max planck institut
informatik

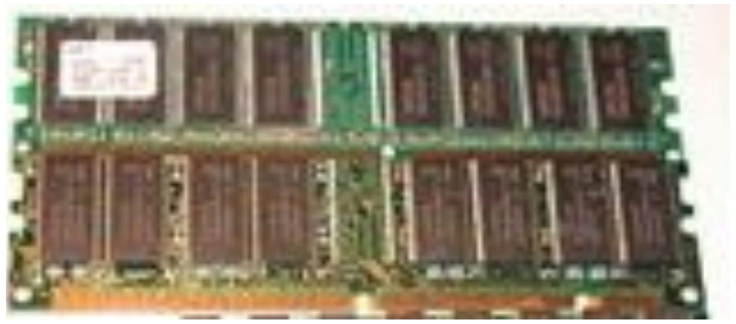
Attention-based Networks

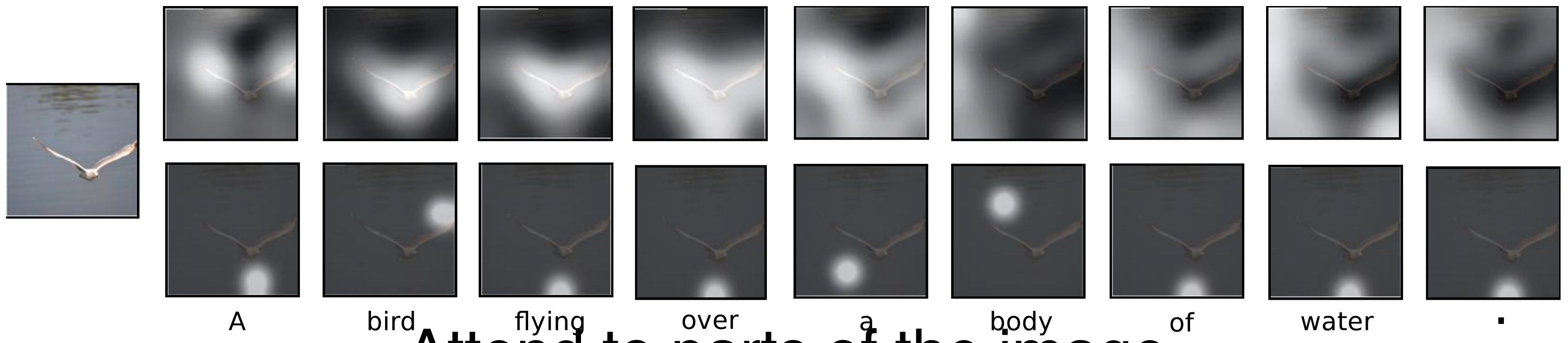
M. Malinowski

Why attention?

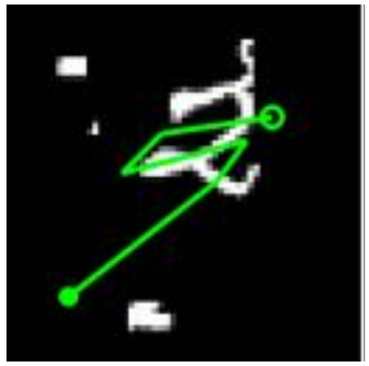
- Long term memories - attending to memories
  - Dealing with gradient vanishing problem
- Exceeding limitations of a global representation
  - Attending/focusing to smaller parts of data
    - patches in images
    - words or phrases in sentences
- Decoupling representation from a problem
  - Different problems required different sizes of representations
    - LSTM with longer sentences requires larger vectors
- Overcoming computational limits for visual data
  - Focusing only on the parts of images
  - Scalability independent of the size of images
- Adds some interpretability to the models (error inspection)

Plan

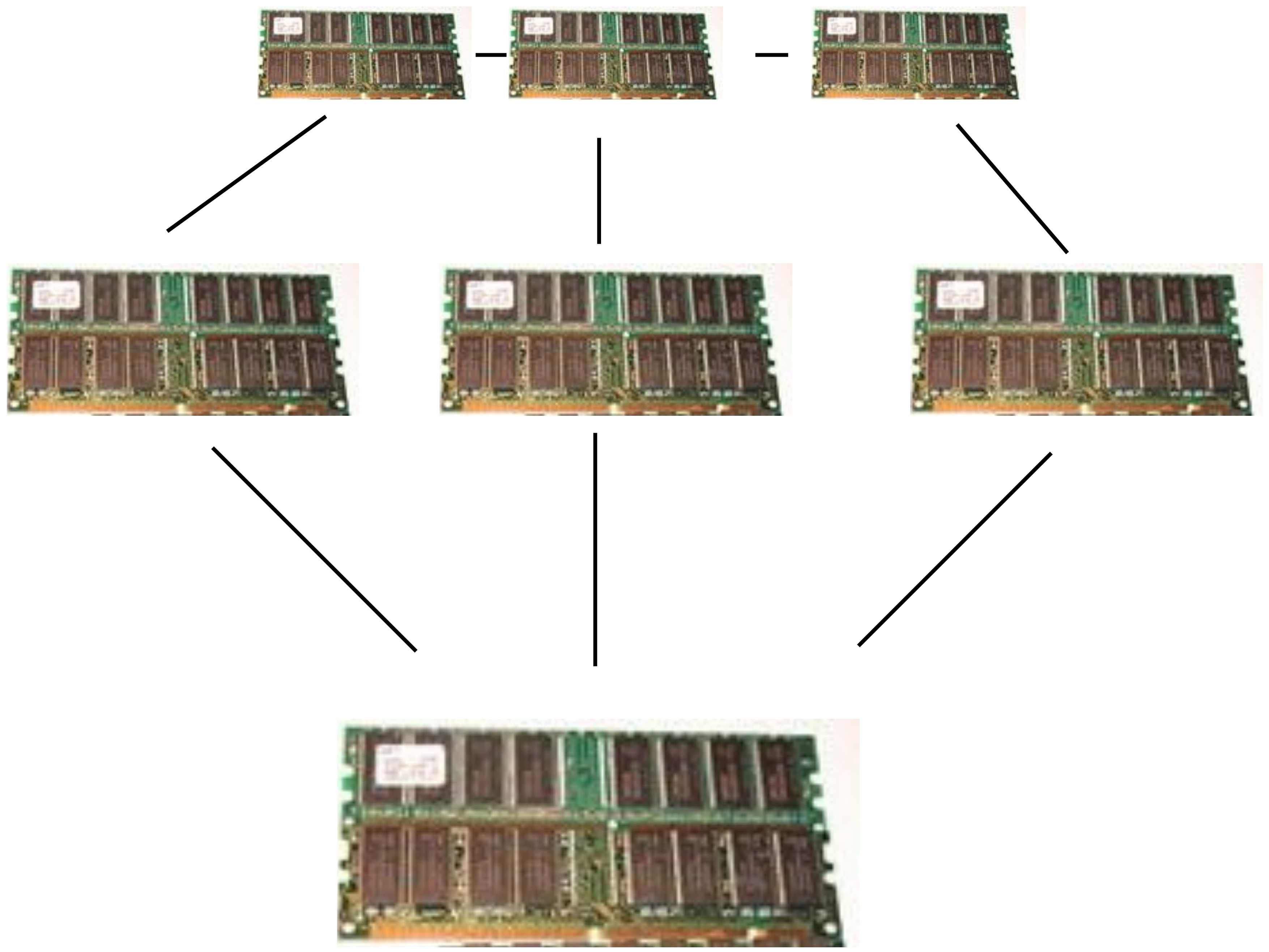

Attend to memory cells



Attend to parts of the image

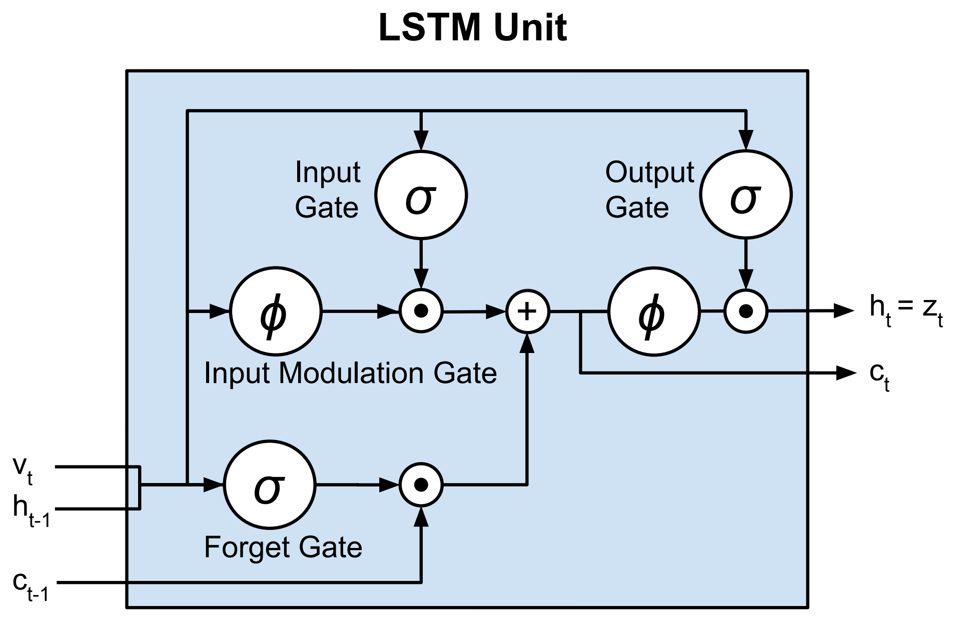

Glimpse-driven mechanism

Memory Networks



Motivation and task

- New class of networks that combine inference with long-term memories
  - LSTM is a subclass
  - But the class is much broader
- The long-term memories can be read from or written to
  - Long-term memories == Knowledge base
  - We want to store information
  - We want to retrieve information



Story (2: 2 supporting facts)	Support	Hop 1	Hop 2	Hop 3
John dropped the milk.		0.06	0.00	0.00
John took the milk there.	yes	0.88	1.00	0.00
Sandra went back to the bathroom.		0.00	0.00	0.00
John moved to the hallway.	yes	0.00	0.00	1.00
Mary went back to the bedroom.		0.00	0.00	0.00
Where is the milk? Answer: hallway Prediction: hallway				

IGOR

- Components (IGOR)

I **component:** Component I can make use of standard pre-processing, e.g., parsing, coreference and entity resolution for text inputs. It could also encode the input into an internal feature representation, e.g., convert from text to a sparse or dense feature vector.

