CRIPAC
智能感知与计算研究中心
Center for Research on Intelligent Perception and Computing

中国科学院自动化研究所
Institute of Automation
Chinese Academy of Sciences

2018

"Deep Learning Lecture"

Lecture 11：Attention and Memory

Wei Wang

Center for Research on Intelligent Perception and Computing (CRIPAC)

National Laboratory of Pattern Recognition (NLPR)

Institute of Automation, Chinese Academy of Science (CASIA)

Outline

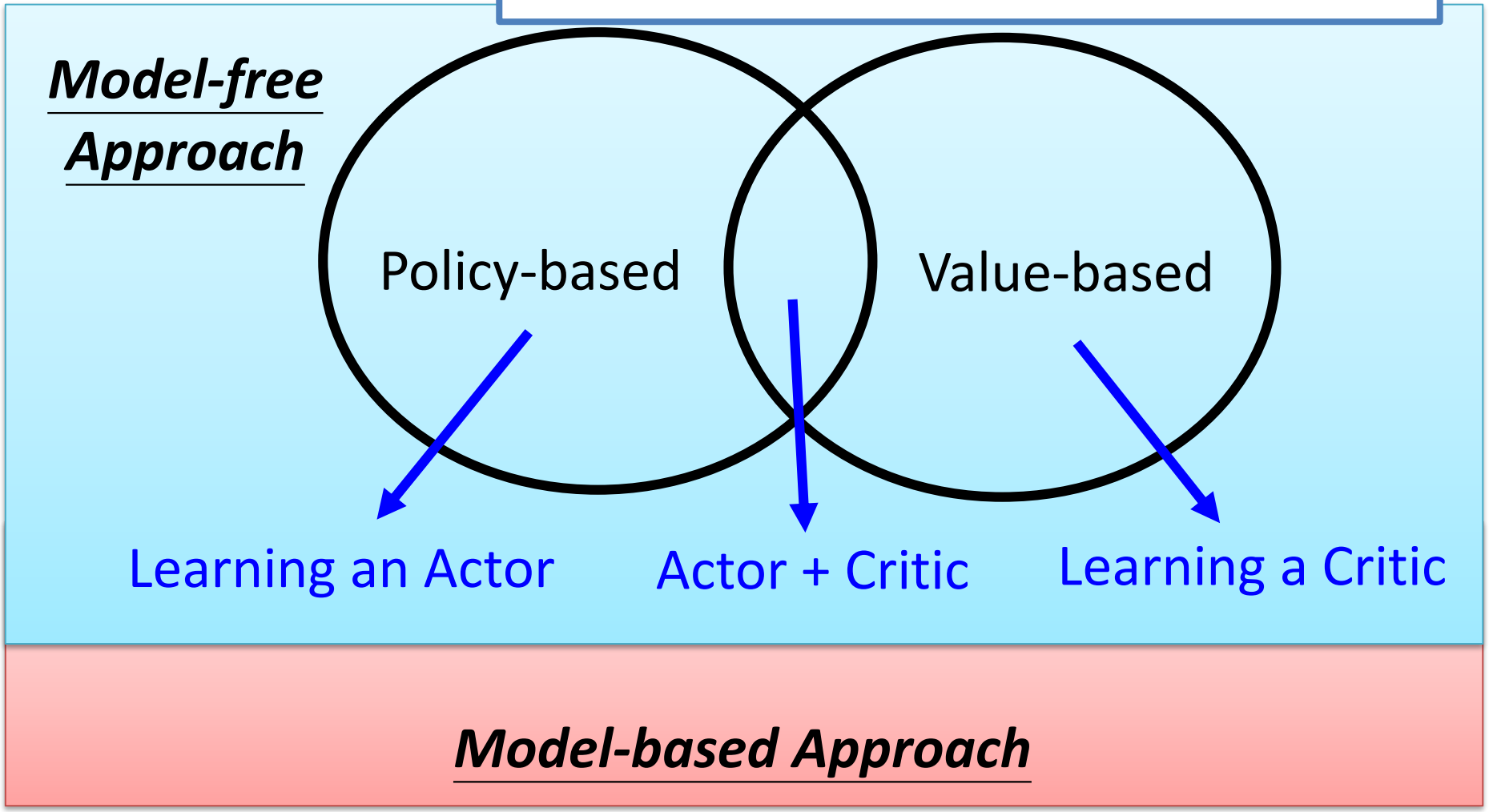
1. Course Review
2. Attention Models
3. Memory Models

Scenario of Reinforcement Learning

https://yoast.com/how-to-clean-site-structure/

Methods of Reinforcement Learning

Alpha Go: policy-based + value-based + model-based



Policy Gradient

$$\bar{R}_\theta = \sum_\tau R(\tau)P(\tau|\theta) \quad \nabla\bar{R}_\theta = ?$$

$$\nabla\bar{R}_\theta = \sum_\tau R(\tau)\nabla P(\tau|\theta) = \sum_\tau R(\tau)P(\tau|\theta)\frac{\nabla P(\tau|\theta)}{P(\tau|\theta)}$$

$R(\tau)$ do not have to be differentiable

It can even be a black box.

$$= \sum_\tau R(\tau)P(\tau|\theta)\nabla logP(\tau|\theta)$$

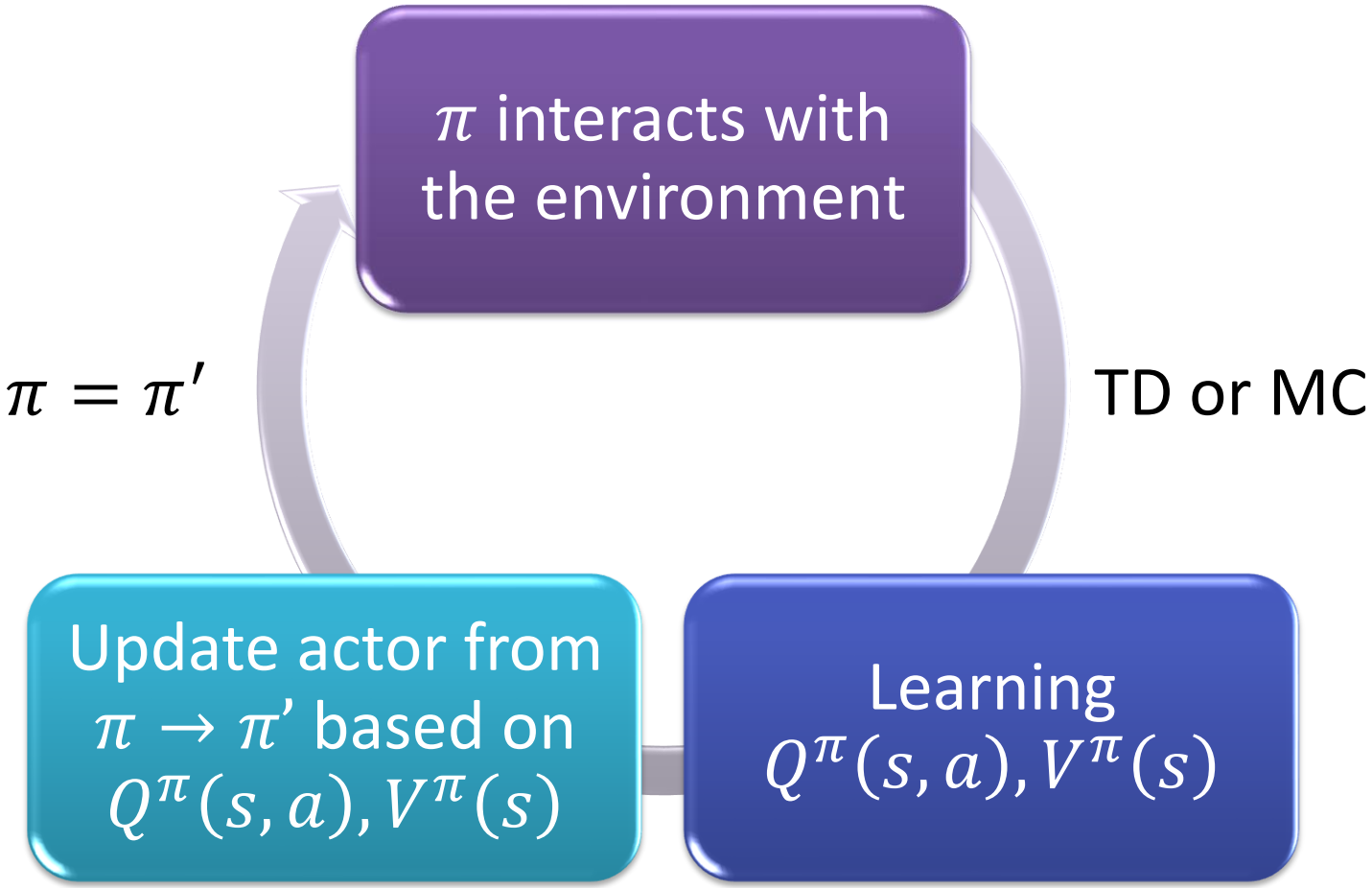
$$\frac{dlog\big(f(x)\big)}{dx} = \frac{1}{f(x)}\frac{df(x)}{dx}$$

$$\approx \frac{1}{N}\sum_{n=1}^{N} R(\tau^n)\nabla logP(\tau^n|\theta)$$

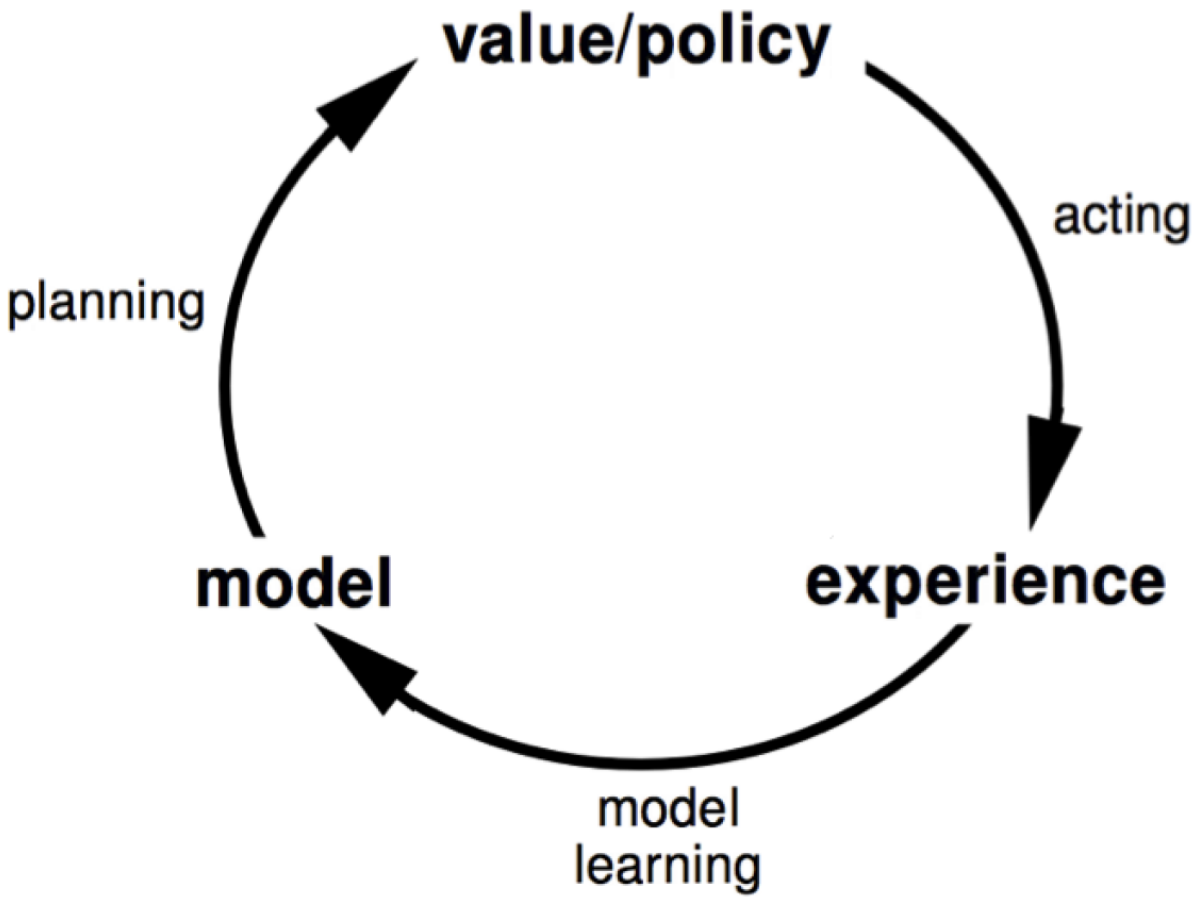
Use π_θ to play the game N times,

Obtain $\{\tau^1, \tau^2, \cdots, \tau^N\}$

Actor-Critic



Model-Based RL

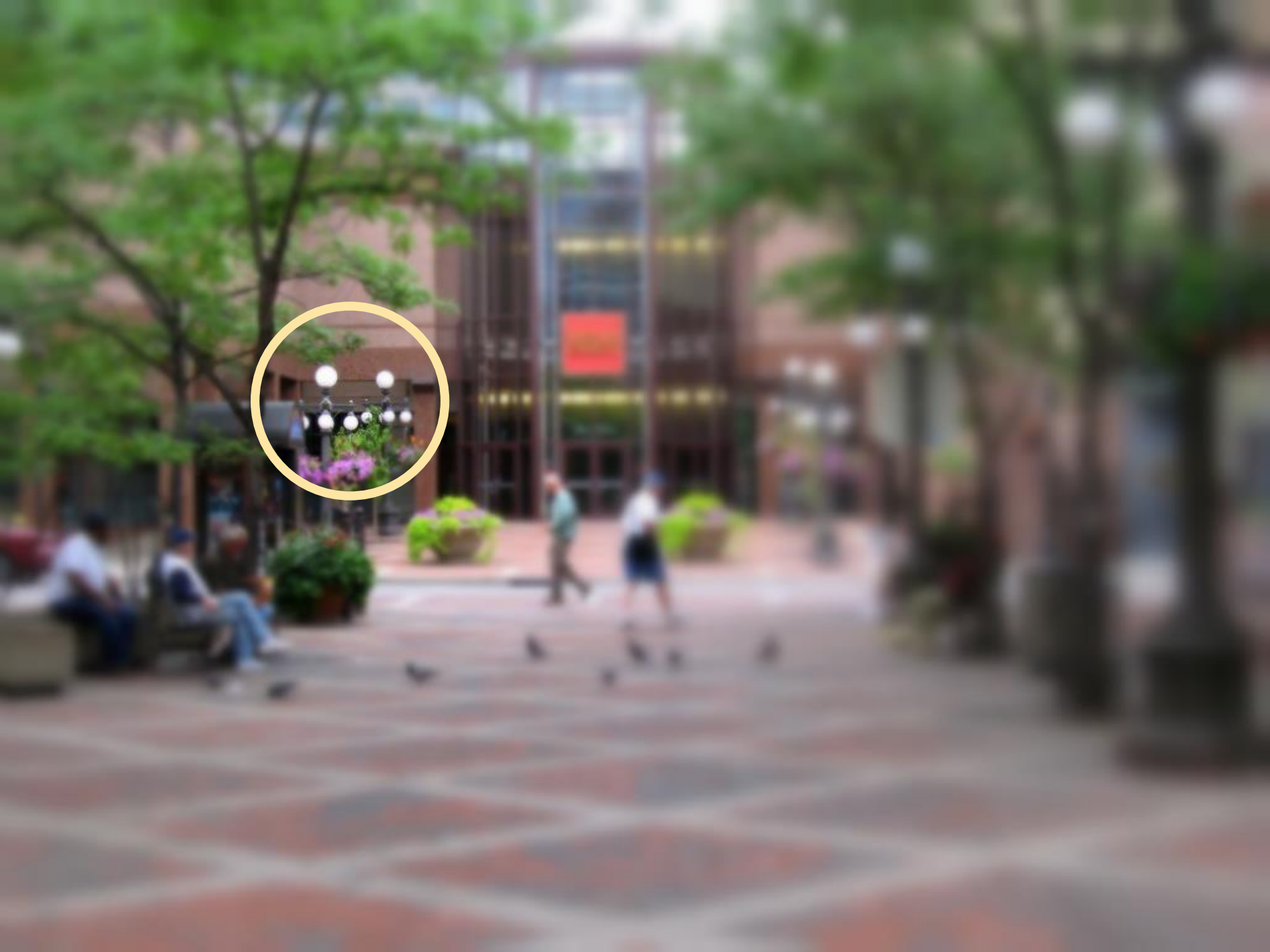


Outline

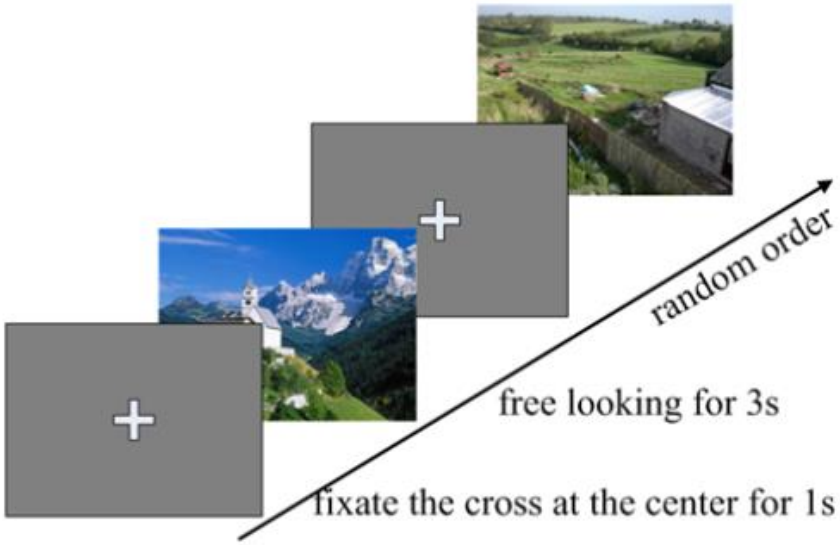
1. Course Review
2. Attention Models
3. Memory Models


WELLS FARGO

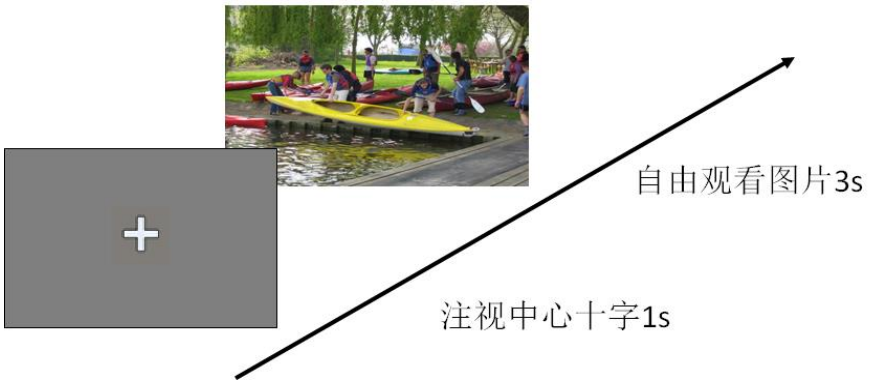

WELLS
FARGO



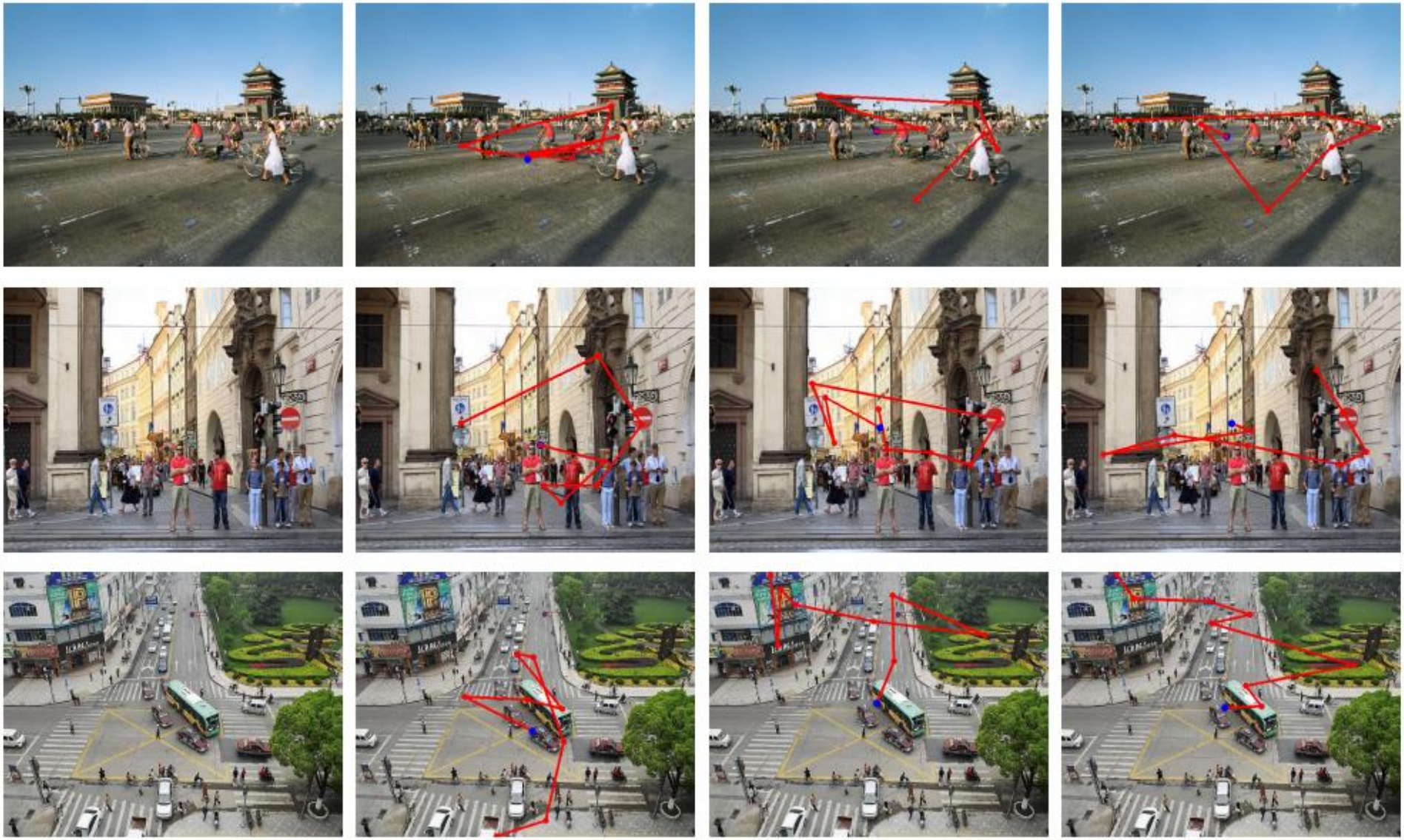
Eye Tracking



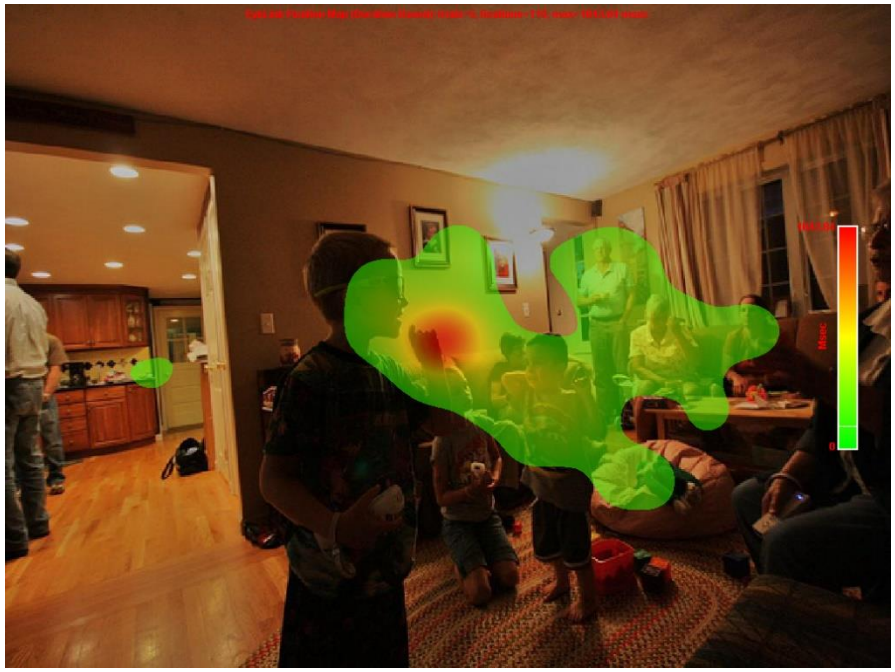
任务：让被试看完一副图片后，用文字对图片内容进行描述

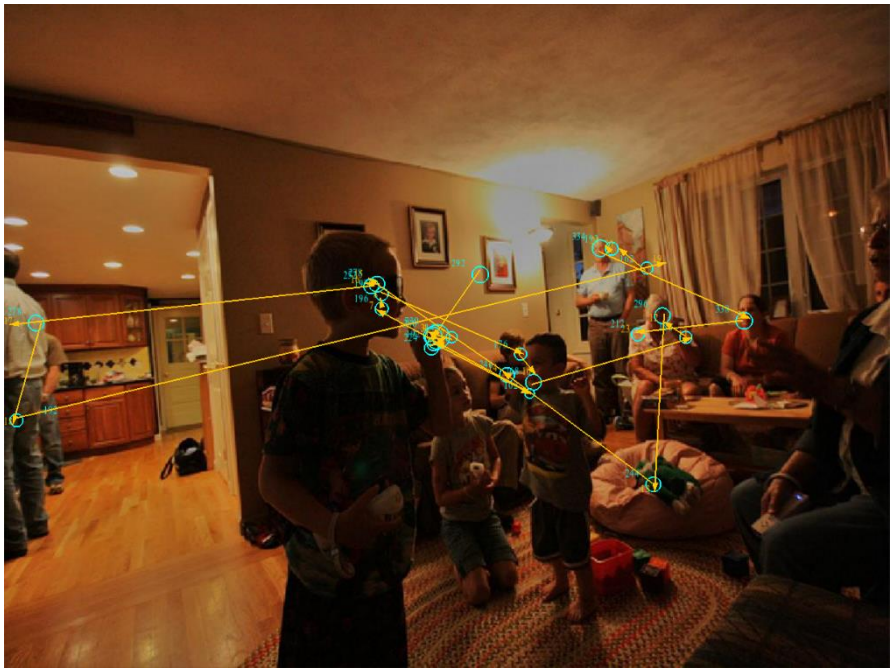


Task-Free Saccadic Scanpath



Task-Driven Saccadic Scanpath





1. 房间里有一群人在嬉闹
2. 晚饭过后一家人在客厅里享受悠闲的时光
3. 在一个灯光昏暗的房间里有许多小孩在玩耍
4. 屋子里一群孩子围在一起玩耍
5. 许多人在房间里聚会

Attention

Everyone knows what attention is. It is the taking possession by the mind, in clear and vivid form, of one out of what seem several simultaneously possible objects or trains of thought. Focalization, concentration, of consciousness are of its essence. It implies withdrawal from some things in order to deal effectively with others.

--- Williams James

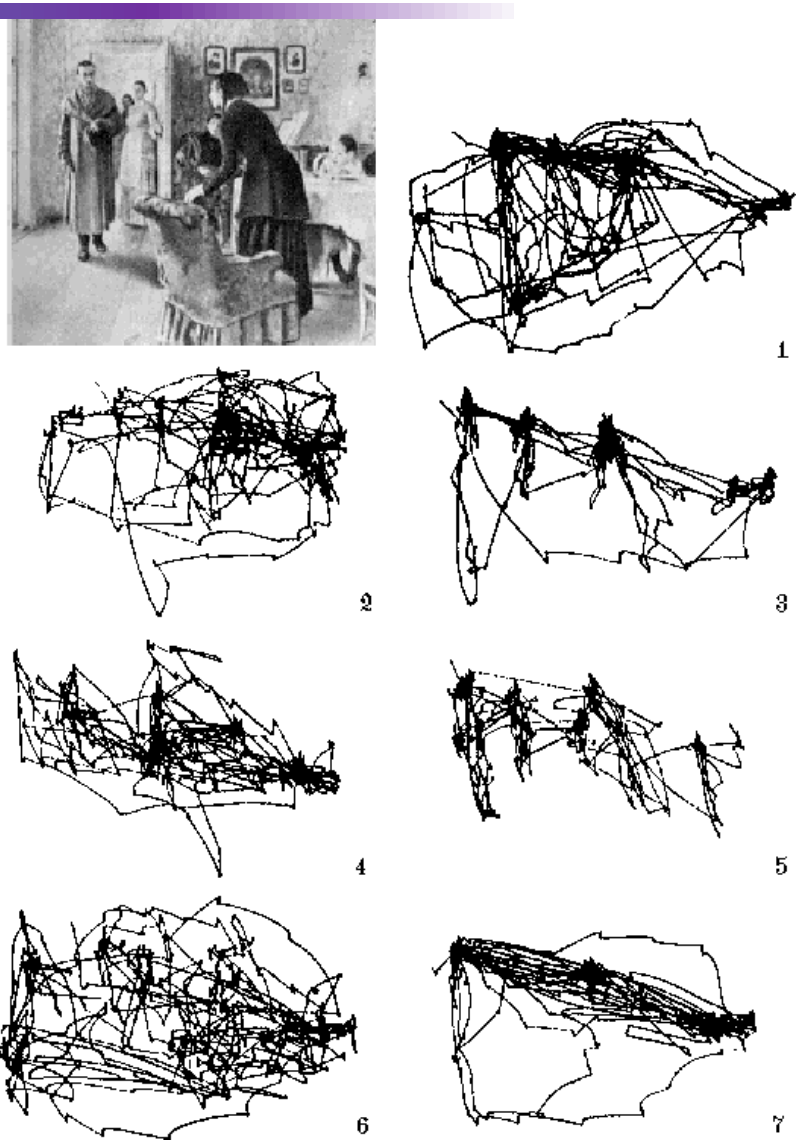
注意在人眼视觉信息处理过程中体现为信息的选择与过滤，从而减少向高级皮层传递的信息，避免冗余或者噪声信息对高级视觉任务的干扰，这种选择过程发生在视觉通道的多个阶段，而且这种选择过程既有各阶段视觉特征的刺激，也有高级皮层向下的反馈调节，对选择性注意的研究不仅有理论上的意义，而且有现实的迫切需求

Attention and Task

Yarbus demonstrated how eye movements changed depending on the question asked of the subject:

1. No question asked
2. Judge economic status
3. "What were they doing before the visitor arrived?"
4. "What clothes are they wearing?"
5. "Where are they?"
6. "How long is it since the visitor has seen the family?"
7. Estimate how long the "unexpected visitor" had been away from the family

Each recording lasted
3 minutes



Yarbus, A. L. (1967). **Eye Movements and Vision**. New York: Plenum.

Attention and Object Representation

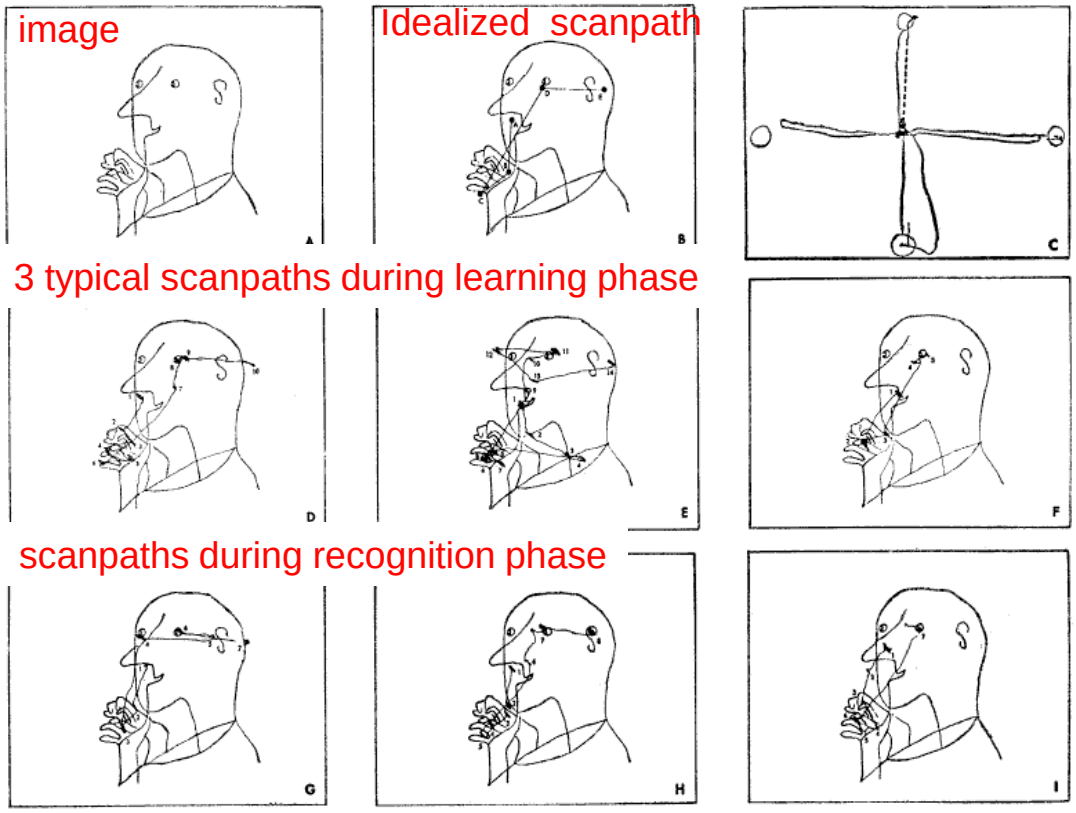
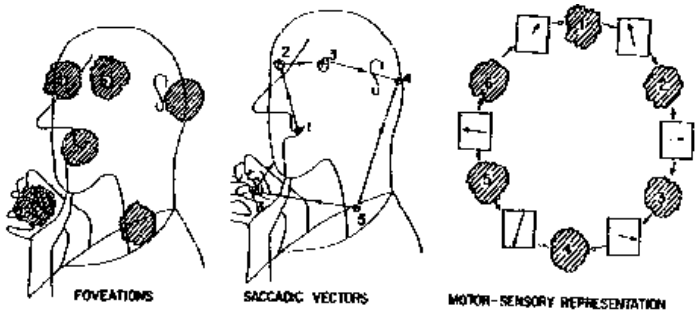


Fig. 1. Example of scanpaths in eye movements during learning and recognition phases (subject J.M.).

Reported a connection between the eye movement patterns observed during learning of a visual pattern and the subsequent viewing of that pattern.

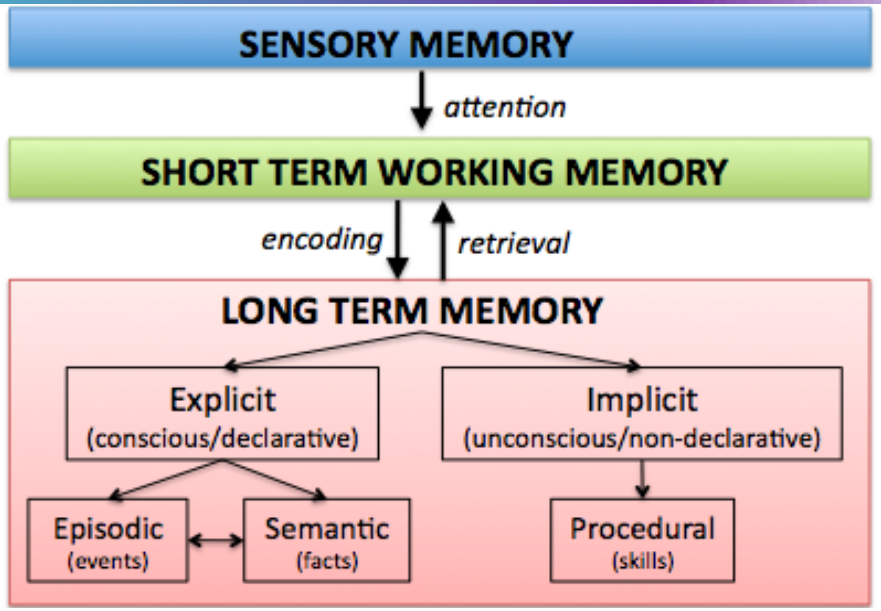
During learning, subjects followed a characteristic scanpath. When later presented with the pattern again, subjects usually followed a very similar scanpath for at least the first few fixations.

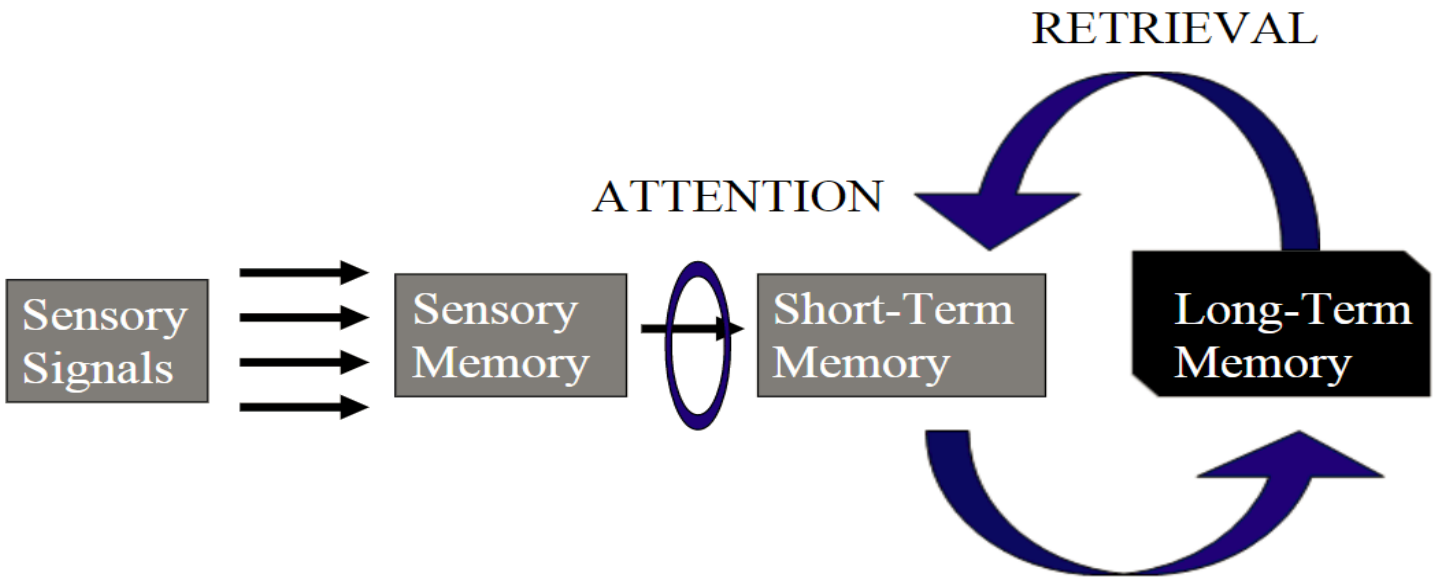
This suggested that the internal representation of a pattern in memory is a network of features, and thus attention shifts move from feature to feature.



Noton, D., Stark, L. (1971). Scanpaths in Eye Movements during Pattern Perception, *Science 171(3968)*, p308-311.

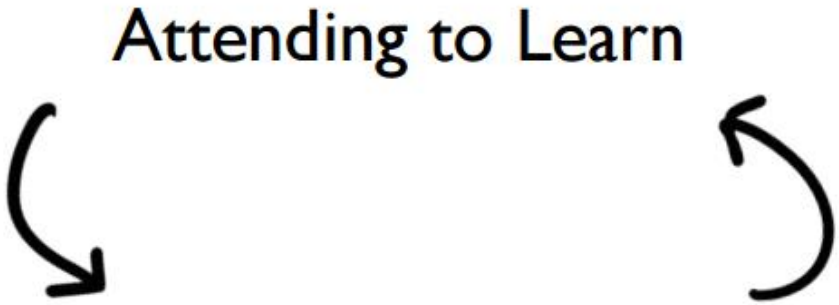
Attention and Memory

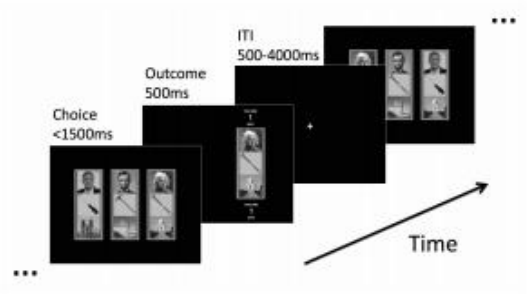




https://memorisingmusic.files.wordpress.com/2013/03/long-termmemory.png?w=640

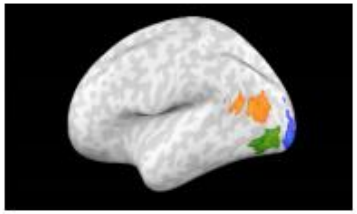
Attention and Learning



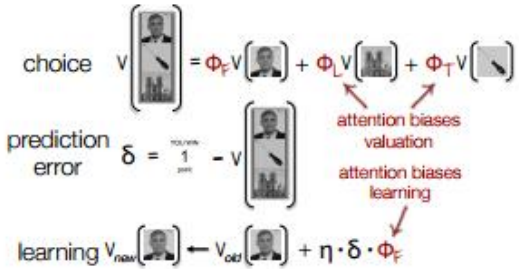


Behavioral Experiments

Eye-Tracking



fMRI



Computational Modeling

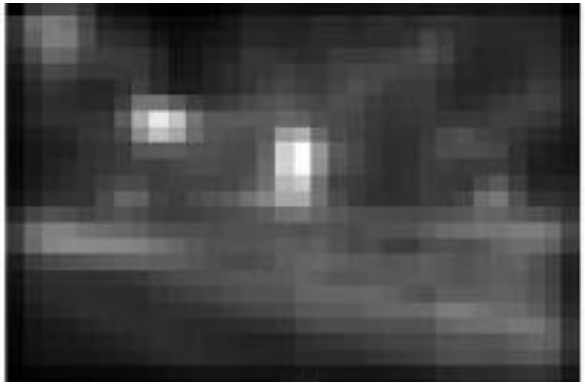
YC Leong, A Radulescu, R Daniel, V DeWoskin, & Y Niv (2017). Dynamic interaction between reinforcement learning and attention in multidimensional environments. Neuron.

