted from the lower convolutional layers

$$\mathbf{a} = \{\mathbf{a}_1, \ldots, \mathbf{a}_L\}, \quad \mathbf{a}_i \in \mathbb{R}^D$$

- to capture multiple objects inside an image

Caption generation

- LSTM training in a seq2seq manner

$$y = \{\mathbf{y}_1, \ldots, \mathbf{y}_C\}, \quad \mathbf{y}_i \in \mathbb{R}^K$$

- Attention mechanism involved at each step

$$\hat{\mathbf{z}}_t = \sum_i s_{t,i}\mathbf{a}_i \quad \text{(hard attention)}$$

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

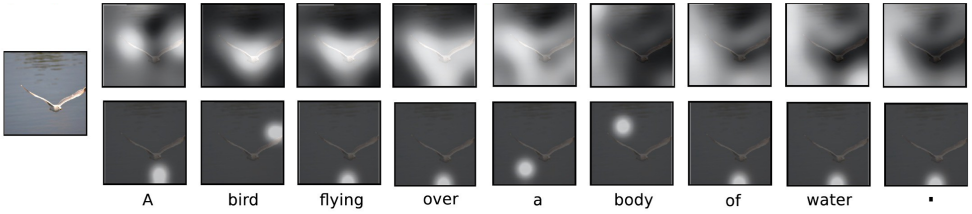
Machine Intelligence Lab

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

$$\alpha_{t,i} = \frac{\exp(e_{t,i})}{\sum_{k=1}^{L} \exp(e_{t,k})}$$

+ $e_{t,i}$: weight for location i, f_{att}: multilayer perceptron
+ softmax: a valid multinomial distribution parameter

- at each time step, the LSTM is fed a feature from different image location to generate the corresponding word



Deterministic ‘soft’ attention

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

Stochastic ‘hard’ attention

$$\begin{aligned} p\left(s_{t,i} = 1 | s_{j<t}, \mathbf{a}\right) &= \alpha_{t,i} \\ \hat{\mathbf{z}}_t &= \sum_i s_{t,i}\mathbf{a}_i \end{aligned}$$

Machine Intelligence Lab

- Objective function in hard attention
  - variational lower bound on the marginal log-likelihood $\log p(\mathbf{y}|\mathbf{a})$

$$\mathcal{L}_s = \sum_s p(s|\mathbf{a}) \log p(\mathbf{y}|s, \mathbf{a}) \leq \log \sum_s p(s|\mathbf{a})\, p(\mathbf{y}|s, \mathbf{a}) = \log p(\mathbf{y}|\mathbf{a})$$

- Learning for parameter W (via Monte-Carlo estimator of the gradient)

$$\frac{\partial \mathcal{L}_s}{\partial W} = \sum_s p(s|\mathbf{a}) \left[\frac{\partial \log p(\mathbf{y}|s, \mathbf{a})}{\partial W} + \log p(\mathbf{y}|s, \mathbf{a}) \frac{\partial \log p(s|\mathbf{a})}{\partial W} \right]$$

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

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

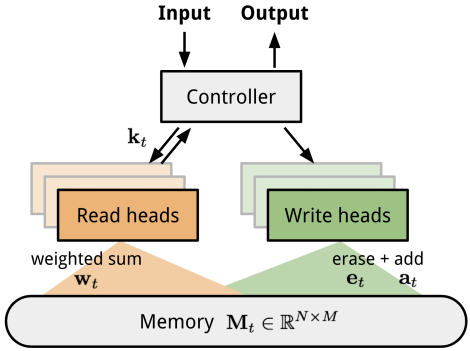
- Moving average baseline to reduce the variance in the Monte-Carlo estimator of the gradient
  - Upon seeing the k-th mini-batch

$$b_k = 0.9 \times b_{k-1} + 0.1 \times \log p(\mathbf{y}|\tilde{s}_k, \mathbf{a})$$