G **component:** The simplest form of G is to store $I(x)$ in a "slot" in the memory:

$$\mathbf{m}_{S(x)} = I(x), \tag{1}$$

where $S(.)$ is a function selecting the slot. That is, G updates the index $S(x)$ of $\mathbf{m}$, but all other

More sophisticated versions can update all memories based on a new evidence.
If memory is huge, we can organize this memory differently according to S(.) (organize memories according to topics).
The selection function S can also be responsible for 'forgetting' by replacing the current memories.

O **and** R **components:** The O component is typically responsible for reading from memory and performing inference, e.g., calculating what are the relevant memories to perform a good response. The R component then produces the final response given O. For example in a question answering setup O finds relevant memories, and then R produces the actual wording of the answer, e.g., R could be an RNN that is conditioned on the output of O. Our hypothesis is that without conditioning on such memories, such an RNN will perform poorly.

1. Convert x to an internal feature representation $I(x)$.
2. Update memories $\mathbf{m}_i$ given the new input: $\mathbf{m}_i = G(\mathbf{m}_i, I(x), \mathbf{m}), \ \forall i$.
3. Compute output features o given the new input and the memory: $o = O(I(x), \mathbf{m})$.
4. Finally, decode output features o to give the final response: $r = R(o)$.

MemNN - training

- Supervision with the supporting sentences
- Max-margin loss

$$\sum_{\bar{f}\neq f_1} \max(0, \gamma - s_O(x, f_1) + s_O(x, \bar{f})) +$$

$$\sum_{\bar{f}'\neq f_2} \max(0, \gamma - s_O([x, \mathbf{m}_{o_1}], f_2]) + s_O([x, \mathbf{m}_{o_1}], \bar{f}'])) +$$

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

  - ‘Bad’ sentences are sampled for the speed reason
- Additional ‘tricks’
  - Segmenter to decides when a sentence should be written to
  - Time stamps
  - Dealing with unknown words

End-to-end Memory Networks

- Solves the severe limitation of Memory Network
  - Supervision of whether a sentence is important or not
- If we transform the separated steps of the memory network into an end-to-end formulation we could use the error signal form the task to train the whole network
- IGOR
  - I - Content-based addressing

$$m_i = \sum_j A x_{ij} \quad x_i = \{x_{i1}, x_{i2}, ..., x_{in}\} \qquad u = \sum_j B q_j \quad \text{question vector } q$$

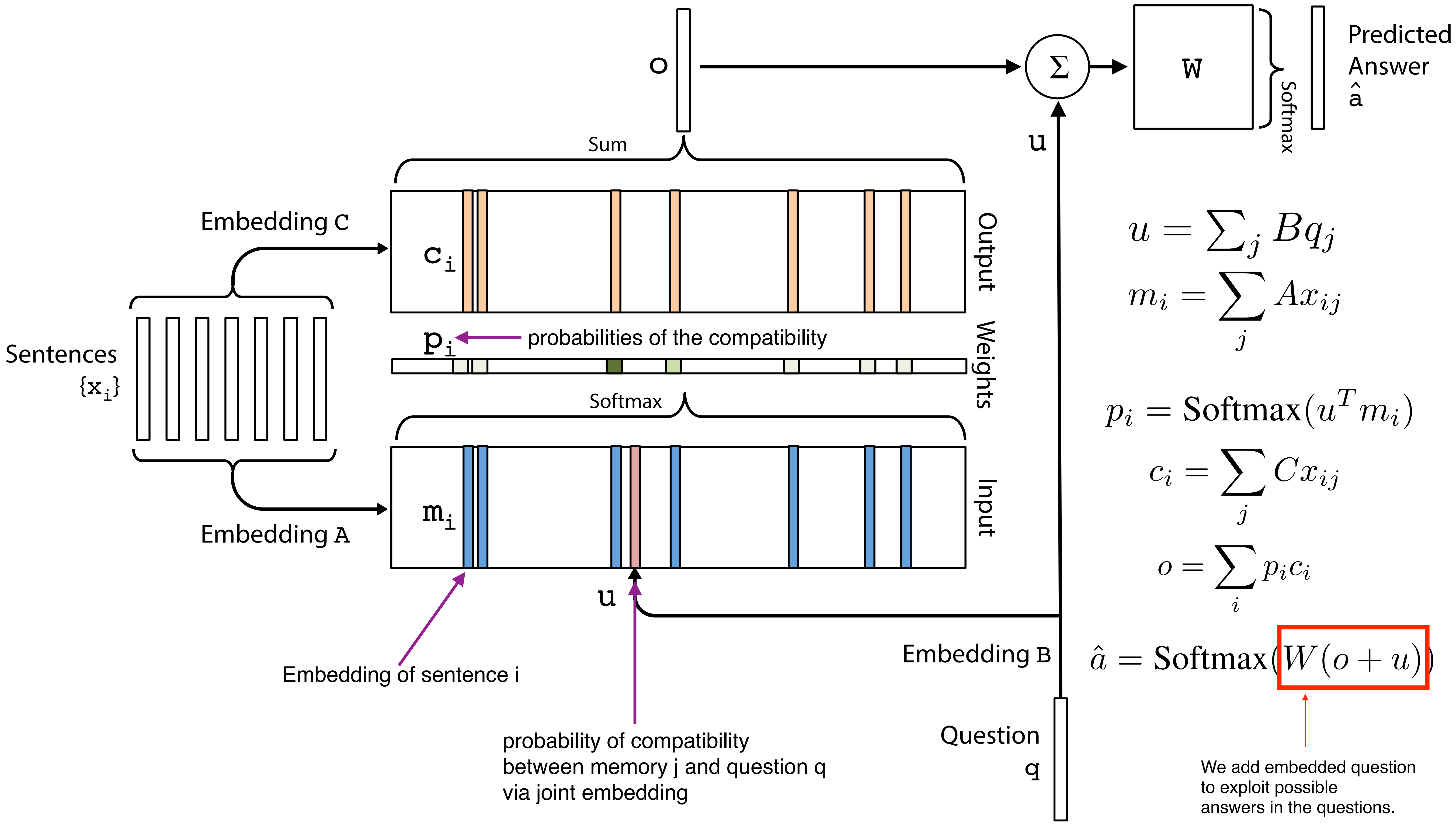
  - O - ‘Soft’ attention mechanism while reading the memory