Computational Models of Visual Attention

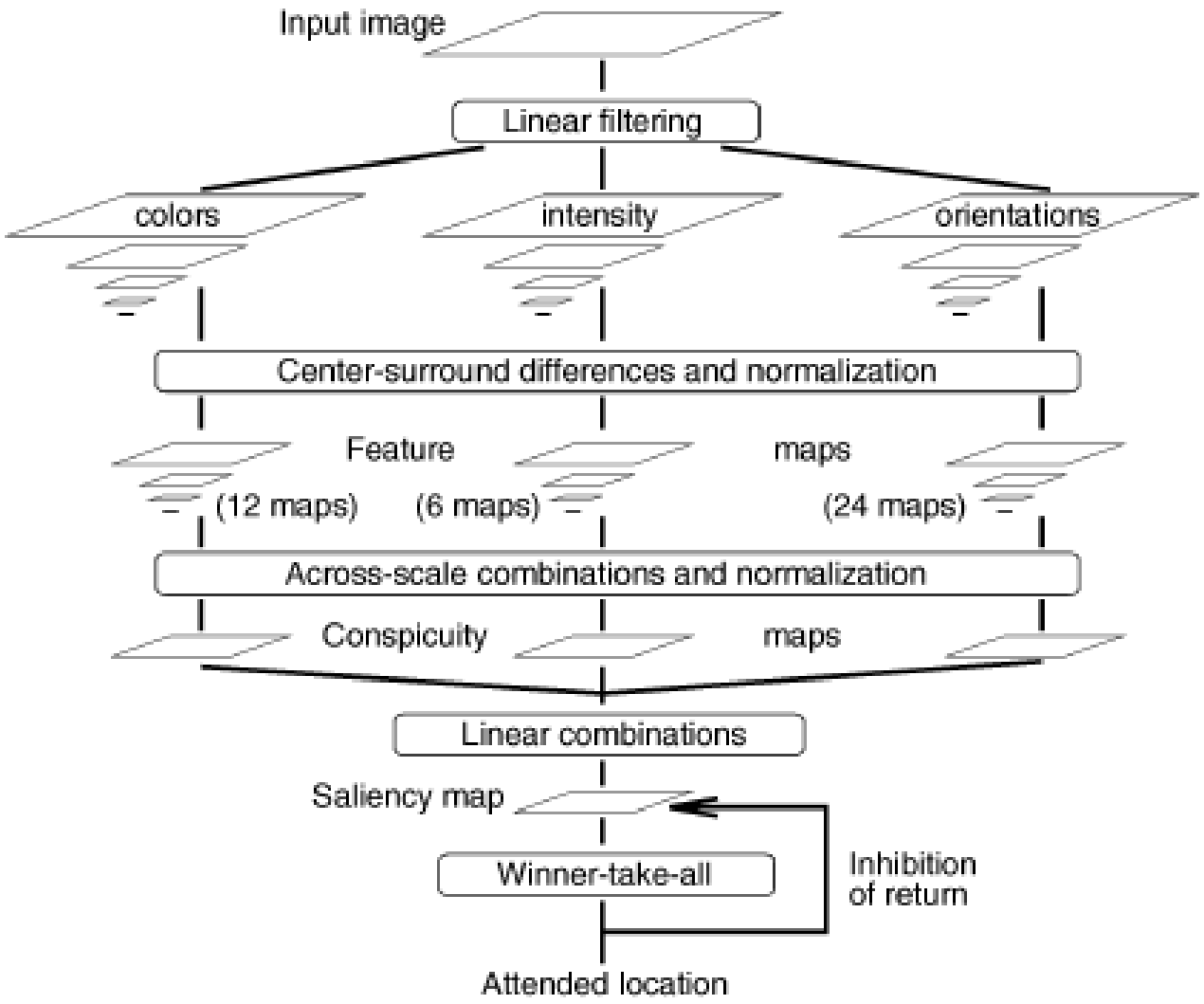
Visual Saliency

A bottom up saliency map in the primary visual cortex (V1 saliency)

Li Zhaoping@UCL

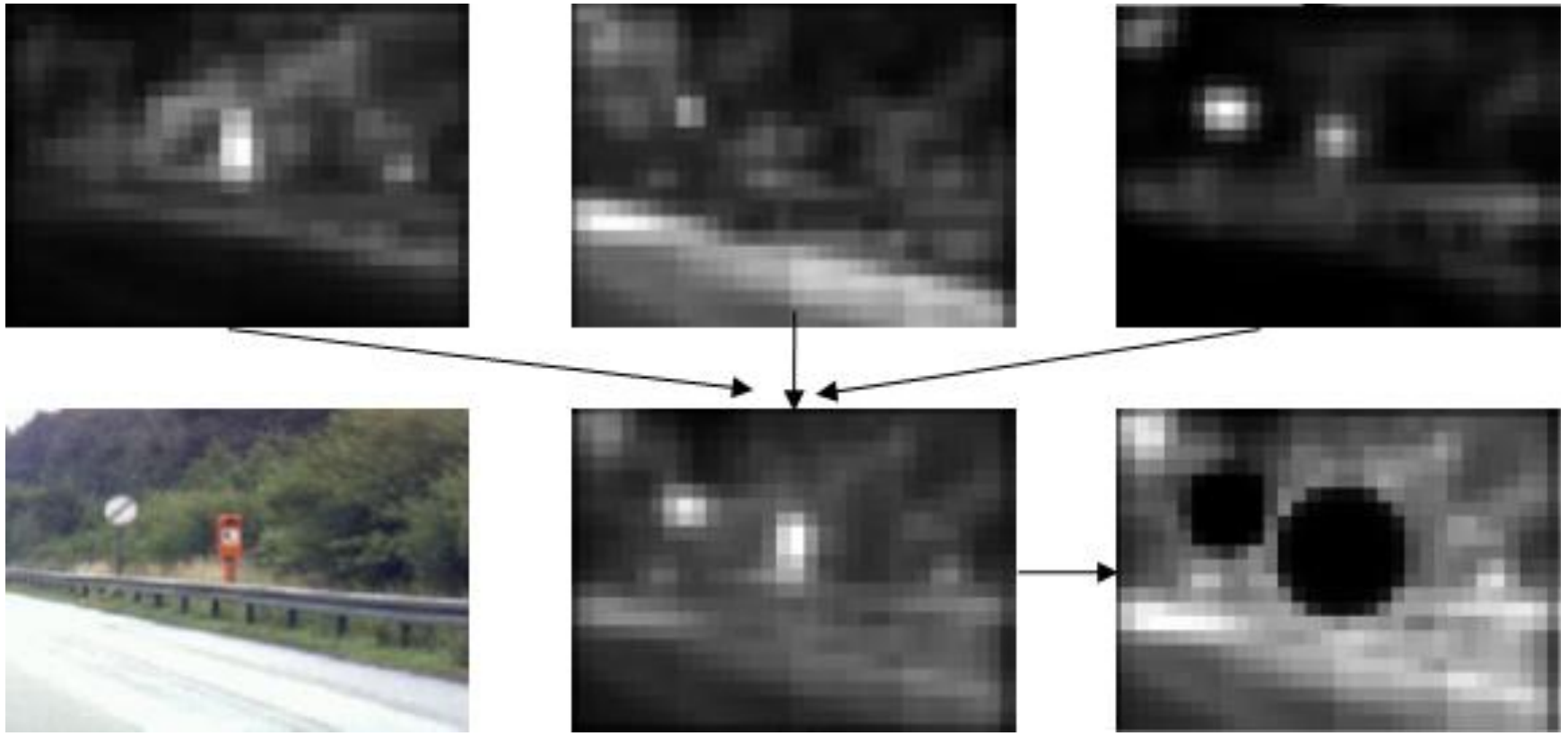


Center-Surround Difference



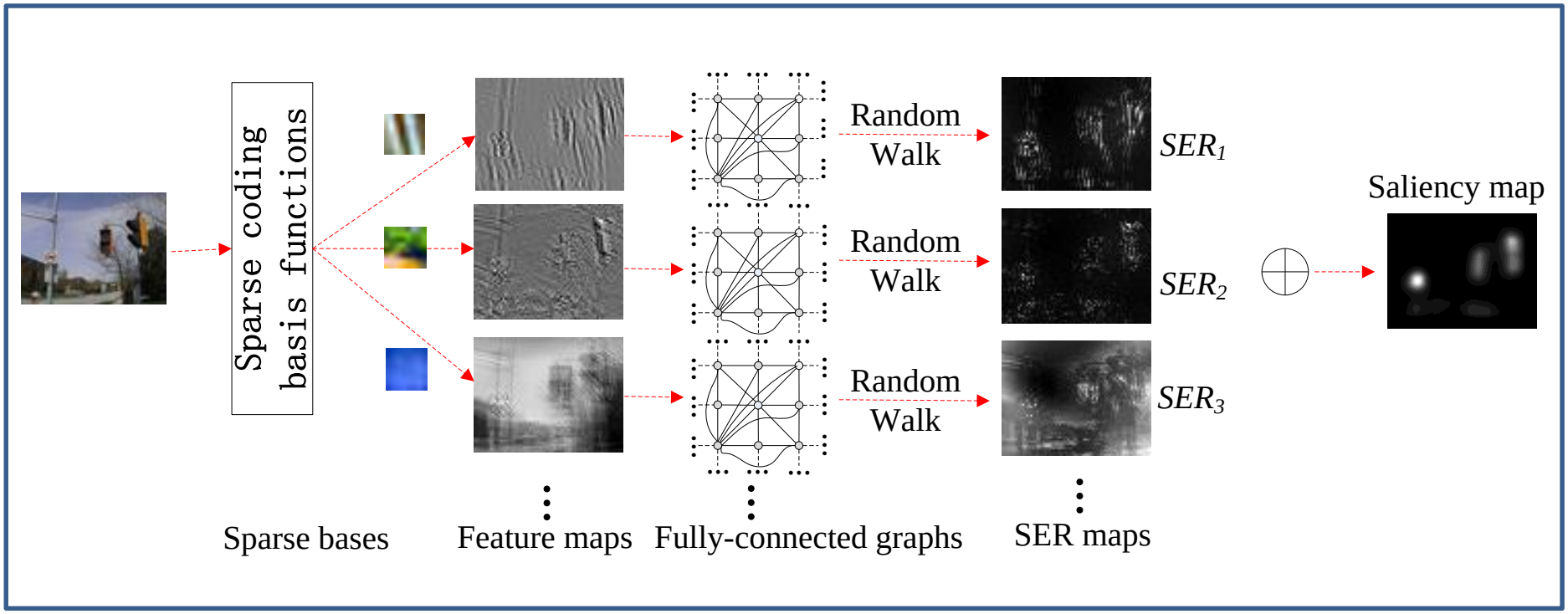
Itti et al. saliency-based visual attention for rapid scene analysis. TPAMI 1998

Center-Surround Difference



$$S = \frac{1}{3}\left(\mathcal{N}(\bar{I}) + \mathcal{N}(\bar{C}) + \mathcal{N}(\bar{O})\right)$$

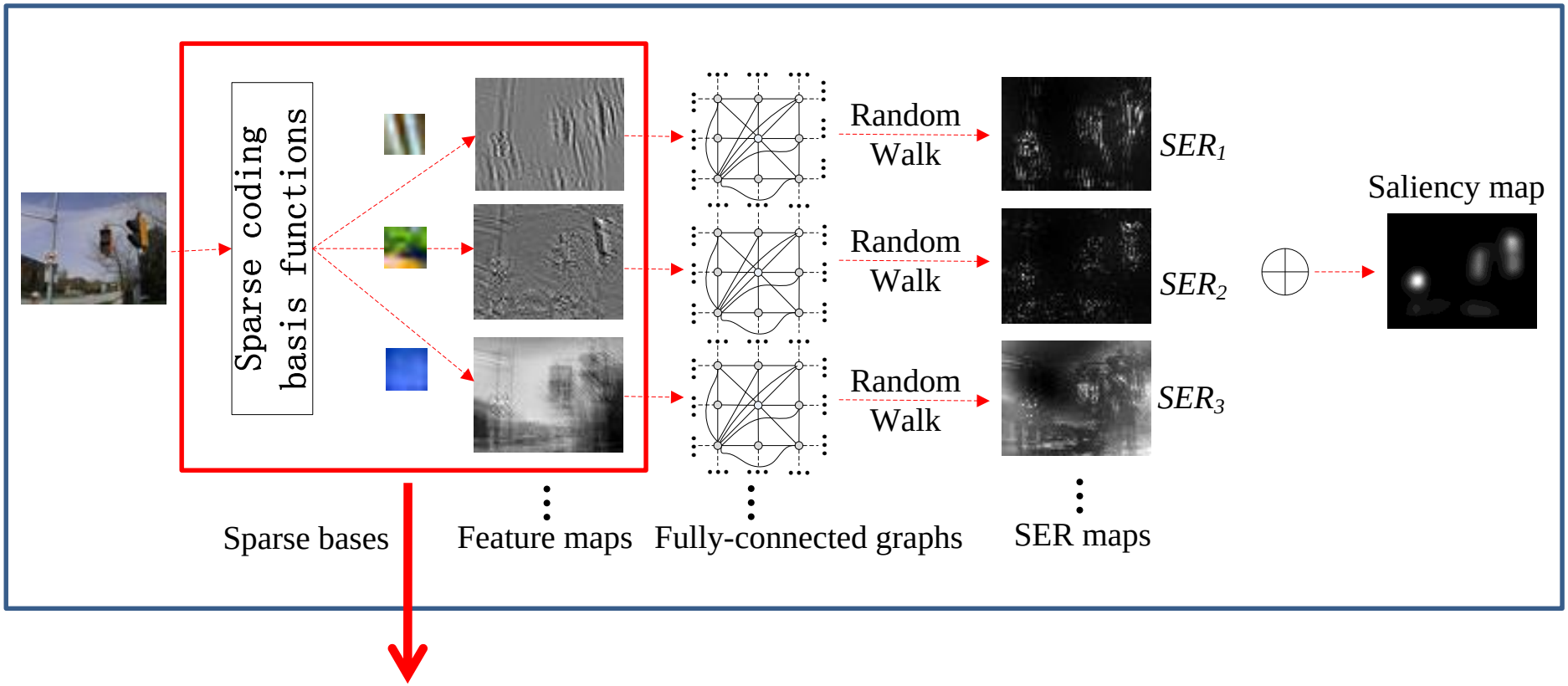
Site Entropy Rate



The procedure of computing the saliency map of an image

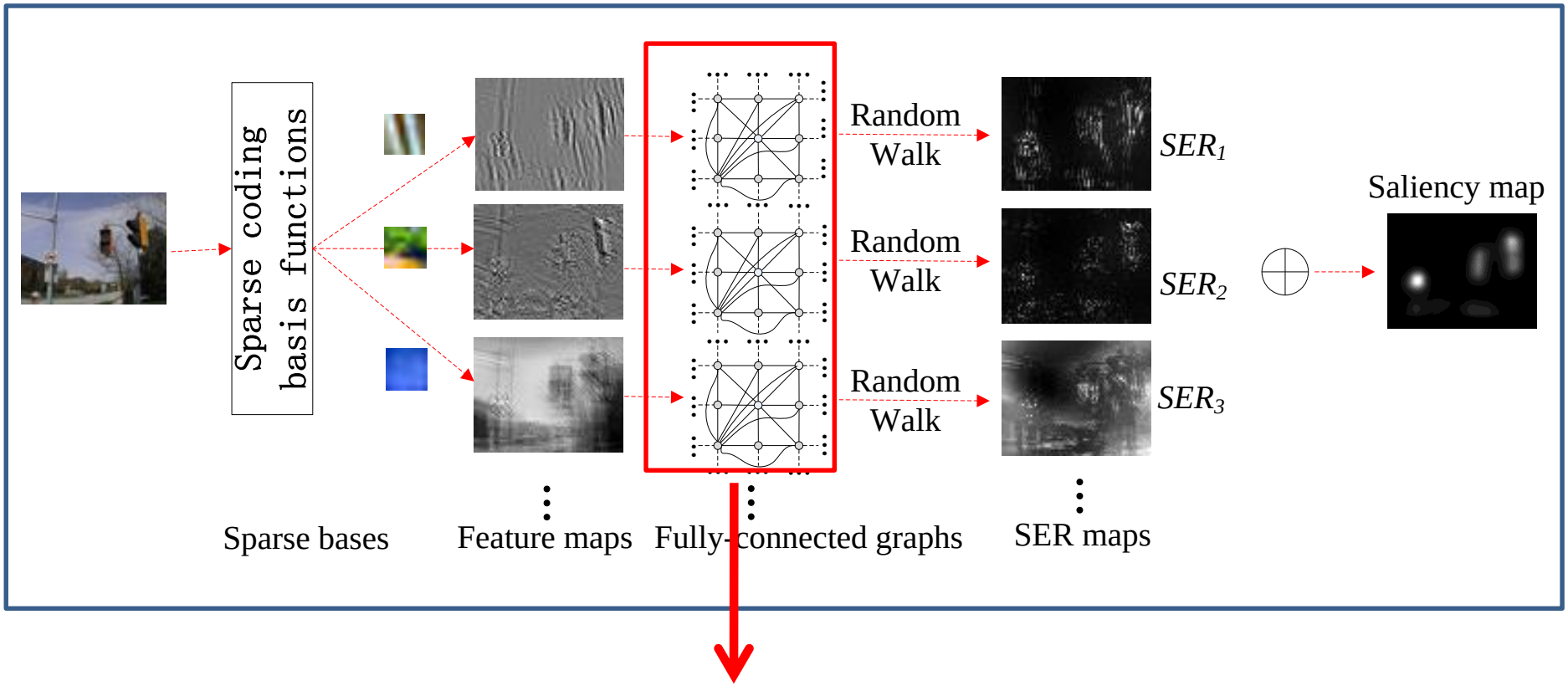
Wei Wang, Yizhou Wang, Qingming Huang, Wen Gao, "Measuring Visual Saliency by Site Entropy Rate", IEEE Computer Vision and Pattern Recognition, CVPR, San Francisco, Jun, 2010. (oral, google citation: 180)

Site Entropy Rate



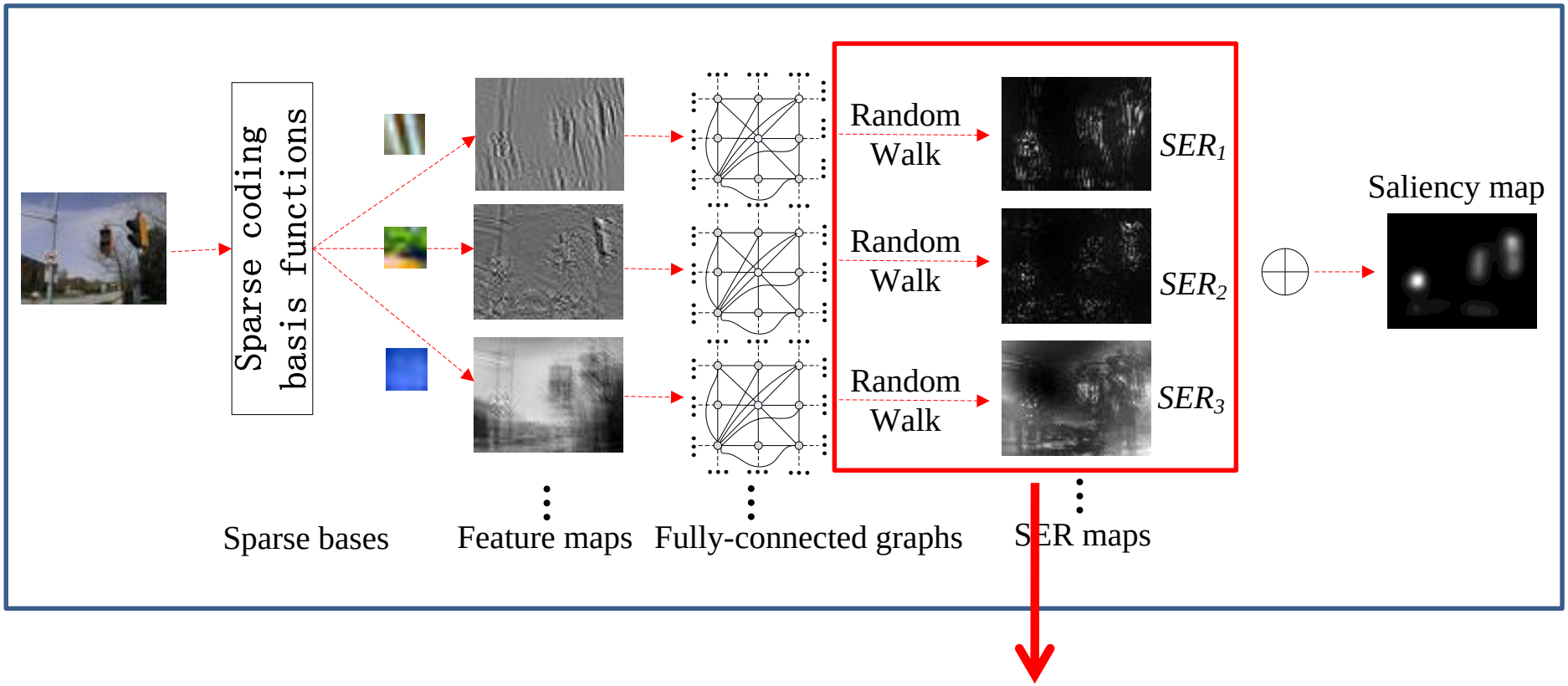
① Filter an input image into multi-band filter response maps using number of learned sparse coding basis functions.

Site Entropy Rate



② To simulate the cortical connectivity, a fully-connected graph representation is adopted for each feature map.

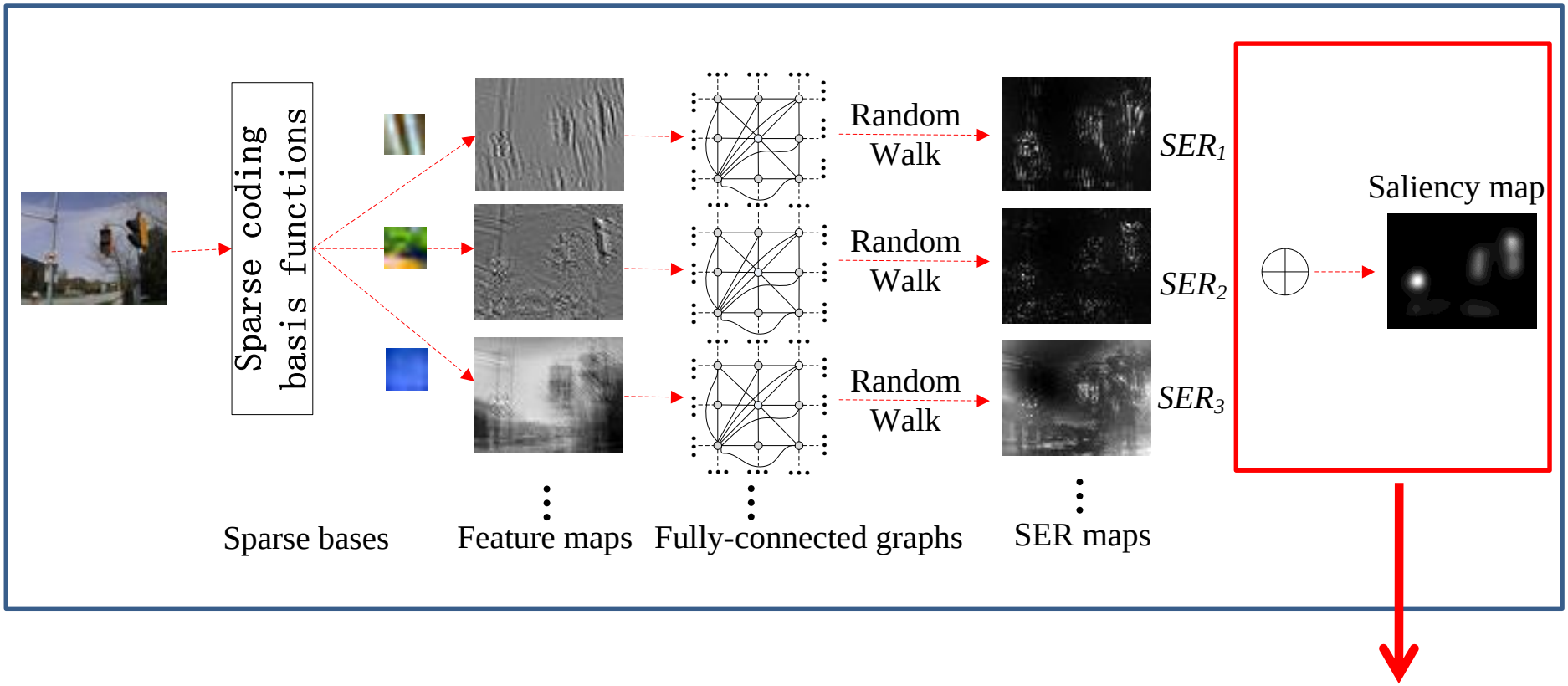
Site Entropy Rate



③ A random walk is adopted on each sub-band graph.

Site Entropy Rate (SER) is proposed to measure the average information from a node to all the others.

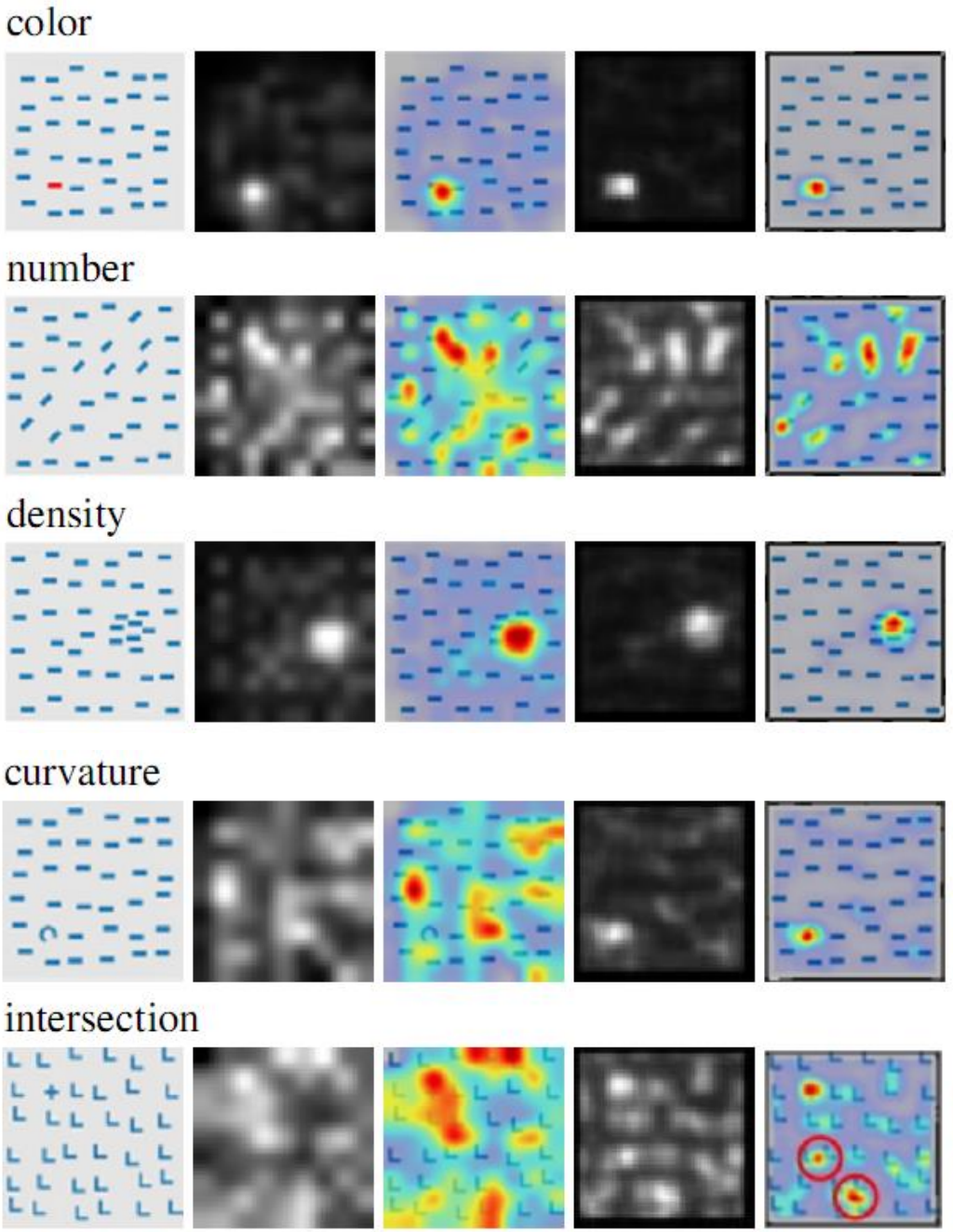
Site Entropy Rate



④ The saliency map is computed by summing over all the sub-band SER maps.

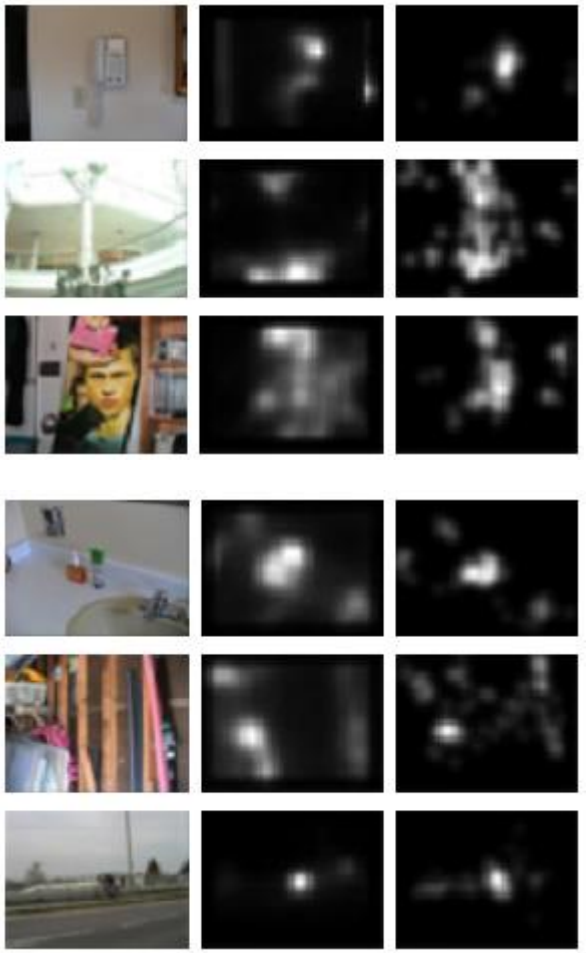
Experiments on Psychological Stimuli

■Comparison results of five psychological stimuli between our model and Itti et al.'s [PAMI1998]



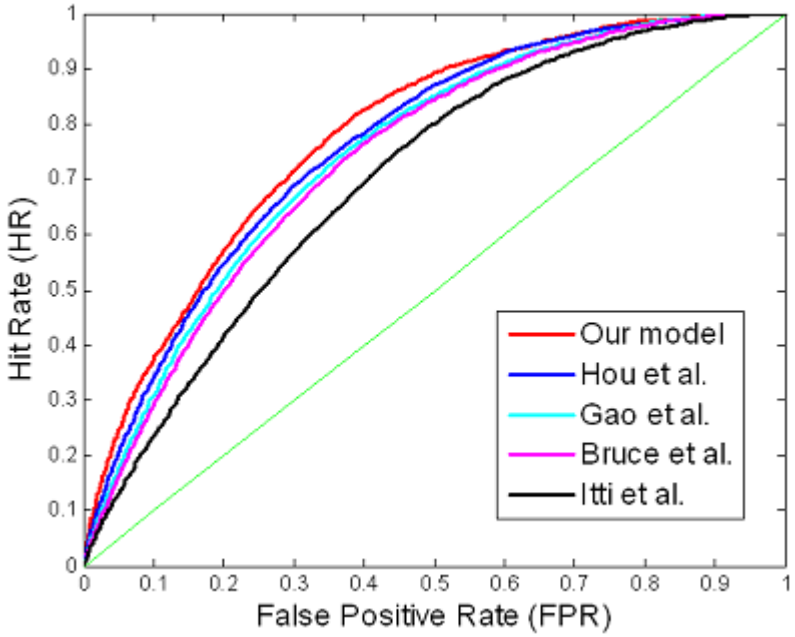
Experiments on Color Image Data Set

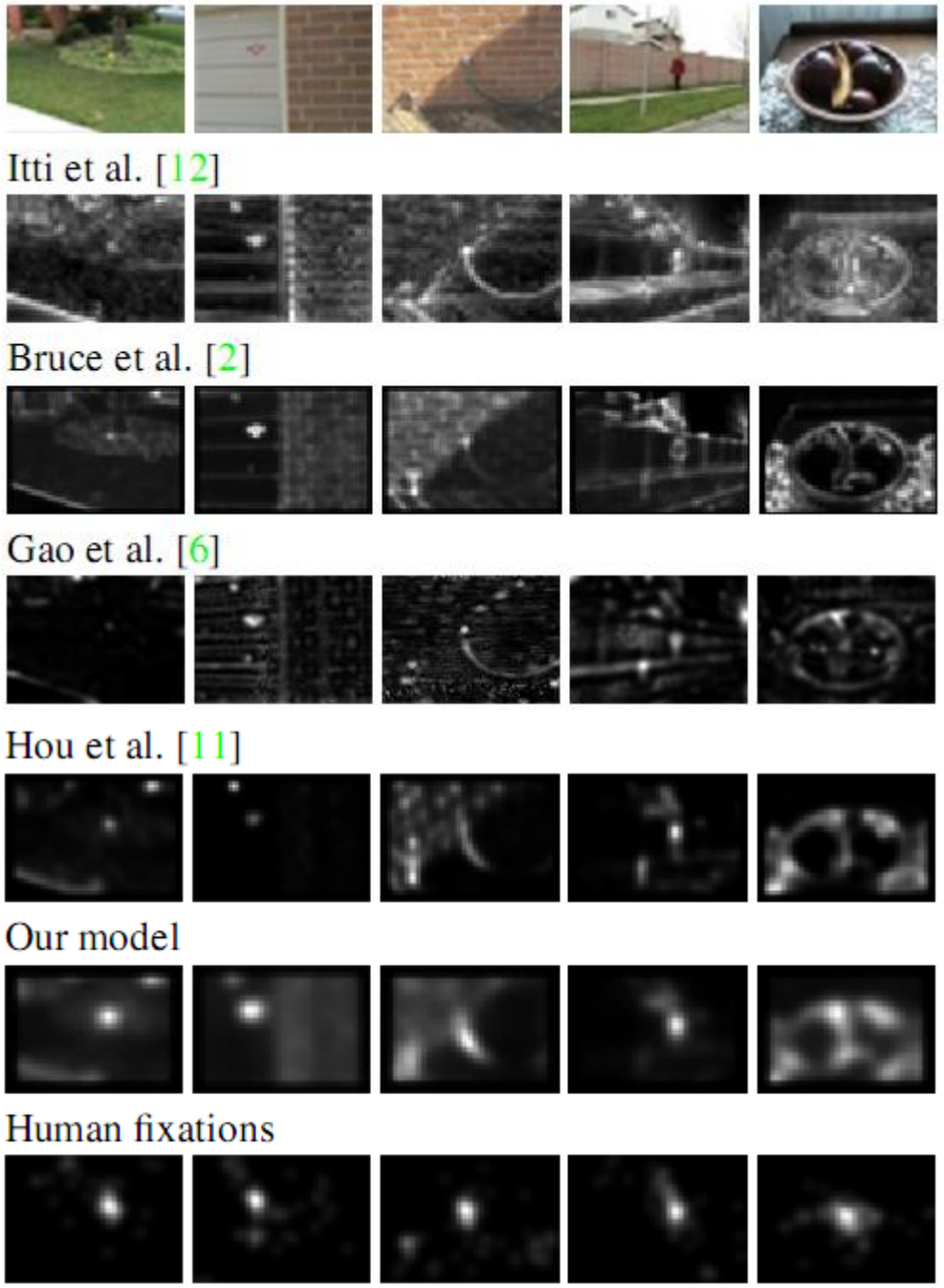
- N. Bruce's dataset [NIPS06]
  - 120 color images (indoor and outdoor scenes)
  - Eye fixations recorded from 20 subjects
- Evaluation method
  - quantitative comparison between human fixation density map and saliency map
  - ROC curve/area



Comparison Results

	ROC Area
Itti et al. [12]	0.7031
Bruce et al. [2]	0.7522
Gao et al. [6]	0.7644
Hou et al. [11]	0.7808
Our model	0.8049

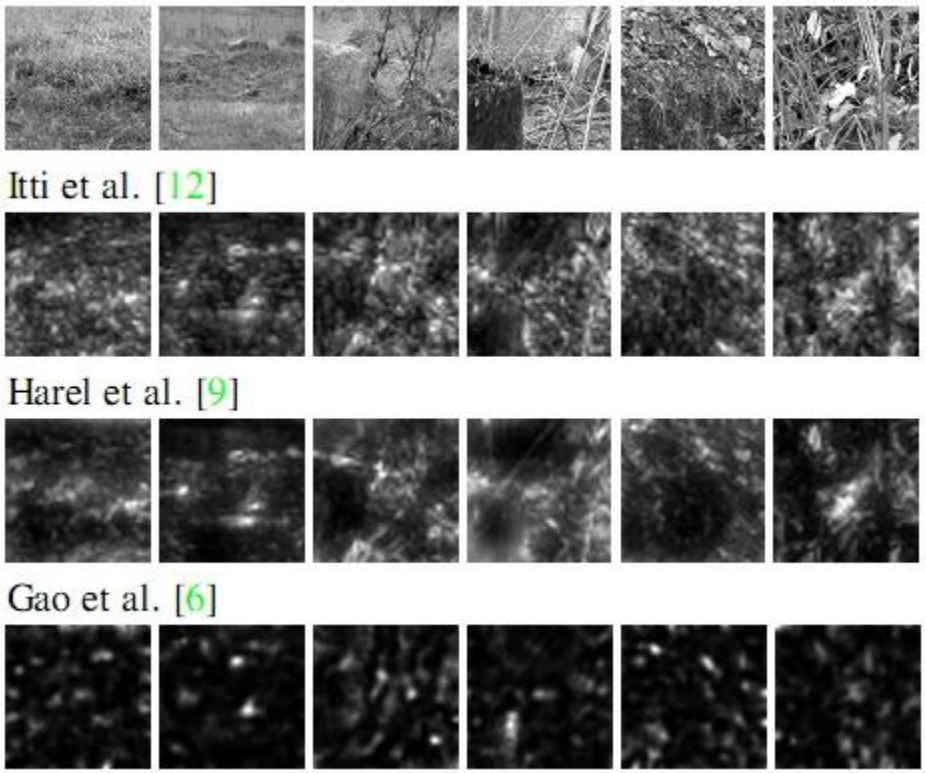


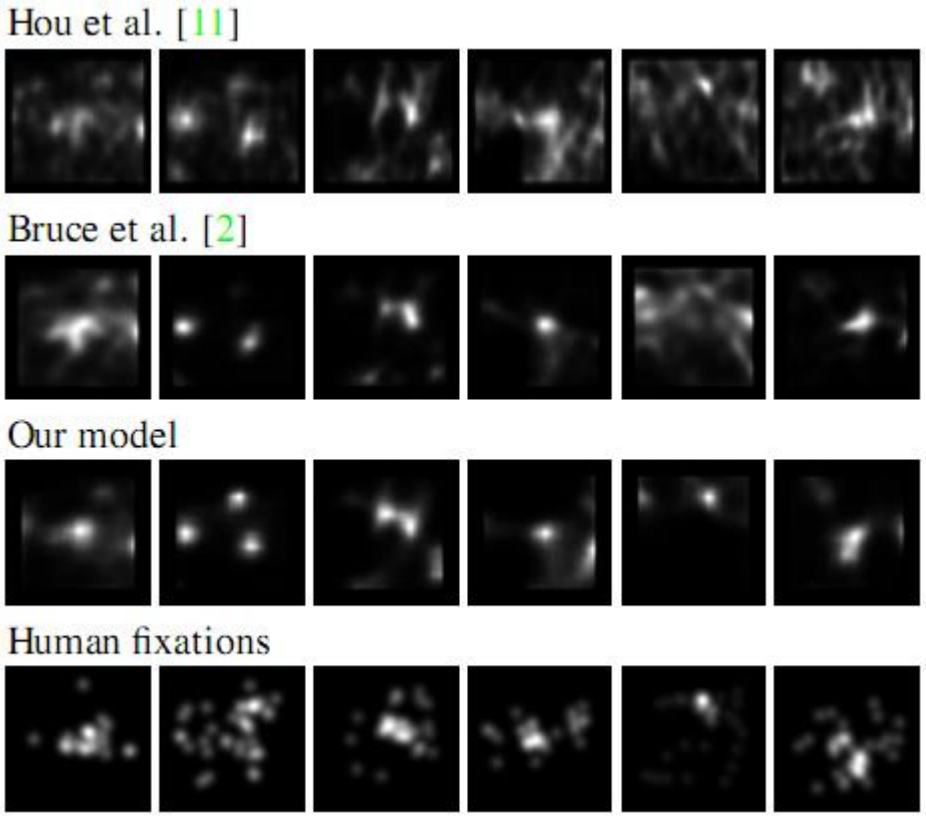


Experiments on Gray Image Data Set

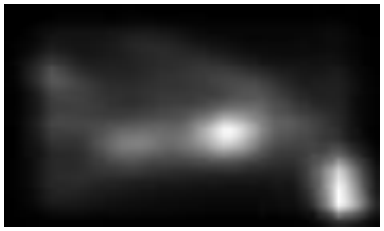
- W. Einhäuser et al's dataset [Vision Research 06]
  - 108 gray images (natural scenes)
  - Fixations from 7 subjects