Neural Turing Machines

Coupling a neural network with external memory storage

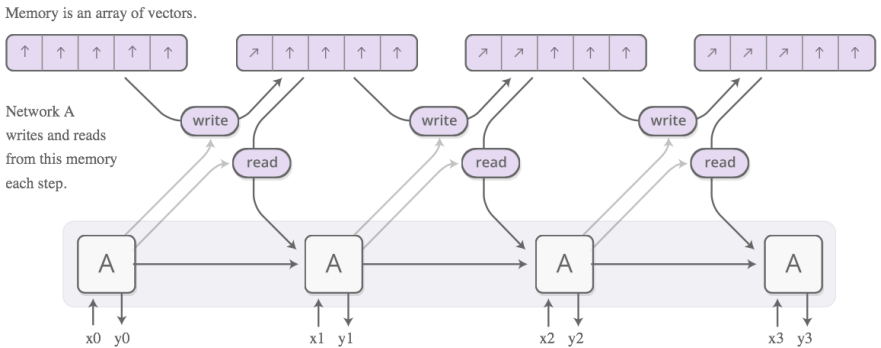
- **Controller**: in charge of executing operations on memory
  - any type of neural network
  - processes the input and interacts with the memory bank accordingly to generate output
  - interaction: handled by a set of parallel 'read' and 'write' heads
- **Memory**: stores processed information
  - an array of M-dimensional vectors

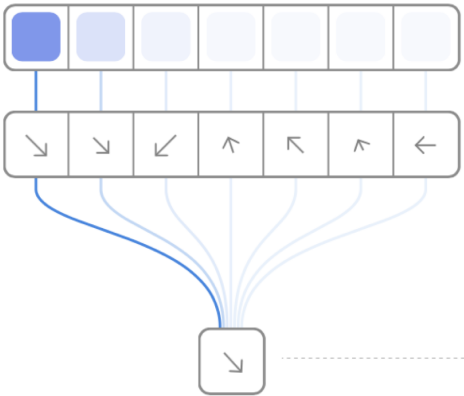


Reading and Writing

- Challenge: learning where to read and write
  - need to be differentiable w.r.t. the location we read from and write to

"at every step, read and write everywhere, just to different extents"

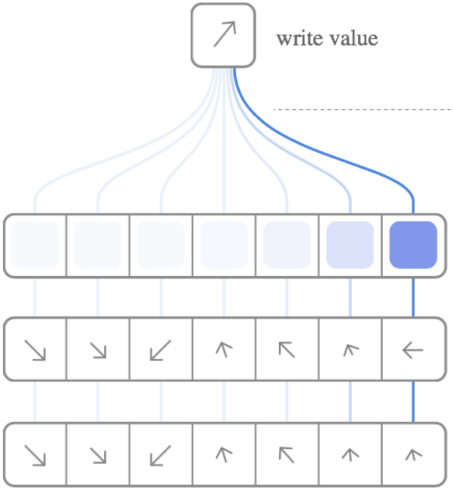




The RNN gives an attention distribution which describe how we spread out the amount we care about different memory positions.

The read result is a weighted sum.

$$\mathbf{r} = \sum_i w_t(i)\mathbf{M}_t(i)$$



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.

erase: $\tilde{\mathbf{M}}_t(i) = \mathbf{M}_{t-1}(i)\,[1 - w_t(i)\mathbf{e}_t]$

add: $\mathbf{M}_t(i) = \tilde{\mathbf{M}}_t(i) + w_t(i)\mathbf{a}_t$

Attention mechanism

★ Content-based addressing

- allows NTMs to search through their memory and focus on places that match what they're looking for

$$\mathbf{w}_t^c = \text{softmax}\left(\beta_t \cdot \cos\left(\mathbf{k}_t, \mathbf{M}_t\right)\right)$$

- $\mathbf{k}_t$: key vector extracted by the controller from the input and memory rows
- $\cos\left(\mathbf{k}_t, \mathbf{M}_t\right)$: cosine similarity between $\mathbf{k}_t$ and the rows of $\mathbf{M}_t$
- β_t: a strength multiplier to amplify or attenuate the focus of the distribution

• Interpolation

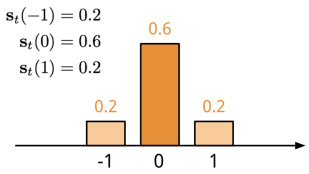
- to blend the newly generated content-based attention vector with the attention weights in the last time step

$$\mathbf{w}_t^g = g_t \mathbf{w}_t^c + \left(1 - g_t\right) \mathbf{w}_{t-1}$$

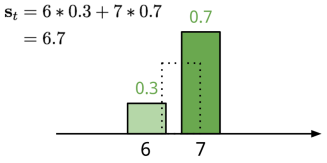
★ Location-based addressing

- allows relative movement in memory, enabling the NTM to loop
- sums up the values at different positions in the attention vector, weighted by a weighting distribution over allowable integer shifts
- equivalent to a 1-d convolution with a kernel $\mathbf{s}_t$, a function of the position offset

When $\mathbf{s}_t$ corresponds to the shift weighting distribution at positions (-1, 0, 1).