$$p_i = \text{Softmax}(u^T m_i) = \text{Softmax}(q^T B^T \sum_j A x_{ij}).$$

$$o = \sum_i p_i c_i = \sum_i \sum_j p_i C x_{ij} \quad c_i = \sum_j C x_{ij}$$

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

End-to-end Memory Networks



$$u = \sum_j Bq_j$$

$$m_i = \sum_j Ax_{ij}$$

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

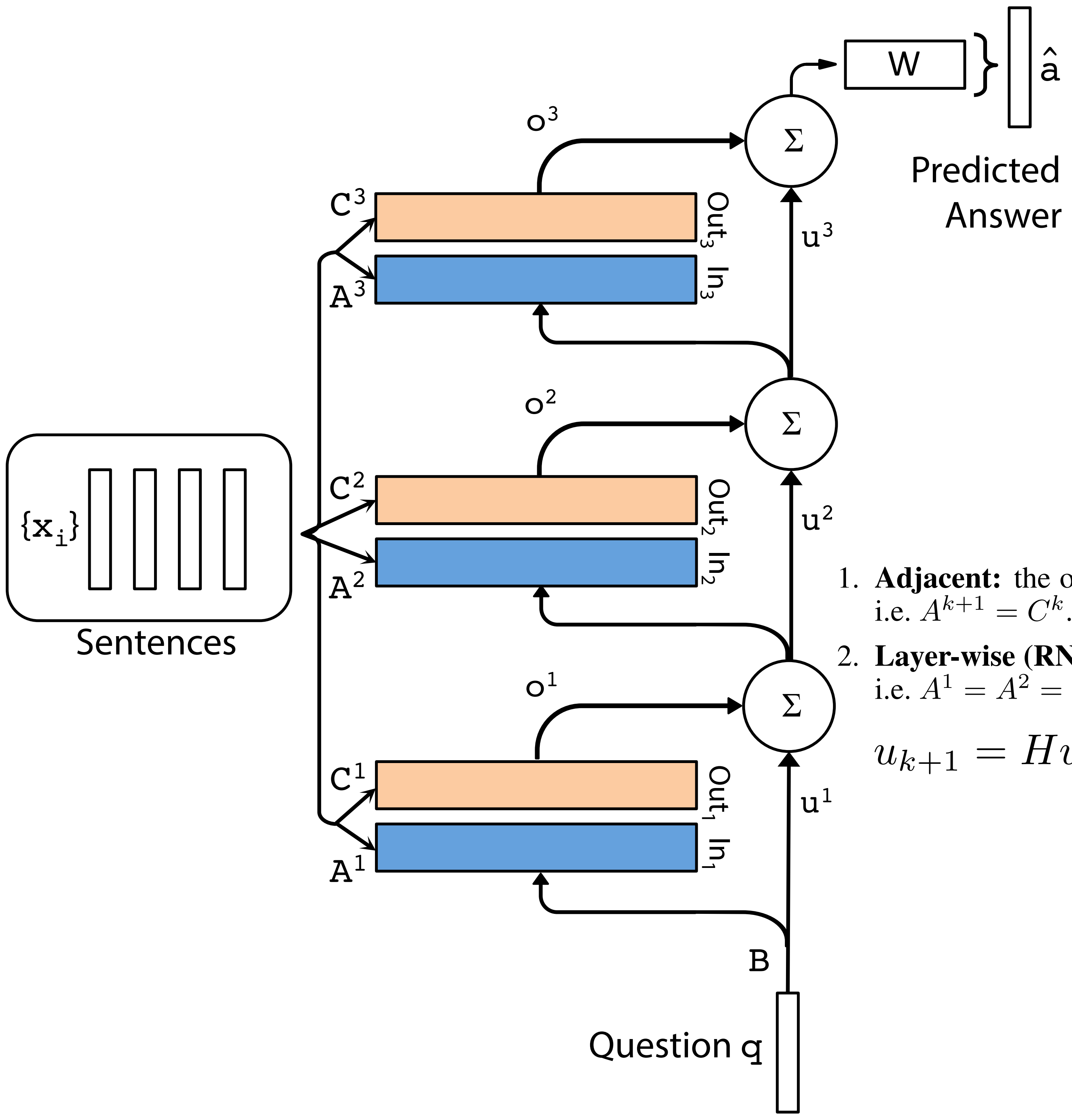
$$c_i = \sum_j Cx_{ij}$$

$$o = \sum_i p_i c_i$$

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

We add embedded question to exploit possible answers in the questions.

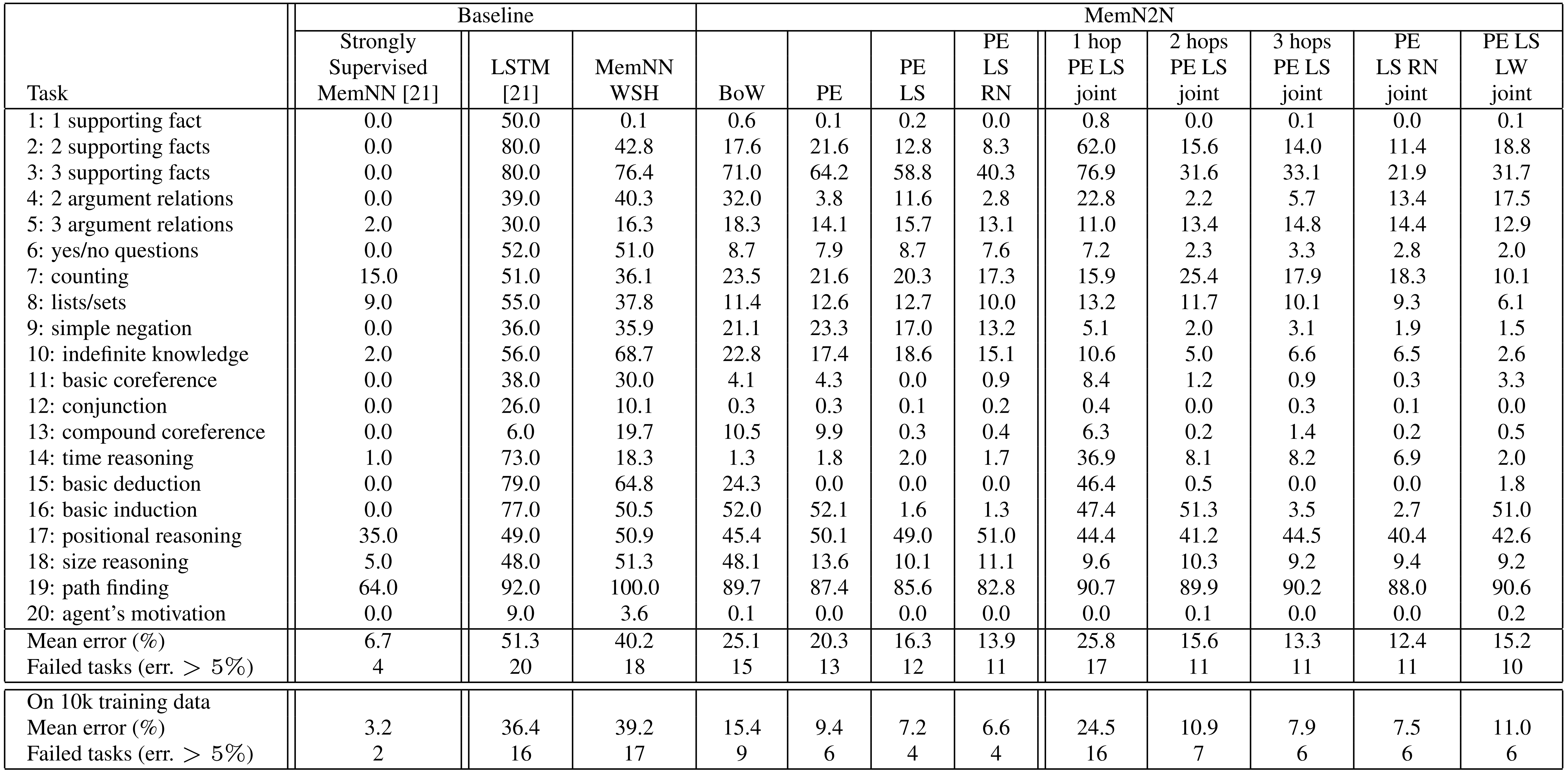
End-to-end Memory Networks



1. **Adjacent:** the output embedding for one layer is the input embedding for the one above, i.e. $A^{k+1} = C^k$.
2. **Layer-wise (RNN):** the input and output embeddings are the same across different layers, i.e. $A^1 = A^2 = A^3$ and $C^1 = C^2 = C^3$.

$$u_{k+1} = Hu_k + o_k$$

End-to-end Memory Networks



	Baseline			MemN2N								
Task	Strongly Supervised MemNN [21]	LSTM [21]	MemNN WSH	BoW	PE	PE LS	PE LS RN	1 hop PE LS joint	2 hops PE LS joint	3 hops PE LS joint	PE LS RN joint	PE LS LW joint
1: 1 supporting fact	0.0	50.0	0.1	0.6	0.1	0.2	0.0	0.8	0.0	0.1	0.0	0.1
2: 2 supporting facts	0.0	80.0	42.8	17.6	21.6	12.8	8.3	62.0	15.6	14.0	11.4	18.8
3: 3 supporting facts	0.0	80.0	76.4	71.0	64.2	58.8	40.3	76.9	31.6	33.1	21.9	31.7
4: 2 argument relations	0.0	39.0	40.3	32.0	3.8	11.6	2.8	22.8	2.2	5.7	13.4	17.5
5: 3 argument relations	2.0	30.0	16.3	18.3	14.1	15.7	13.1	11.0	13.4	14.8	14.4	12.9
6: yes/no questions	0.0	52.0	51.0	8.7	7.9	8.7	7.6	7.2	2.3	3.3	2.8	2.0
7: counting	15.0	51.0	36.1	23.5	21.6	20.3	17.3	15.9	25.4	17.9	18.3	10.1
8: lists/sets	9.0	55.0	37.8	11.4	12.6	12.7	10.0	13.2	11.7	10.1	9.3	6.1
9: simple negation	0.0	36.0	35.9	21.1	23.3	17.0	13.2	5.1	2.0	3.1	1.9	1.5
10: indefinite knowledge	2.0	56.0	68.7	22.8	17.4	18.6	15.1	10.6	5.0	6.6	6.5	2.6
11: basic coreference	0.0	38.0	30.0	4.1	4.3	0.0	0.9	8.4	1.2	0.9	0.3	3.3
12: conjunction	0.0	26.0	10.1	0.3	0.3	0.1	0.2	0.4	0.0	0.3	0.1	0.0
13: compound coreference	0.0	6.0	19.7	10.5	9.9	0.3	0.4	6.3	0.2	1.4	0.2	0.5
14: time reasoning	1.0	73.0	18.3	1.3	1.8	2.0	1.7	36.9	8.1	8.2	6.9	2.0
15: basic deduction	0.0	79.0	64.8	24.3	0.0	0.0	0.0	46.4	0.5	0.0	0.0	1.8
16: basic induction	0.0	77.0	50.5	52.0	52.1	1.6	1.3	47.4	51.3	3.5	2.7	51.0
17: positional reasoning	35.0	49.0	50.9	45.4	50.1	49.0	51.0	44.4	41.2	44.5	40.4	42.6
18: size reasoning	5.0	48.0	51.3	48.1	13.6	10.1	11.1	9.6	10.3	9.2	9.4	9.2
19: path finding	64.0	92.0	100.0	89.7	87.4	85.6	82.8	90.7	89.9	90.2	88.0	90.6
20: agent's motivation	0.0	9.0	3.6	0.1	0.0	0.0	0.0	0.0	0.1	0.0	0.0	0.2
Mean error (%)	6.7	51.3	40.2	25.1	20.3	16.3	13.9	25.8	15.6	13.3	12.4	15.2
Failed tasks (err. $> 5\%$)	4	20	18	15	13	12	11	17	11	11	11	10
On 10k training data												
Mean error (%)	3.2	36.4	39.2	15.4	9.4	7.2	6.6	24.5	10.9	7.9	7.5	11.0
Failed tasks (err. $> 5\%$)	2	16	17	9	6	4	4	16	7	6	6	6