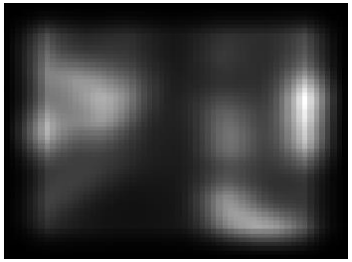
	ROC area
Harel et al. [9]	0.5028
Gao et al. [6]	0.5203
Itti et al. [12]	0.5241
Hou et al. [11]	0.6094
Bruce et al. [2]	0.6420
Our model	0.6537

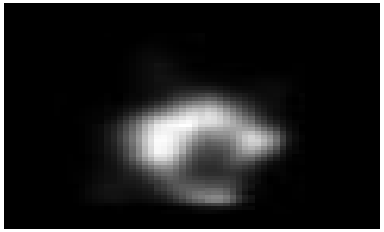


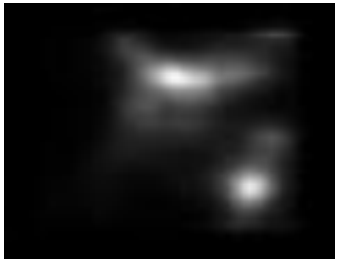


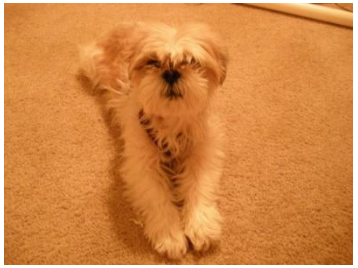
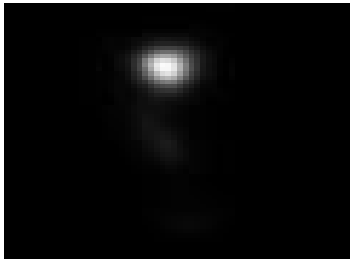
More Results

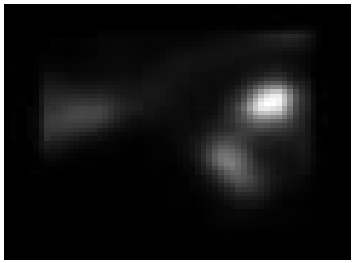


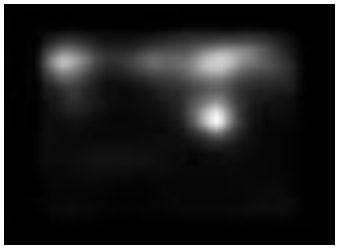


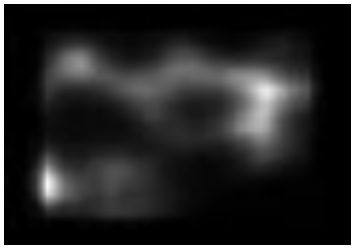


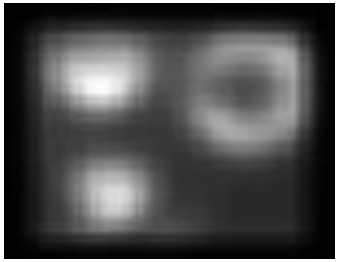


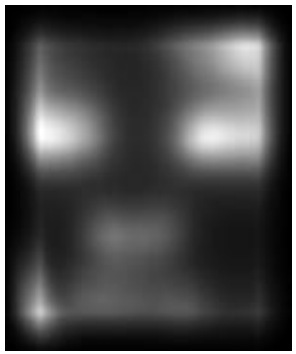



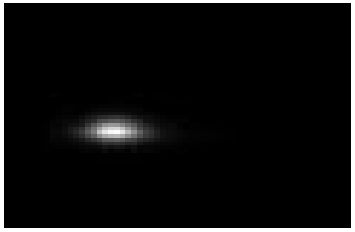


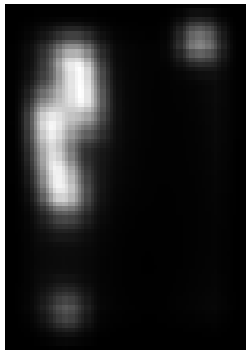










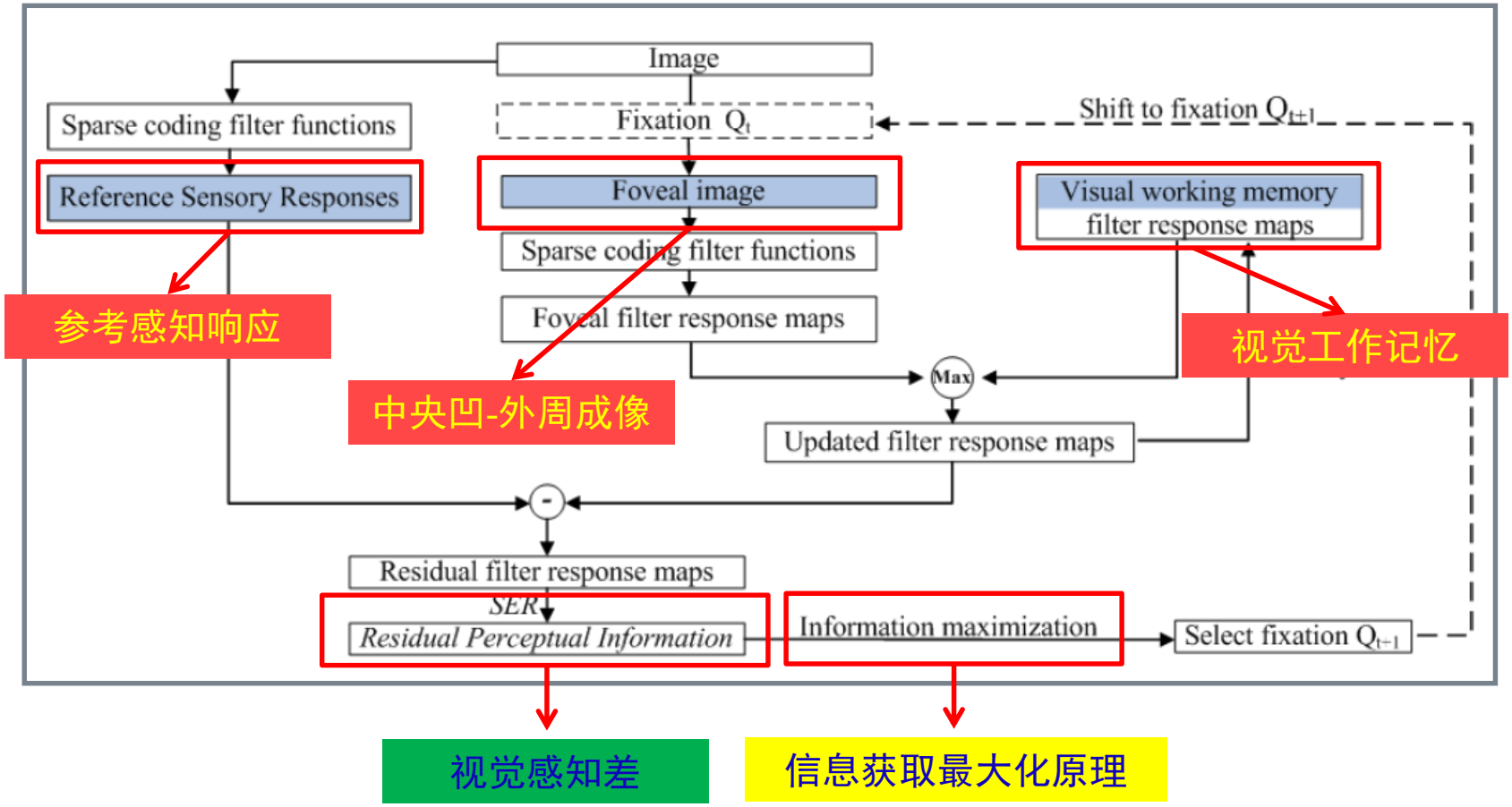


Saccadic Scanpath

Dynamic Visual Attention

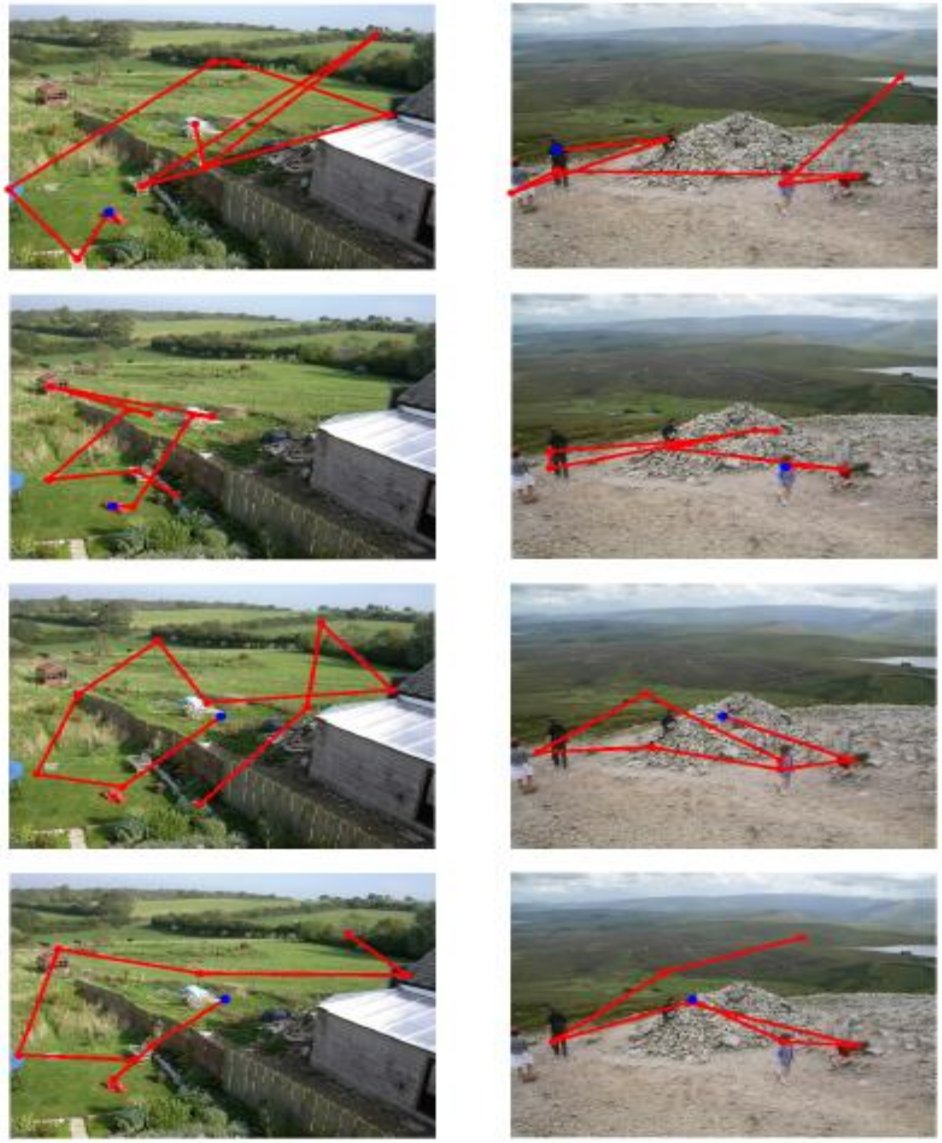
Computational models：

One principle + One mechanism + Three factors



Wei Wang, Cheng Chen, Yizhou Wang, Tingting Jiang, Fang Fang, Yuan Yao, Simulating Human Saccadic Scanpath on Natural Images, in Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, CVPR, Colorado Springs, USA, Jun.21-25, 2011

Dynamic Visual Attention



(a) Itti et al. (PAMI 1998)

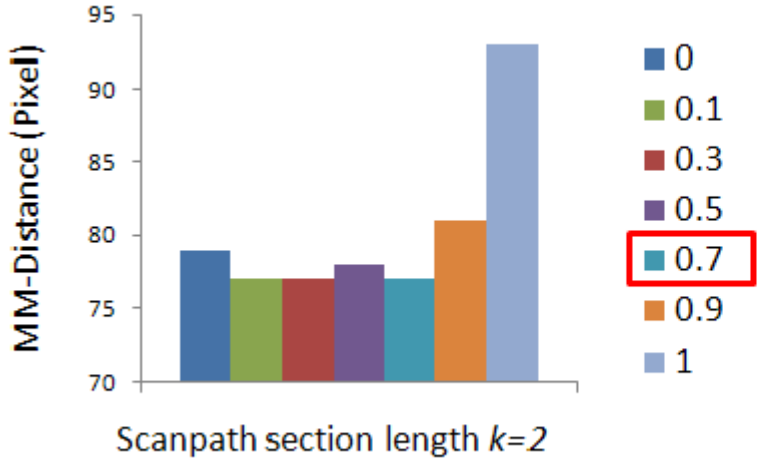
(b) Wang et al. (CVPR 2010)

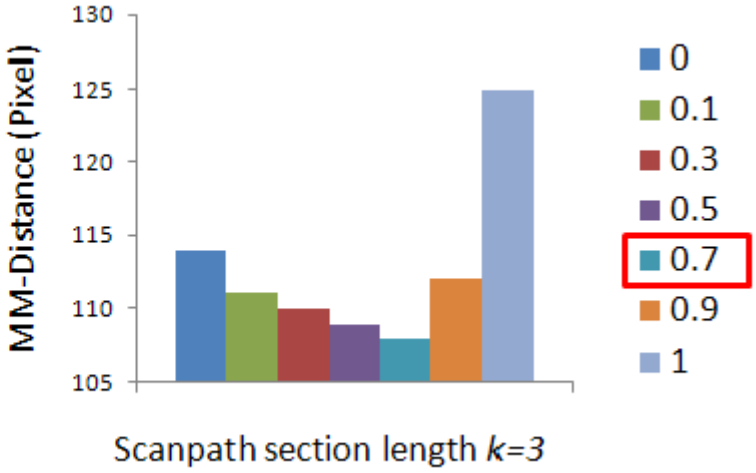
(c) Our model

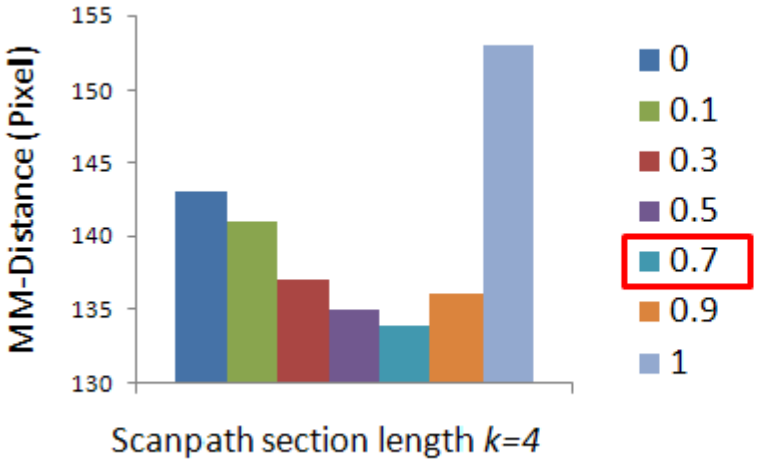
(d) Eye tracking data

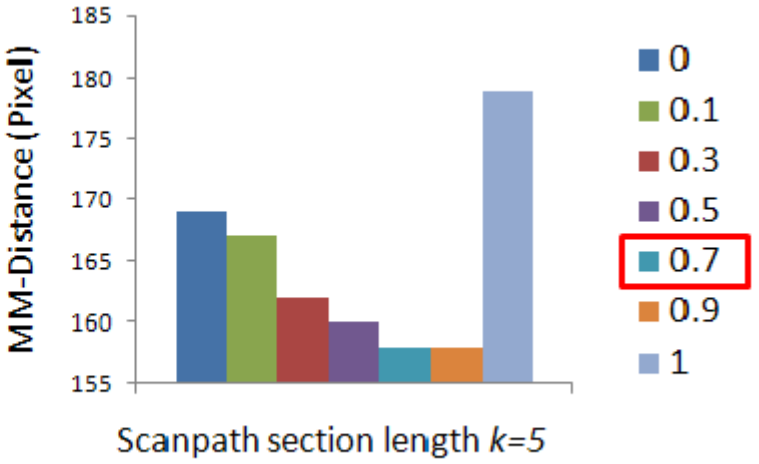
Dynamic Visual Attention

Forgetting factor in working memory









Neural Attention Model

2014

On Learning Where To Look

Marc'Aurelio Ranzato
Google Inc.*
Mountain View CA, U.S.A.

2014

On Learning Where To Look

-

Recurrent Models of Visual Attention

Volodymyr Mnih Nicolas Heess Alex Graves Koray Kavukcuoglu
Google DeepMind

2014

On Learning Where To Look

Recurrent Models of Visual Attention

Deep Networks with Internal Selective Attention through Feedback Connections

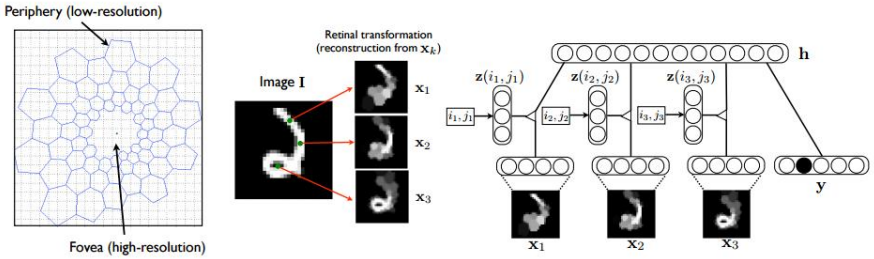
A version of this paper was submitted to ICML 2014 on 31-01-2014.

Marijn Stollenga marijn@idsia.ch*,
Jonathan Masci jonathan@idsia.ch*,
Faustino Gomez tino@idsia.ch, and
Juergen Schmidhuber juergen@idsia.ch

- They all aim to solve two major challenges
  - scalability
  - variability of appearance

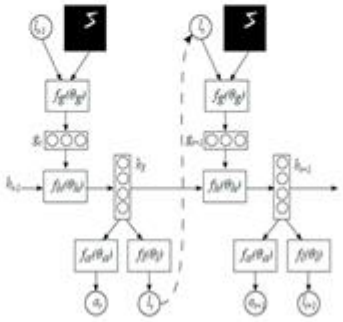
Learning to combine foveal glimpses with a third-order Boltzmann machine

Hugo Larochelle and Geoffrey Hinton
Department of Computer Science, University of Toronto
6 King's College Rd, Toronto, ON, Canada, M5S 3G4
{larocheh,hinton}@cs.toronto.edu



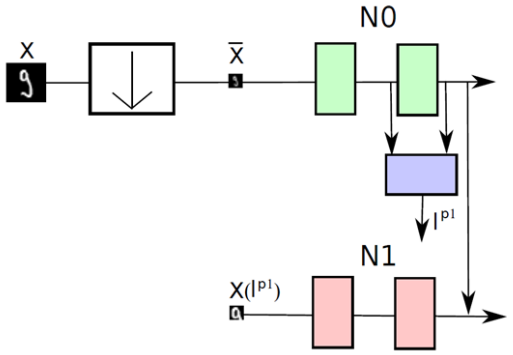
Recurrent Models of Visual Attention

Volodymyr Mnih **Nicolas Heess** **Alex Graves** **Koray Kavukcuoglu**
Google DeepMind



On Learning Where To Look

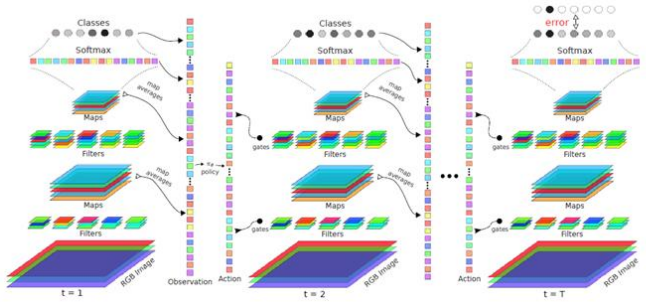
Marc'Aurelio Ranzato
Google Inc.*
Mountain View CA, U.S.A.



Deep Networks with Internal Selective Attention through Feedback Connections

A version of this paper was submitted to ICML 2014 on 31-01-2014.

Marijn Stollenga marijn@idsia.ch*,
Jonathan Masci jonathan@idsia.ch*,
Faustino Gomez tino@idsia.ch, and
Juergen Schmidhuber juergen@idsia.ch



2015

MULTIPLE OBJECT RECOGNITION WITH VISUAL ATTENTION

Jimmy Lei Ba*
University of Toronto
jimmy@psi.utoronto.ca

Volodymyr Mnih
Google DeepMind
vmnih@google.com

Koray Kavukcuoglu
Google DeepMind
korayk@google.com

ABSTRACT

We present an attention-based model for recognizing multiple objects in images. The proposed model is a deep recurrent neural network trained with reinforcement learning to attend to the most relevant regions of the input image. We show that the model learns to both localize and recognize multiple objects despite being given only class labels during training. We evaluate the model on the challenging task of transcribing house number sequences from Google Street View images and show that it is both more accurate than the state-of-the-art convolutional networks and uses fewer parameters and less computation.

2015

MULTIPLE OBJECT RECOGNITION WITH VISUAL ATTENTION

Spatial Transformer Networks

Max Jaderberg **Karen Simonyan** **Andrew Zisserman** **Koray Kavukcuoglu**

Google DeepMind, London, UK
{jaderberg,simonyan,zisserman,korayk}@google.com

Abstract

Convolutional Neural Networks define an exceptionally powerful class of models, but are still limited by the lack of ability to be spatially invariant to the input data in a computationally and parameter efficient manner. In this work we introduce a new learnable module, the *Spatial Transformer*, which explicitly allows the spatial manipulation of data within the network. This differentiable module can be inserted into existing convolutional architectures, giving neural networks the ability to actively spatially transform feature maps, conditional on the feature map itself, without any extra training supervision or modification to the optimisation process. We show that the use of spatial transformers results in models which learn invariance to translation, scale, rotation and more generic warping, resulting in state-of-the-art performance on several benchmarks, and for a number of classes of transformations.

2015

MULTIPLE OBJECT RECOGNITION WITH VISUAL ATTENTION

Spatial Transformer Networks

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

Kelvin Xu KELVIN.XU@UMONTREAL.CA
Jimmy Lei Ba JIMMY@PSI.UTORONTO.CA
Ryan Kiros RKIROS@CS.TORONTO.EDU
Kyunghyun Cho KYUNGHYUN.CHO@UMONTREAL.CA
Aaron Courville AARON.COURVILLE@UMONTREAL.CA
Ruslan Salakhutdinov RSALAKHU@CS.TORONTO.EDU
Richard S. Zemel ZEMEL@CS.TORONTO.EDU
Yoshua Bengio FIND-ME@THE.WEB

Abstract

Inspired by recent work in machine translation and object detection, we introduce an attention

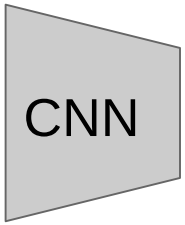
Figure 1. Our model learns a words/image alignment. The visualized attentional maps (3) are explained in section 3.1 & 5.4

Recall: RNN for Captioning

Image:
H x W x 3

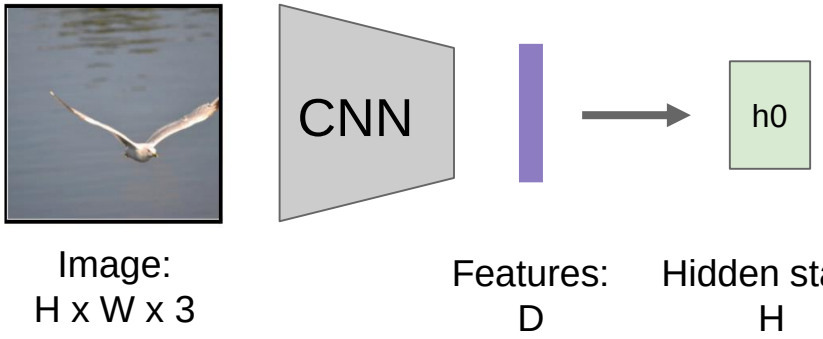
Recall: RNN for Captioning

Image:
H x W x 3

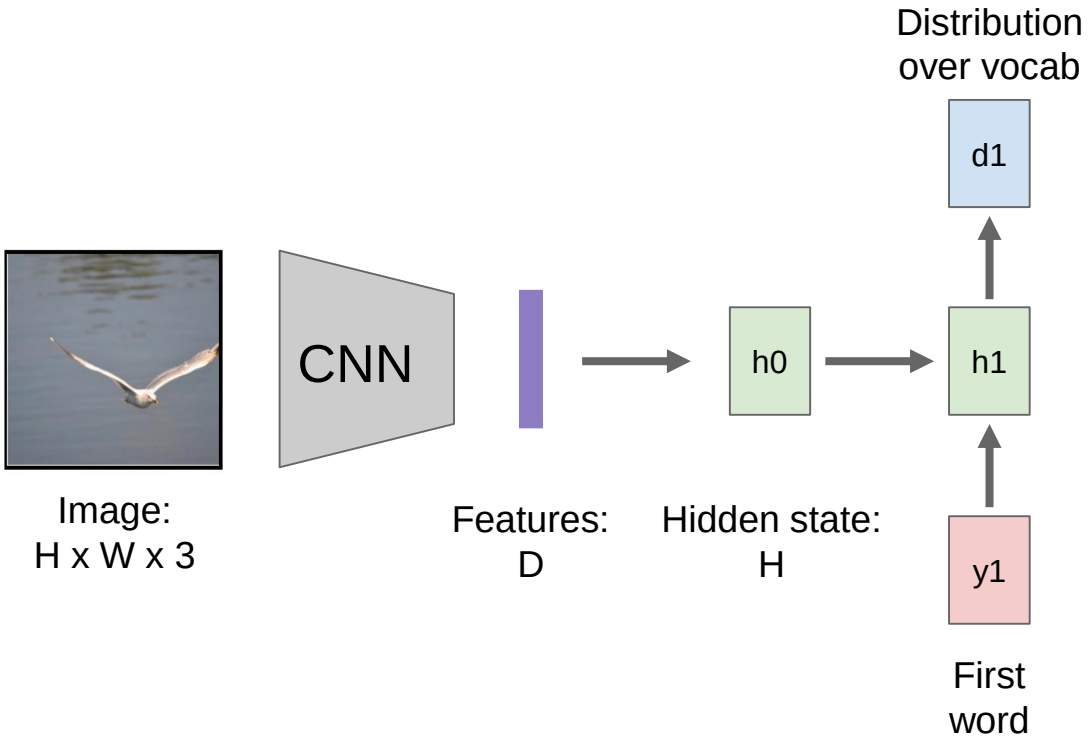


Features:
D

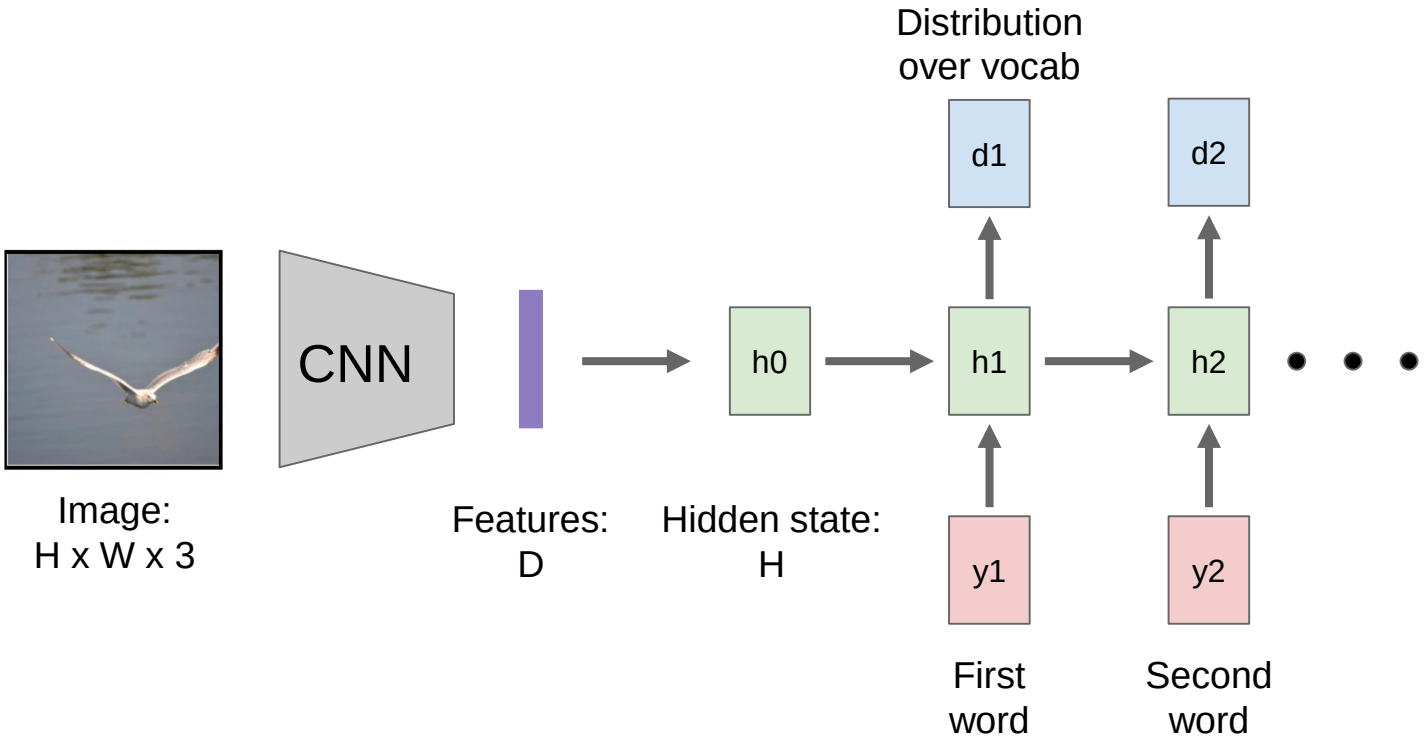
Recall: RNN for Captioning



Recall: RNN for Captioning



Recall: RNN for Captioning



Recall: RNN for Captioning

Image:
H x W x 3

CNN

Features:
D

h0

Hidden state:
H

Distribution
over vocab

d1

h1

y1

First
word

d2

h2

y2

Second
word

RNN only looks at
whole image, once

Recall: RNN for Captioning

Image:
H x W x 3

CNN

Features:
D

h0

Hidden state:
H

Distribution
over vocab

d1

h1

y1

First
word

d2

h2

y2

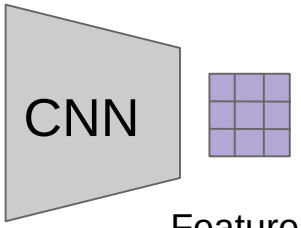
Second
word

RNN only looks at
whole image, once

What if the RNN
looks at different
parts of the image
at each timestep?

Soft Attention for Captioning

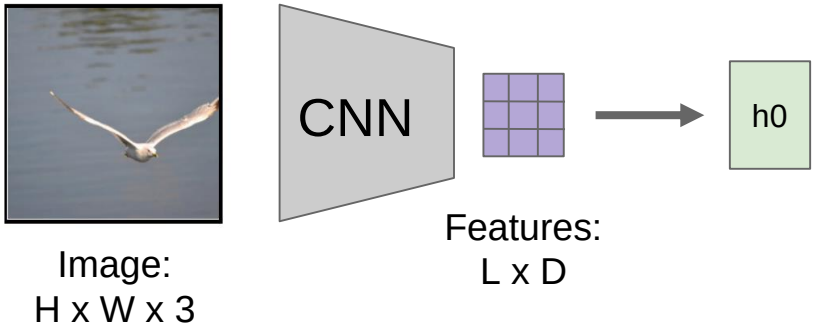
Image:
H x W x 3



Features:
L x D

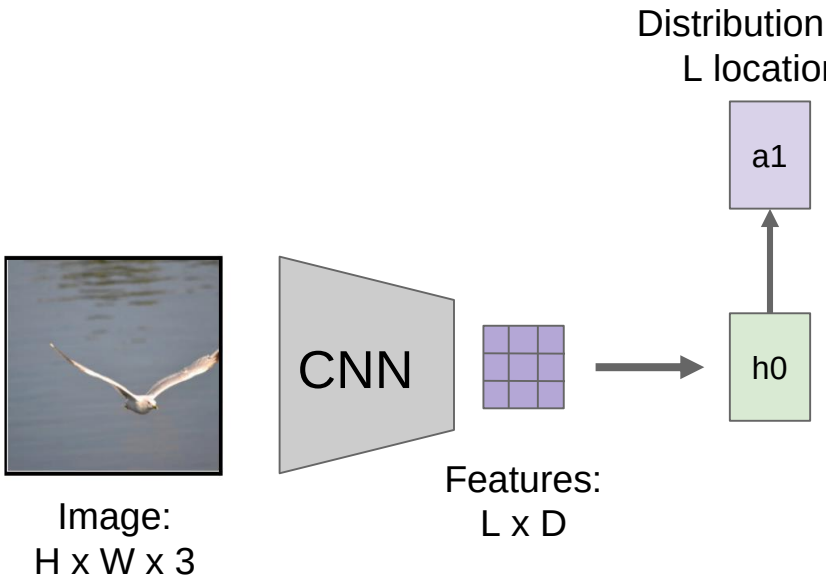
Xu et al, “Show, Attend and Tell: Neural Image Caption Generation with Visual Attention”, ICML 2015

Soft Attention for Captioning



Xu et al, “Show, Attend and Tell: Neural Image Caption Generation with Visual Attention”, ICML 2015

Soft Attention for Captioning



Xu et al, “Show, Attend and Tell: Neural Image Caption Generation with Visual Attention”, ICML 2015

Soft Attention for Captioning

Image:
H x W x 3

CNN

Features:
L x D

Distribution over
L locations

a1

h0

Weighted
features: D

z1

Weighted
combination
of features

Xu et al, “Show, Attend and Tell: Neural Image Caption Generation with Visual Attention”, ICML 2015

Soft Attention for Captioning

Image:
H x W x 3

CNN

Features:
L x D

Distribution over
L locations

a1

h0

h1

z1

y1

Weighted
features: D

Weighted
combination
of features

First word

Xu et al, “Show, Attend and Tell: Neural Image Caption Generation with Visual Attention”, ICML 2015

Soft Attention for Captioning

Image:
H x W x 3

CNN

Features:
L x D

Distribution over
L locations

Distribution
over vocab

a1

a2

d1

h0

h1

Weighted
features: D

z1

y1

Weighted
combination
of features

First word

Xu et al, “Show, Attend and Tell: Neural Image Caption Generation with Visual Attention”, ICML 2015

Soft Attention for Captioning

Image:
H x W x 3

CNN

Features:
L x D

Distribution over
L locations

Distribution
over vocab

a1

a2

d1

h0

h1

Weighted
features: D

z1

y1

z2

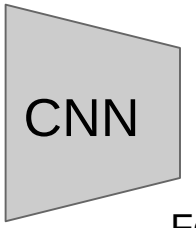
Weighted
combination
of features

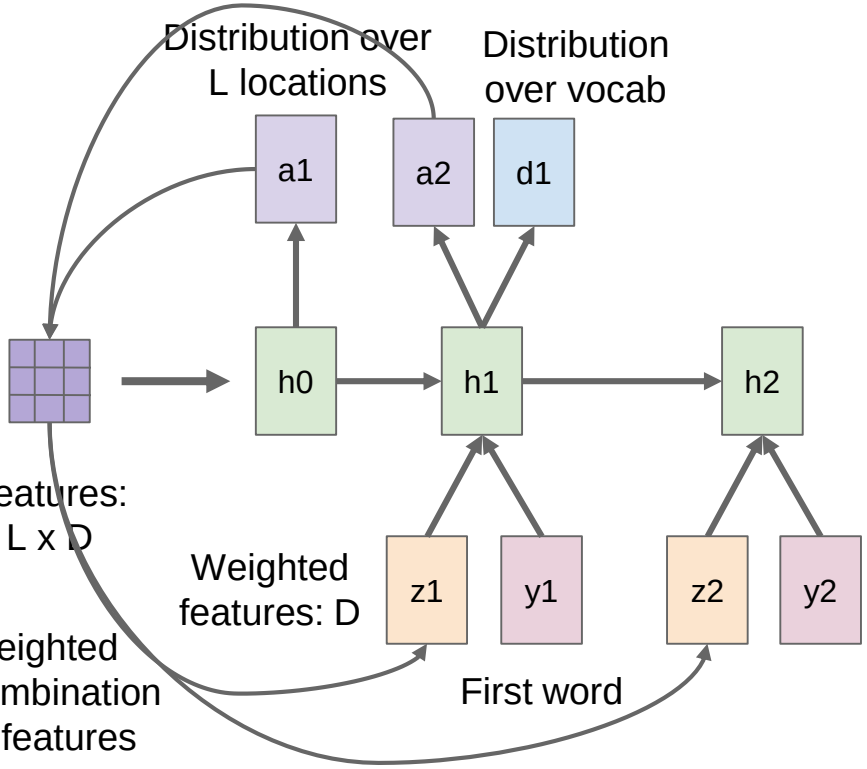
First word

Xu et al, “Show, Attend and Tell: Neural Image Caption Generation with Visual Attention”, ICML 2015

Soft Attention for Captioning

Image:
H x W x 3





Xu et al, “Show, Attend and Tell: Neural Image Caption Generation with Visual Attention”, ICML 2015

Soft Attention for Captioning

Image:
H x W x 3

CNN

Features:
L x D

Distribution over
L locations

Distribution
over vocab

a1 a2 d1 a3 d2

h0 h1 h2

z1 y1 z2 y2

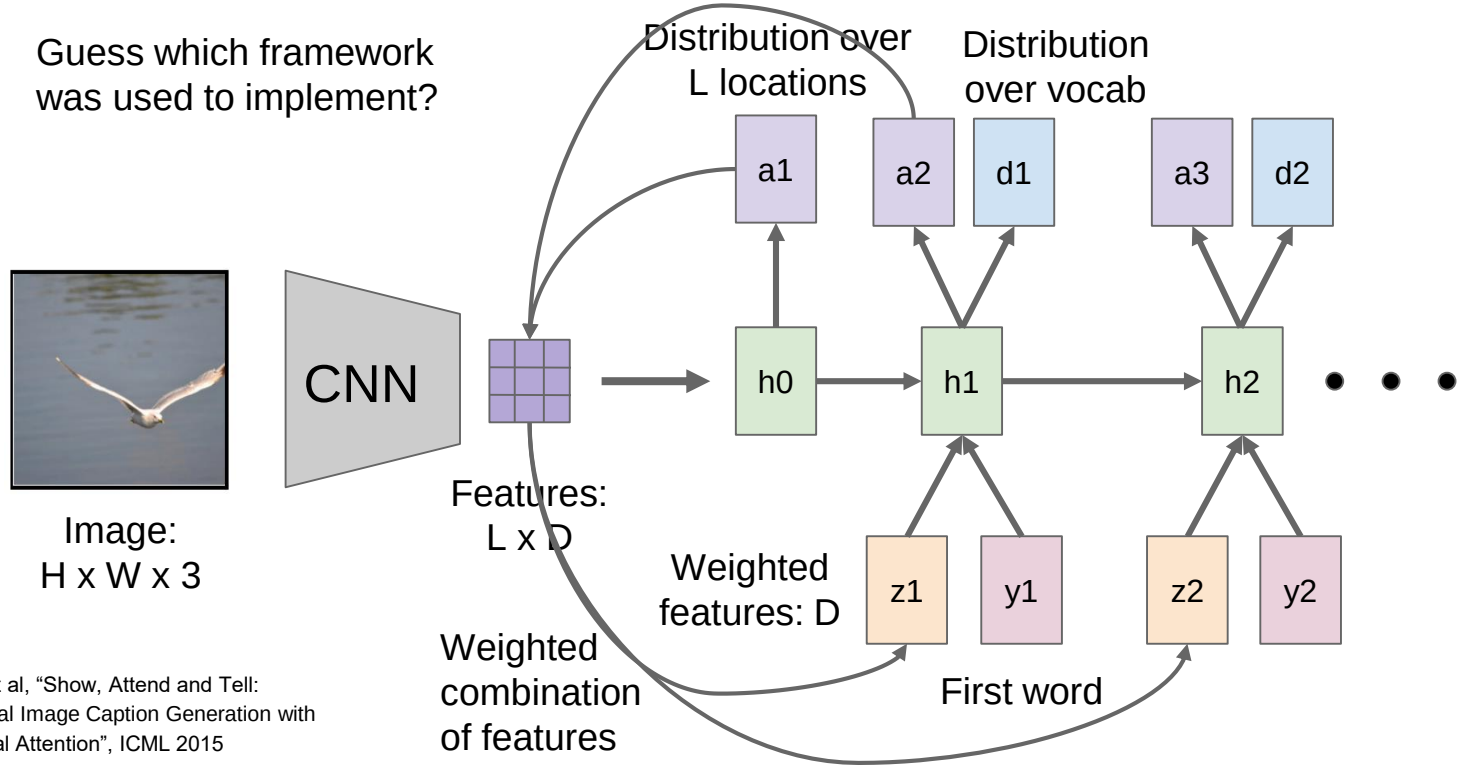
Weighted
features: D

Weighted
combination
of features

First word

Xu et al, “Show, Attend and Tell: Neural Image Caption Generation with Visual Attention”, ICML 2015

Soft Attention for Captioning



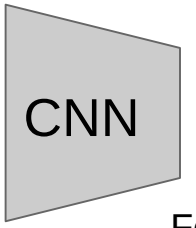
Xu et al, "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention", ICML 2015

Soft Attention for Captioning

Guess which framework was used to implement?