When $\mathbf{s}_t$ corresponds to the lower bound of an uniform distribution of width 1.

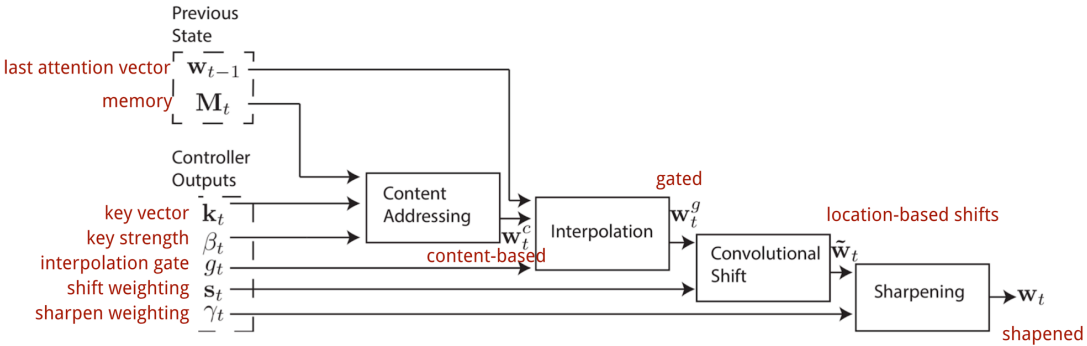


Two ways to represent the shift weighting distribution $\mathbf{s}_t$

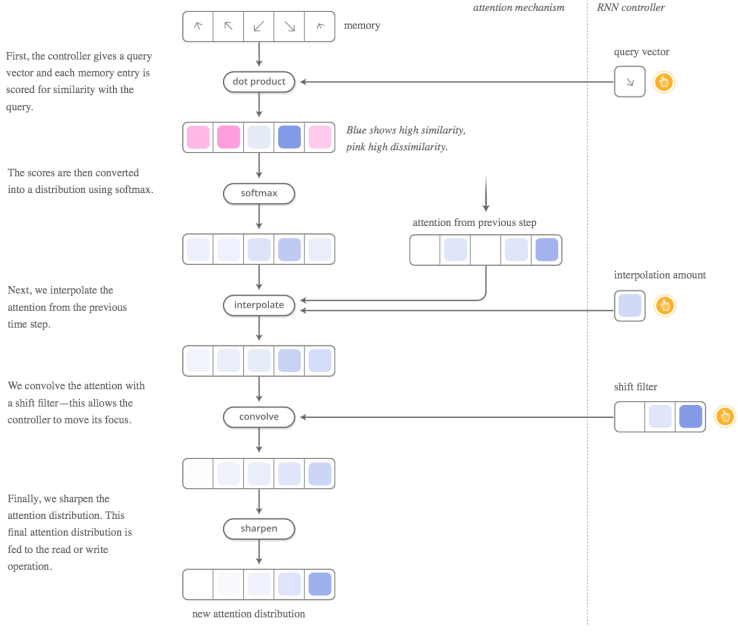
- **Mixture of content-based and location-based addressing**
  - $\gamma_t \geq 1$: sharpening scalar

$$\tilde{\mathbf{w}}_t(i) = \sum_{j=1}^{N} \mathbf{w}_t^g(j) s_t\left(i-j\right) \quad : \text{circular convolution}$$

$$\mathbf{w}_t(i) = \frac{\tilde{\mathbf{w}}_t(i)^{\gamma_t}}{\sum_{j=1}^{N} \tilde{\mathbf{w}}_t(j)^{\gamma_t}} \quad : \text{sharpening}$$



(Flow diagram of the addressing mechanisms [Graves *et al.*, 2014])



Machine Intelligence Lab

Thank you
for your attention!!!

(Q & A)

hisuk (AT) korea.ac.kr

http://milab.korea.ac.kr

Machine Intelligence Lab