Table 1: Test error rates (%) on the 20 QA tasks for models using 1k training examples (mean test errors for 10k training examples are shown at the bottom). Key: BoW = bag-of-words representation; PE = position encoding representation; LS = linear start training; RN = random injection of time index noise; LW = RNN-style layer-wise weight tying (if not stated, adjacent weight tying is used); joint = joint training on all tasks (as opposed to per-task training).

MemNN - architecture

- MemNN components
  - I - BoW embedding
  - G - S(x) returns the next empty memory slot
  - O - finds k supporting memories given x (up to 2 hops here)
    - o_1 = O_1(x,m) = argmax s(x, m_i), s is a similarity measure
    - o_2 = O_2(x,m) = argmax s([x,m_{o_1}], m_i)
    - final output is [x, m_{o_1}, m_{o_2}]
  - R generates single word answers $r = \text{argmax}_{w \in W}\, s_R([x, \mathbf{m}_{o_1}, \mathbf{m}_{o_2}], w)$
  - For s and S_R $s(x,y) = \Phi_x(x)^\top U^\top U \Phi_y(y)$

Story (2: 2 supporting facts)	**Support**	**Hop 1**	**Hop 2**	**Hop 3**
John dropped the milk.		0.06	0.00	0.00
John took the milk there.	yes	0.88	1.00	0.00
Sandra went back to the bathroom.		0.00	0.00	0.00
John moved to the hallway.	yes	0.00	0.00	1.00
Mary went back to the bedroom.		0.00	0.00	0.00
Where is the milk? **Answer: hallway** **Prediction: hallway**				

End-to-end Memory Networks

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				

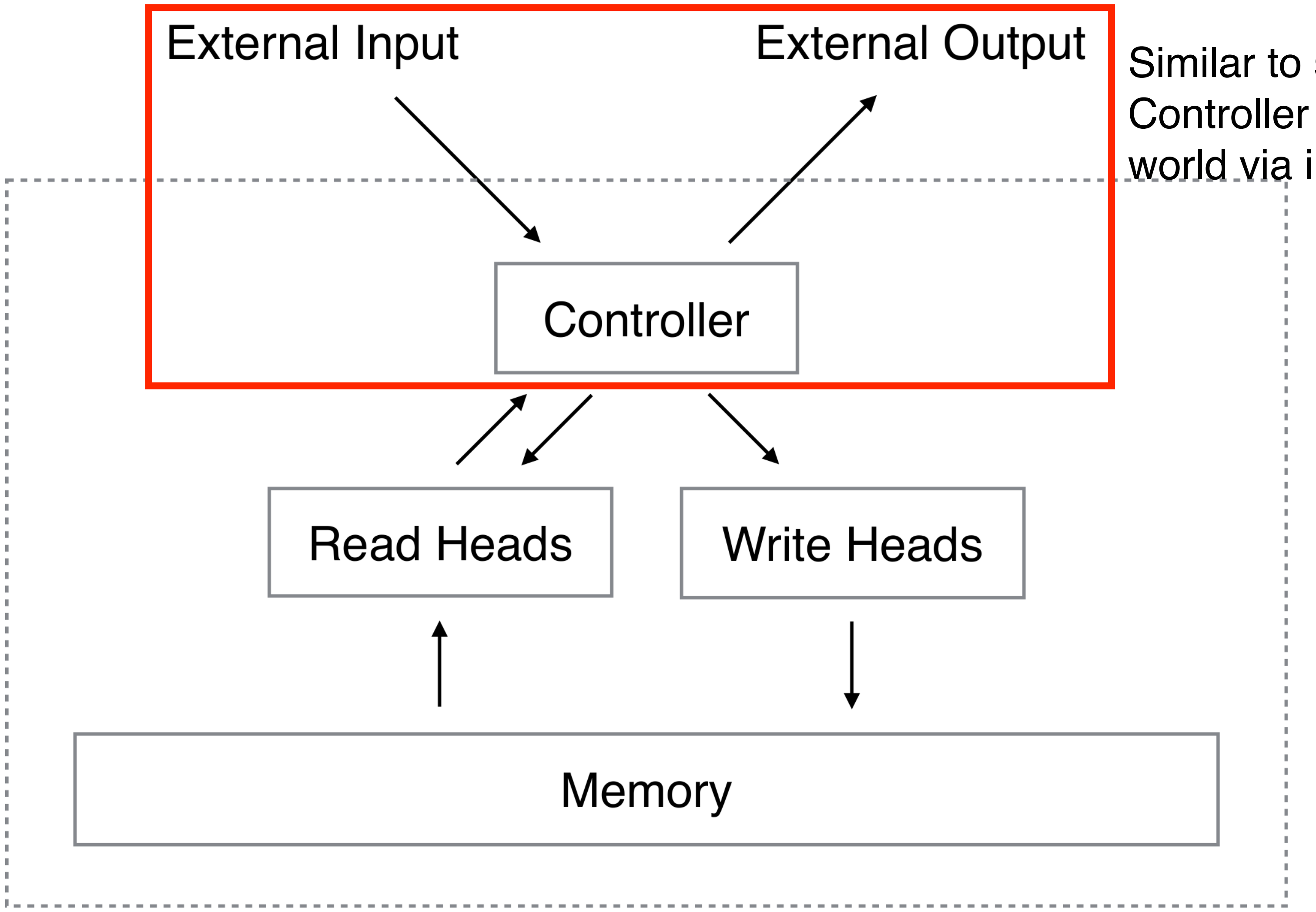
Story (16: basic induction)	Support	Hop 1	Hop 2	Hop 3
Brian is a frog.	yes	0.00	0.98	0.00
Lily is gray.		0.07	0.00	0.00
Brian is yellow.	yes	0.07	0.00	1.00
Julius is green.		0.06	0.00	0.00
Greg is a frog.	yes	0.76	0.02	0.00
What color is Greg? Answer: yellow Prediction: yellow				

Story (2: 2 supporting facts)	Support	Hop 1	Hop 2	Hop 3
John dropped the milk.		0.06	0.00	0.00
John took the milk there.	yes	0.88	1.00	0.00
Sandra went back to the bathroom.		0.00	0.00	0.00
John moved to the hallway.	yes	0.00	0.00	1.00
Mary went back to the bedroom.		0.00	0.00	0.00
Where is the milk? Answer: hallway Prediction: hallway				

Story (18: size reasoning)	Support	Hop 1	Hop 2	Hop 3
The suitcase is bigger than the chest.	yes	0.00	0.88	0.00
The box is bigger than the chocolate.		0.04	0.05	0.10
The chest is bigger than the chocolate.	yes	0.17	0.07	0.90
The chest fits inside the container.		0.00	0.00	0.00
The chest fits inside the box.		0.00	0.00	0.00
Does the suitcase fit in the chocolate? Answer: no Prediction: no				

Neural Turing Machines

- Extend the capabilities of neural nets by coupling them to external memory resources
    - enrich RNN by a large addressable memory
    - Differentiable model of attention
- Infers simple algorithms like copying



Similar to standard Neural Nets, Controller interacts with the external world via input/output vectors