Crazy RNN = **Theano**

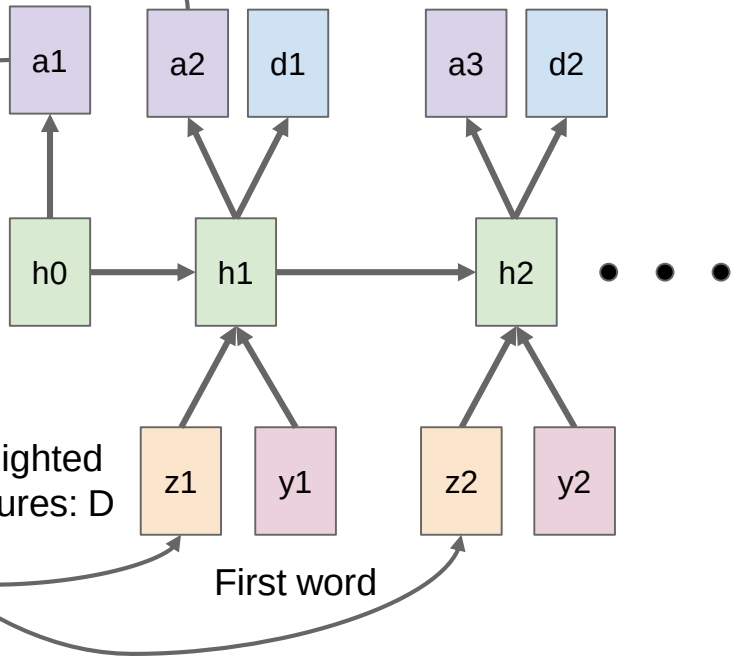
Image:
H x W x 3



Features:
L x D

Distribution over
L locations

Distribution
over vocab



Weighted
features: D

Weighted
combination
of features

First word

Xu et al, “Show, Attend and Tell: Neural Image Caption Generation with Visual Attention”, ICML 2015

Soft vs Hard Attention

Image:
H x W x 3

CNN

a	b
c	d

Grid of features
(Each D-dimensional)

From RNN:

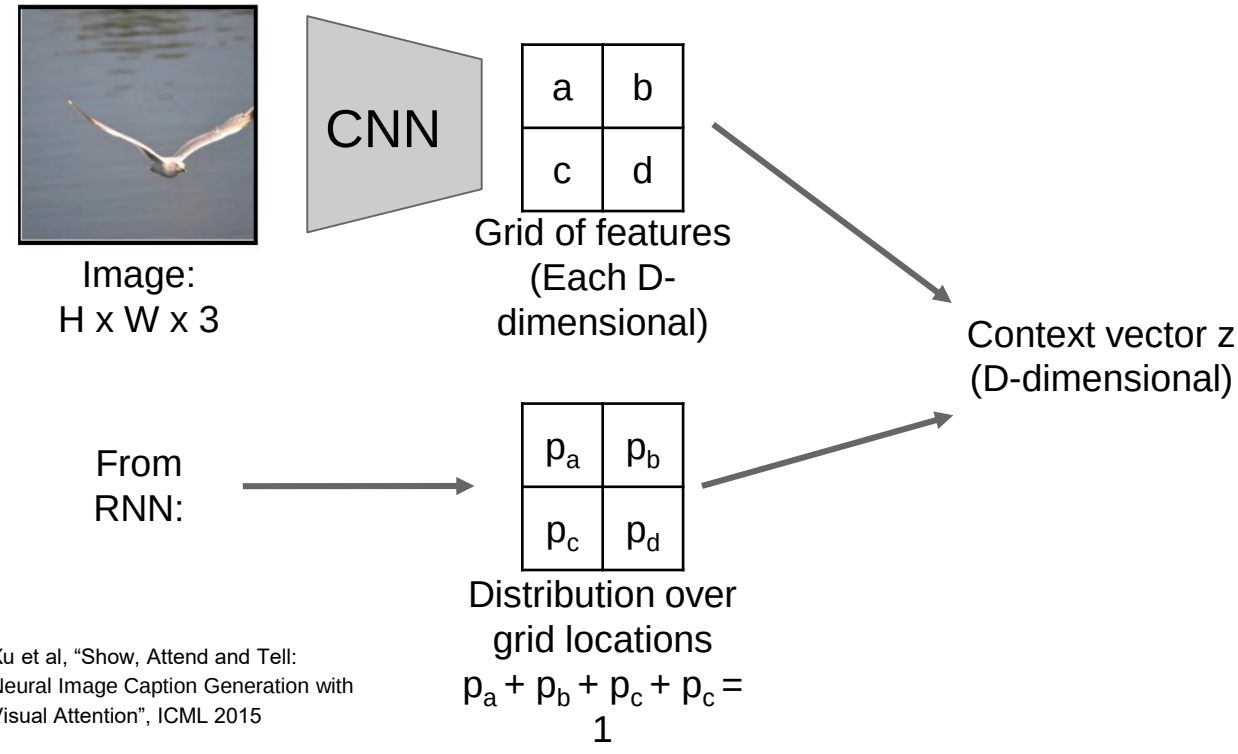
p_a	p_b
p_c	p_d

Distribution over grid locations

$p_a + p_b + p_c + p_c = 1$

Xu et al, “Show, Attend and Tell: Neural Image Caption Generation with Visual Attention”, ICML 2015

Soft vs Hard Attention



Xu et al, “Show, Attend and Tell: Neural Image Caption Generation with Visual Attention”, ICML 2015

Soft vs Hard Attention

Image:
H x W x 3

CNN

a	b
c	d

Grid of features
(Each D-dimensional)

Context vector z
(D-dimensional)

From RNN:

p_a	p_b
p_c	p_d

Distribution over grid locations

$p_a + p_b + p_c + p_c = 1$

Xu et al, “Show, Attend and Tell: Neural Image Caption Generation with Visual Attention”, ICML 2015

Soft attention:
Summarize ALL locations

$z = p_a a + p_b b + p_c c + p_d d$

Derivative dz/dp is nice!
Train with gradient descent

Soft vs Hard Attention

Image:
H x W x 3

CNN

a	b
c	d

Grid of features
(Each D-dimensional)

Context vector z
(D-dimensional)

From RNN:

p_a	p_b
p_c	p_d

Distribution over grid locations
$p_a + p_b + p_c + p_c = 1$

Xu et al, "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention", ICML 2015

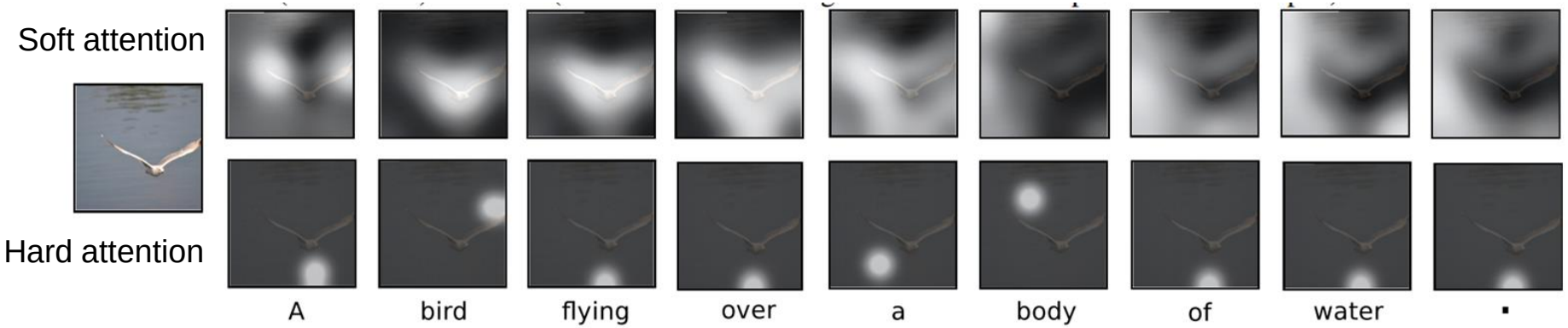
Soft attention:
Summarize ALL locations
$z = p_a a + p_b b + p_c c + p_d d$

Derivative dz/dp is nice!
Train with gradient descent

Hard attention:
Sample ONE location according to p, z = that vector

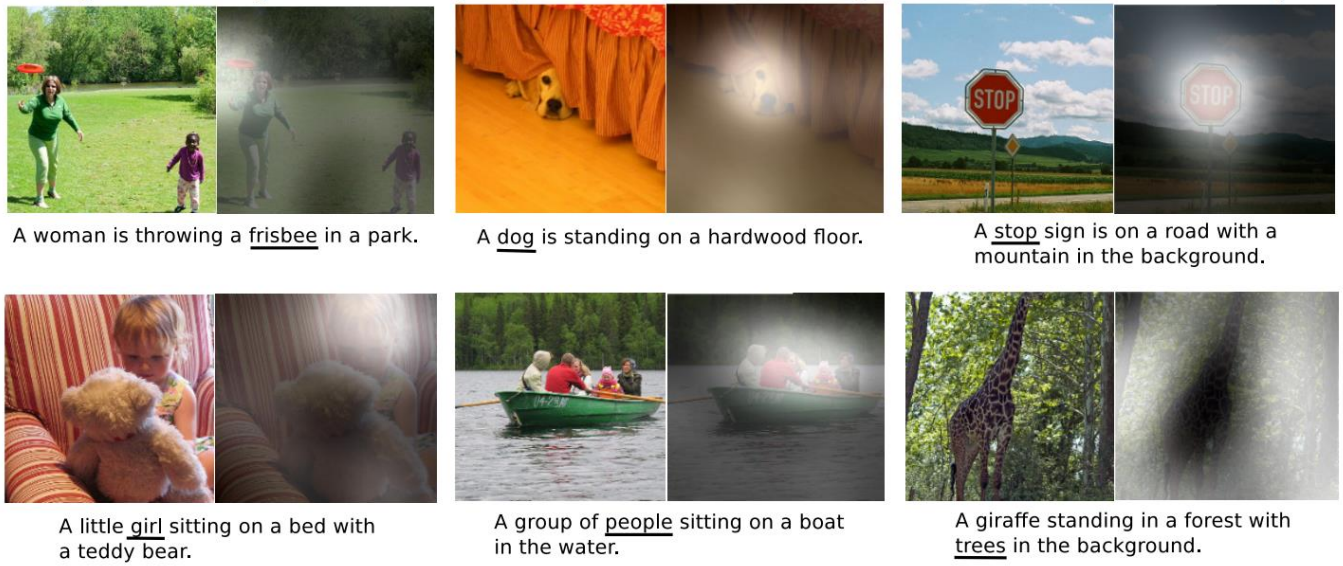
With argmax, dz/dp is zero almost everywhere …
Can't use gradient descent; need reinforcement learning

Soft Attention for Captioning



Xu et al, “Show, Attend and Tell: Neural Image Caption Generation with Visual Attention”, ICML 2015

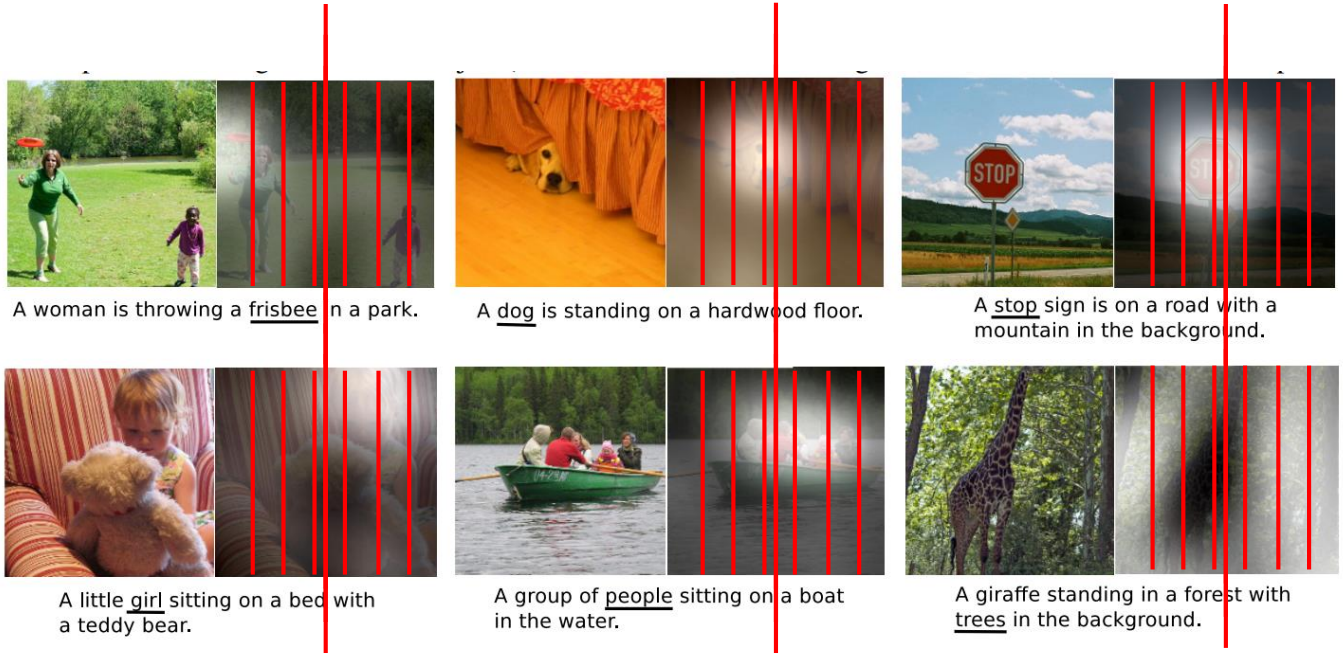
Soft Attention for Captioning



Xu et al, “Show, Attend and Tell: Neural Image Caption Generation with Visual Attention”, ICML 2015

Soft Attention for Captioning

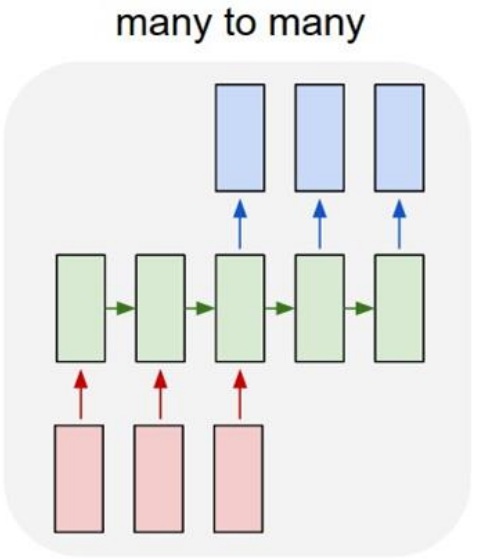
Attention constrained to fixed grid! We'll come back to this ….



Xu et al, "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention", ICML 2015

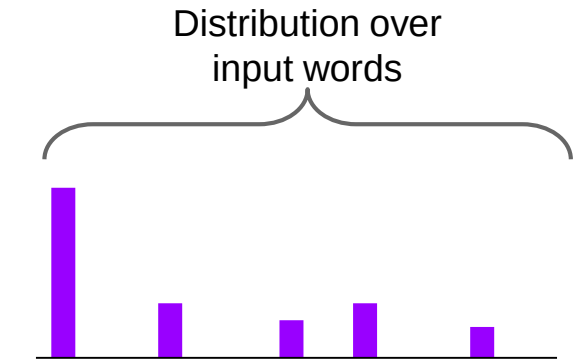
Soft Attention for Translation

“Mi gato es el mejor” -> “My cat is the best”

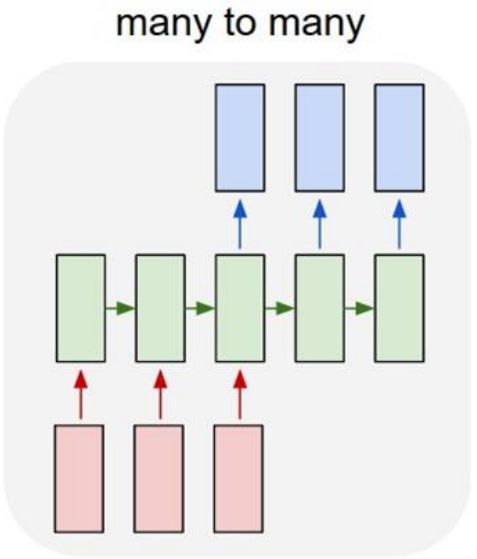


Bahdanau et al, “Neural Machine Translation by Jointly Learning to Align and Translate”, ICLR 2015

Soft Attention for Translation

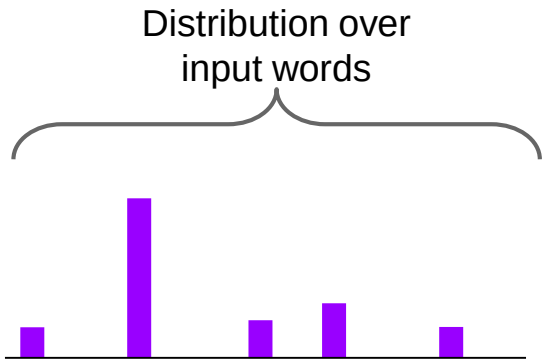


“Mi gato es el mejor” -> “My cat is the best”

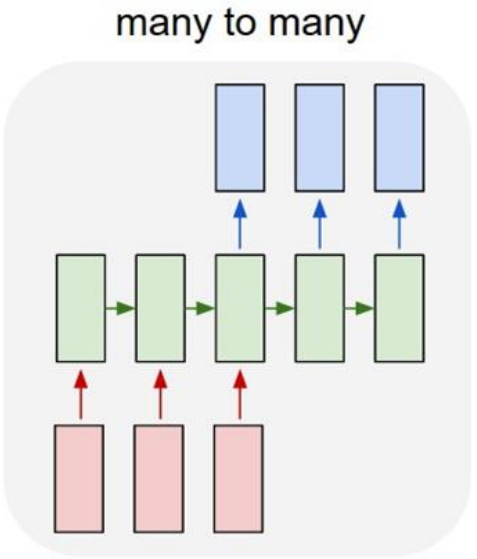


Bahdanau et al, “Neural Machine Translation by Jointly Learning to Align and Translate”, ICLR 2015

Soft Attention for Translation

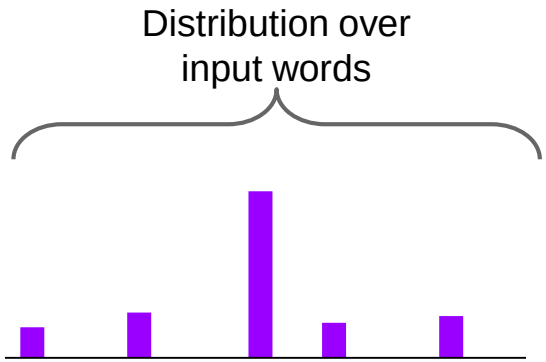


“Mi gato es el mejor” -> “My cat is the best”

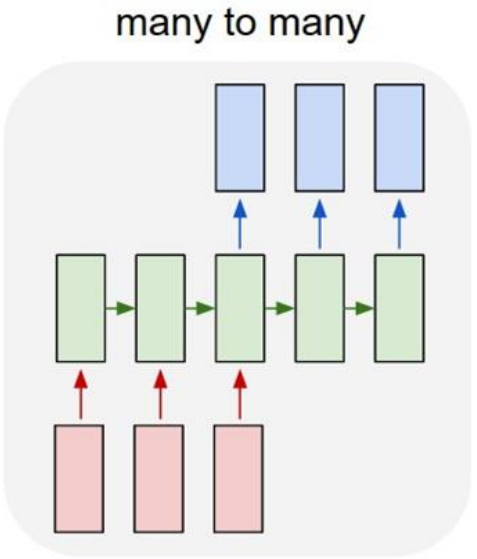


Bahdanau et al, “Neural Machine Translation by Jointly Learning to Align and Translate”, ICLR 2015

Soft Attention for Translation



“Mi gato es el mejor” -> “My cat is the best”



Bahdanau et al, “Neural Machine Translation by Jointly Learning to Align and Translate”, ICLR 2015

Soft Attention for Everything!

Machine Translation, attention over input:

- Luong et al, “Effective Approaches to Attention-based Neural Machine Translation,” EMNLP 2015

Speech recognition, attention over input sounds:

- Chan et al, “Listen, Attend, and Spell”, arXiv 2015
- Chorowski et al, “Attention-based models for Speech Recognition”, NIPS 2015

+Local+Global: Someone is frying a fish in a pot

Video captioning, attention over input frames:

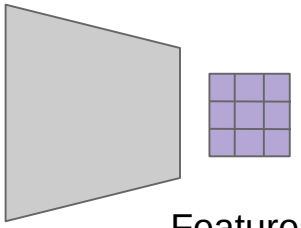
- Yao et al, “Describing Videos by Exploiting Temporal Structure”, ICCV 2015

Image, question to answer, attention over image:

- Xu and Saenko, “Ask, Attend and Answer: Exploring Question-Guided Spatial Attention for Visual Question Answering”, arXiv 2015
- Zhu et al, “Visual7W: Grounded Question Answering in Images”, arXiv 2015

Attending to arbitrary regions?

Image:
H x W x 3

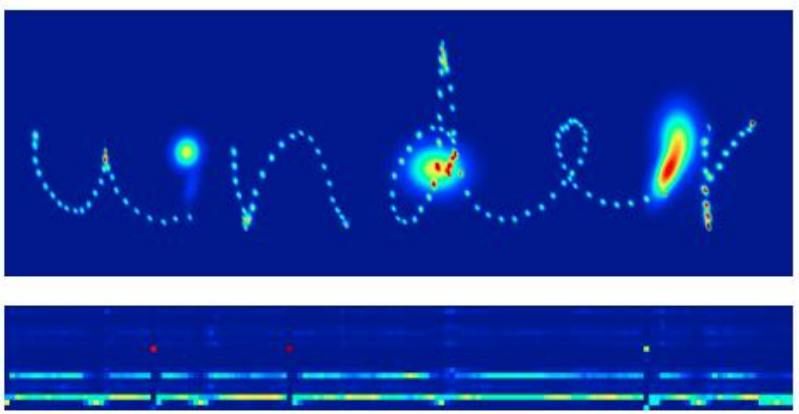


Features:
L x D

Attention mechanism from Show, Attend, and Tell only lets us softly attend to fixed grid positions … can we do better?

Attending to Arbitrary Regions

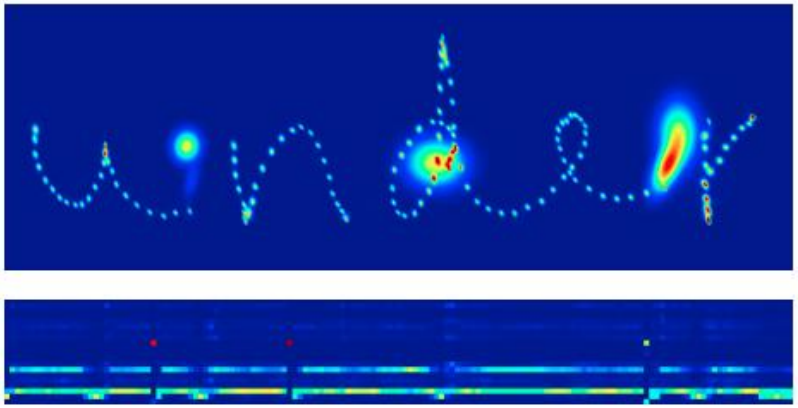
- Read text, generate handwriting using an RNN
- Attend to arbitrary regions of the **output** by predicting params of a mixture model



Graves, “Generating Sequences with Recurrent Neural Networks”, arXiv 2013

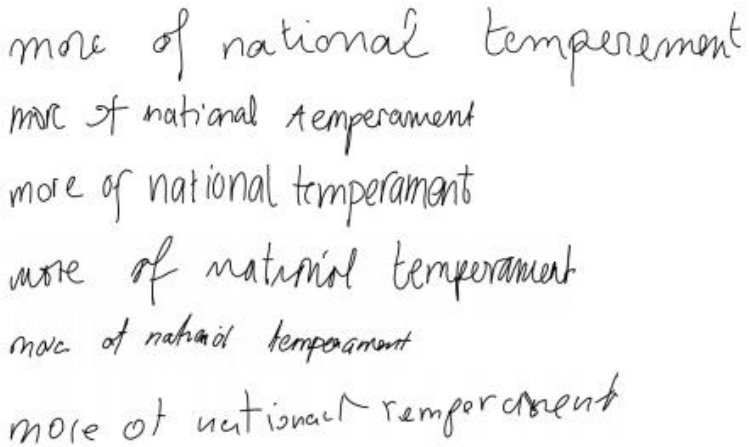
Attending to Arbitrary Regions

- Read text, generate handwriting using an RNN
- Attend to arbitrary regions of the **output** by predicting params of a mixture model



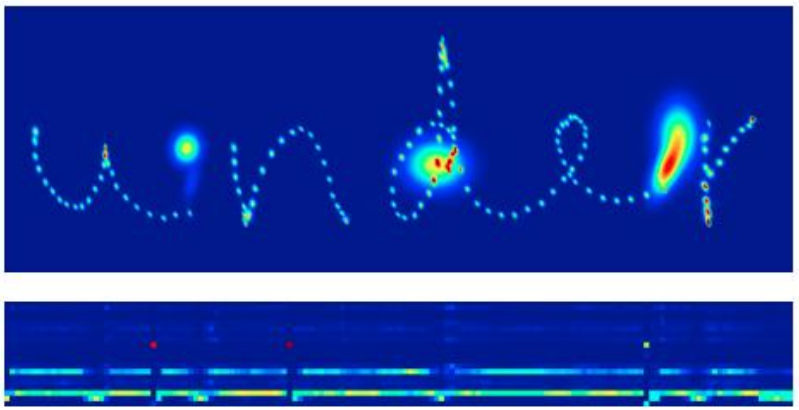
Graves, “Generating Sequences with Recurrent Neural Networks”, arXiv 2013

Which are real and which are generated?



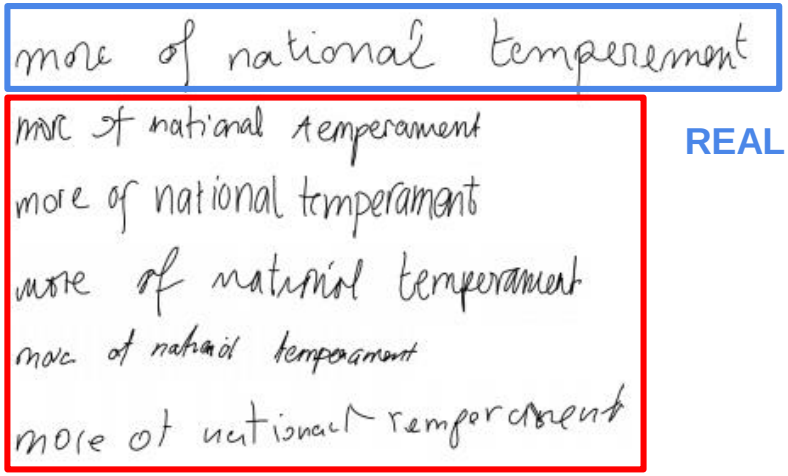
Attending to Arbitrary Regions

- Read text, generate handwriting using an RNN
- Attend to arbitrary regions of the **output** by predicting params of a mixture model



Graves, “Generating Sequences with Recurrent Neural Networks”, arXiv 2013

Which are real and which are generated?

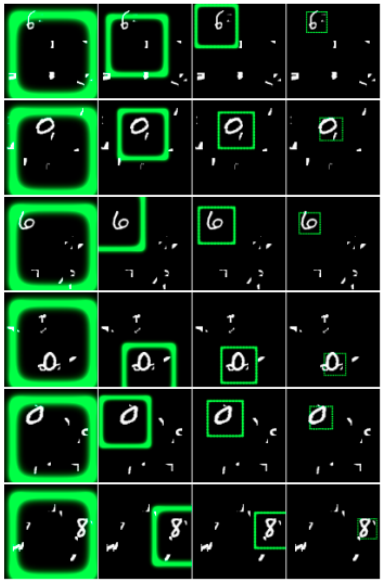


REAL

GENERATED

Attending to Arbitrary Regions: DRAW

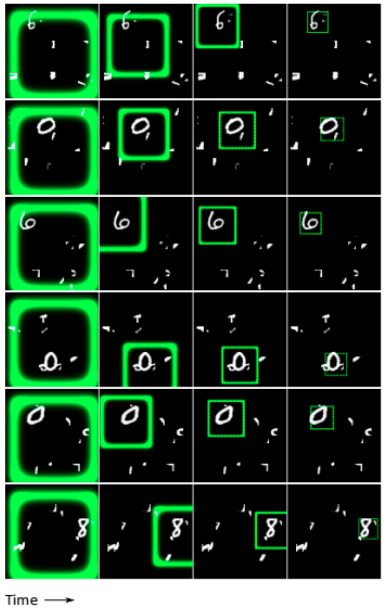
Classify images by attending to arbitrary regions of the *input*



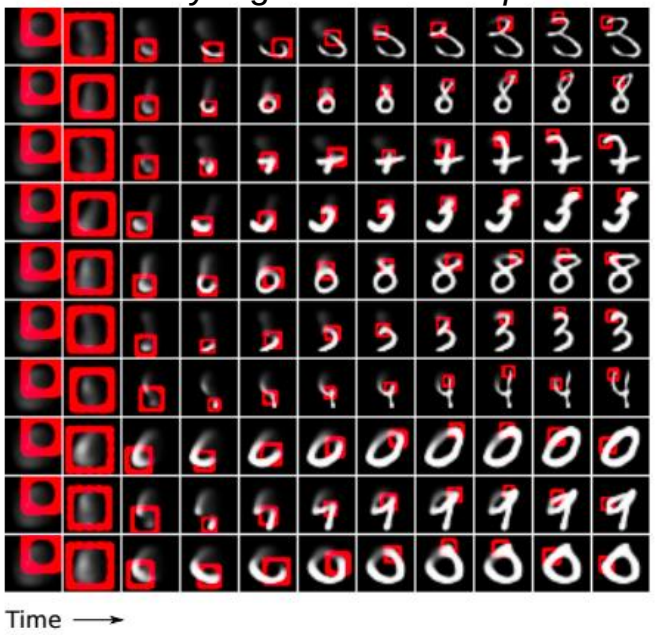
Gregor et al, “DRAW: A Recurrent Neural Network For Image Generation”, ICML 2015

Attending to Arbitrary Regions: DRAW

Classify images by attending to arbitrary regions of the *input*



Generate images by attending to arbitrary regions of the *output*



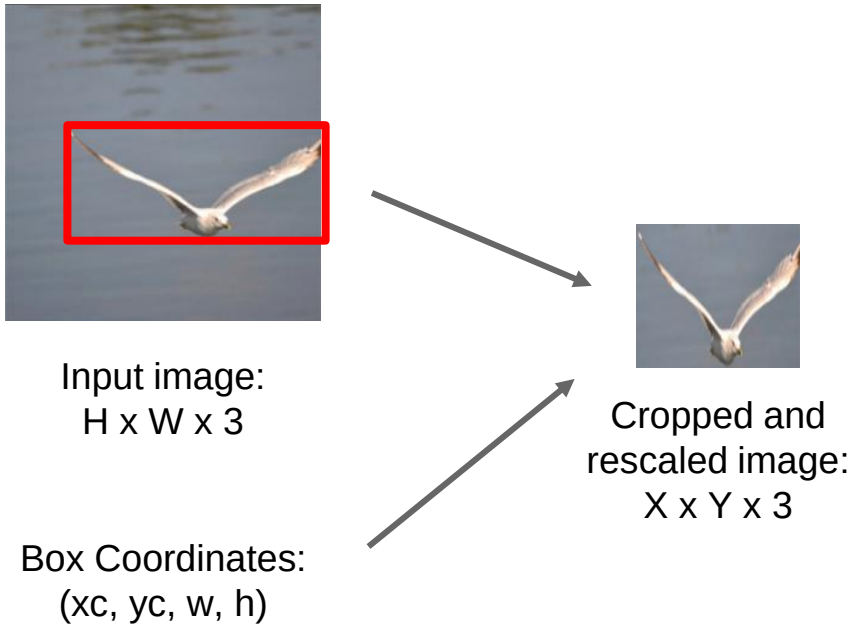
Gregor et al, “DRAW: A Recurrent Neural Network For Image Generation”, ICML 2015

Attending to Arbitrary Regions:

Attention mechanism similar to DRAW, but easier to explain

Jaderberg et al, “Spatial Transformer Networks”, NIPS 2015

Spatial Transformer Networks



Jaderberg et al, “Spatial Transformer Networks”, NIPS 2015

Spatial Transformer Networks

Input image:
H x W x 3

Can we make this function differentiable?

Cropped and rescaled image:
X x Y x 3

Box Coordinates:
(xc, yc, w, h)

Jaderberg et al, “Spatial Transformer Networks”, NIPS 2015

Spatial Transformer Networks

Input image:
H x W x 3

Can we make this function differentiable?

Cropped and rescaled image:
X x Y x 3

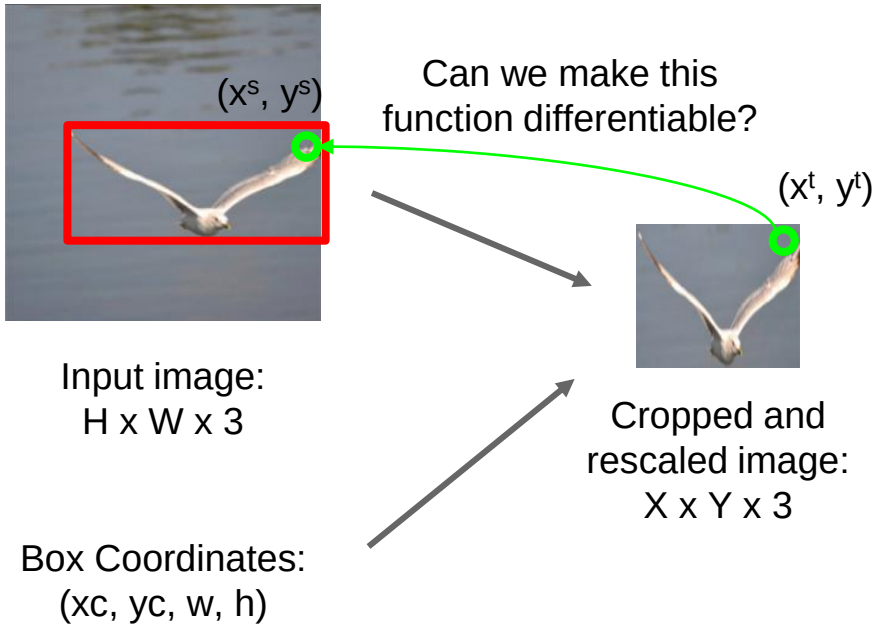
Box Coordinates:
(xc, yc, w, h)

Idea: Function mapping *pixel coordinates* (xt, yt) of output to *pixel coordinates* (xs, ys) of input

$$\begin{pmatrix} x_i^s \\ y_i^s \end{pmatrix} = \begin{bmatrix} \theta_{11} & \theta_{12} & \theta_{13} \\ \theta_{21} & \theta_{22} & \theta_{23} \end{bmatrix} \begin{pmatrix} x_i^t \\ y_i^t \\ 1 \end{pmatrix}$$

Jaderberg et al, “Spatial Transformer Networks”, NIPS 2015

Spatial Transformer Networks

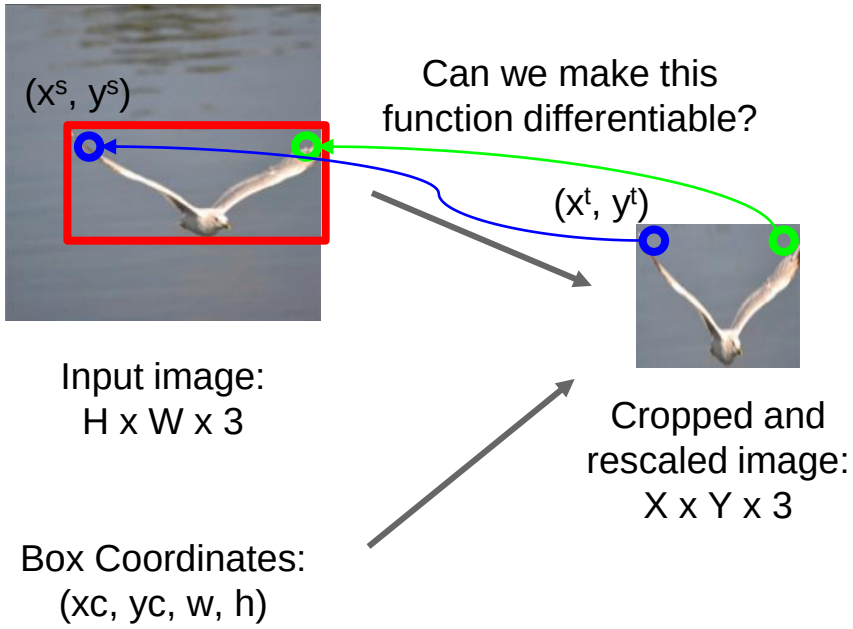


Idea: Function mapping *pixel coordinates* (xt, yt) of output to *pixel coordinates* (xs, ys) of input

$$\begin{pmatrix} x_i^s \\ y_i^s \end{pmatrix} = \begin{bmatrix} \theta_{11} & \theta_{12} & \theta_{13} \\ \theta_{21} & \theta_{22} & \theta_{23} \end{bmatrix} \begin{pmatrix} x_i^t \\ y_i^t \\ 1 \end{pmatrix}$$

Jaderberg et al, “Spatial Transformer Networks”, NIPS 2015

Spatial Transformer Networks

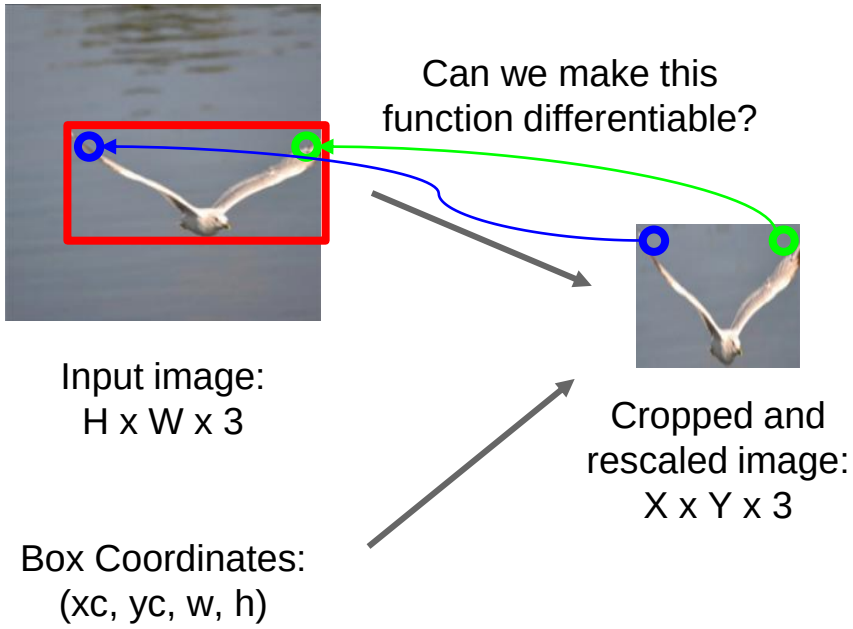


Idea: Function mapping *pixel coordinates* (xt, yt) of output to *pixel coordinates* (xs, ys) of input

$$\begin{pmatrix} x_i^s \\ y_i^s \end{pmatrix} = \begin{bmatrix} \theta_{11} & \theta_{12} & \theta_{13} \\ \theta_{21} & \theta_{22} & \theta_{23} \end{bmatrix} \begin{pmatrix} x_i^t \\ y_i^t \\ 1 \end{pmatrix}$$