Neural Turing Machines

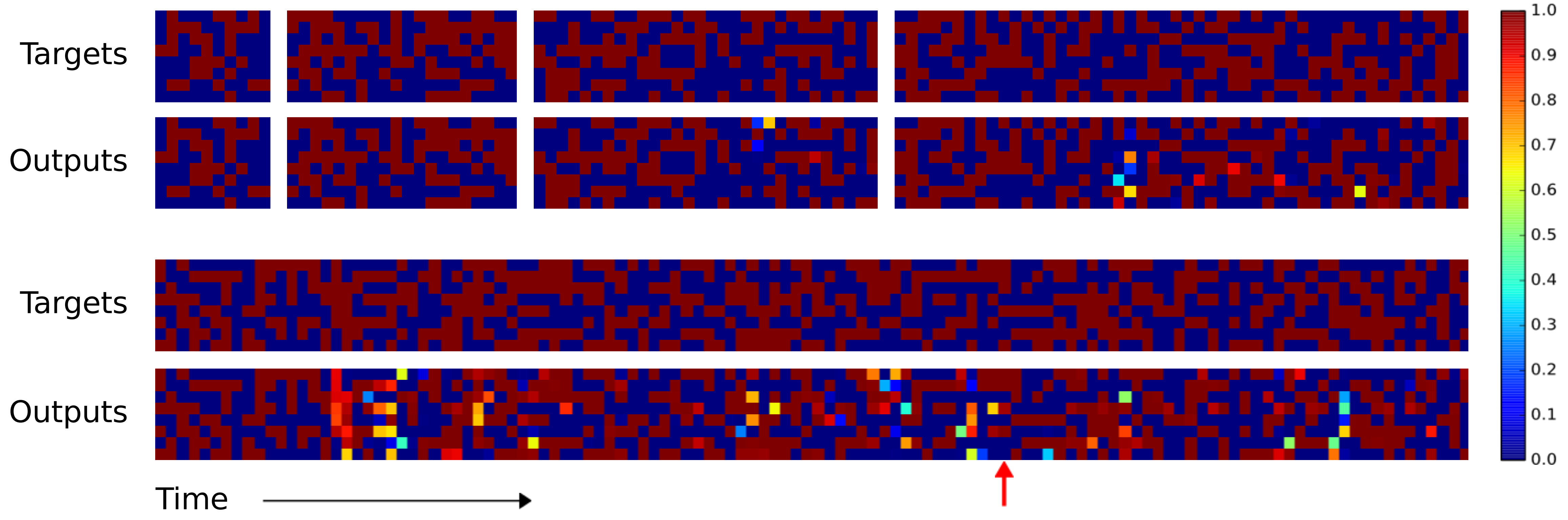
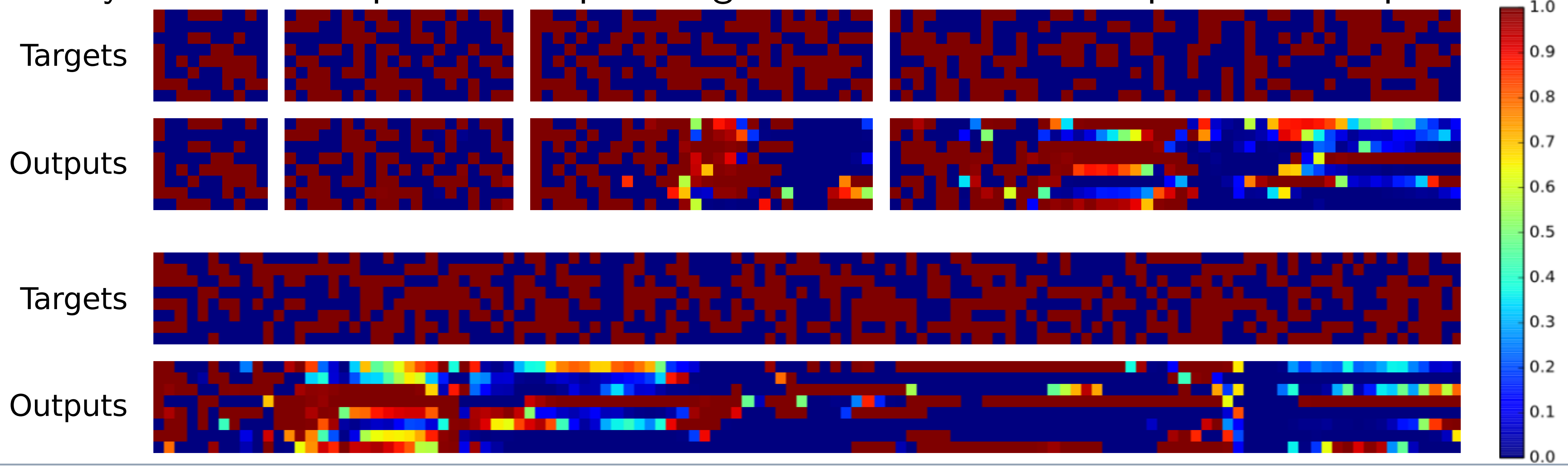
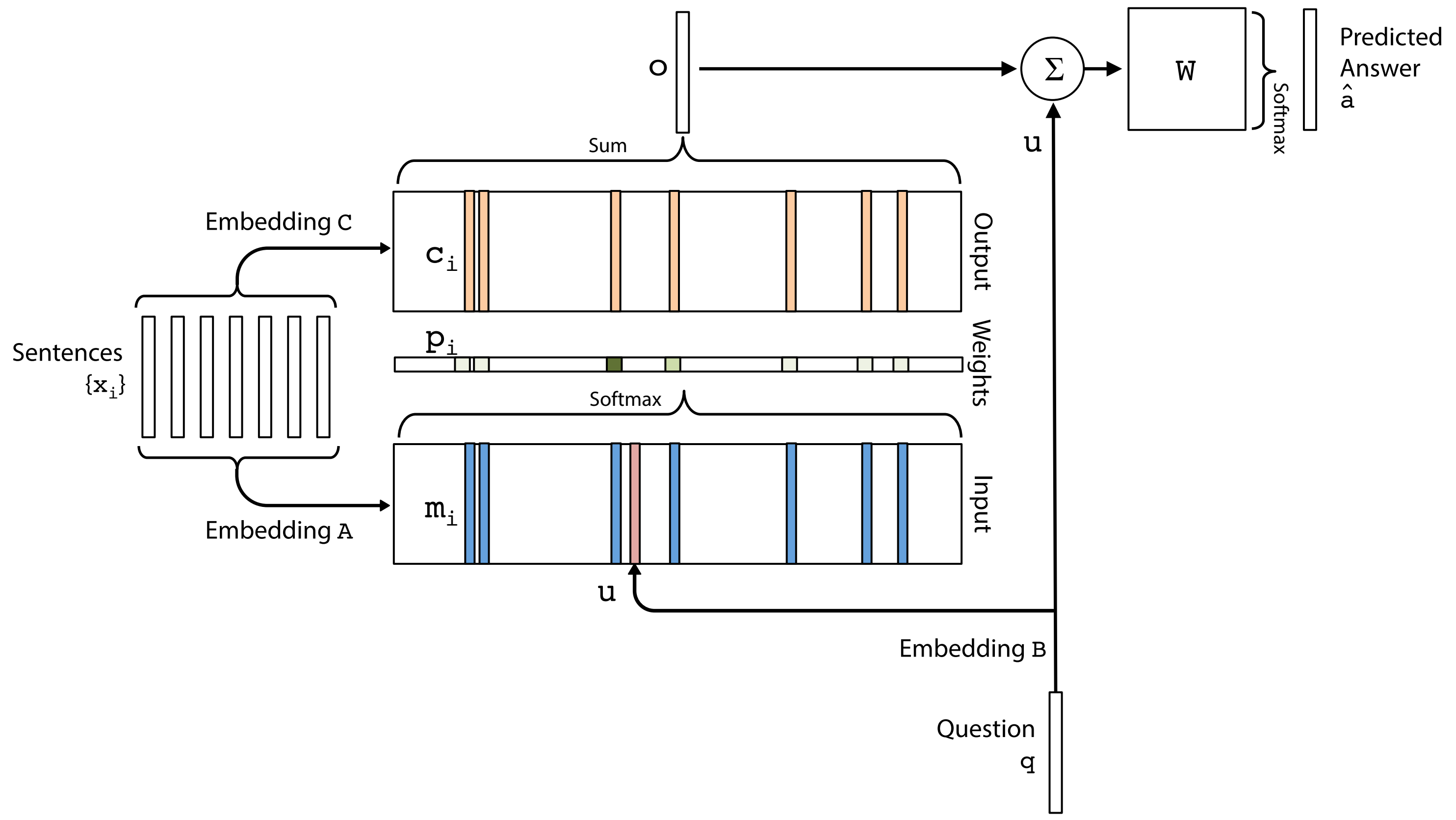


Figure 4: NTM Generalisation on the Copy Task. The four pairs of plots in the top row depict network outputs and corresponding copy targets for test sequences of length 10, 20, 30, and 50, respectively. The plots in the bottom row are for a length 120 sequence. The network was only trained on sequences of up to length 20. The first four sequences are reproduced with

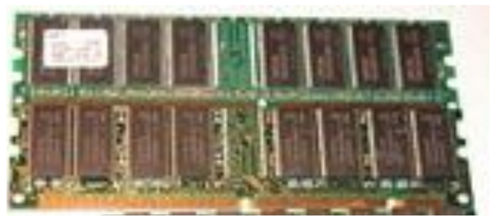


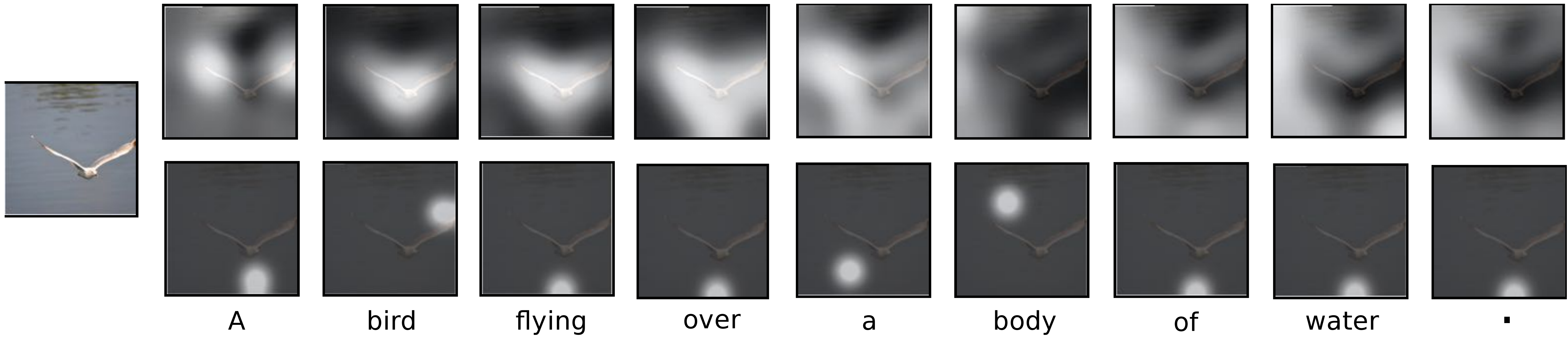
Memory Networks - summary

- Memory Networks that broadens LSTM class
  - Networks with long-term dependencies
  - Attention ‘distribution’ over data points
  - So far specific architectures tailored to QA task
  - Some empirical evidence that the gradient vanishing problem or capacity limitations can be overcome by having an external memory



Show, attend and tell

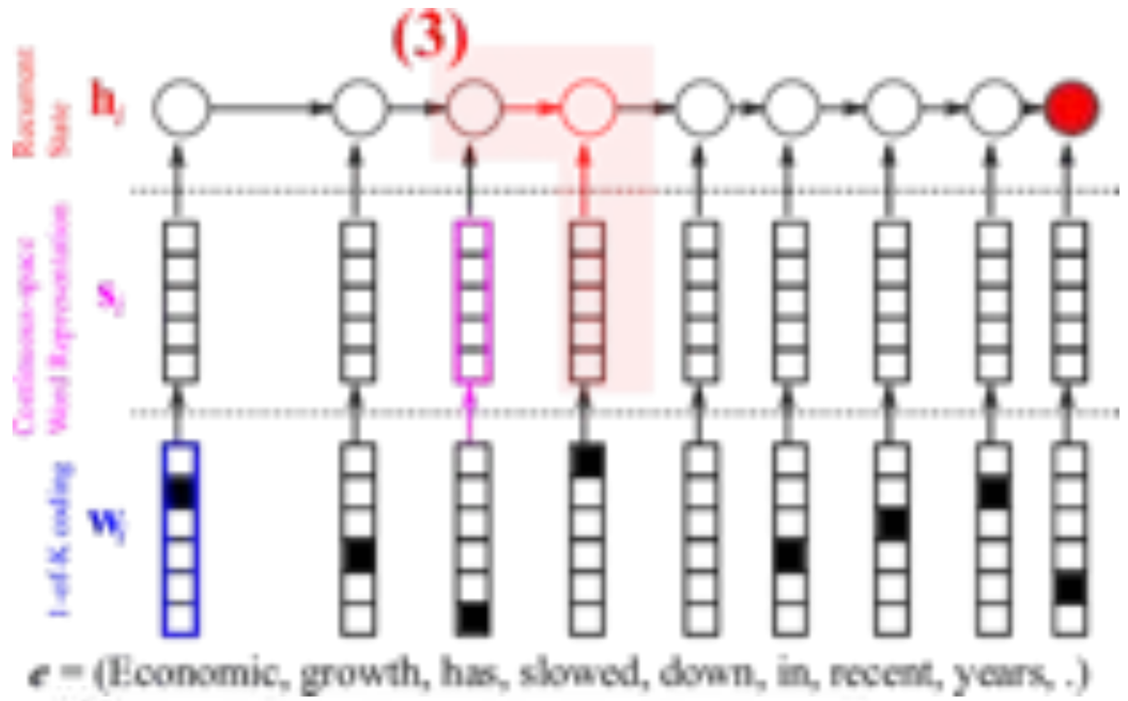


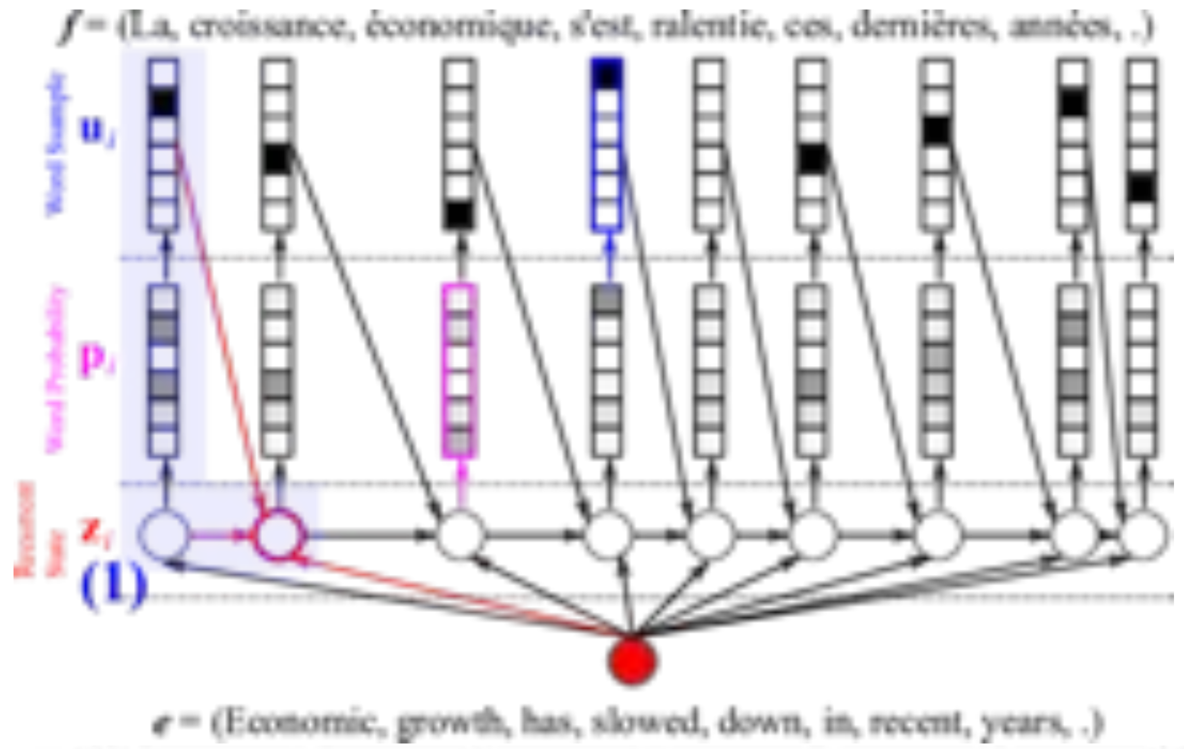


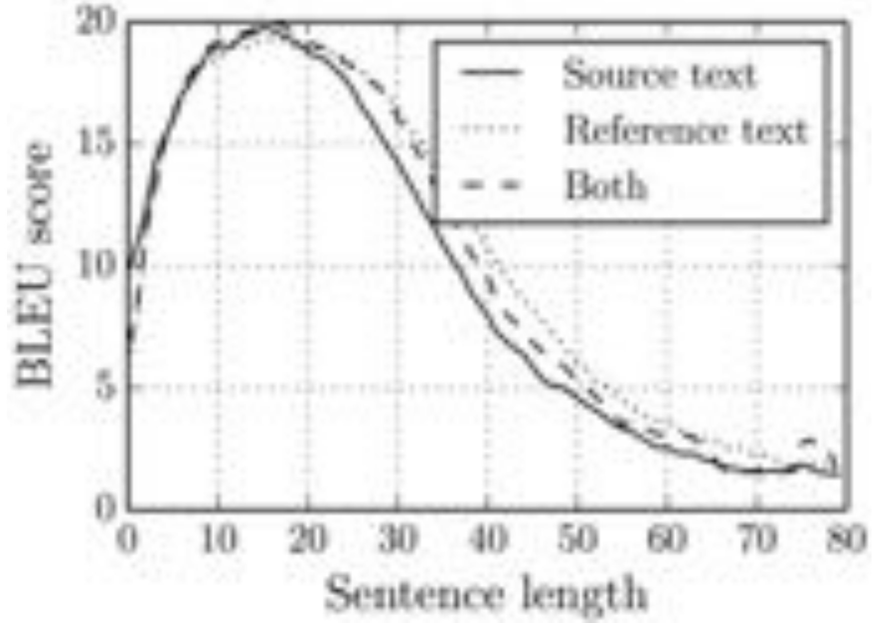
Attend to parts of the image

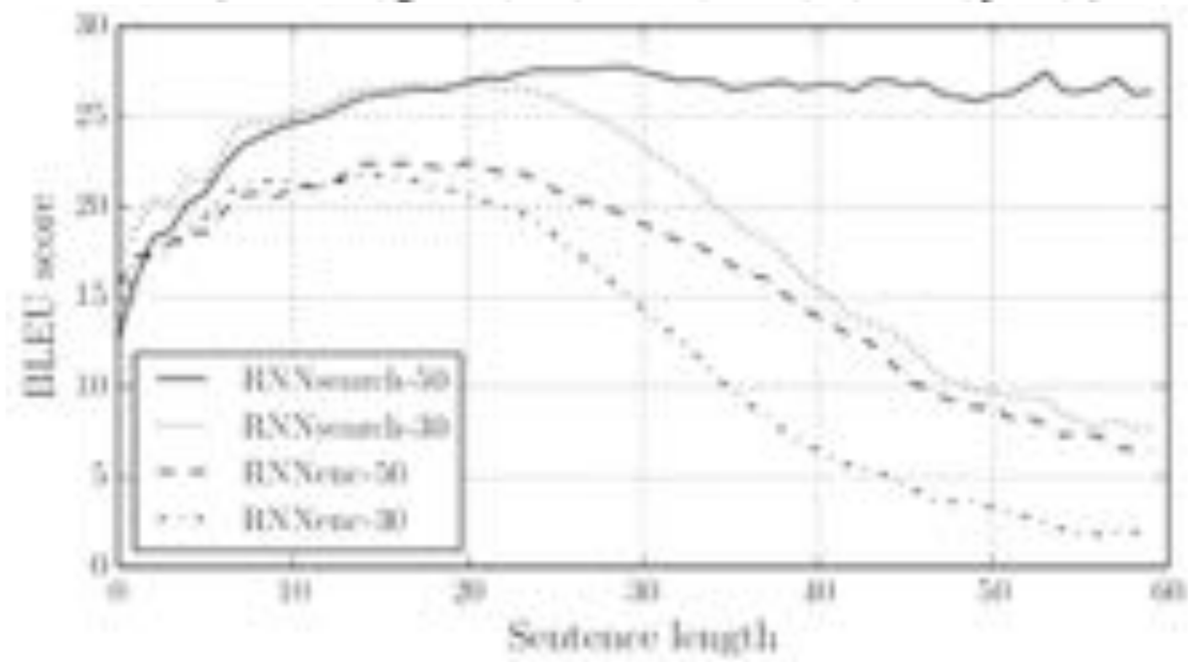
Motivation (Show, attend and tell …)

- Motivation
  - Increase the capacity of the encoder that compress the input into a single vector


(3)
Recurrent State
h_i
Continuous-space Word Representation
s_i
1-of-K coding
w_i
e = (Economic, growth, has, slowed, down, in, recent, years, .)


f = (La, croissance, économique, s'est, ralentie, ces, dernières, années, .)
Word Sample
u_i
Word Probability
p_i
Recurrent State
z_i
(1)
e = (Economic, growth, has, slowed, down, in, recent, years, .)

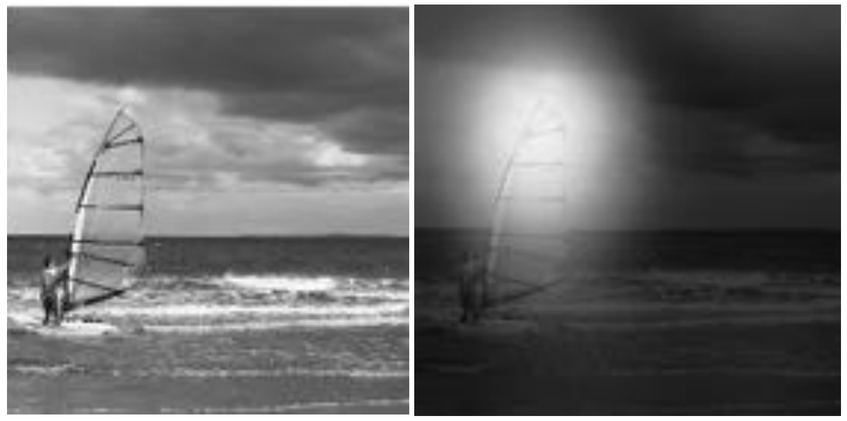




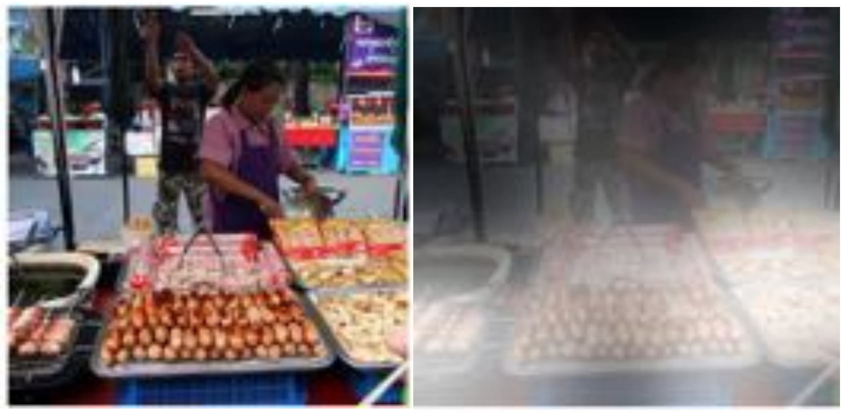
[1] D. Bahdanu et. al. "Neural Machine Translation by Jointly Learning to Align and Translate"

Motivation (Show, attend and tell …)

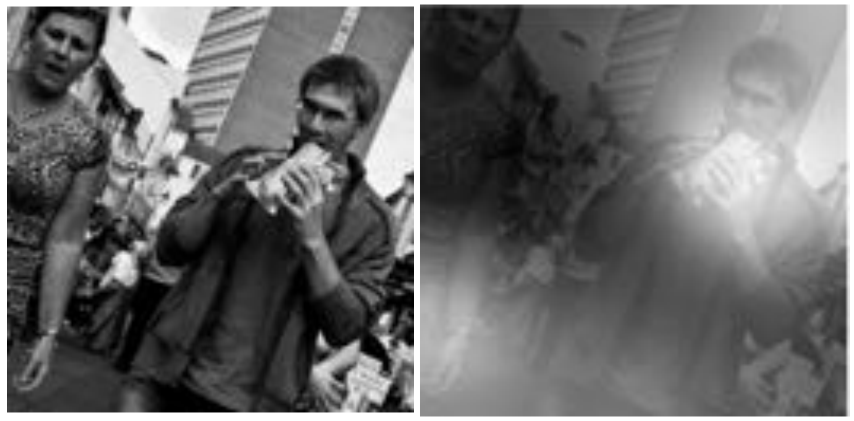
- Motivation
  - Increase the capacity of the encoder that compress the input into a single vector
  - Increase interpretability - errors inspection
- Two attention mechanism
  - 'Soft' deterministic trained via backprop
  - 'Hard' stochastic trained via variational lower bound
- Language generation task



A person is standing on a beach with a surfboard.



A woman is sitting at a table with a large pizza.



A man is talking on his cell phone while another man watches.

Extension of LSTM via the context vector

- Extract L D-dimensional annotations $a = \{\mathbf{a}_1, \ldots, \mathbf{a}_L\}, \ \mathbf{a}_i \in \mathbb{R}^D$
  - Lower convolutional layer to have the correspondence between the feature vectors and portions of the 2-D image

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

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

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

E - embedding matrix
y - captions
h - previous hidden state
z - context vector, a dynamic representation of the relevant part of the image input at time t

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

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

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

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