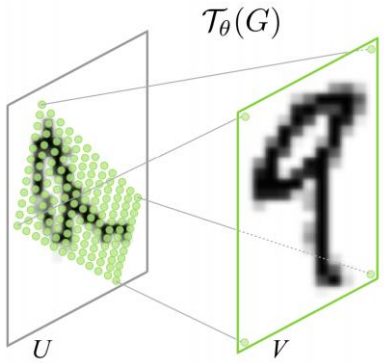
Jaderberg et al, “Spatial Transformer Networks”, NIPS 2015

Spatial Transformer Networks



Idea: Function mapping *pixel coordinates* (xt, yt) of output to *pixel coordinates* (xs, ys) of input

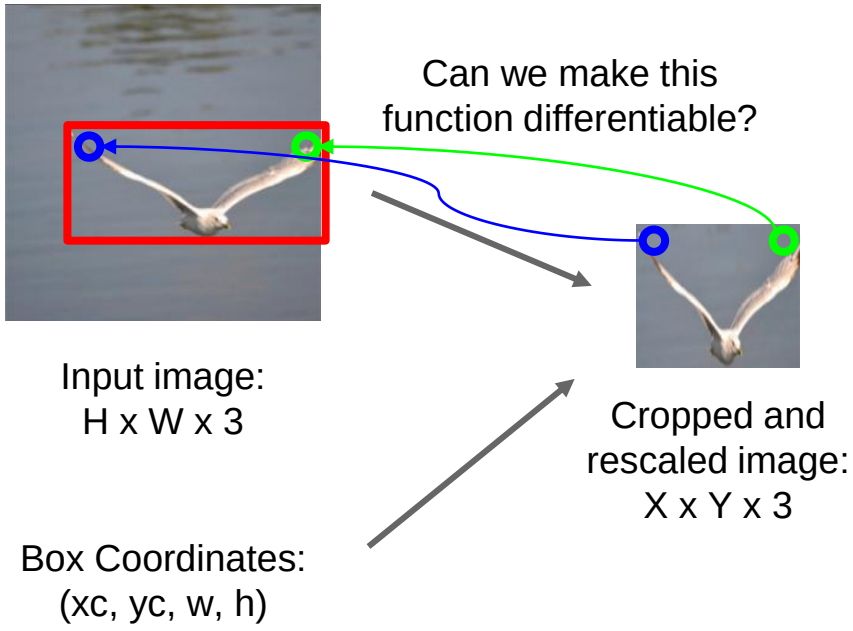
$$\begin{pmatrix} x_i^s \\ y_i^s \end{pmatrix} = \begin{bmatrix} \theta_{11} & \theta_{12} & \theta_{13} \\ \theta_{21} & \theta_{22} & \theta_{23} \end{bmatrix} \begin{pmatrix} x_i^t \\ y_i^t \\ 1 \end{pmatrix}$$



Repeat for all pixels in *output* to get a **sampling grid**

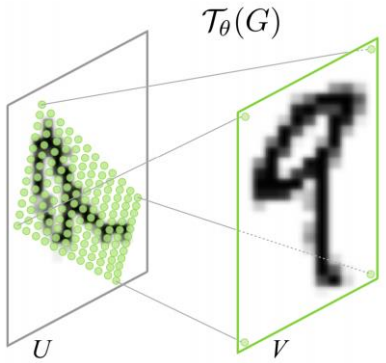
Jaderberg et al, “Spatial Transformer Networks”, NIPS 2015

Spatial Transformer Networks



Idea: Function mapping *pixel coordinates* (xt, yt) of output to *pixel coordinates* (xs, ys) of input

$$\begin{pmatrix} x_i^s \\ y_i^s \end{pmatrix} = \begin{bmatrix} \theta_{11} & \theta_{12} & \theta_{13} \\ \theta_{21} & \theta_{22} & \theta_{23} \end{bmatrix} \begin{pmatrix} x_i^t \\ y_i^t \\ 1 \end{pmatrix}$$

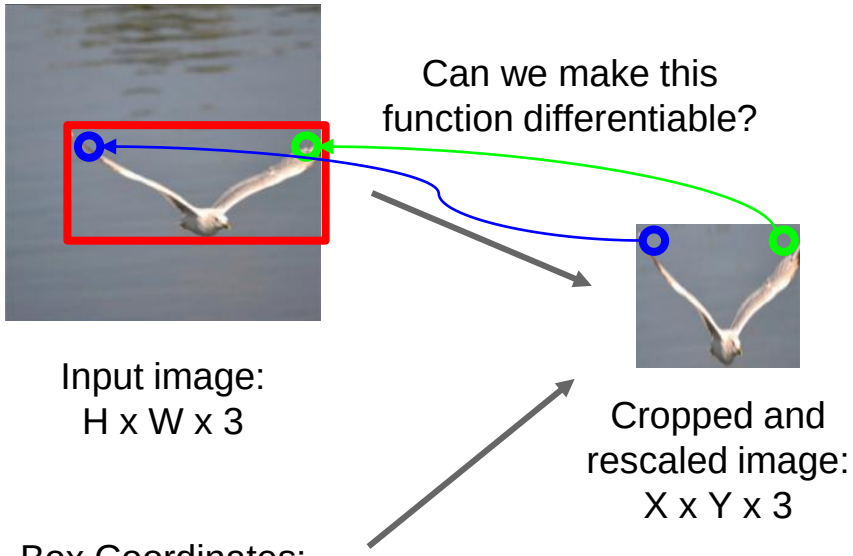


Repeat for all pixels in *output* to get a **sampling grid**

Then use **bilinear interpolation** to compute output

Jaderberg et al, “Spatial Transformer Networks”, NIPS 2015

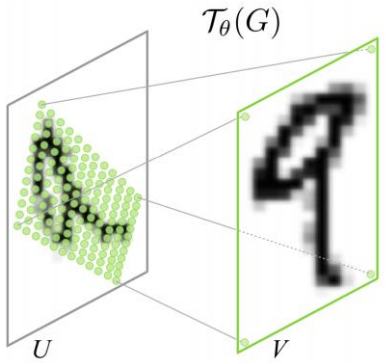
Spatial Transformer Networks



Idea: Function mapping *pixel coordinates* (xt, yt) of output to *pixel coordinates* (xs, ys) of input

Network attends to input by predicting θ

$$\begin{pmatrix} x_i^s \\ y_i^s \end{pmatrix} = \begin{bmatrix} \theta_{11} & \theta_{12} & \theta_{13} \\ \theta_{21} & \theta_{22} & \theta_{23} \end{bmatrix} \begin{pmatrix} x_i^t \\ y_i^t \\ 1 \end{pmatrix}$$

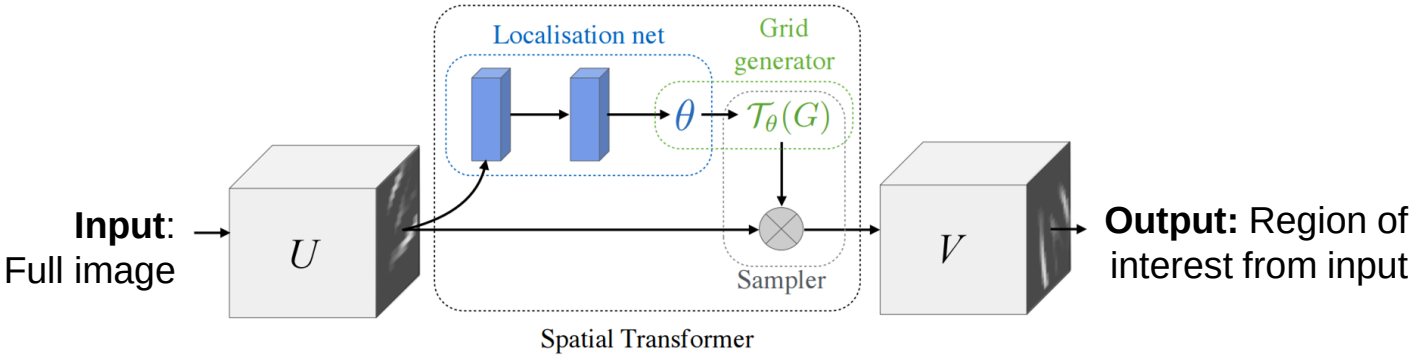


Repeat for all pixels in *output* to get a **sampling grid**

Then use **bilinear interpolation** to compute output

Jaderberg et al, “Spatial Transformer Networks”, NIPS 2015

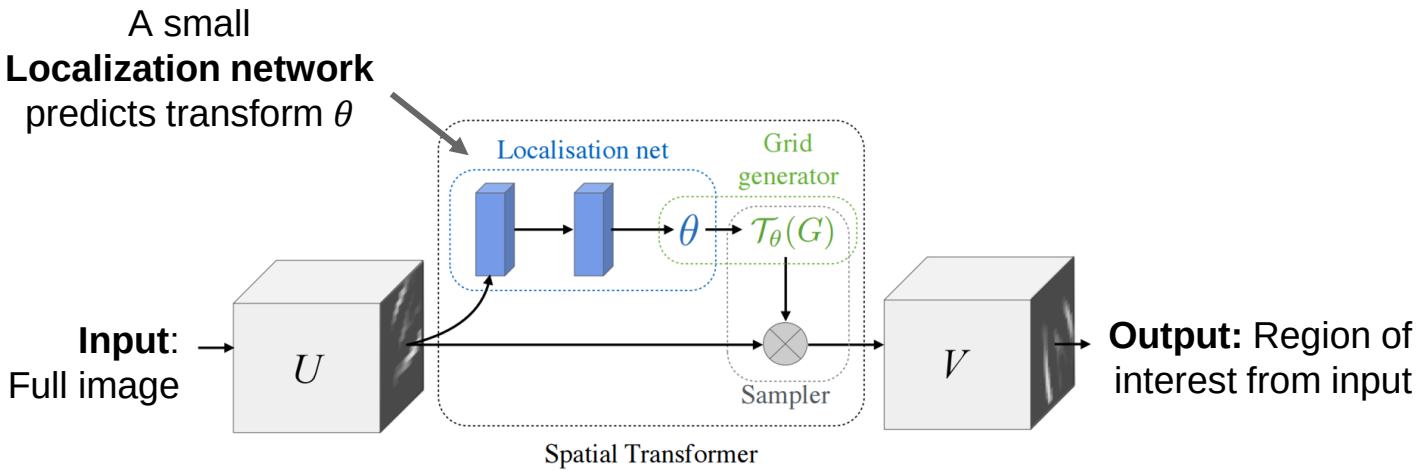
Spatial Transformer Networks



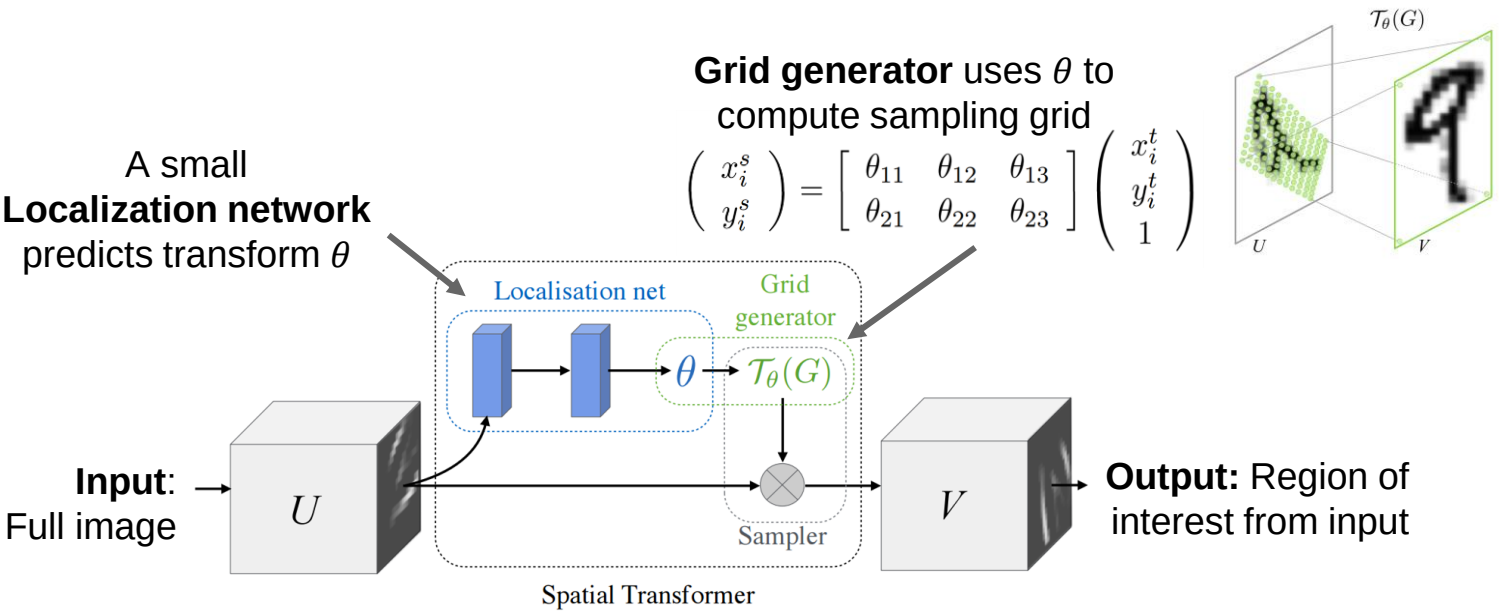
Input: Full image

Output: Region of interest from input

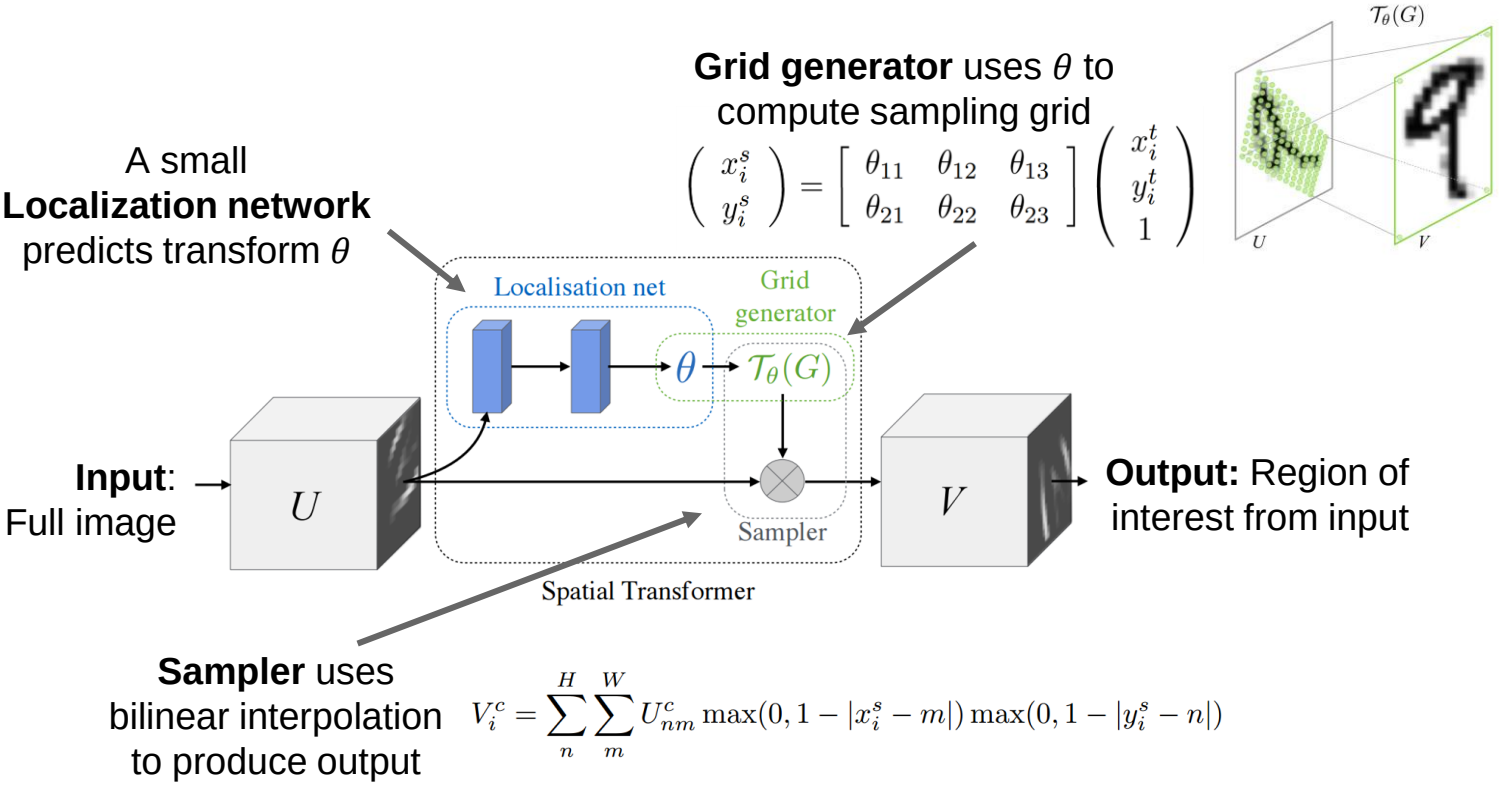
Spatial Transformer Networks



Spatial Transformer Networks

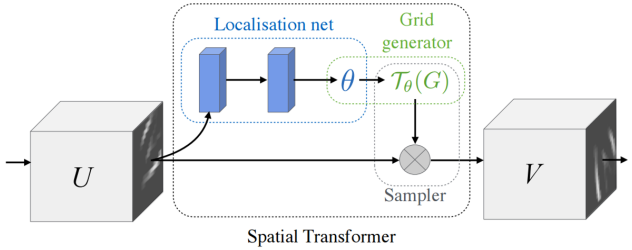


Spatial Transformer Networks

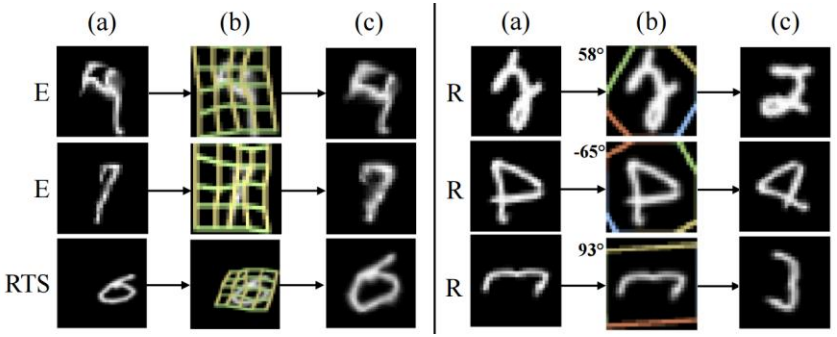


Spatial Transformer Networks

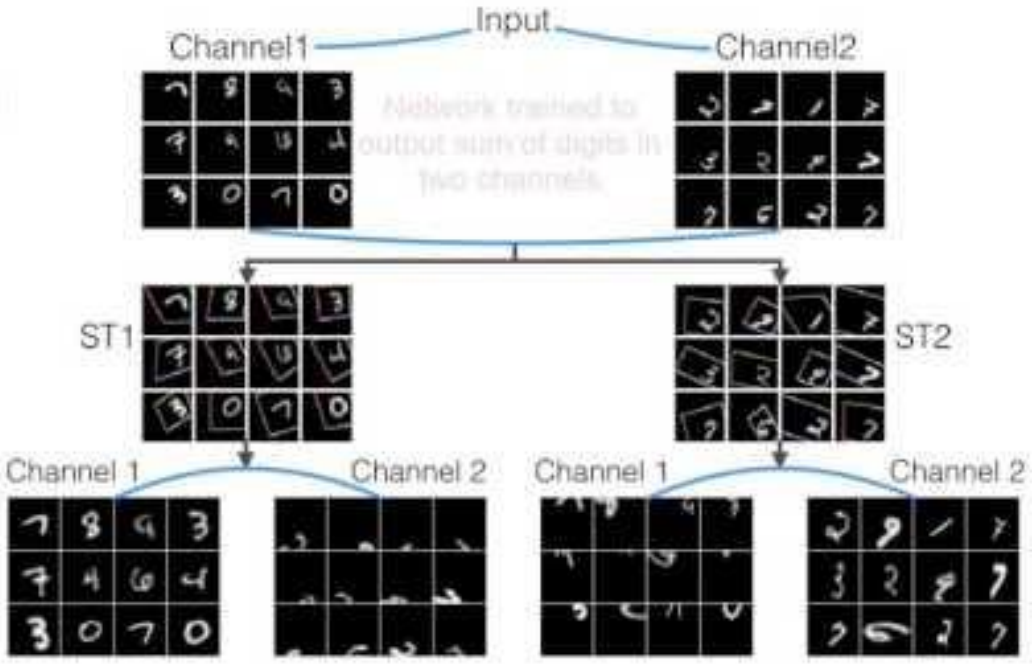
Differentiable “attention / transformation” module



Insert spatial transformers into a classification network and it learns to attend and transform the input



MNIST Addition



Attention Recap

- Soft attention:
  - Easy to implement: produce distribution over input locations, reweight features and feed as input
  - Attend to arbitrary input locations using spatial transformer networks
- Hard attention:
  - Attend to a single input location
  - Can’t use gradient descent!
  - Need **reinforcement learning!**

Outline

1. Course Review
2. Attention Models
3. Memory Models

Story Comprehension

Joe went to the kitchen. Fred went to the kitchen.
Joe picked up the milk. Joe travelled to his office.
Joe left the milk. Joe went to the bathroom.

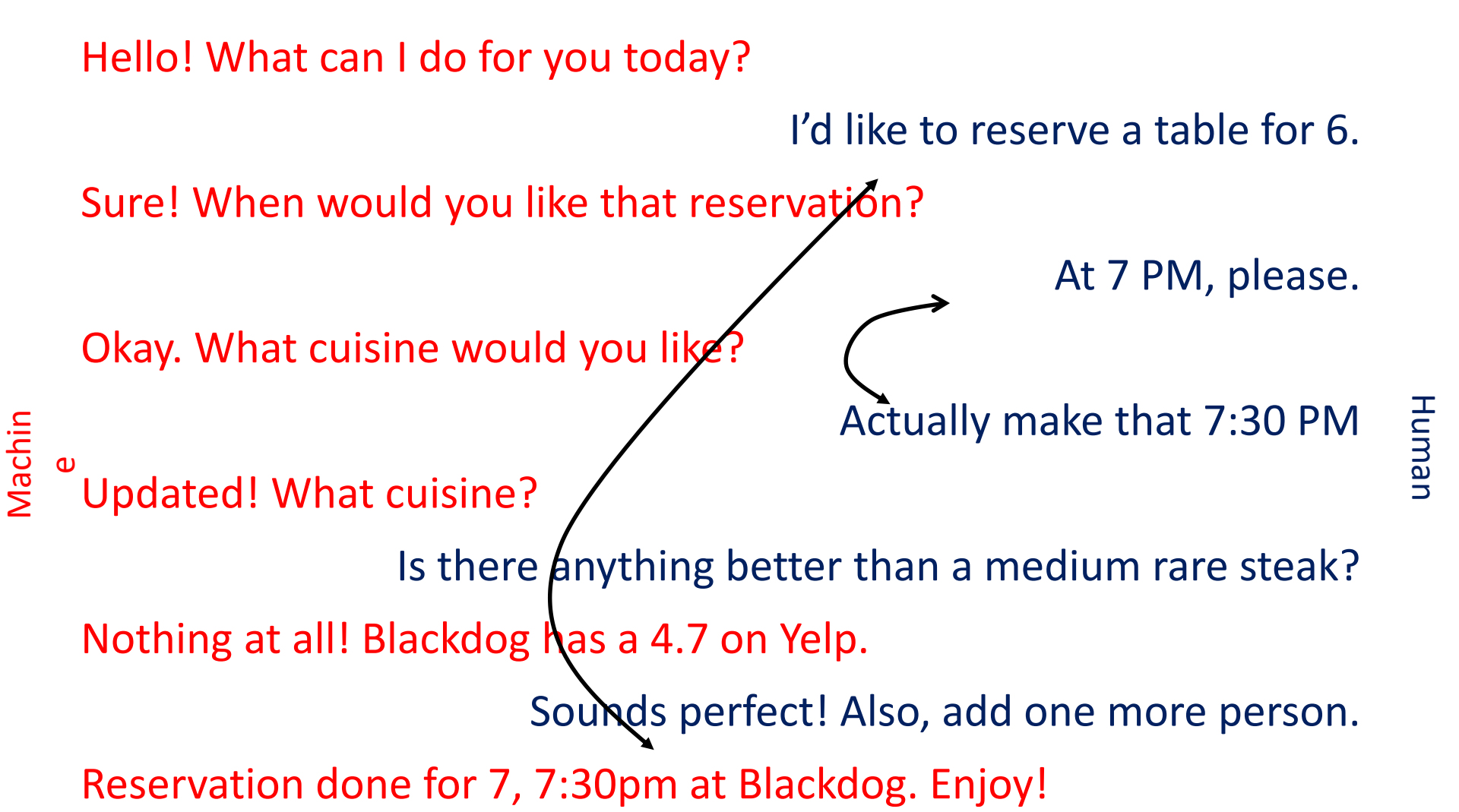
Questions from Joe's angry mother:

Q1 : Where is Joe?

Q2 : Where is the milk now?

Q3 : Where was Joe before the office?

Dialogue System



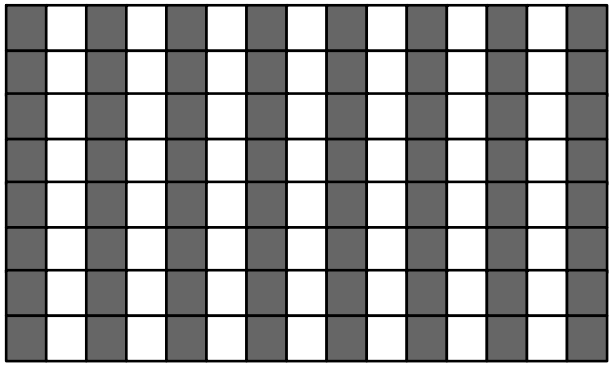
ML models need memory

Deeper AI tasks require explicit memory and multi-hop reasoning over it

- RNNs have short memory
- Cannot increase memory without increasing number of parameters
- Need for compartmentalized memory
- Read/Write should be asynchronous

Memory Networks

- Class of Models with memory m - Array of objects m_i



Each memory here is a dense vector

Four Components :

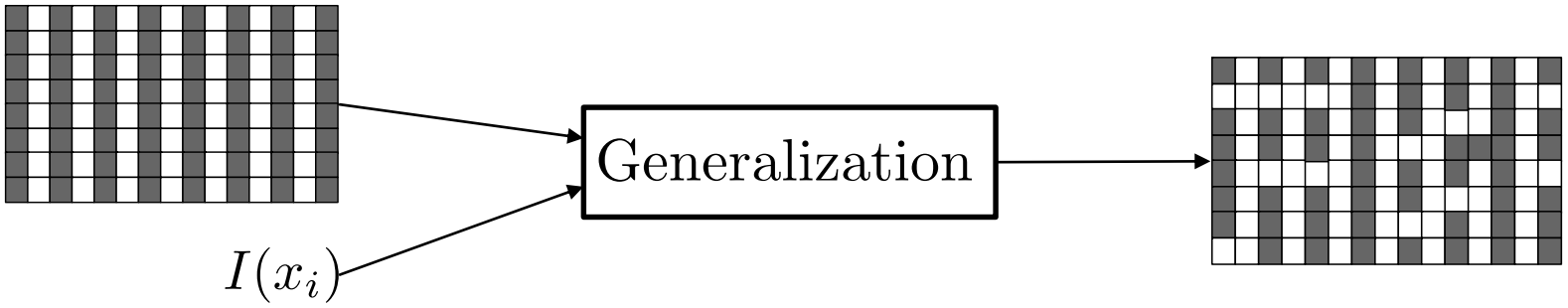
I - Input Feature Map : Input manipulation
G - Generalization : Memory Manipulation
O - Output Feature Map : Output representation generator
R - Response : Response Generator

Memory Networks

1. Input Feature Map

- Imagine input as a sequence of sentences x_i

2. Update Memories

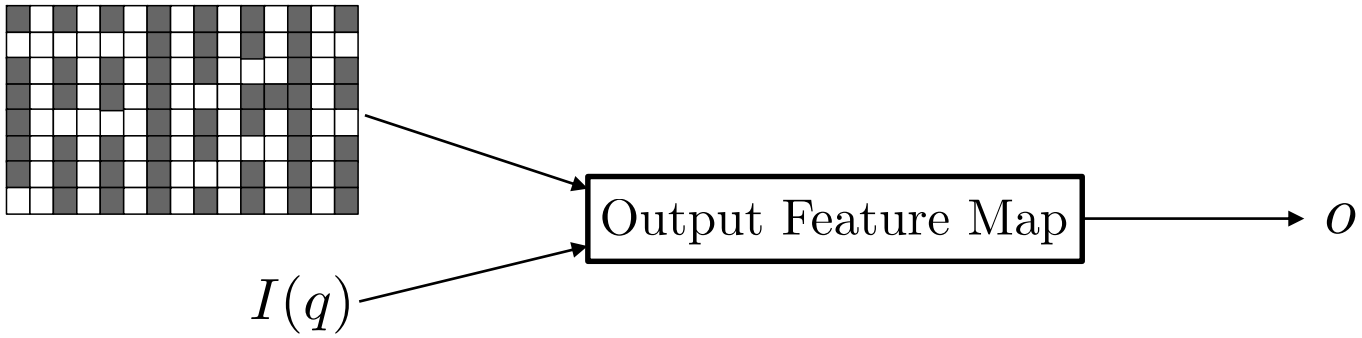


Memory Networks, Weston et. al., ICLR 2015

Memory Networks

3. Output Representation

- Say if q is a question, compute output representation



4. Generate Answer Response

Memory Networks, Weston et. al., ICLR 2015

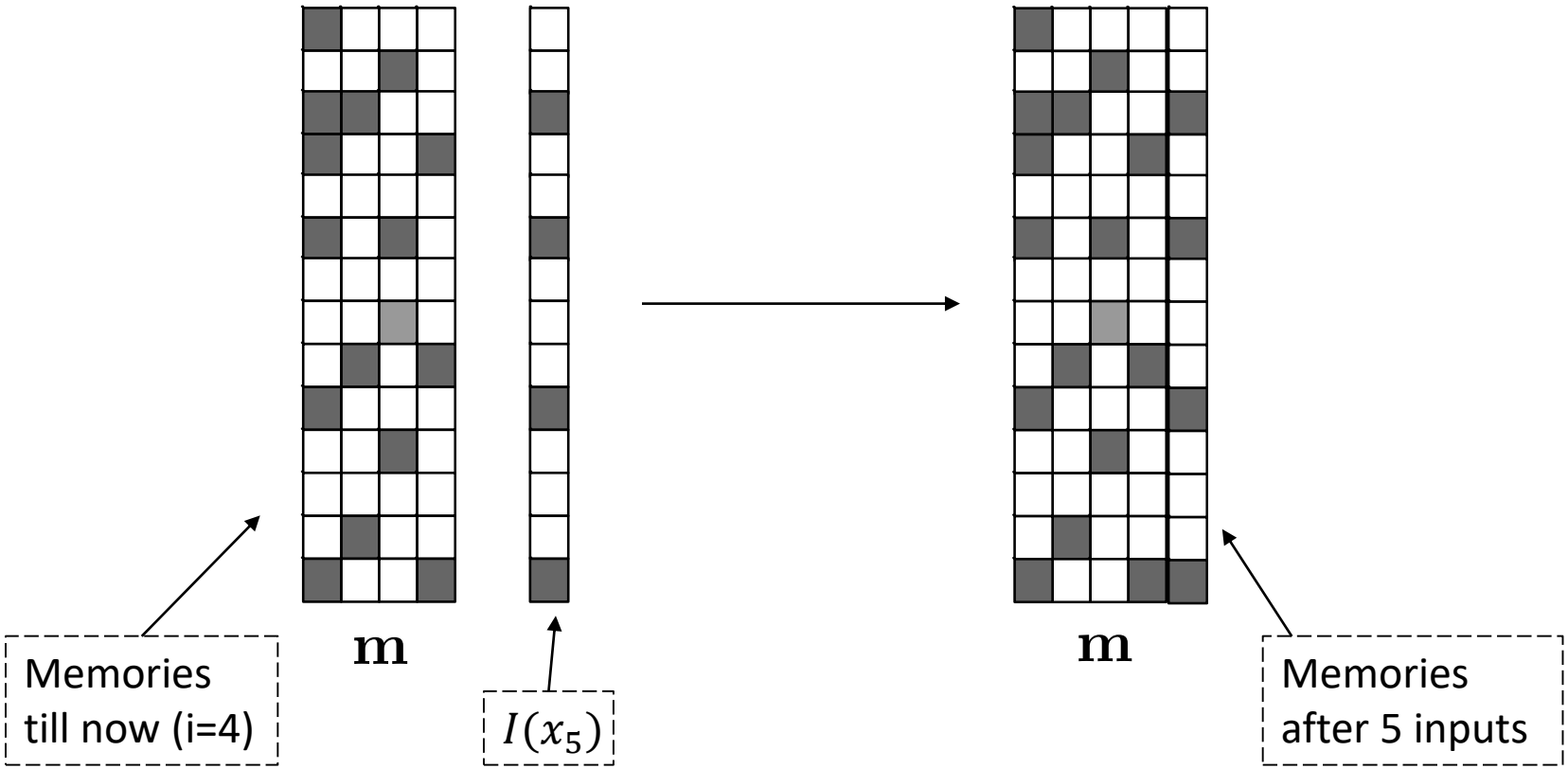
Simple MemNN for Text

1. **Input Feature Map** - Bag-of-Words representation

Simple MemNN for Text

2. Generalization : Store input in new memory

$$m_i = I(x_i)$$



Simple MemNN for Text

3. **Output:** Using $k = 2$ memory hops with query x

1st Max scoring memory index

$$o_1 = O_1(x, \mathbf{m}) = \underset{i=1,\ldots,N}{argmax}\ s_O(I(x), m_i)$$

Score all memories against input

$$o_2 = O_2(x, \mathbf{m}) = \underset{i=1,\ldots,N}{argmax}\ s_O([I(x), m_{o_1}], m_i)$$

2nd Max scoring memory index

Score all memories against input & o_1

4. **Response** - Single Word Answer

Max scoring word

$$r = argmax_{w \in W}\ s_R([I(x), m_{o_1}, m_{o_2}], w)$$

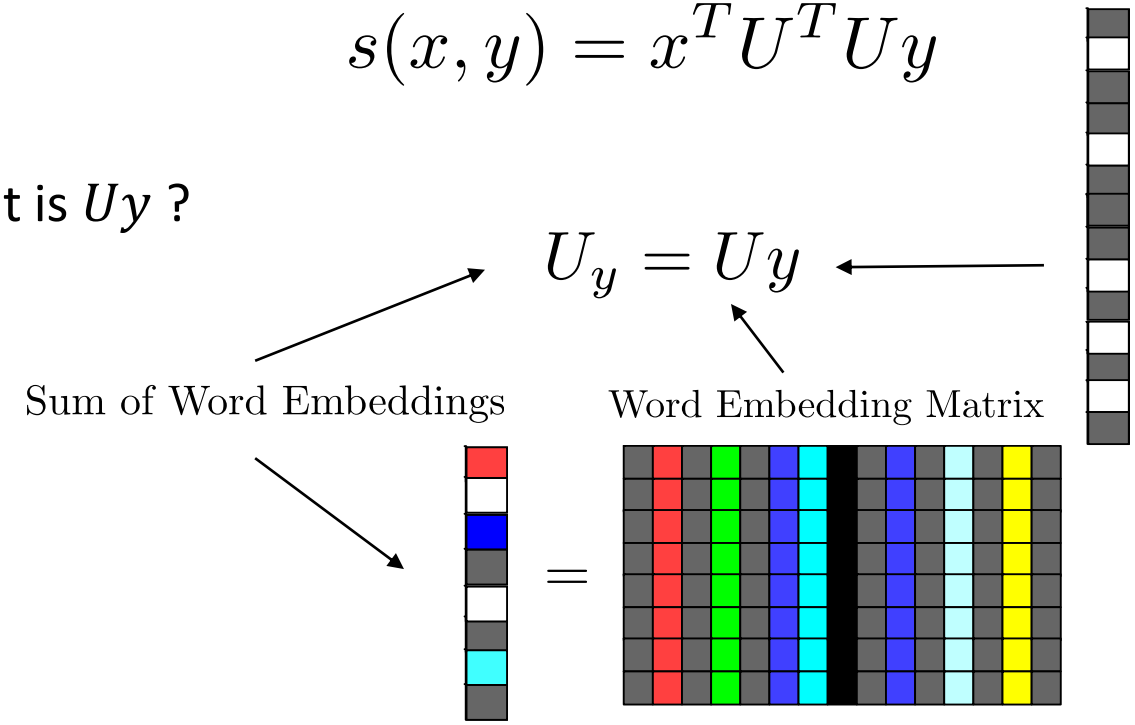
Score all words against query and 2 supporting memories

Scoring Function

- Scoring Function is an embedding model

$$s(x, y) = x^T U^T U y$$

- What is Uy ?

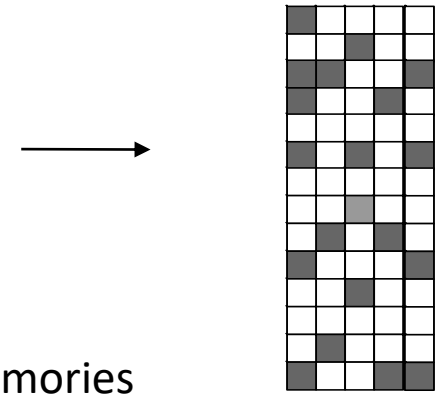


Scoring Function is just dot-product between sum of word embeddings!!!

Memory Networks, Weston et. al., ICLR 2015

Scoring Function

Joe went to the kitchen.
Fred went to the kitchen.
Joe picked up the milk.
Joe travelled to his office.
Joe left the milk. Joe went to the bathroom.

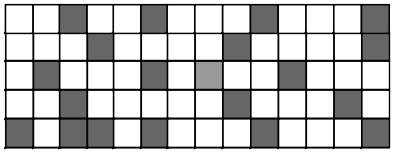


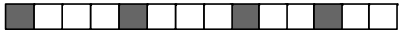
Input Sentences

Memories

Where is the milk now?

Question

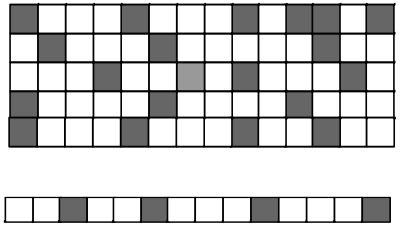


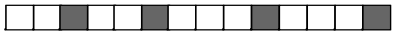


1st supporting memory

Where is the milk now?

Question

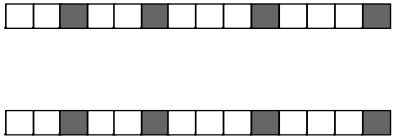




2nd supporting memory

Where is the milk now?