$$\boxed{p(\mathbf{y}_t | \mathbf{a}, \mathbf{y}_1^{t-1}) \propto \exp(\mathbf{L}_o(\mathbf{E}\mathbf{y}_{t-1} + \mathbf{L}_h \mathbf{h}_t + \mathbf{L}_z \hat{\mathbf{z}}_t))}$$

$f_{\text{att}}(\mathbf{a}_i, \mathbf{h}_{t-1})$ is MLP conditioned on the previous hidden state

Hard attention

We have two sequences
'i' that runs over localizations
't' that runs over words

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

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

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

Stochastic decisions are discrete here, so derivatives are zero.

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

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

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

$$= \log p(\mathbf{y} \mid \mathbf{a})$$

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

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

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

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

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

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

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

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

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

Soft attention

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

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

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

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

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

....

Theoretical arguments

- $\mathbb{E}_{p(s_t|a)}[\mathbf{h}_t]$ equals to computing $\mathbf{h}_t$ using a single forward prop with the expected context vector $\mathbb{E}_{p(s_t|a)}[\hat{\mathbf{z}}_t]$
- Normalized Weighted Geometric Mean approximation [1] $NWGM[p(y_t = k \mid \mathbf{a})] \approx \mathbb{E}[p(y_t = k \mid \mathbf{a})]$
- Finally

$$NWGM[p(y_t = k \mid \mathbf{a})] = \frac{\prod_i \exp(n_{t,k,i})^{p(s_{t,i}=1|a)}}{\sum_j \prod_i \exp(n_{t,j,i})^{p(s_{t,i}=1|a)}} = \frac{\exp(\mathbb{E}_{p(s_t|a)}[n_{t,k}])}{\sum_j \exp(\mathbb{E}_{p(s_t|a)}[n_{t,j}])}$$

$$\mathbb{E}[\mathbf{n}_t] = \mathbf{L}_o(\mathbf{E}\mathbf{y}_{t-1} + \mathbf{L}_h\mathbb{E}[\mathbf{h}_t] + \mathbf{L}_z\mathbb{E}[\hat{\mathbf{z}}_t])$$

[1] P. Baldi et. al. “The dropout learning algorithm”