Question

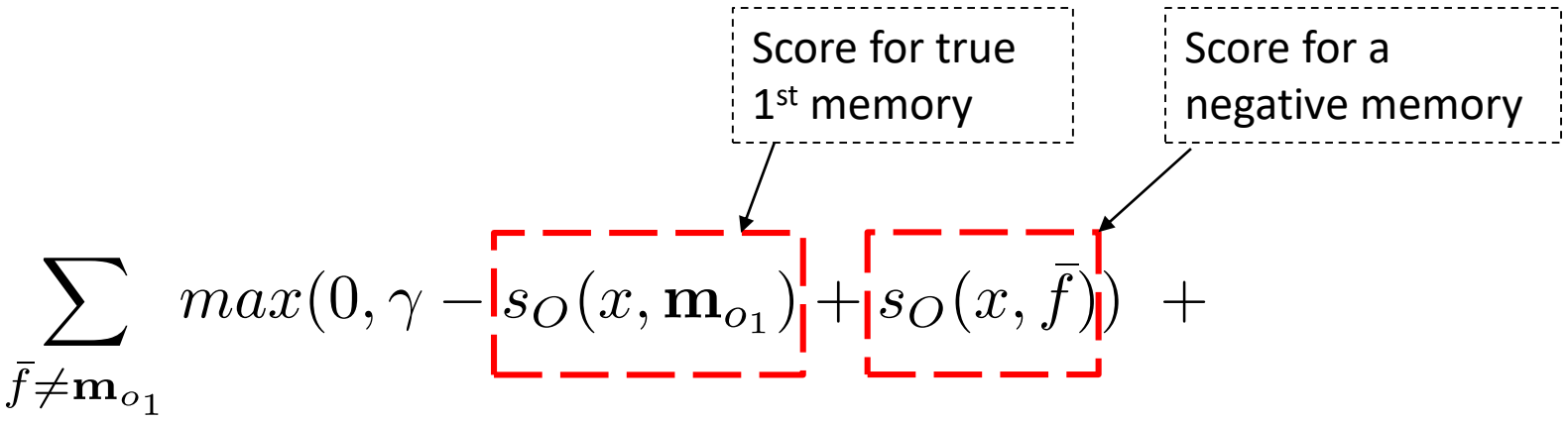


Office

Response

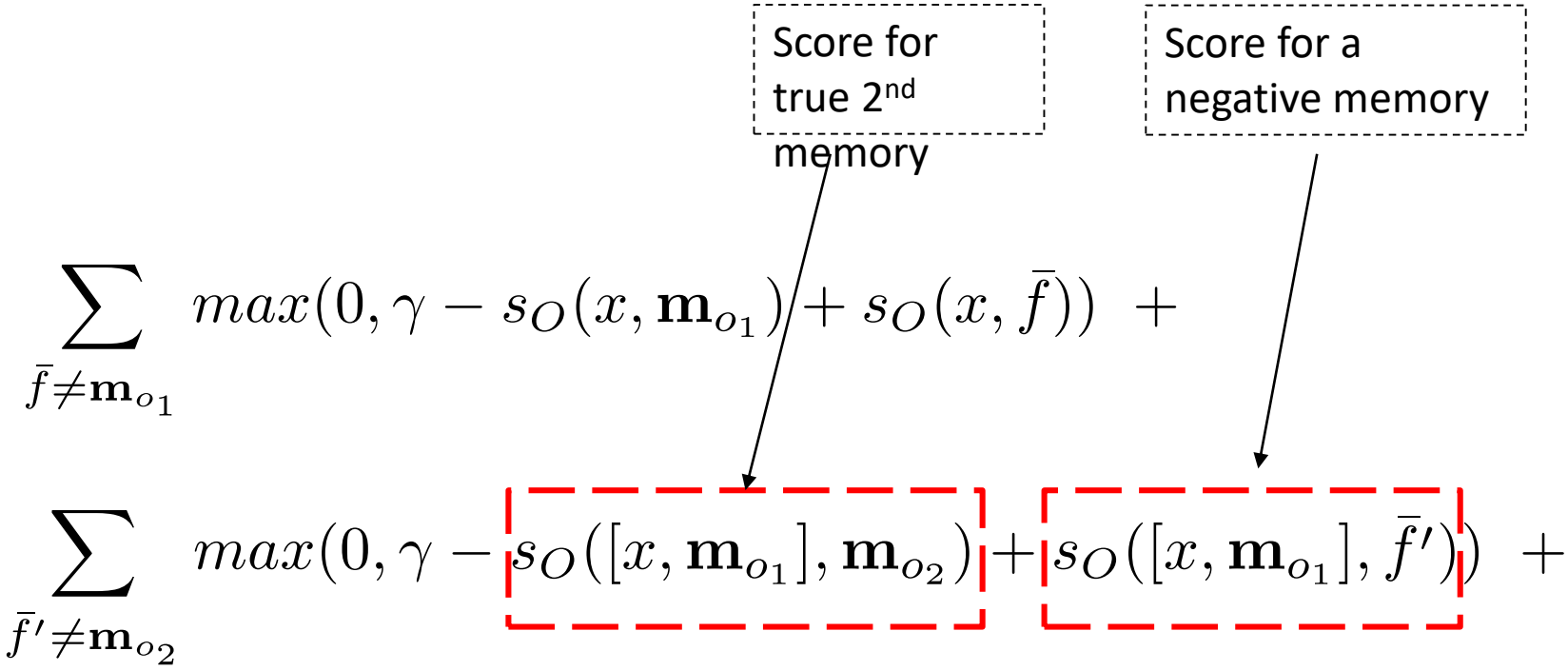
Memory Networks, Weston et. al., ICLR 2015

Training Objective



Memory Networks, Weston et. al., ICLR 2015

Training Objective



Memory Networks, Weston et. al., ICLR 2015

Training Objective

Score for true response

Score for a negative response

$$\sum_{\bar{f} \neq \mathbf{m}_{o_1}} max(0, \gamma - s_O(x, \mathbf{m}_{o_1}) + s_O(x, \bar{f})) \; +$$

$$\sum_{\bar{f}' \neq \mathbf{m}_{o_2}} max(0, \gamma - s_O([x, \mathbf{m}_{o_1}], \mathbf{m}_{o_2}) + s_O([x, \mathbf{m}_{o_1}], \bar{f}')) \; +$$

$$\sum_{\bar{r} \neq r} max(0, \gamma - s_R([x, \mathbf{m}_{o_1}, \mathbf{m}_{o_2}], r) + s_R([x, \mathbf{m}_{o_1}, \mathbf{m}_{o_2}], \bar{r}))$$

Memory Networks, Weston et. al., ICLR 2015

Experiment

- Large – Scale QA
  - 14M Statements – *(subject, relation, object)*
  - Memory Hops; $k = 1$
  - Only re-ranked candidates from other system

Stored as memories

Output is highest scoring memory

Method	F1
Fader et. al. 2013	0.54
Bordes et. al. 2014b	0.73
Memory Networks (This work)	0.72

Why does Memory Network perform exactly as previous model?

Memory Networks, Weston et. al., ICLR 2015

Experiment

- Large – Scale QA
  - 14M Statements – *(subject, relation, object)*
  - Memory Hops; $k = 1$
  - Only re-ranked candidates from other system

Method	F1
Fader et al. 2013	0.54
Bordes et al. 2014b	0.73
Memory Networks (This work)	0.72

USELESS EXPERIMENT

Why does Memory Networks not perform as well?

Useful Experiment

- Simulated World QA
  - 4 characters, 3 objects, 5 rooms
  - 7k statements, 3k questions for training and same for testing
  - Difficulty 1 (5) – Entity in question is mentioned in last 1 (5) sentences
  - For $k = 2$, annotation has intermediate best *memories* as well

Joe went to the garden then Fred picked up the milk; Joe moved to the bathroom and Fred dropped the milk, and then Dan moved to the living_room.
Where is Dan? A: living room I believe
Where is Joe? A: the bathroom
Fred moved to the bedroom and Joe went to the kitchen then Joe took the milk there and Dan journeyed to the bedroom; Joe discarded the milk.
Where is the milk now ? A: the milk is in the kitchen
Where is Dan now? A: I think he is in the bedroom
Joe took the milk there, after that Mike travelled to the office, then Joe went to the living_room, next Dan went back to the kitchen and Joe travelled to the office.
Where is Joe now? A: I think Joe is in the office

Memory Networks, Weston et. al., ICLR 2015

Limitations

- Simple BOW representation
- Simulated Question Answering dataset is too trivial
- Strong supervision i.e. for intermediate memories is needed

Memory Networks, Weston et. al., ICLR 2015

End-to-End Memory Networks (MemN2N)

- What if the annotation is:
  - Input sentences $x_1, x_2, \ldots, x_n$
  - Query q
  - Answer a

Joe went to the kitchen. Fred went to the kitchen. Joe picked up the milk. Joe travelled to his office. Joe left the milk. Joe went to the bathroom.

Where is the milk now?

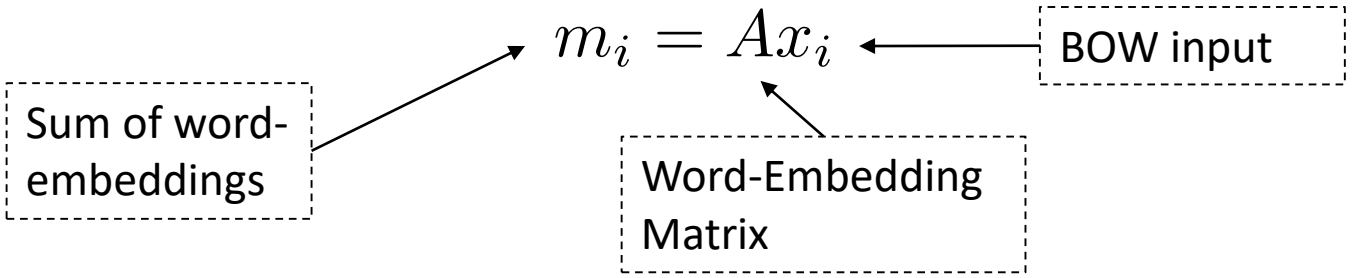
Office

- Model performs by:
  - Generating memories from inputs
  - Transforming query into suitable representation
  - Process query and memories jointly using multiple hops to produce the answer
  - Backpropagate through the whole procedure

End-To-End Memory Networks, Sukhbaatar et. al., NIPS 2015

End-to-End Memory Networks (MemN2N)

1. Convert input to memories $x_i \rightarrow m_i$



2. Transform query q into same representation space

$$u = Bq$$

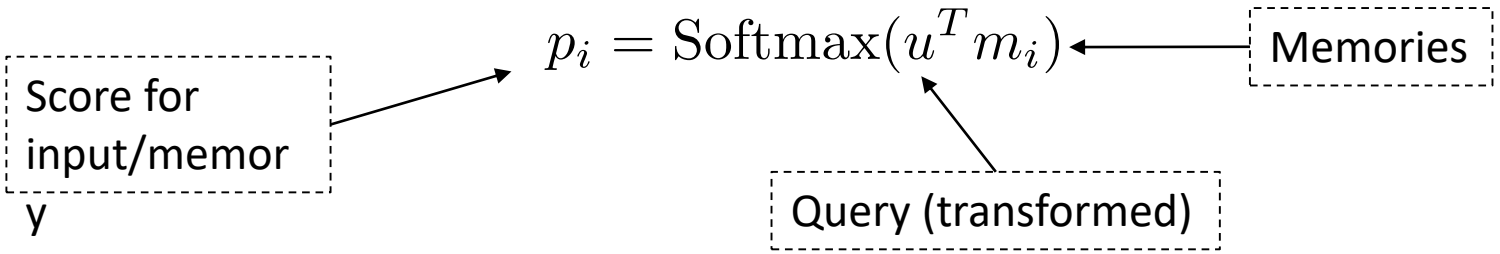
3. Output Vectors $x_i \rightarrow c_i$

$$c_i = Cx_i$$

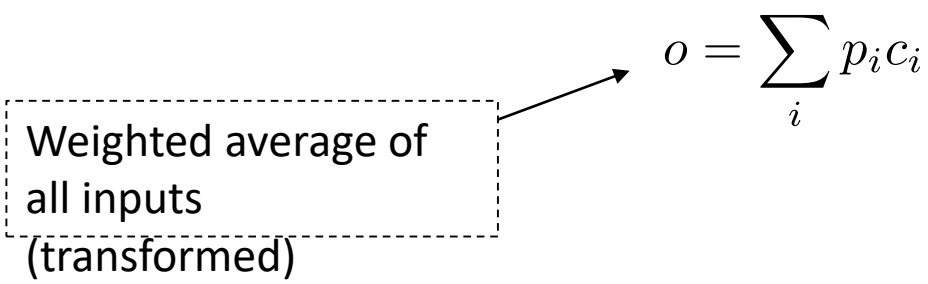
End-To-End Memory Networks, Sukhbaatar et. al., NIPS 2015

End-to-End Memory Networks (MemN2N)

3. Scoring memories against query



4. Generate output



End-To-End Memory Networks, Sukhbaatar et. al., NIPS 2015

End-to-End Memory Networks (MemN2N)

5. Generating Response

$$\hat{a} = \text{Softmax}(W(u + o))$$

Distribution over response words

Query

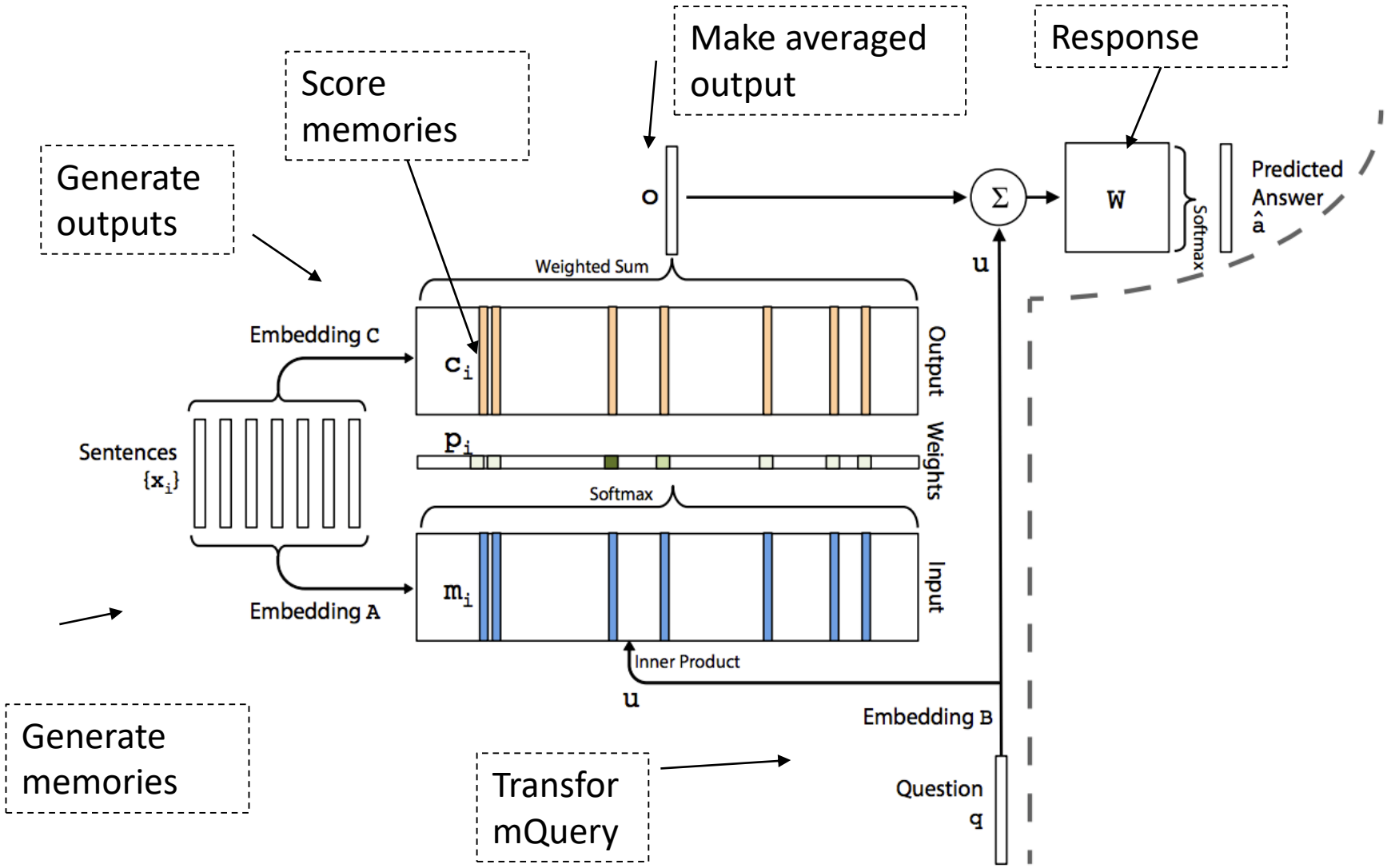
Averaged-output

Training Objective – Maximum Likelihood / Cross Entropy

$$\hat{\Theta} = \text{argmax} \sum_{s=1}^{N} \log P(\hat{a}_s)$$

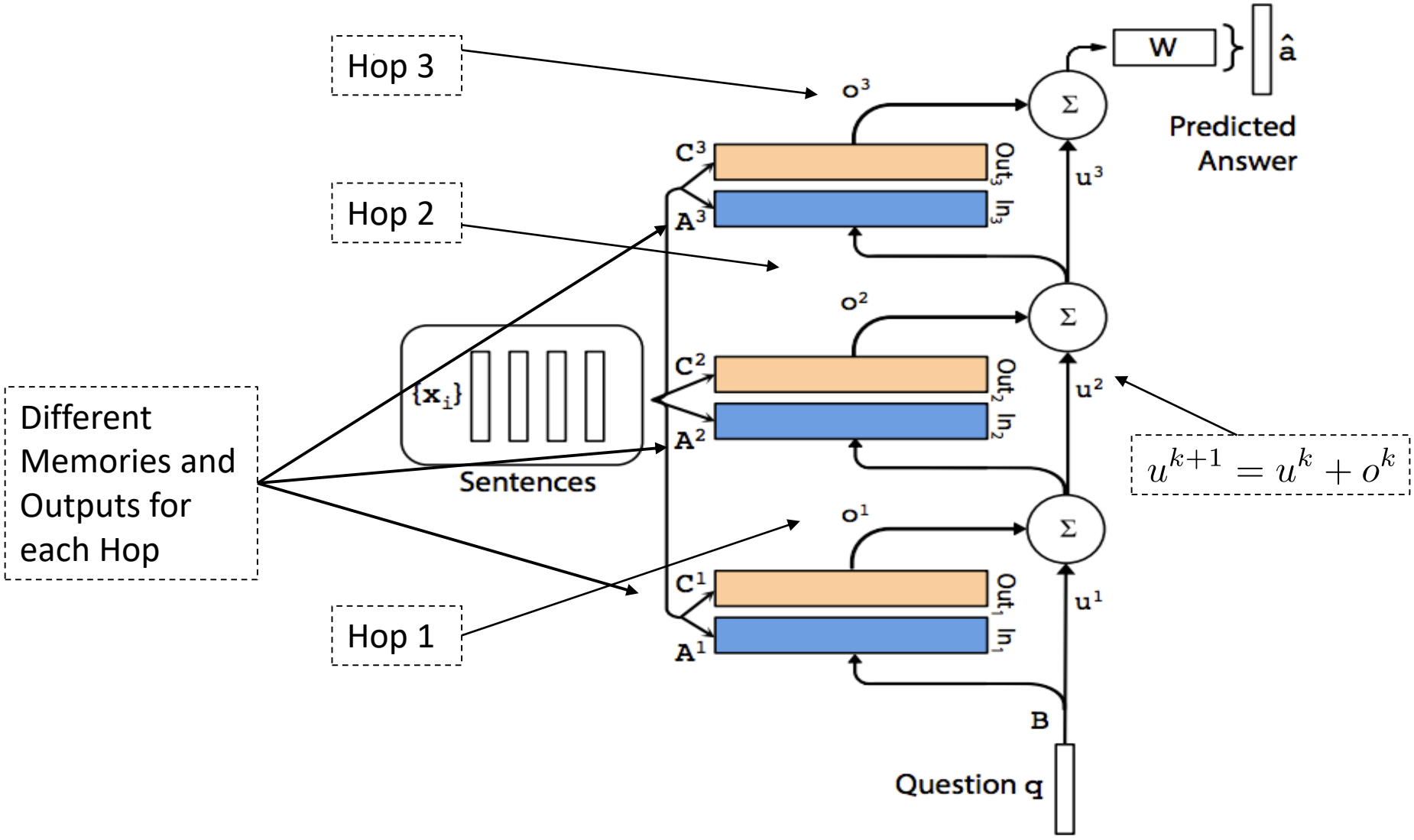
End-To-End Memory Networks, Sukhbaatar et. al., NIPS 2015

End-to-End Memory Networks (MemN2N)



End-To-End Memory Networks, Sukhbaatar et. al., NIPS 2015

Multi-hop MemN2N



End-To-End Memory Networks, Sukhbaatar et. al., NIPS 2015

Experiments

- Simulated World QA
  - 20 Tasks from bAbI dataset - 1K and 10K instances per task
  - Vocabulary = 177 words only!!!!!
  - 60 epochs
  - Learning Rate annealing
  - Linear Start with different learning rate
  - *"Model diverged very often, hence trained multiple models"*

End-To-End Memory Networks, Sukhbaatar et. al., NIPS 2015

Experiments

Story (1: 1 supporting fact)	Support	Hop 1	Hop 2	Hop 3
Daniel went to the bathroom.		0.00	0.00	0.03
Mary travelled to the hallway.		0.00	0.00	0.00
John went to the bedroom.		0.37	0.02	0.00
John travelled to the bathroom.	yes	0.60	0.98	0.96
Mary went to the office.		0.01	0.00	0.00
Where is John? Answer: bathroom		**Prediction: bathroom**		

Story (16: basic induction)	Support	Hop 1	Hop 2	Hop 3
Brian is a frog.	yes	0.00	0.98	0.00
Lily is gray.		0.07	0.00	0.00
Brian is yellow.	yes	0.07	0.00	1.00
Julius is green.		0.06	0.00	0.00
Greg is a frog.	yes	0.76	0.02	0.00
What color is Greg? Answer: yellow		**Prediction: yellow**		

	MemNN	MemN2N
Error % (1k)	**6.7**	**12.4**
Error % (10k)	**3.2**	**7.5**

End-To-End Memory Networks, Sukhbaatar et. al., NIPS 2015

Movie Trivia Time

- Which was *Stanley Kubricks*'s first movie?

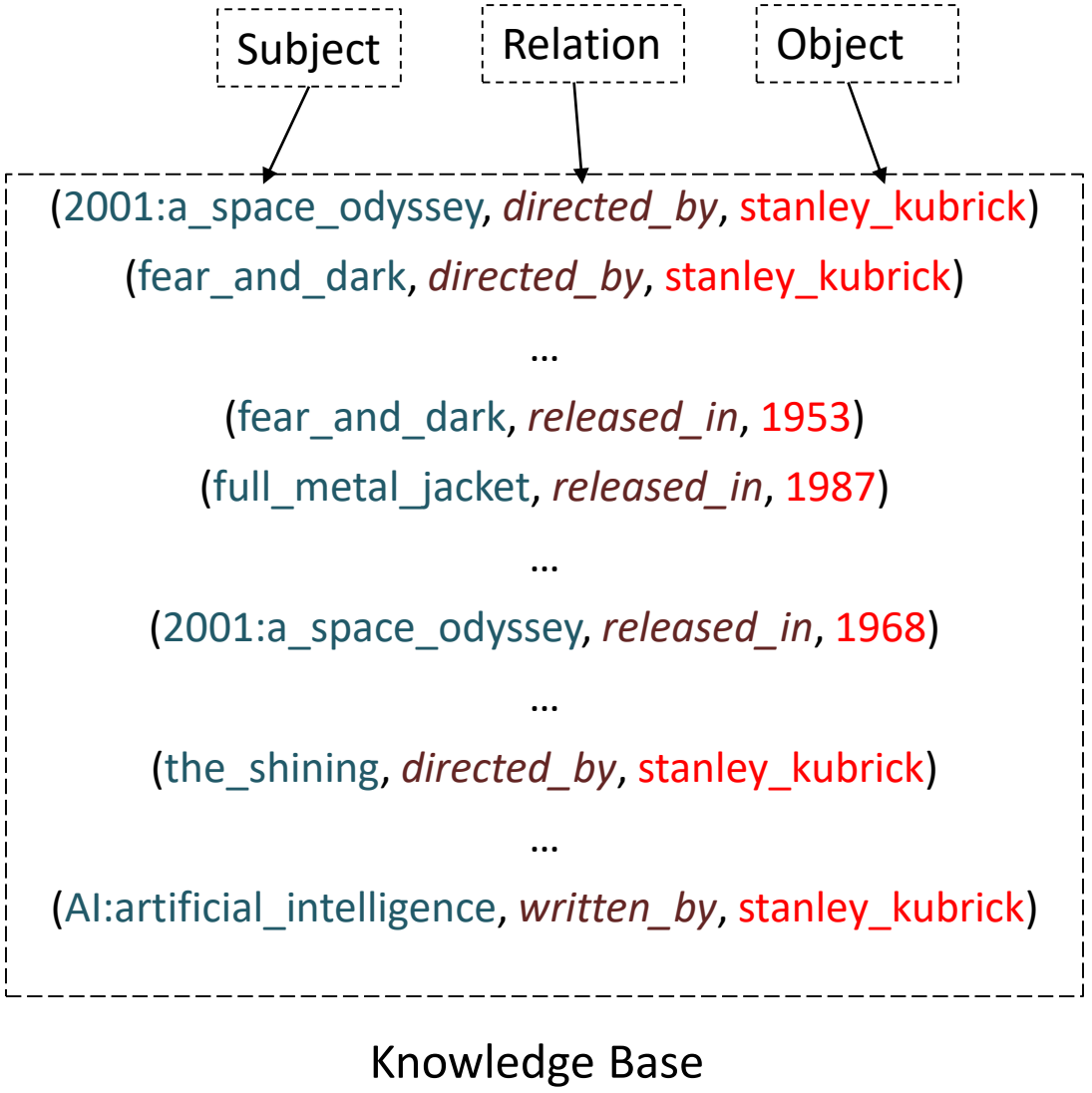
  Fear and Desire

- When did *2001:A Space Odyssey* release?

  1968

- After *The Shining*, which movie did its director direct?

  Full Metal Jacket



Knowledge Base

Combine using Memory Networks?

Knowledge Base?

(2001:a_space_odyssey, *directed_by*, stanley_kubrick)

(fear_and_dark, *directed_by*, stanley_kubrick)

…

(fear_and_dark, *released_in*, 1953)

(full_metal_jacket, *released_in*, 1987)

…

(2001:a_space_odyssey, *released_in*, 1968)

…

(the_shining, *directed_by*, stanley_kubrick)

…

(AI:artificial_intelligence, *written_by*, stanley_kubrick)

Incomplete!

Textual Knowledge?

WIKIPEDIA

Too Challenging!

Key-Value MemNNs for Reading Documents

- Structured Memories as Key-Value Pairs
  - Regular MemNNs have single vector for each memory
  - Key more related to question and values to answer

$$\text{Memories} = (k_1, v_1),\ (k_2, v_2),\ \ldots,\ (k_N, v_N)$$

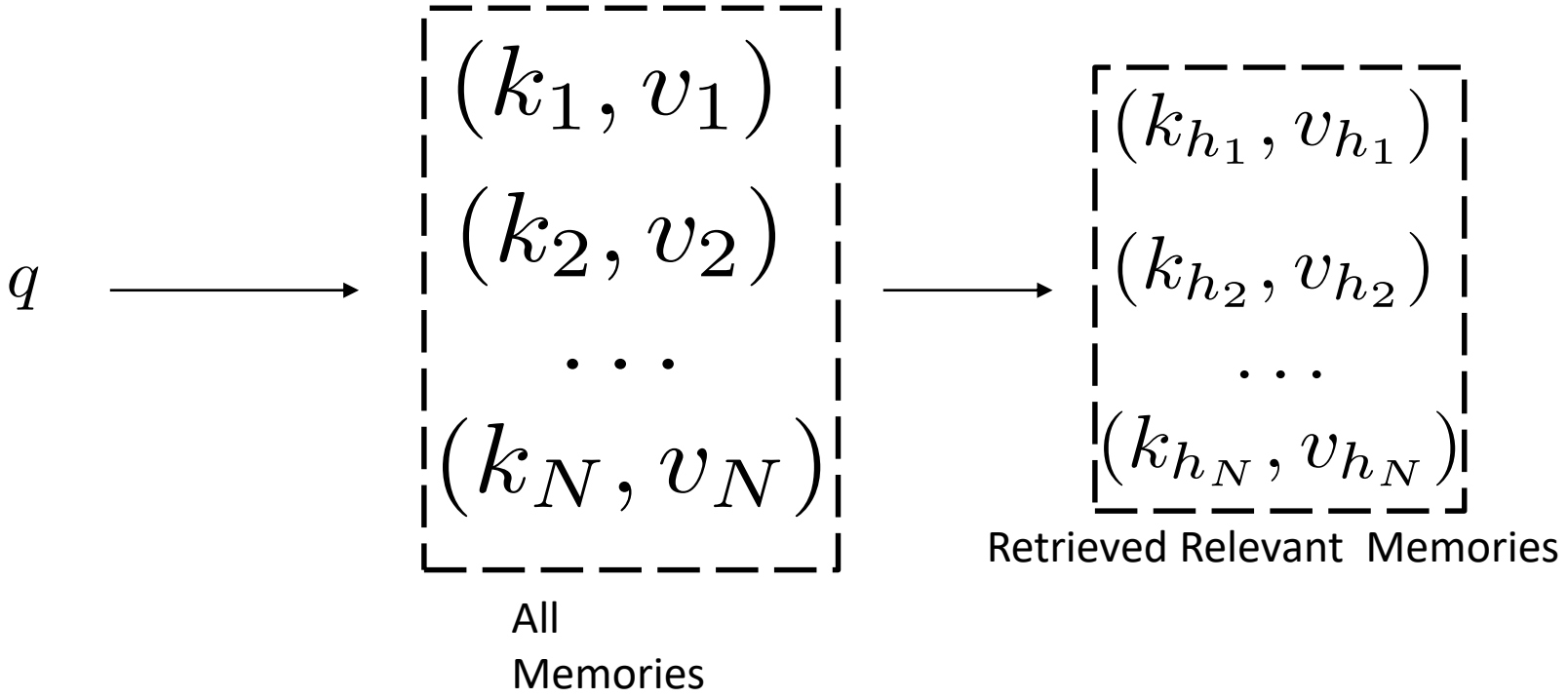
(*k: Kubrick's first movie was*, *v: Fear and Dark*)

Keys and Values can be Words, Sentences, Vectors etc.

Key-Value Memory Networks for Directly Reading Documents, Miller et. al., EMNLP 2016

Key-Value MemNNs

1. Retrieve relevant memories using Hashing Techniques



Use inverted index, locality sensitive hashing, something sensible

Key-Value Memory Networks for Directly Reading Documents, Miller et. al., EMNLP 2016

Key-Value MemNNs

2. Score Memory-Keys

$$p_i = \text{Softmax}(u^T m_i)$$

$$p_{h_i} = \text{Softmax}(A\Phi_Q(q) \cdot A\Phi_K(k_{h_i}))$$

Distribution over Memory-Keys

Dot-Prod

Sum of Embeddings

BOW

Key

3. Generate Output

$$o = \sum_i p_{h_i} A\Phi_V(v_{h_i})$$

$$o = \sum_i p_i c_i$$

Weighted average of Memory-values

KV-MemNN - Multiple Hops

In the j^{th} hop:

Query representation :

$$q_j = R_j(A\Phi_Q(q_{j-1}) + o)$$

Key Addressing

$$p_{h_i} = \text{Softmax}(A\Phi_Q(q_j) \cdot A\Phi_K(k_{h_i}))$$

Generate Response

$$\hat{a} = \text{Softmax}(A\Phi_Q(q_{H+1}) \cdot B\Phi_Y(y_i))$$

Final Hop

Key-Value Memory Networks for Directly Reading Documents, Miller et. al., EMNLP 2016

KV-MemNN – What to store in memories?

1. KB Based :

Key: (subject, relation); Value: Object

K: (2001:a_space_odyssey, *directed_by)*; V: stanley_kubrick

2. Document Based

For each entity in document, extract 5-word window around it

Key: window; Value: Entity

K: screenplay written by and; V: Hampton

Key-Value Memory Networks for Directly Reading Documents, Miller et. al., EMNLP 2016

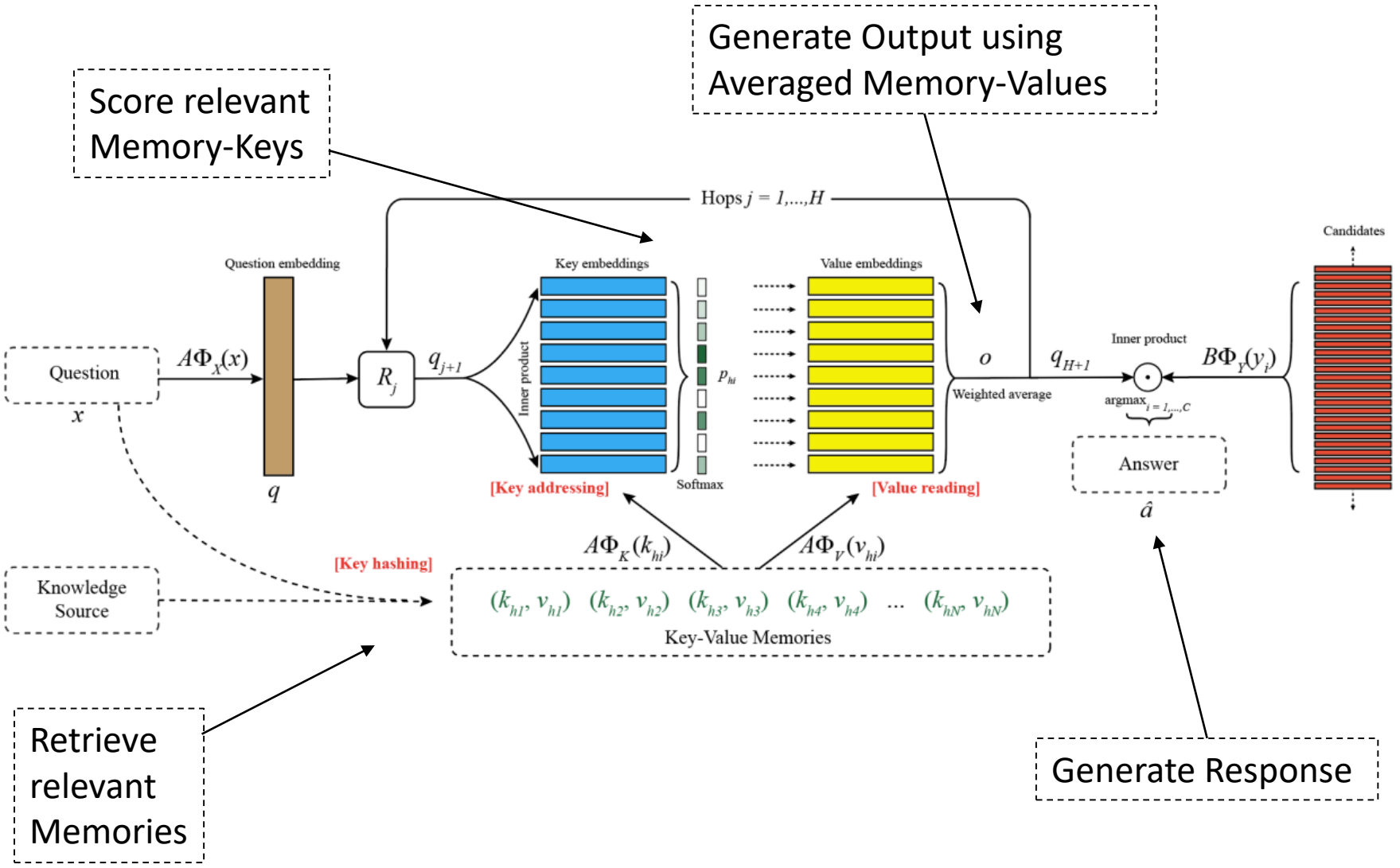
KV-MemNN – Experiments

- WikiMovies Benchmark
  - Total 100K QA-pairs
  - 10% for testing

Method	KB	Doc
E2E Memory Network	78.5	69.9
Key-Value Memory Network	93.9	76.2

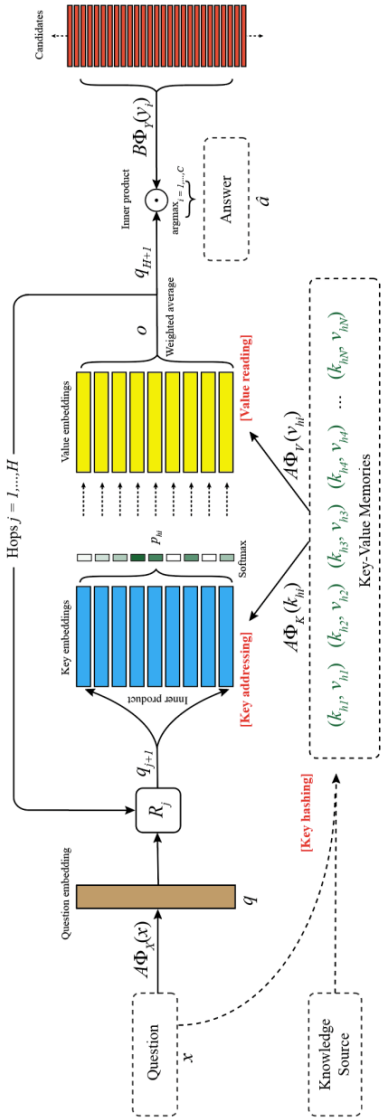
Key-Value Memory Networks for Directly Reading Documents, Miller et. al., EMNLP 2016

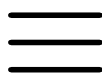
KV-MemNN

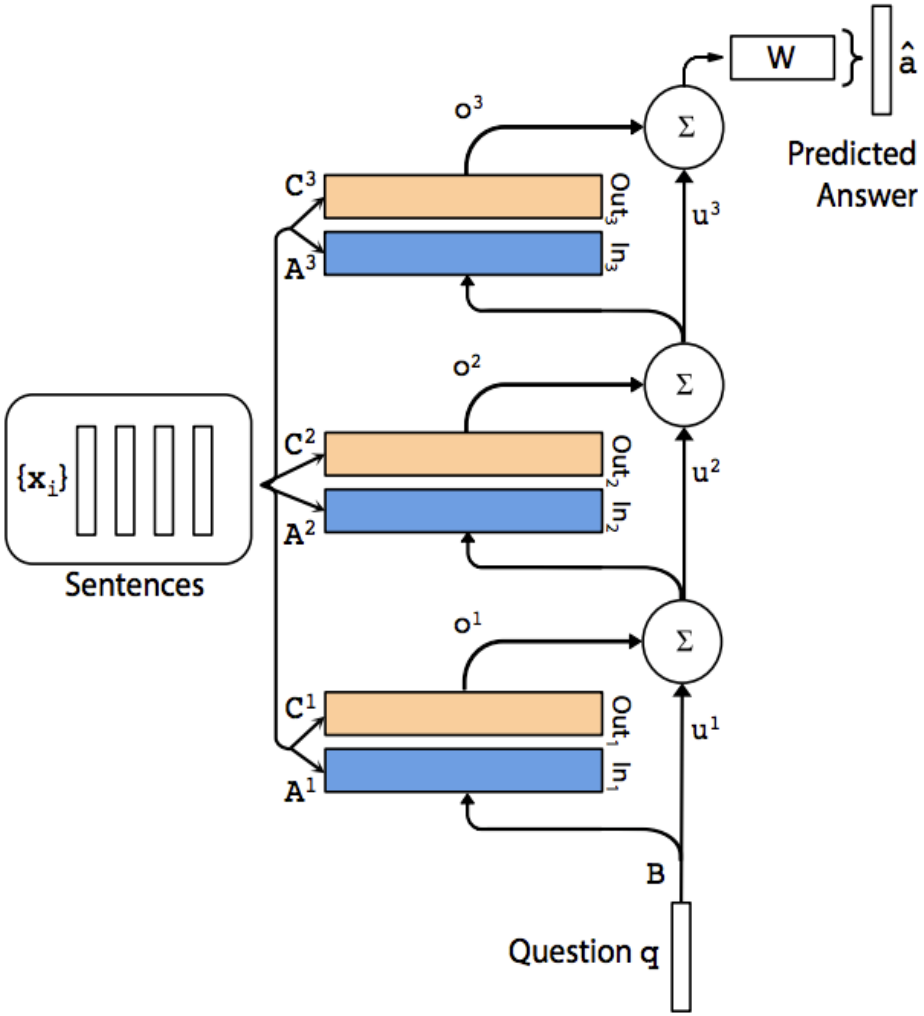


Key-Value Memory Networks for Directly Reading Documents, Miller et. al., EMNLP 2016

KV-MemNN





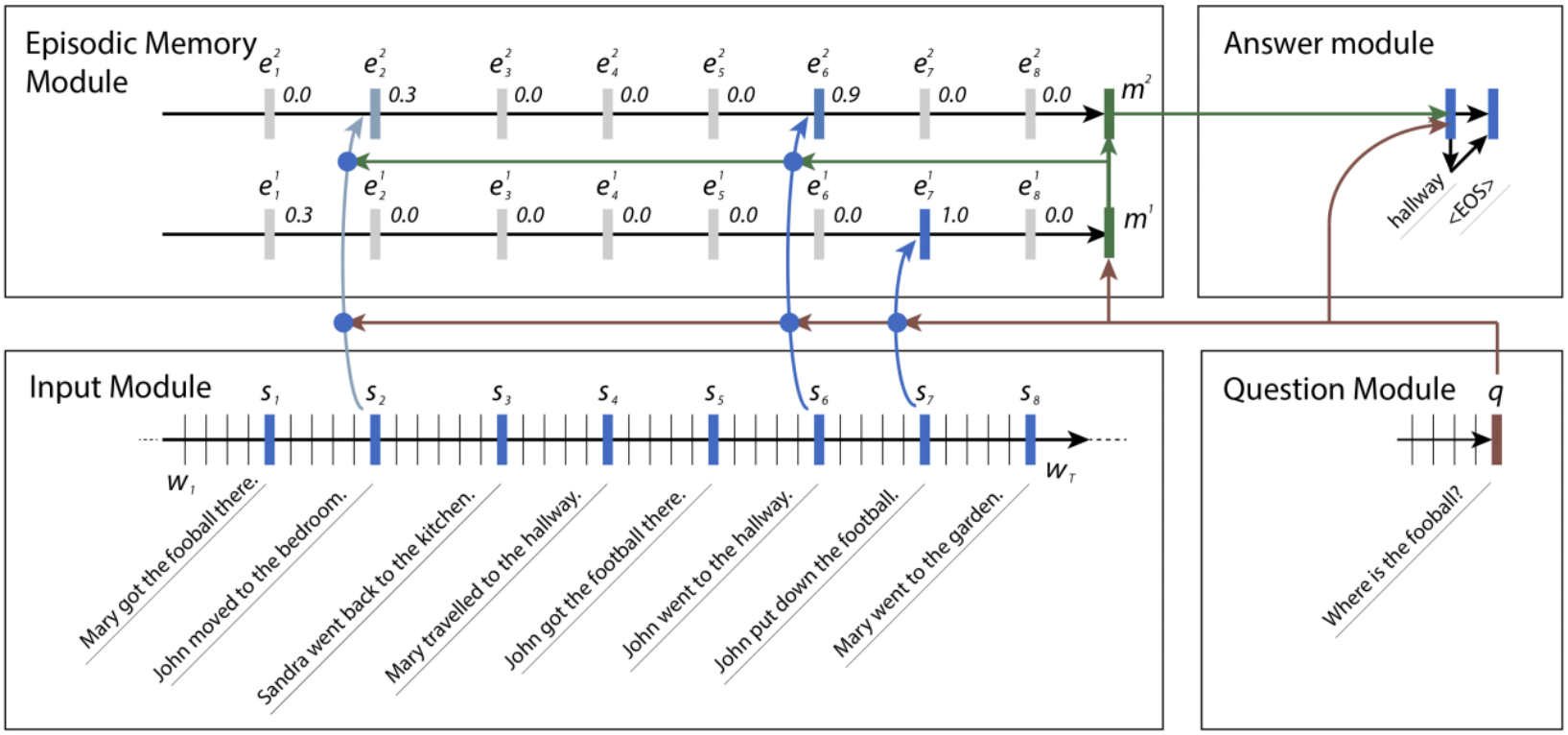


Key-Value Memory Networks for Directly Reading Documents, Miller et. al., EMNLP 2016

CNN : Computer Vision :: RNN : NLP

Key-Value Memory Networks for Directly Reading Documents, Miller et. al., EMNLP 2016

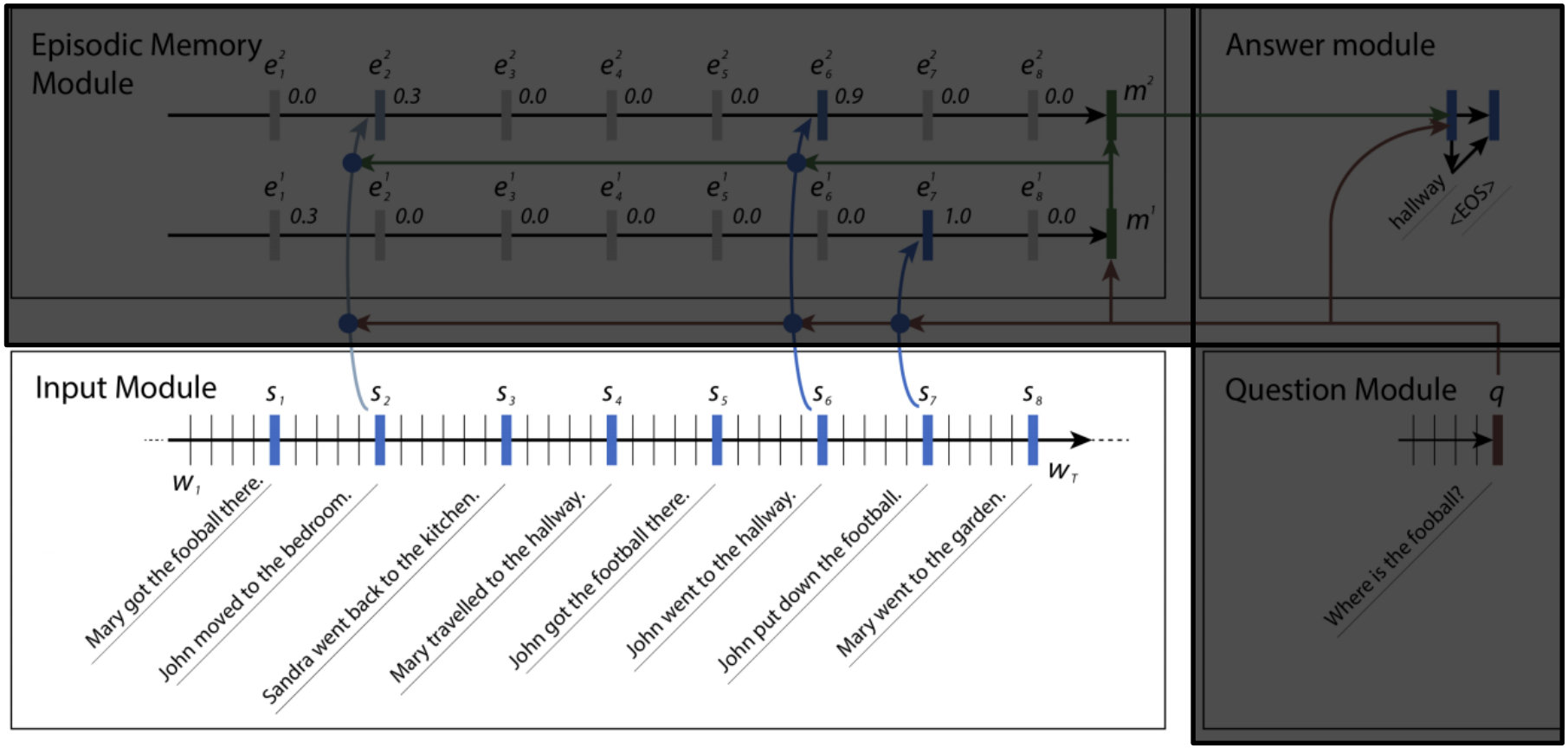
Dynamic Memory Networks – The Beast



Use RNNs, specifically GRUs for every module

Ask Me Anything: Dynamic Memory Networks for Natural Language Processing, Kumar et. al. ICML 2016

Dynamic Memory Networks

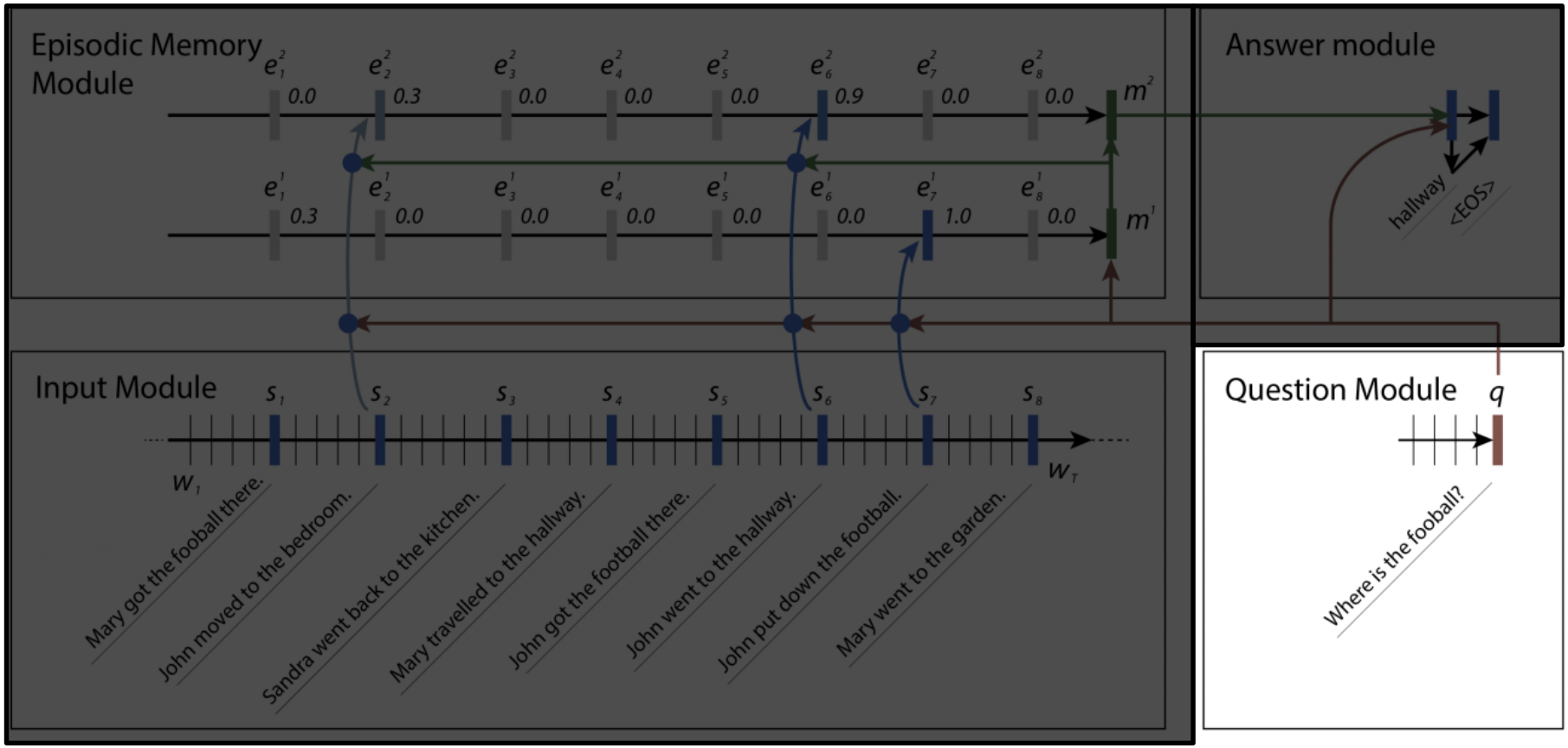


Final GRU Output for t^{th} sentence

$$c_t = \mathrm{GRU}(w_t^i, c_t^{i-1})$$

Ask Me Anything: Dynamic Memory Networks for Natural Language Processing, Kumar et. al. ICML 2016

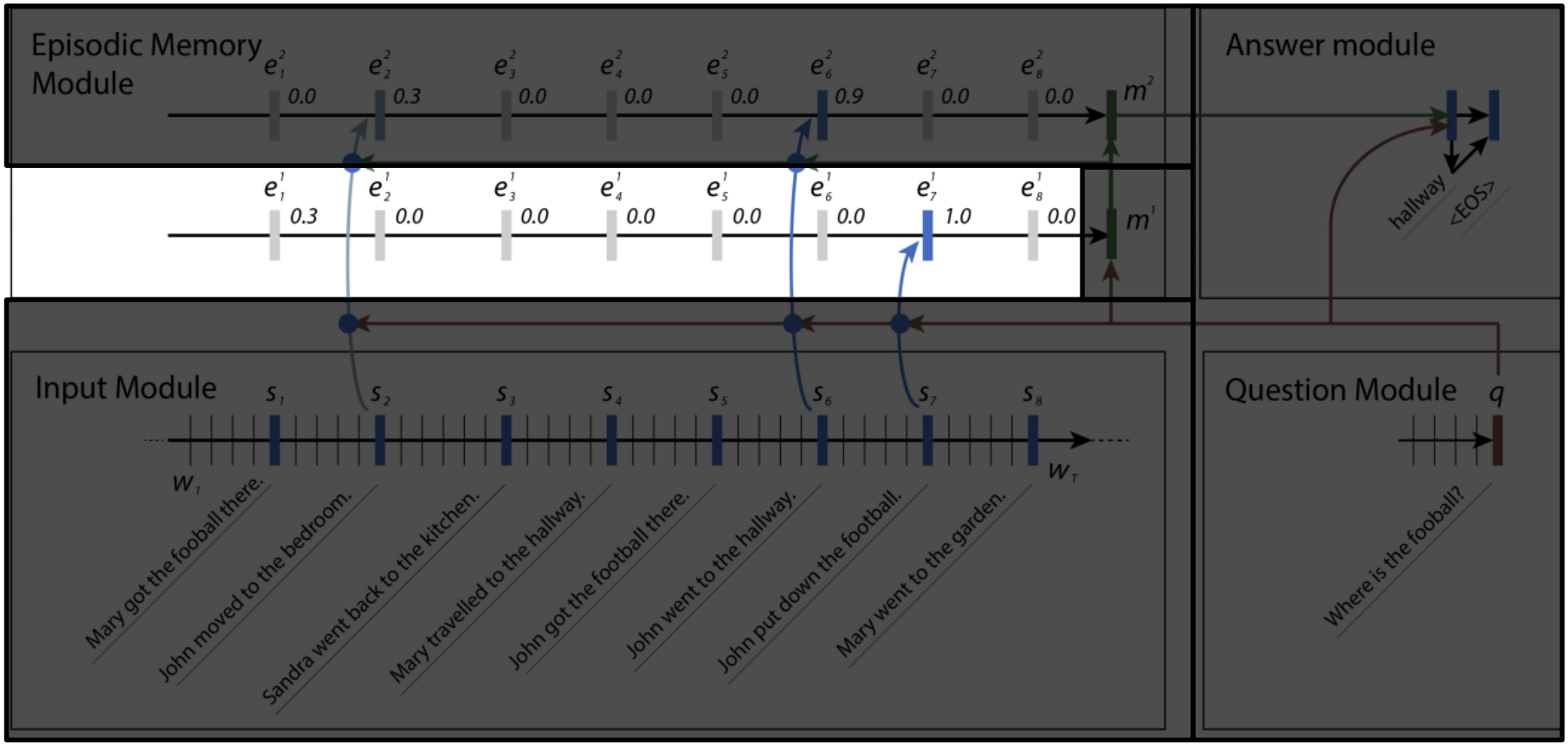
Dynamic Memory Networks



$$q = \mathrm{GRU}(q_w^i, q^{i-1})$$

Ask Me Anything: Dynamic Memory Networks for Natural Language Processing, Kumar et. al. ICML 2016

Dynamic Memory Networks



$Hop = i$

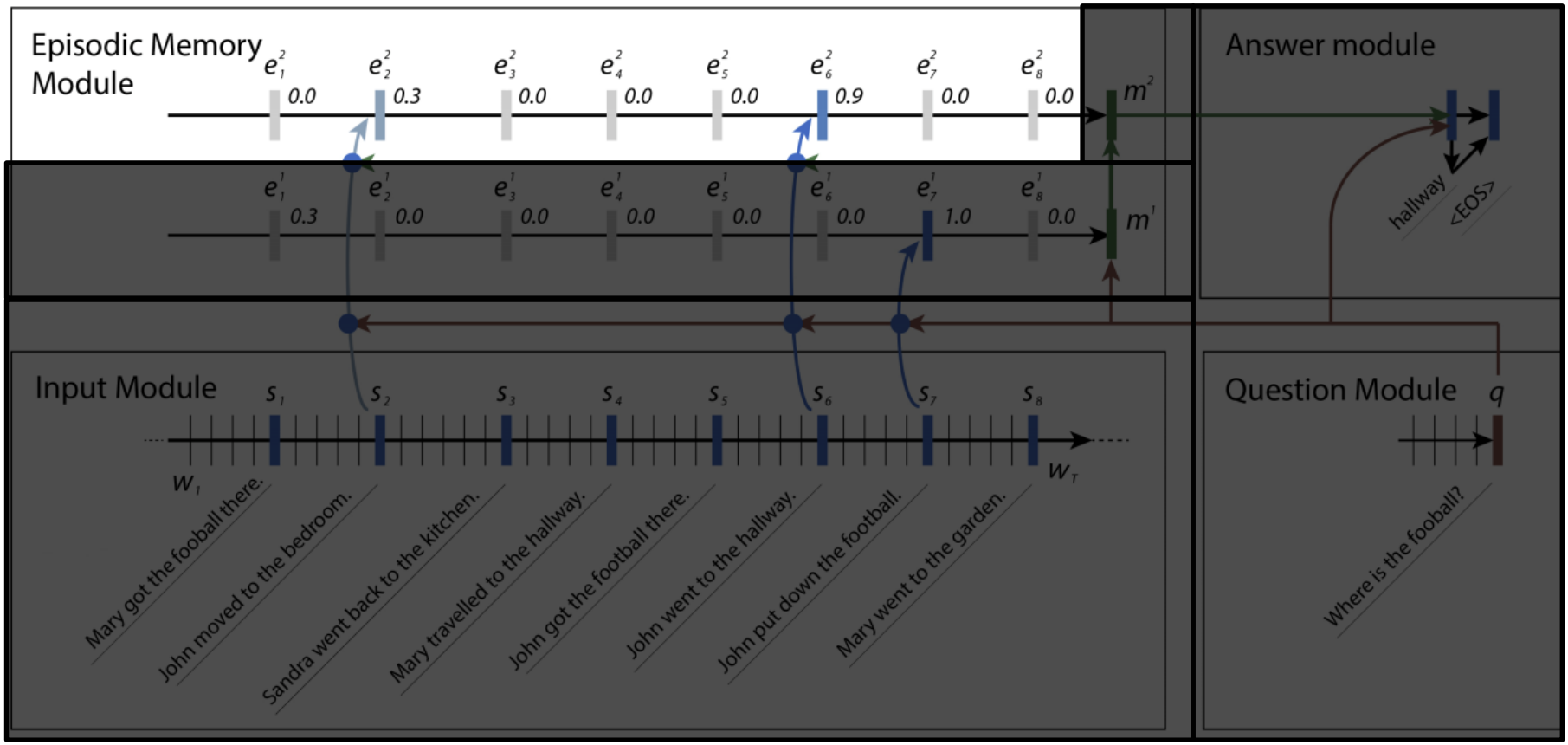
$i = 1$

$$h_t^i = g_t^i \mathrm{GRU}(c_t, h_{t-1}^i) + (1 - g_t^i) h_{t-1}^i$$

$$e^i = h_{T_C}^i$$

Ask Me Anything: Dynamic Memory Networks for Natural Language Processing, Kumar et. al. ICML 2016

Dynamic Memory Networks



$Hop = i$

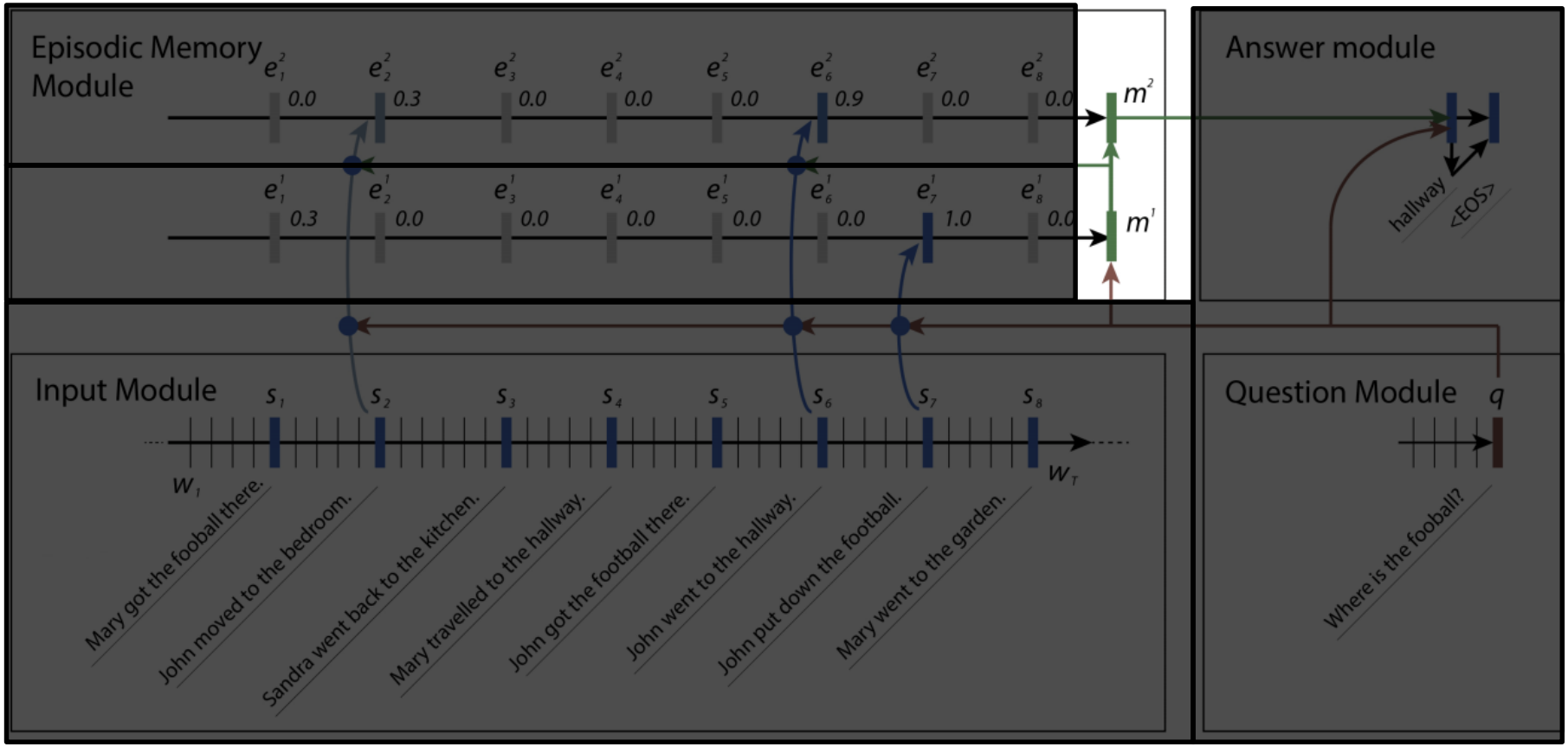
$i = 2$

$$h_t^i = g_t^i \mathrm{GRU}(c_t, h_{t-1}^i) + (1 - g_t^i) h_{t-1}^i$$

$$e^i = h_{T_C}^i$$

Ask Me Anything: Dynamic Memory Networks for Natural Language Processing, Kumar et. al. ICML 2016

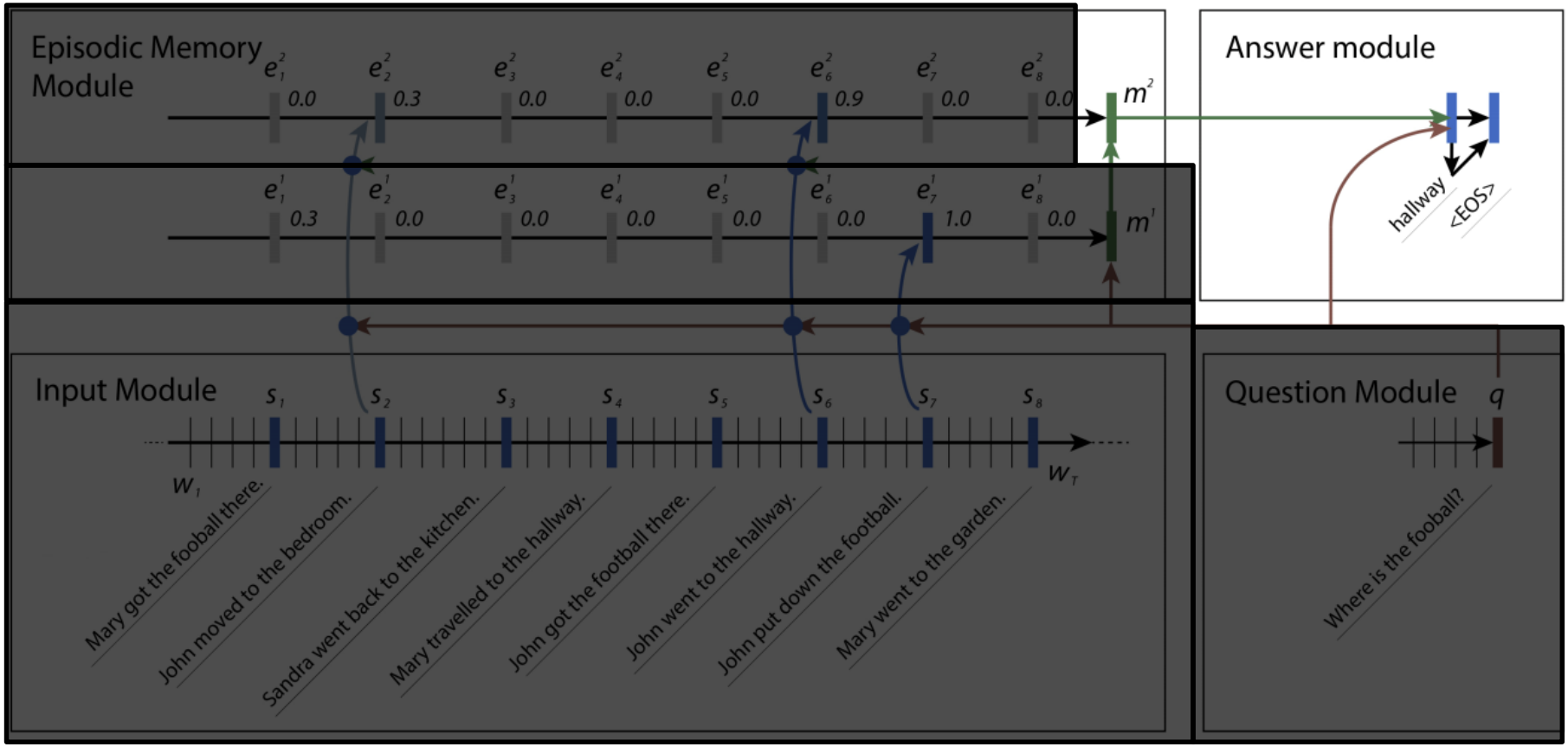
Dynamic Memory Networks



$$m^i = \text{GRU}(e^i, m^{i-1})$$

Ask Me Anything: Dynamic Memory Networks for Natural Language Processing, Kumar et. al. ICML 2016

Dynamic Memory Networks

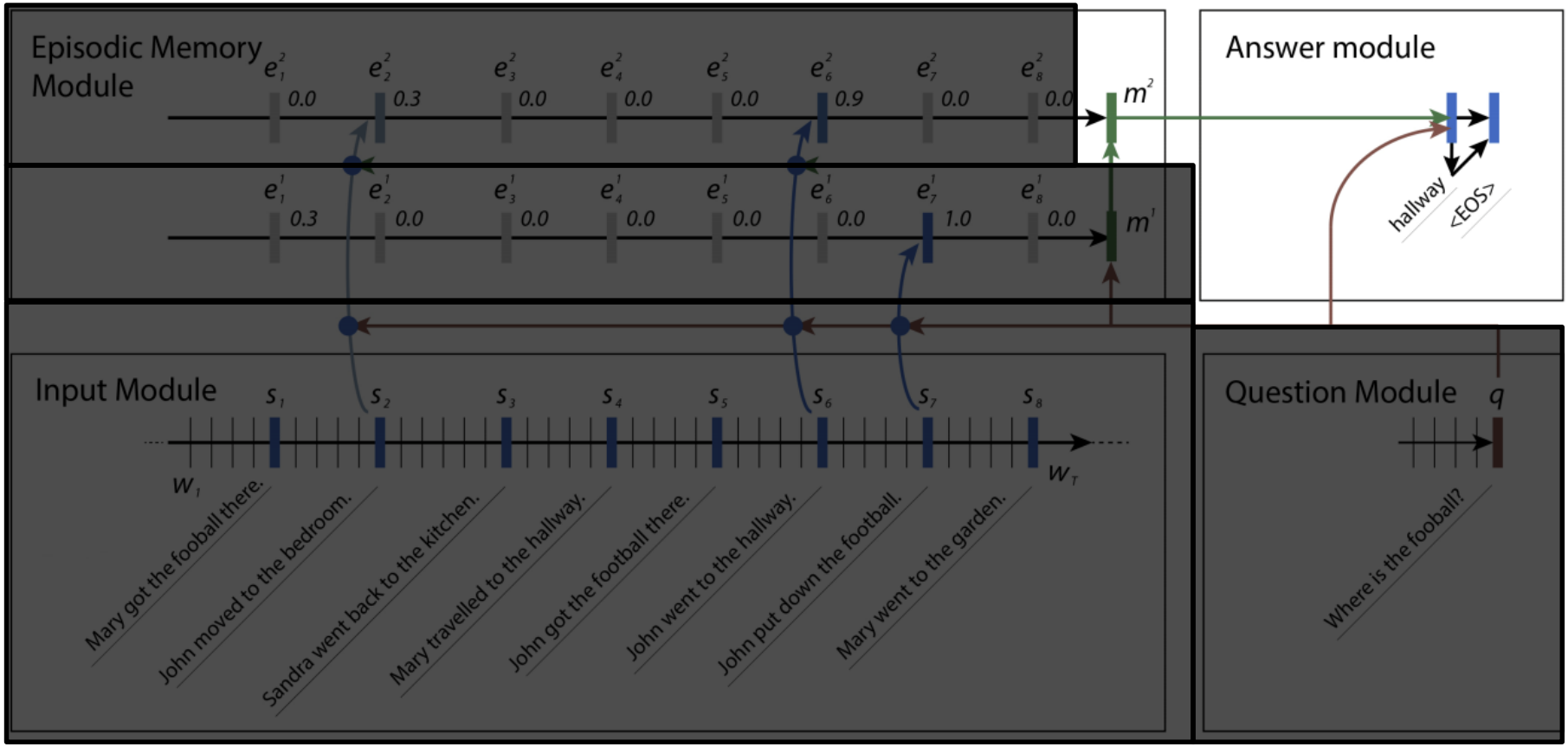


$$y_t = \text{Softmax}(W^{(a)}\alpha_t) \qquad \alpha_0 = m^{T_m}$$

$$\alpha_t = \text{GRU}([y_{t-1}, q], \alpha_{t-1})$$

Ask Me Anything: Dynamic Memory Networks for Natural Language Processing, Kumar et. al. ICML 2016

Dynamic Memory Networks



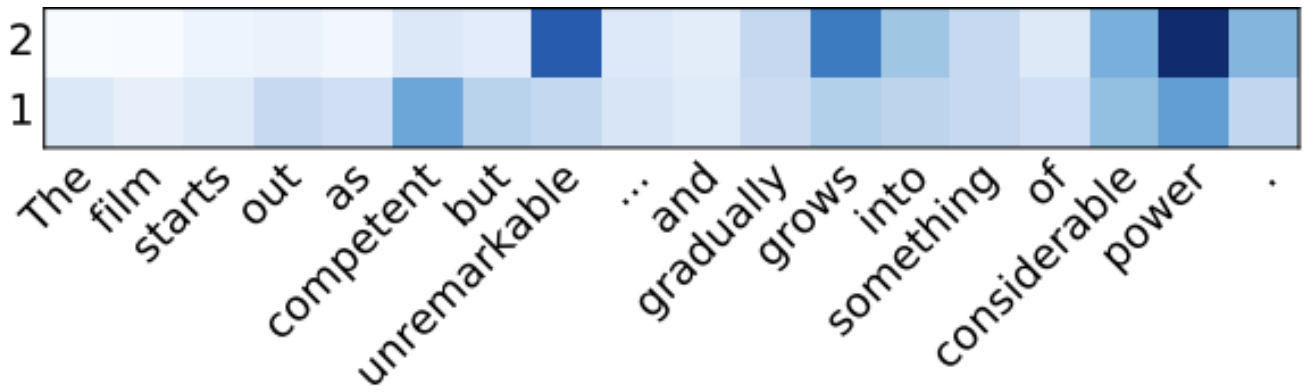
How many GRUs were used with 2 hops?

Ask Me Anything: Dynamic Memory Networks for Natural Language Processing, Kumar et. al. ICML 2016

DMN – Qualitative Results

Question: Where was Mary before the Bedroom?
Answer: Cinema.

Facts	Episode 1	Episode 2	Episode 3
Yesterday Julie traveled to the school.			
Yesterday Marie went to the cinema.		■	
This morning Julie traveled to the kitchen.			
Bill went back to the cinema yesterday.			
Mary went to the bedroom this morning.	■		
Julie went back to the bedroom this afternoon.			
[done reading]			■



Ask Me Anything: Dynamic Memory Networks for Natural Language Processing, Kumar et. al. ICML 2016

Algorithm Learning

Neural Turing Machine

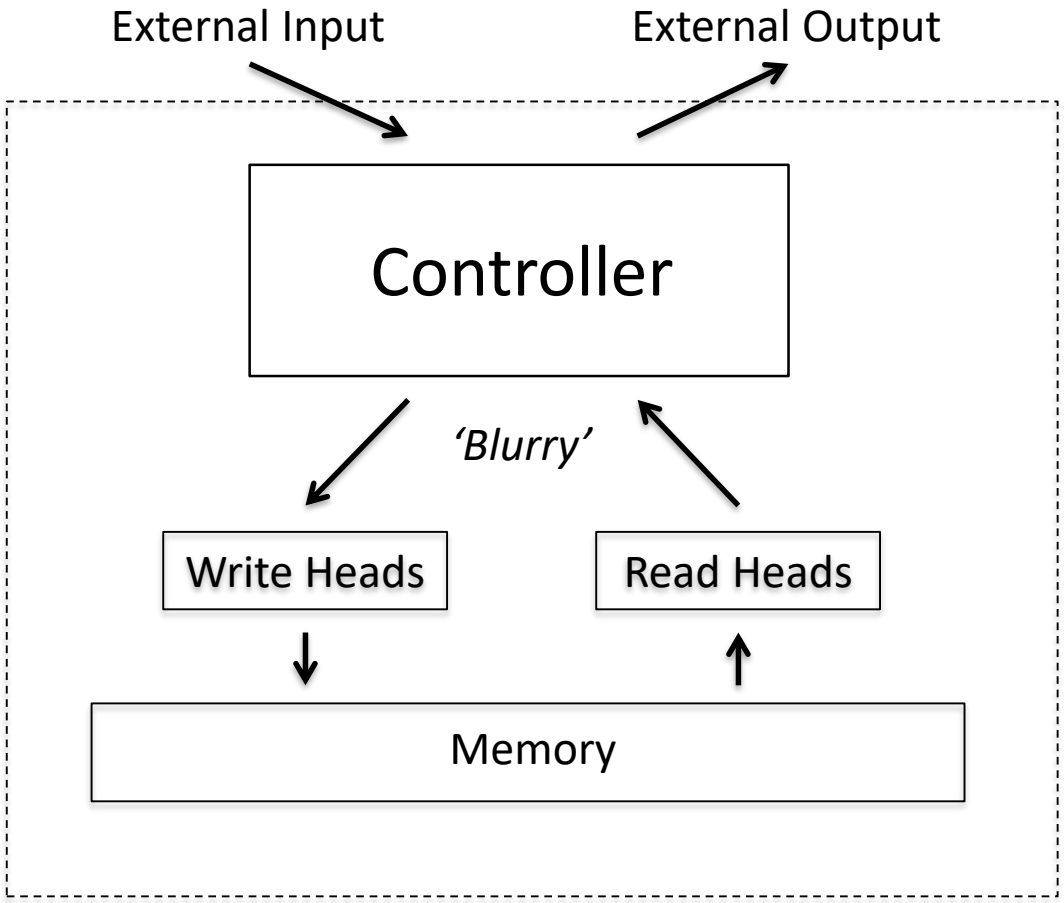
Copy Task: Implement the Algorithm

Given a list of numbers at input, reproduce the list at output

Neural Turing Machine Learns:

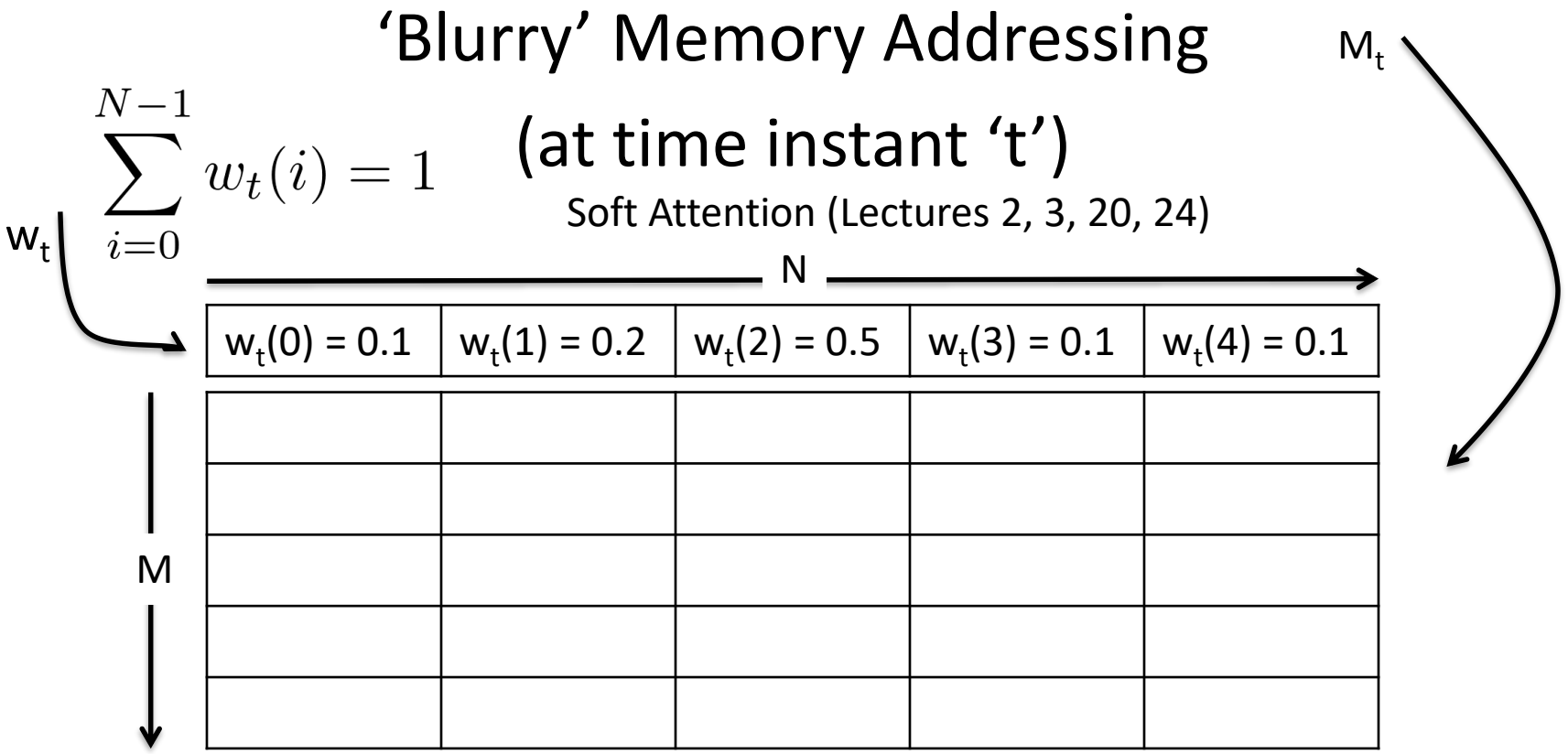
1. What to write to memory
2. When to write to memory
3. When to stop writing
4. Which memory cell to read from
5. How to convert result of read into final output

Neural Turing Machine



Neural Turing Machines, Graves et. al., arXiv:1410.5401

Neural Turing Machine



Neural Turing Machines, Graves et. al., arXiv:1410.5401

Neural Turing Machine

More formally,

Blurry Read Operation

Given: M_t (memory matrix) of size NxM

w_t (weight vector) of length N

t (time index)

$$r_t = \sum_{i=0}^{N-1} w_t(i)\mathbf{M}_t(i)$$

Neural Turing Machines, Graves et. al., arXiv:1410.5401

Neural Turing Machines: Blurry Writes

Blurry Write Operation

Decomposed into blurry erase + blurry add

Given: M_t (memory matrix) of size NxM

w_t (weight vector) of length N

t (time index)

e_t (erase vector) of length M

a_t (add vector) of length M

$$\mathbf{M}_t(i) = \underbrace{\mathbf{M}_{t-1}(i)(1 - w_t(i)\mathbf{e}_t)}_{\text{Erase Component}} + \underbrace{w_t(i)\mathbf{a}_t}_{\text{Add Component}}$$

Neural Turing Machines, Graves et. al., arXiv:1410.5401

Neural Turing Machines: Erase

$$\mathbf{M}_t(i) = \mathbf{M}_{t-1}(i)(1 - w_t(i)\mathbf{e}_t)$$

1xN

	$w_1(0) = 0.1$	$w_1(1) = 0.2$	$w_1(2) = 0.5$	$w_1(3) = 0.1$	$w_1(4) = 0.1$	e_1
M_0	5	7	9	2	12	1.0
	11	6	3	1	2	0.7
	3	7	3	10	6	0.2
	4	2	5	9	9	0.5
	3	5	12	8	4	0.0

Mx1

Neural Turing Machines, Graves et. al., arXiv:1410.5401

Neural Turing Machines: Erase

$$\mathbf{M}_t(i) = \mathbf{M}_{t-1}(i)(1 - w_t(i)\mathbf{e}_t)$$

$w_1(0) = 0.1$	$w_1(1) = 0.2$	$w_1(2) = 0.5$	$w_1(3) = 0.1$	$w_1(4) = 0.1$

4.5	5.6	4.5	1.8	10.8
10.23	5.16	1.95	0.93	1.86
2.94	6.72	2.7	9.8	5.88
3.8	1.8	3.75	8.55	8.55
3	5	12	8	4

Neural Turing Machines, Graves et. al., arXiv:1410.5401

Neural Turing Machines: Addition

$$\mathbf{M}_t(i) = \mathbf{M}_{t-1}(i)(1 - w_t(i)\mathbf{e}_t) + w_t(i)\mathbf{a}_t$$

$w_1(0) = 0.1$	$w_1(1) = 0.2$	$w_1(2) = 0.5$	$w_1(3) = 0.1$	$w_1(4) = 0.1$	a_1
4.5	5.6	4.5	1.8	10.8	3
10.23	5.16	1.95	0.93	1.86	4
2.94	6.72	2.7	9.8	5.88	-2
3.8	1.8	3.75	8.55	8.55	0
3	5	12	8	4	2

Neural Turing Machines, Graves et. al., arXiv:1410.5401

Neural Turing Machines: Blurry Writes

$$\mathbf{M}_t(i) = \mathbf{M}_{t-1}(i)(1 - w_t(i)\mathbf{e}_t) + w_t(i)\mathbf{a}_t$$

M_1 ⇨

4.8	6.2	6	2.1	11.1
10.63	5.96	3.95	1.33	2.26
2.74	6.32	1.7	9.6	5.68
3.8	1.8	3.75	8.55	8.55
3.2	5.4	13	8.2	4.2

Neural Turing Machines, Graves et. al., arXiv:1410.5401

Neural Turing Machines: Demo

Demonstration: Training on Copy Task

Figure from Snips AI's Medium Post

Neural Turing Machines, Graves et. al., arXiv:1410.5401

Neural Turing Machines: Attention Model

Generating w_t

Content Based

Example: QA Task

- Score sentences by similarity with Question
- Weights as softmax of similarity scores

Location Based

Example: Copy Task

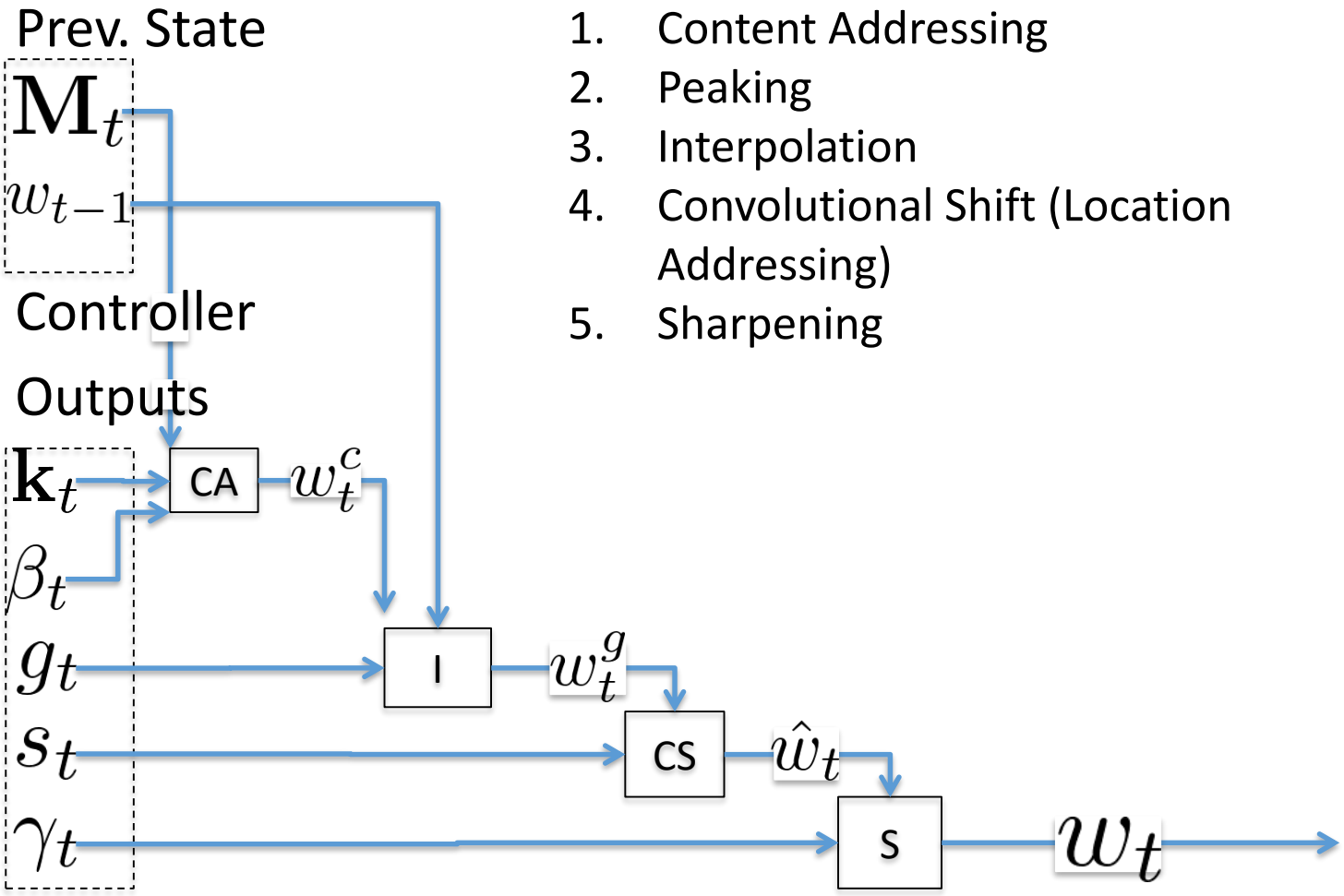
- Move to address (i+1) after writing to index (i)
- Weights ≈ Transition probabilities

Neural Turing Machines, Graves et. al., arXiv:1410.5401

Neural Turing Machines: Attention Model

Steps for generating w_t

1. Content Addressing
2. Peaking
3. Interpolation
4. Convolutional Shift (Location Addressing)
5. Sharpening



Neural Turing Machines, Graves et. al., arXiv:1410.5401

Neural Turing Machines: Attention Model

Prev. State

$\mathbf{M}_t$

Controller

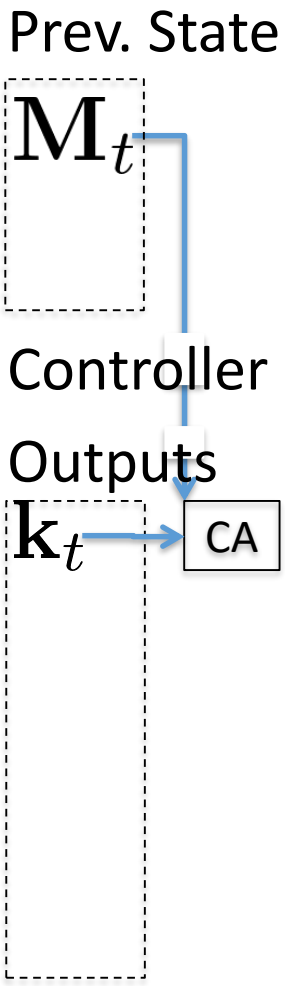
Outputs

$\mathbf{k}_t$

$\mathbf{k}_t$:Vector (length M) produced by Controller

Neural Turing Machines, Graves et. al., arXiv:1410.5401

Neural Turing Machines: Attention Model

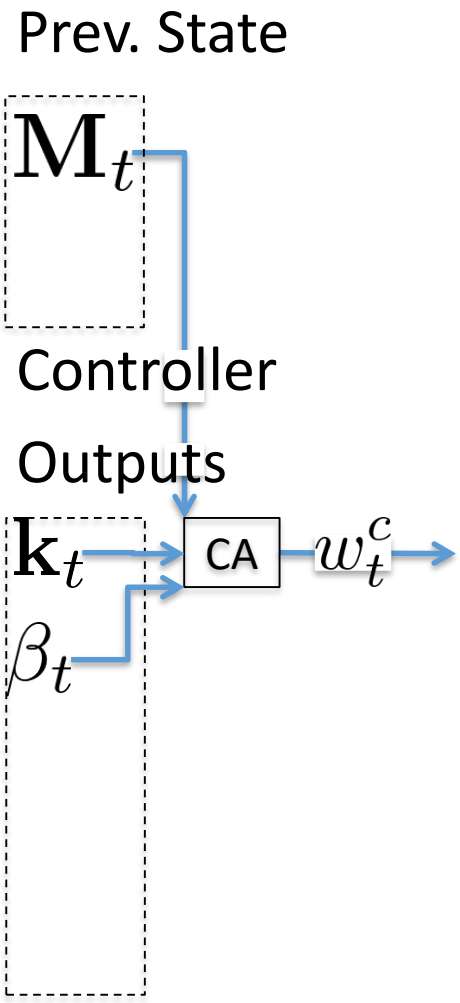


Step 1: Content Addressing (CA)

$$w_t^c(i) = \frac{exp < \mathbf{M}_t(i), \mathbf{k}_t >}{\sum_i exp < \mathbf{M}_t(i), \mathbf{k}_t >}$$

Neural Turing Machines, Graves et. al., arXiv:1410.5401

Neural Turing Machines: Attention Model

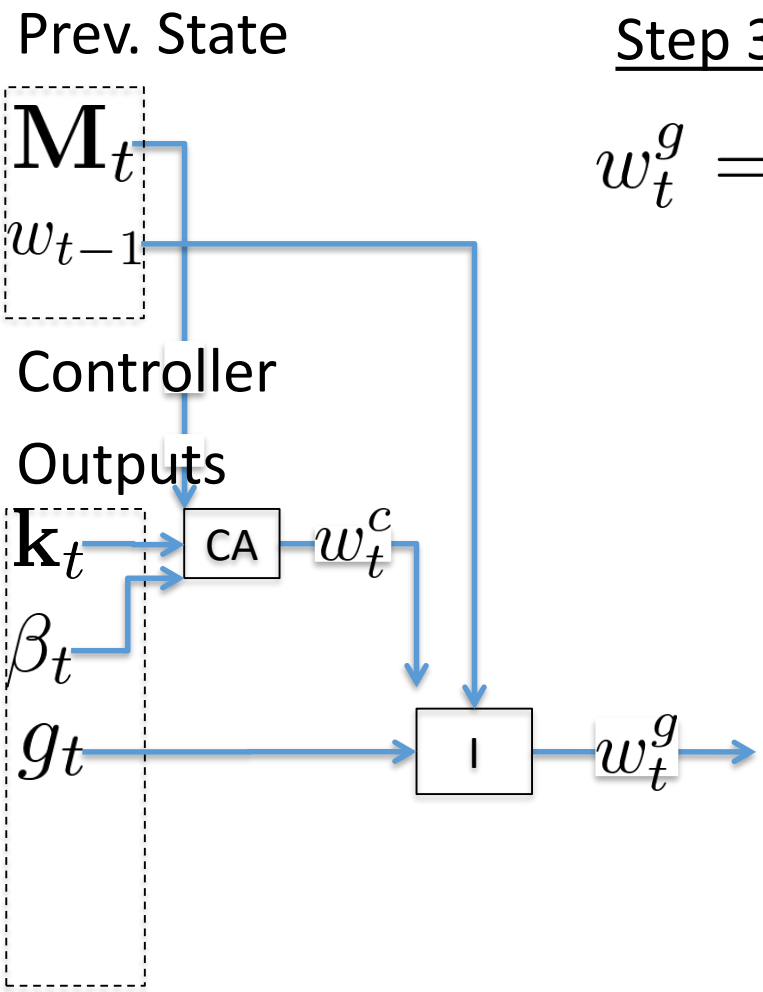


Step 2: Peaking

$$w_t^c(i) = \frac{exp(\beta_t(< \mathbf{M}_t(i), \mathbf{k}_t >))}{\sum_i exp(\beta_t(< \mathbf{M}_t(i), \mathbf{k}_t >))}$$

Neural Turing Machines, Graves et. al., arXiv:1410.5401

Neural Turing Machines: Attention Model

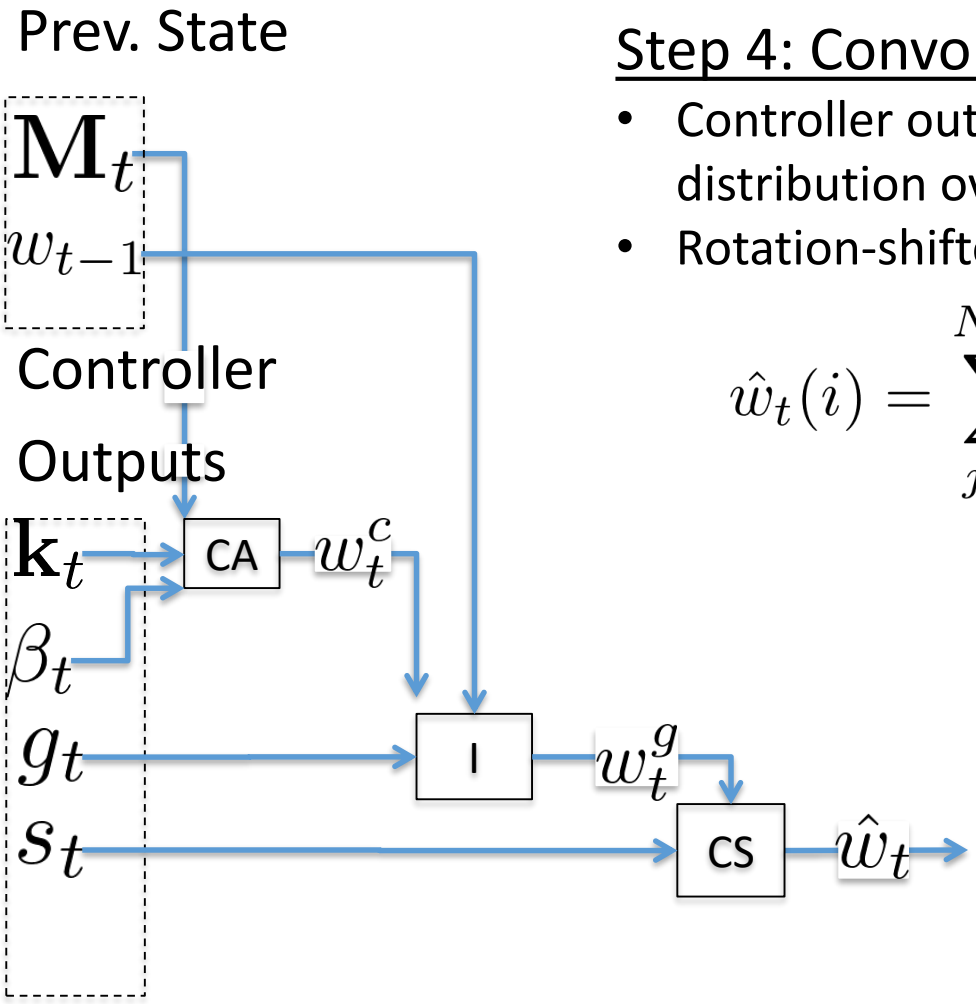


Step 3: Interpolation (I)

$$w_t^g = g_t w_t^c + (1 - g_t) w_{t-1}$$

Neural Turing Machines, Graves et. al., arXiv:1410.5401

Neural Turing Machines: Attention Model



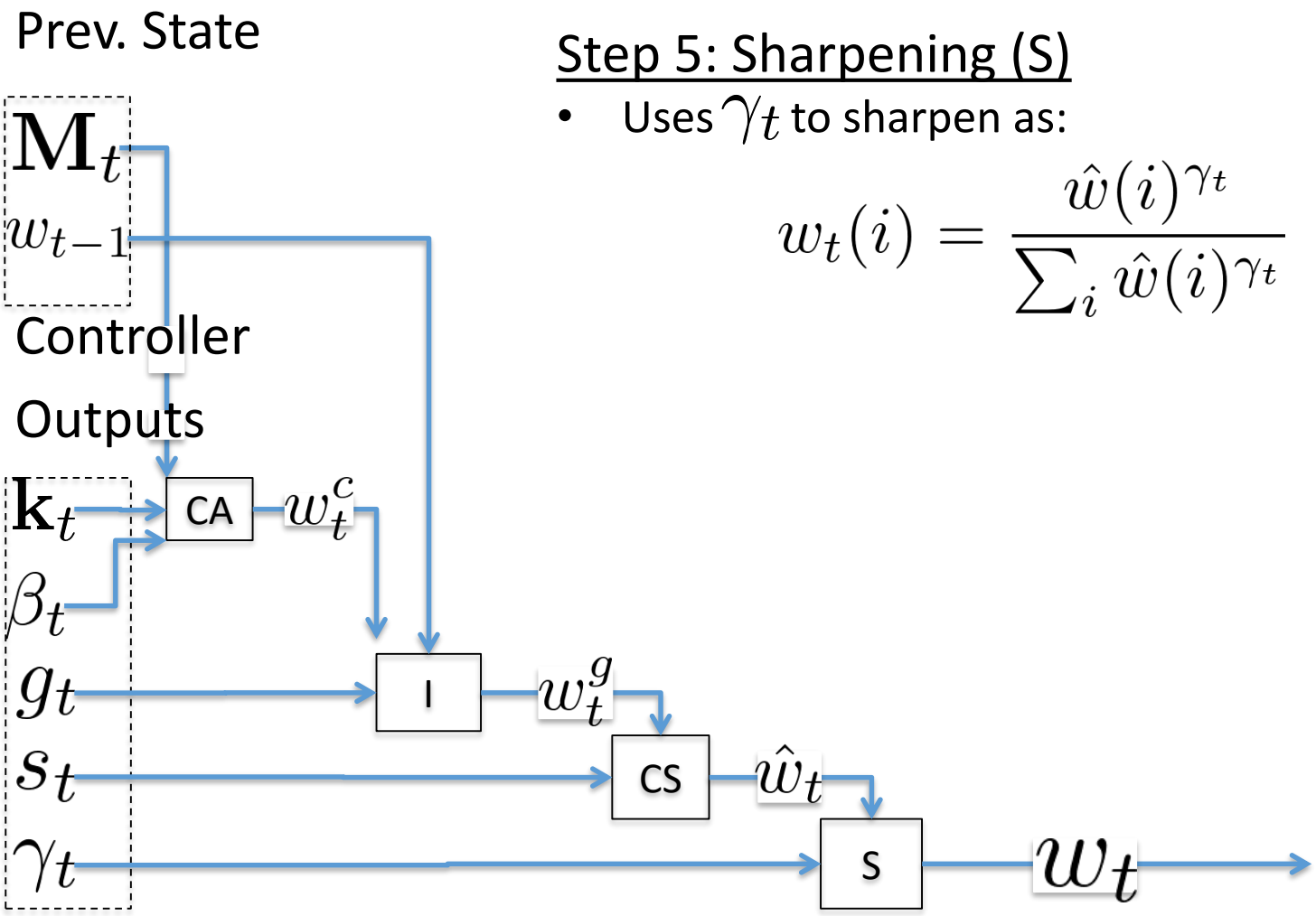
Step 4: Convolutional Shift (CS)

- Controller outputs s_t, a normalized distribution over all N possible shifts
- Rotation-shifted weights computed as:

$$\hat{w}_t(i) = \sum_{j=0}^{N-1} w_t^g(j) s_t(j-i)$$

Neural Turing Machines, Graves et. al., arXiv:1410.5401

Neural Turing Machines: Attention Model



Step 5: Sharpening (S)

- Uses γ_t to sharpen as:

$$w_t(i) = \frac{\hat{w}(i)^{\gamma_t}}{\sum_i \hat{w}(i)^{\gamma_t}}$$

Neural Turing Machines, Graves et. al., arXiv:1410.5401

Neural Turing Machine: Controller Design

- Feed-forward: faster, more transparency & interpretability about function learnt

- LSTM: more expressive power, doesn't limit the number of computations per time step

Both are end-to-end differentiable!

1. Reading/Writing -> Convex Sums
2. w_t generation -> Smooth
3. Controller Networks

Neural Turing Machines, Graves et. al., arXiv:1410.5401

Neural Turing Machine: Network Overview

Unrolled Feed-forward Controller

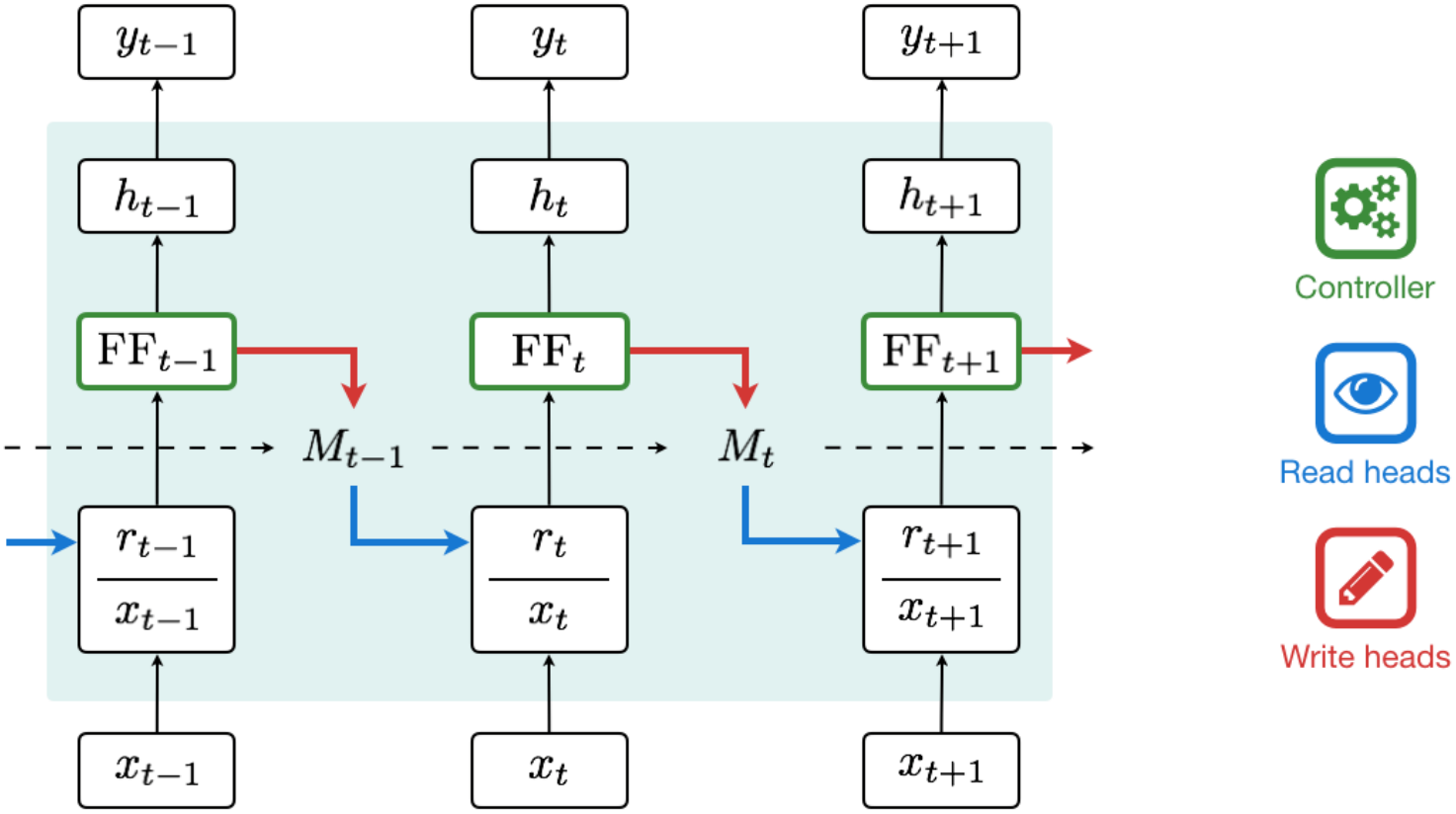


Figure from [Snips AI's Medium Post]

Neural Turing Machines, Graves et. al., arXiv:1410.5401

Neural Turing Machines vs. MemNNs

MemNNs

- Memory is static, with focus on retrieving (reading) information from memory

NTMs

- Memory is continuously written to and read from, with network learning when to perform memory read and write

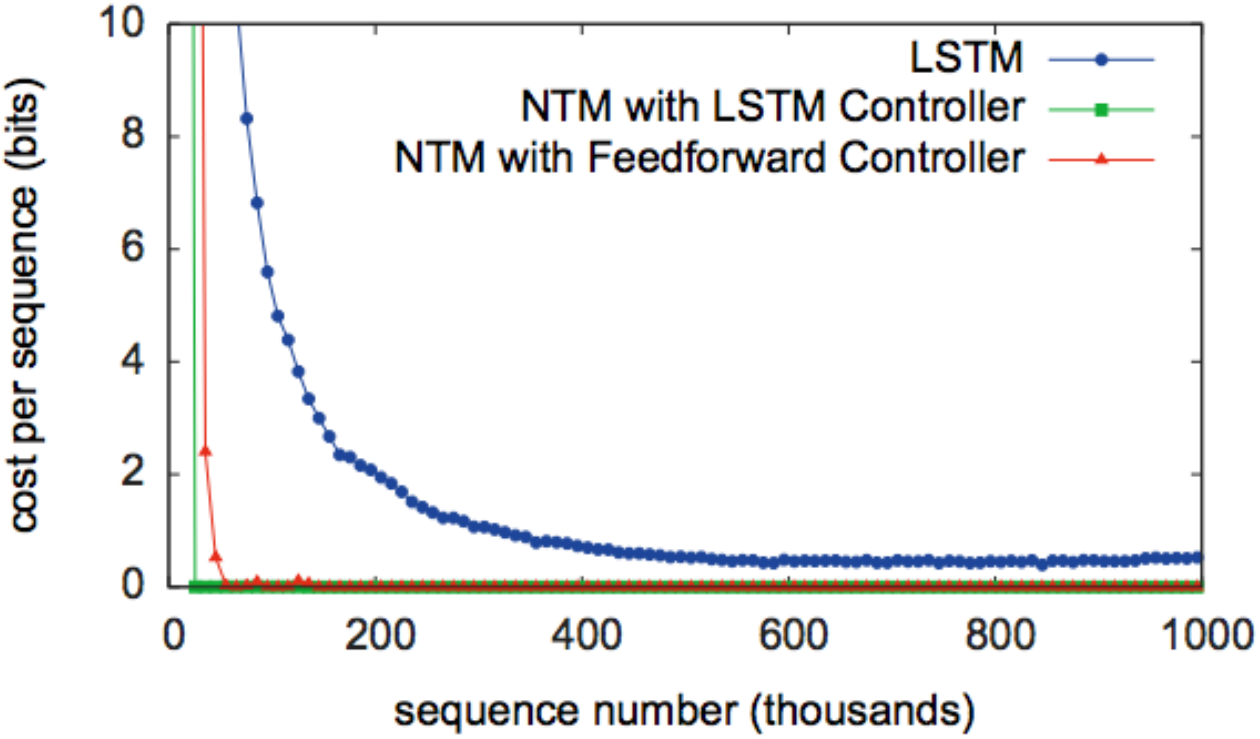
Neural Turing Machines: Experiments

Task	Network Size		Number of Parameters	
	NTM w/ LSTM*	LSTM	NTM w/ LSTM	LSTM
Copy	3 x 100	3 x 256	67K	1.3M
Repeat Copy	3 x 100	3 x 512	66K	5.3M
Associative	3 x 100	3 x 256	70K	1.3M
N-grams	3 x 100	3 x 128	61K	330K
Priority Sort	2 x 100	3 x 128	269K	385K

Neural Turing Machines, Graves et. al., arXiv:1410.5401

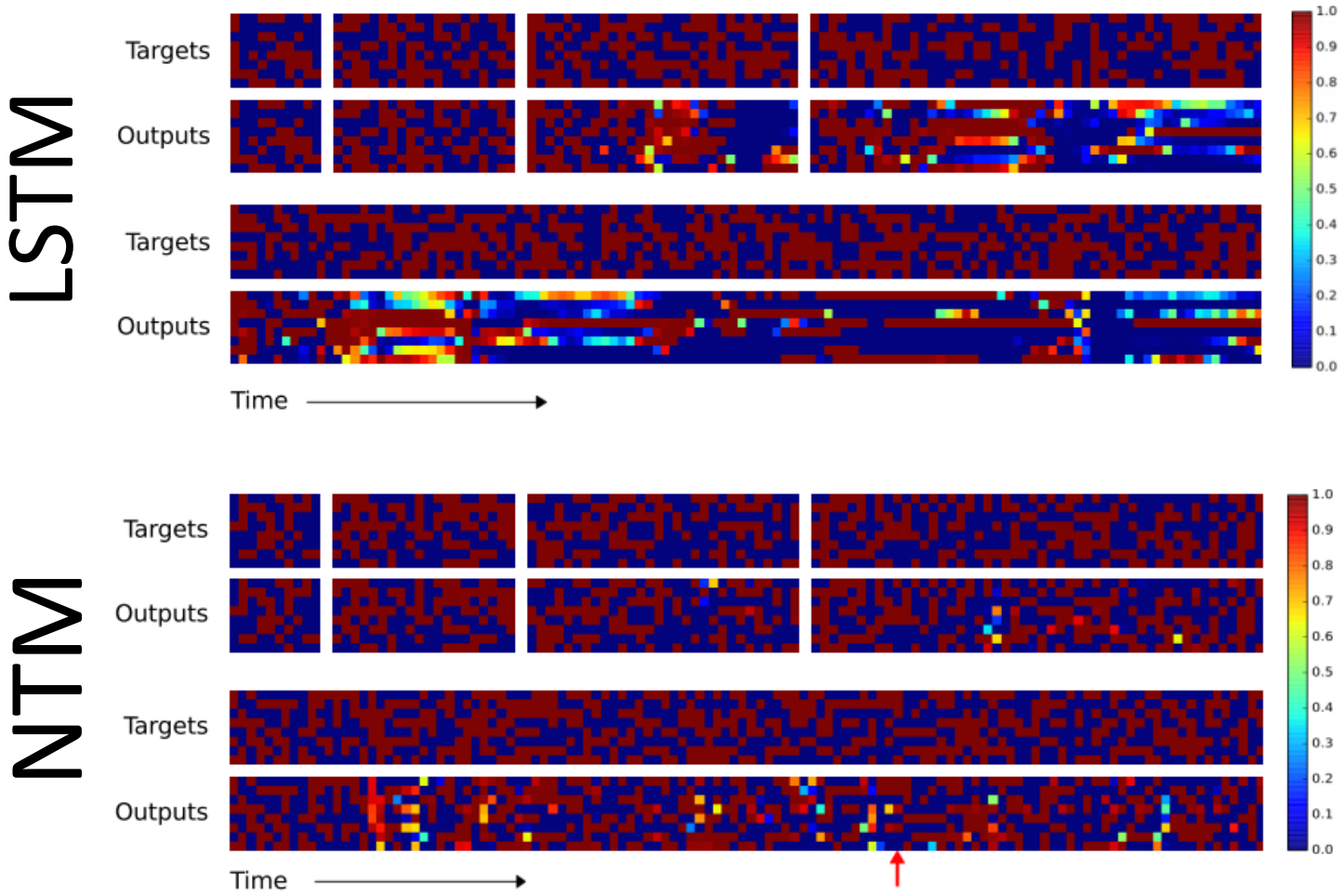
Neural Turing Machines: Copy Learning Curve

Trained on 8-bit sequences, 1<= sequence length <= 20



Neural Turing Machines, Graves et. al., arXiv:1410.5401

Neural Turing Machines: Copy Performance



Neural Turing Machines, Graves et. al., arXiv:1410.5401

Neural Turing Machines triggered an outbreak of Memory Architectures!

Dynamic Neural Turing Machines

Experimented with addressing schemes

- Dynamic Addresses: Addresses of memory locations learnt in training – allows non-linear location-based addressing
- Least recently used weighting: Prefer least recently used memory locations + interpolate with content-based addressing
- Discrete Addressing: Sample the memory location from the content-based distribution to obtain a one-hot address
- Multi-step Addressing: Allows multiple hops over memory

Results: bAbI QA Task

	Location NTM	Content NTM		Soft DNTM	Discrete DNTM
1-step	31.4%	33.6%		29.5%	27.9%
3-step	32.8%	32.7%		24.2%	21.7%

Dynamic Neural Turing Machine with Soft and Hard Addressing Schemes, Gulchere et. al., arXiv:1607.00036

Stack Augmented Recurrent Networks

Learn algorithms based on stack implementations (e.g. learning fixed sequence generators)

Sequence generator	Example
$\{a^n b^n \mid n > 0\}$	aab**ba**aab**bba**b**a**aaaab**bbbb**
$\{a^n b^n c^n \mid n > 0\}$	aaab**bbccca**b**ca**aaaab**bbbbccccc**
$\{a^n b^n c^n d^n \mid n > 0\}$	aab**bccdda**aab**bbccccddda**b**cd**
$\{a^n b^{2n} \mid n > 0\}$	aab**bbba**aab**bbbbbba**b**b**
$\{a^n b^m c^{n+m} \mid n, m > 0\}$	aabc**cca**aabbc**ccccca**bc**c**
$n \in [1, k],\ X \to nXn,\ X \to =$	$(k = 2)$ 12=**21**2122=**2211**1121=**12111**

Uses a stack data structure to store memory (as opposed to a memory matrix)

Inferring Algorithmic Patterns with Stack-Augmented Recurrent Nets, Joulin et. al., arXiv:1503.01007

Stack Augmented Recurrent Networks

- Blurry ‘push’ and ‘pop’ on stack. E.g.:

$$s_t[0] = a[\texttt{Push}](h_t) + a[\texttt{Pop}]s_{t-1}[1]$$

- Some results:

method	$a^n b^n$	$a^n b^n c^n$	$a^n b^n c^n d^n$	$a^n b^{2n}$	$a^n b^m c^{n+m}$
RNN	25%	23.3%	13.3%	23.3%	33.3%
LSTM	100%	100%	68.3%	75%	100%
List RNN 40+5	100%	33.3%	100%	100%	100%
Stack RNN 40+10	100%	100%	100%	100%	43.3%
Stack RNN 40+10 + rounding	100%	100%	100%	100%	100%

Inferring Algorithmic Patterns with Stack-Augmented Recurrent Nets, Joulin et. al., arXiv:1503.01007

Differentiable Neural Computers

Advanced addressing mechanisms:

- Content Based Addressing
- Temporal Addressing
  - Maintains notion of sequence in addressing
  - Temporal Link Matrix *L* (size NxN), L[*i,j*] = degree to which location *I* was written to after location *j.*
- Usage Based Addressing

Hybrid computing using a neural network with dynamic external memory, Graves et. al., Nature *vol.* 538

DNC: Usage Based Addressing

- Writing increases usage of cell, reading decreases usage of cell
- Least used location has highest usage-based weighting
- Interpolate b/w usage & content based weights for final write weights

Hybrid computing using a neural network with dynamic external memory, Graves et. al., Nature *vol.* 538

DNC: Example

Hybrid computing using a neural network with dynamic external memory, Graves et. al., Nature *vol.* 538

DNC: Improvements over NTMs

NTM	DNC
• Large contiguous blocks of memory needed	• Memory locations non-contiguous, usage-based
• No way to free up memory cells after writing	• Regular de-allotment based on usage-tracking

Hybrid computing using a neural network with dynamic external memory, Graves et. al., Nature *vol.* 538

DNC: Experiments

Graph Tasks

Graph Representation: (source, edge, destination) tuples

Types of tasks:

- Traversal: Perform walk on graph given source, list of edges
- Shortest Path: Given source, destination
- Inference: Given source, relation over edges; find destination

Hybrid computing using a neural network with dynamic external memory, Graves et. al., Nature *vol.* 538

DNC: Experiments

Graph Tasks

Training over 3 phases:

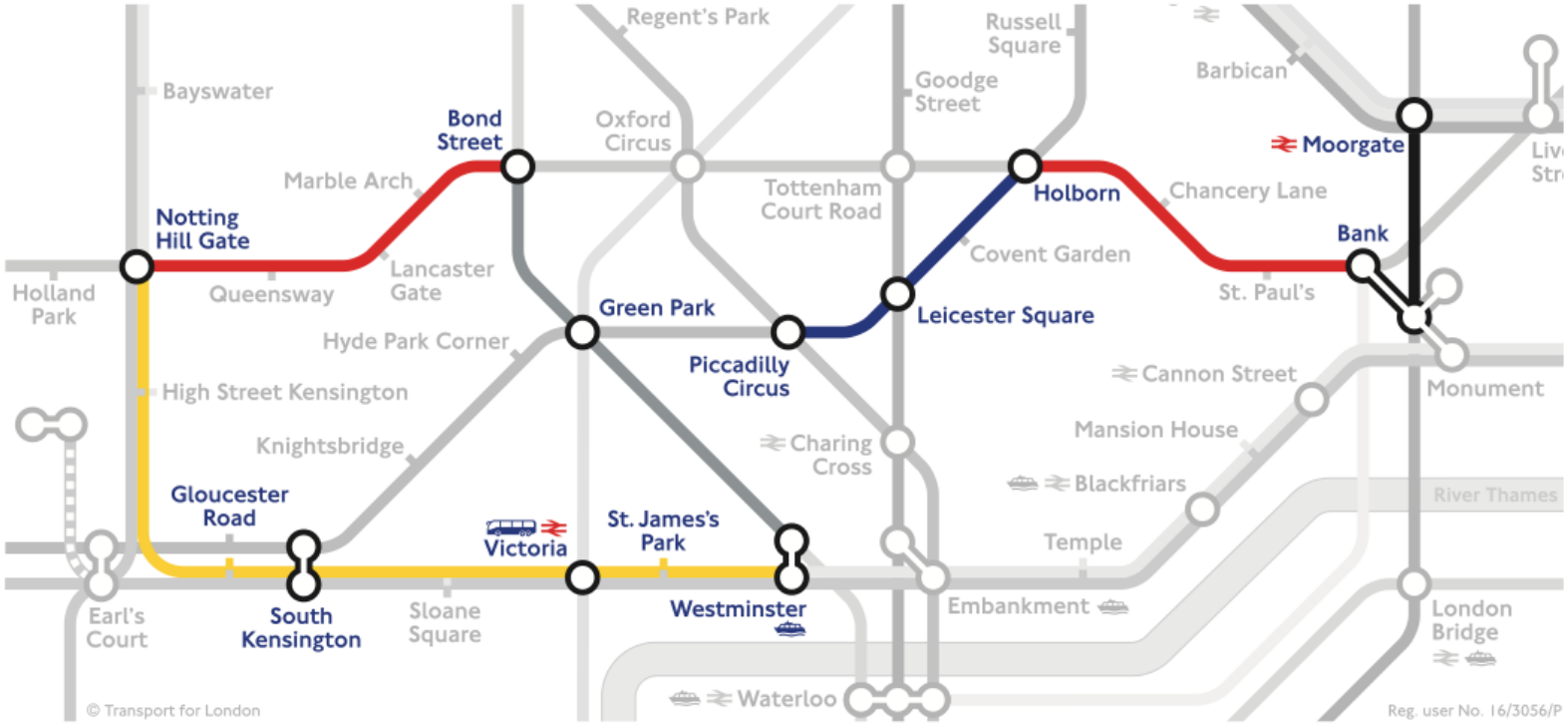
- Graph description phase: (source, edge, destination) tuples fed into the graph
- Query phase: Shortest path (source, ____, destination), Inference (source, hybrid relation, ___), Traversal (source, relation, relation …, ___)
- Answer phase: Target responses provided at output

Trained on random graphs of maximum size 1000

Hybrid computing using a neural network with dynamic external memory, Graves et. al., Nature *vol.* 538

DNC: Experiments

Graph Tasks: London Underground



Hybrid computing using a neural network with dynamic external memory, Graves et. al., Nature *vol.* 538

DNC: Experiments

Graph Tasks: London Underground

Input Phase

(OxfordCircus, TottenhamCtRd, Central)
(TottenhamCtRd, OxfordCircus, Central)
(BakerSt, Marylebone, Circle)
(BakerSt, Marylebone, Bakerloo)
(BakerSt, OxfordCircus, Bakerloo)
⋮
(LeicesterSq, CharingCross, Northern)
(TottenhamCtRd, LeicesterSq, Northern)
(OxfordCircus, PiccadillyCircus, Bakerloo)
(OxfordCircus, NottingHillGate, Central)
(OxfordCircus, Euston, Victoria)

Hybrid computing using a neural network with dynamic external memory, Graves et. al., Nature *vol.* 538

DNC: Experiments

Graph Tasks: London Underground

Traversal Task

(BondSt, _, Central),
(_, _, Circle), (_, _, Circle),
(_, _, Circle), (_, _, Circle),
(_, _, Jubilee), (_, _, Jubilee),

Query Phase

(BondSt, NottingHillGate, Central)
(NottingHillGate, GloucesterRd, Circle)
⋮
(Westminster, GreenPark, Jubilee)
(GreenPark, BondSt, Jubilee)

Answer Phase

Hybrid computing using a neural network with dynamic external memory, Graves et. al., Nature *vol.* 538

DNC: Experiments

Graph Tasks: London Underground

Shortest Path Task

(Moorgate, PiccadillyCircus, _)

(Moorgate, Bank, Northern)
(Bank, Holborn, Central)
(Holborn, LeicesterSq, Piccadilly)
(LeicesterSq, PiccadillyCircus, Piccadilly)

Query Phase

Answer Phase

Hybrid computing using a neural network with dynamic external memory, Graves et. al., Nature *vol.* 538

DNC: Experiments

Graph Tasks: Freya's Family Tree

Differentiable Neural Computer
Family tree inference task
(artistic rendering)

Hybrid computing using a neural network with dynamic external memory, Graves et. al., Nature *vol.* 538

Conclusion

- Machine Learning models require memory and multi-hop reasoning to perform AI tasks better
- Memory Networks for Text are an interesting direction but very simple
- Generic architectures with memory, such as Neural Turing Machine, limited applications shown
- Future directions should be focusing on applying generic neural models with memory to more AI Tasks.

Hybrid computing using a neural network with dynamic external memory, Graves et. al., Nature *vol.* 538

Acknowledgement

Some of the materials in these slides are drawn inspiration from:

- Shubhendu Trivedi and Risi Kondor, University of Chicago, Deep Learning Course

- Hung-yi Lee, National Taiwan University, Machine Learning and having it Deep and Structured course

- Xiaogang Wang, The Chinese University of Hong Kong, Deep Learning Course

- Fei-Fei Li, Standord University, CS231n Convolutional Neural Networks for Visual Recognition course

Next time

- Applications

Questions?

Thank You !

WeChat Group for Deep Learning