Show, attend and tell - reminder of VGG

Key design choices

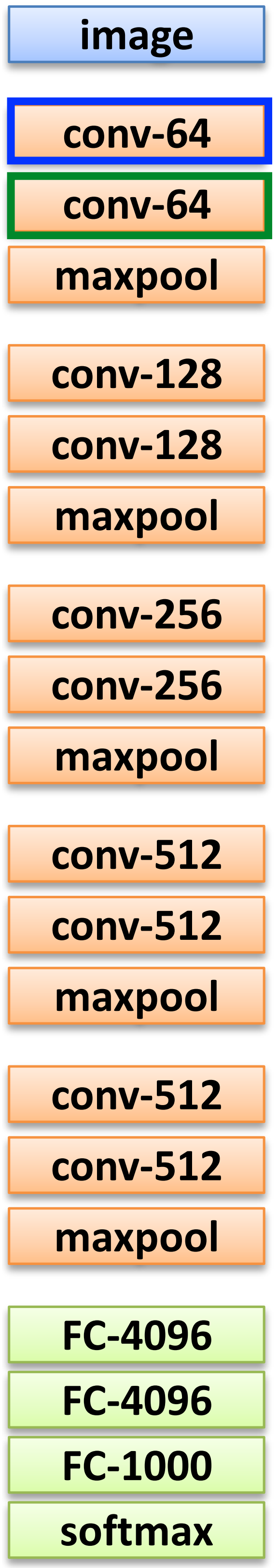
- Small conv kernels (3x3)
- Small stride=1, no information loss
- ReLU
- 5 max-pools (2x reduction)
- 3 fully-connected (FC) layers

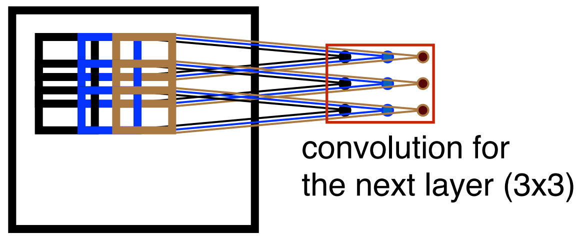
Why 3x3 layers?

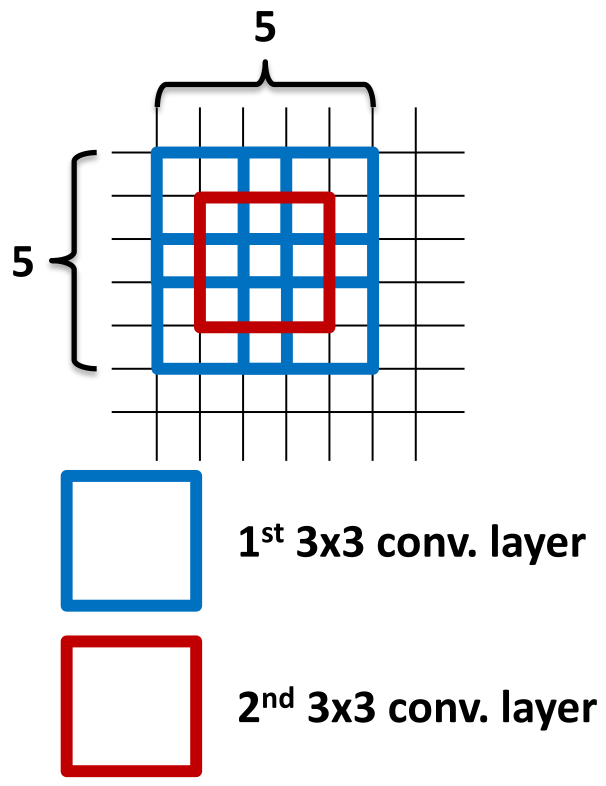
- stacked have large receptive fields
- more non-linearities
- less parameters

Training

- logistic regression
- mini-batch sgd with momentum
- fast convergence (74 epochs)
- most layers are initialized with Gaussian, other (FC layers and top conv 4) with 11 layer net

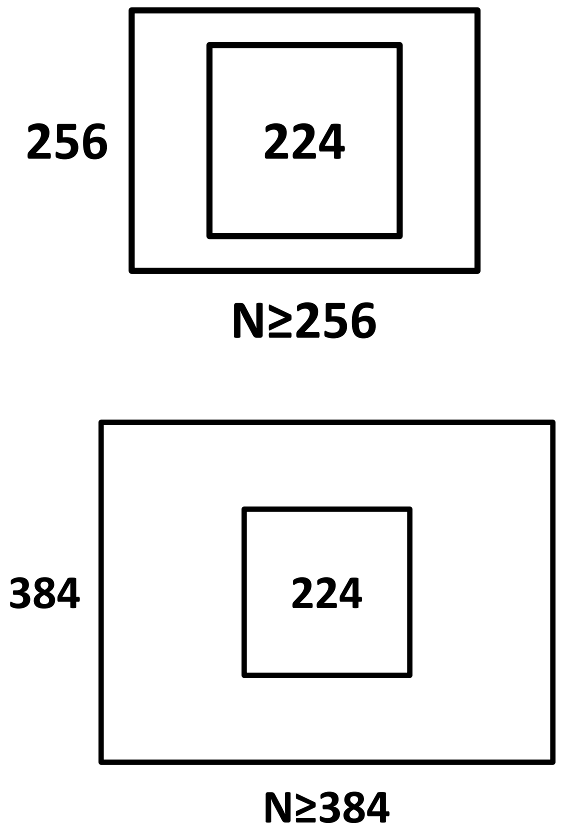






Multi-scale training

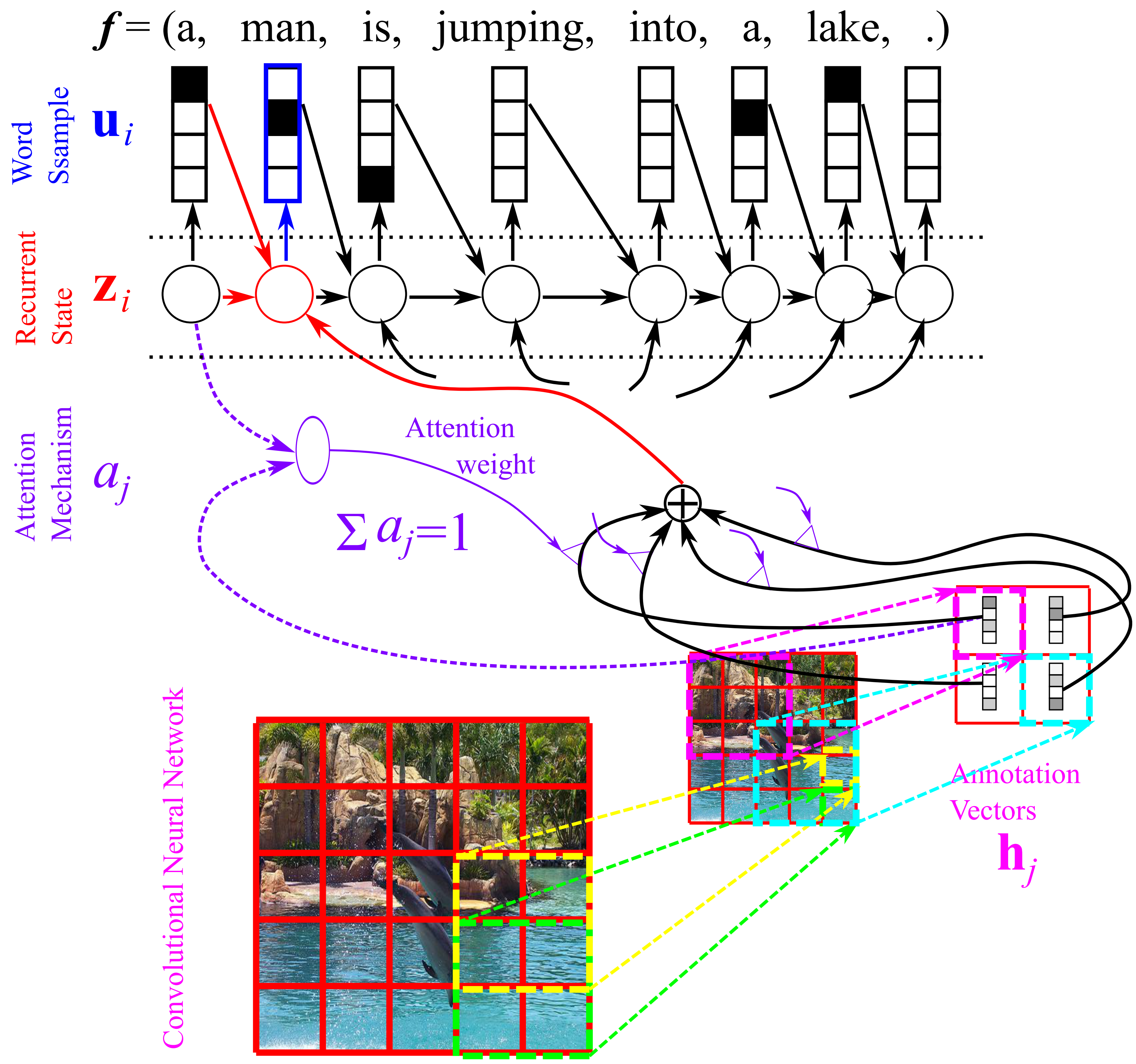
- randomly cropped inputs
- scale jittering



- Standard jittering
  - random horizontal flips
  - random RGB shift

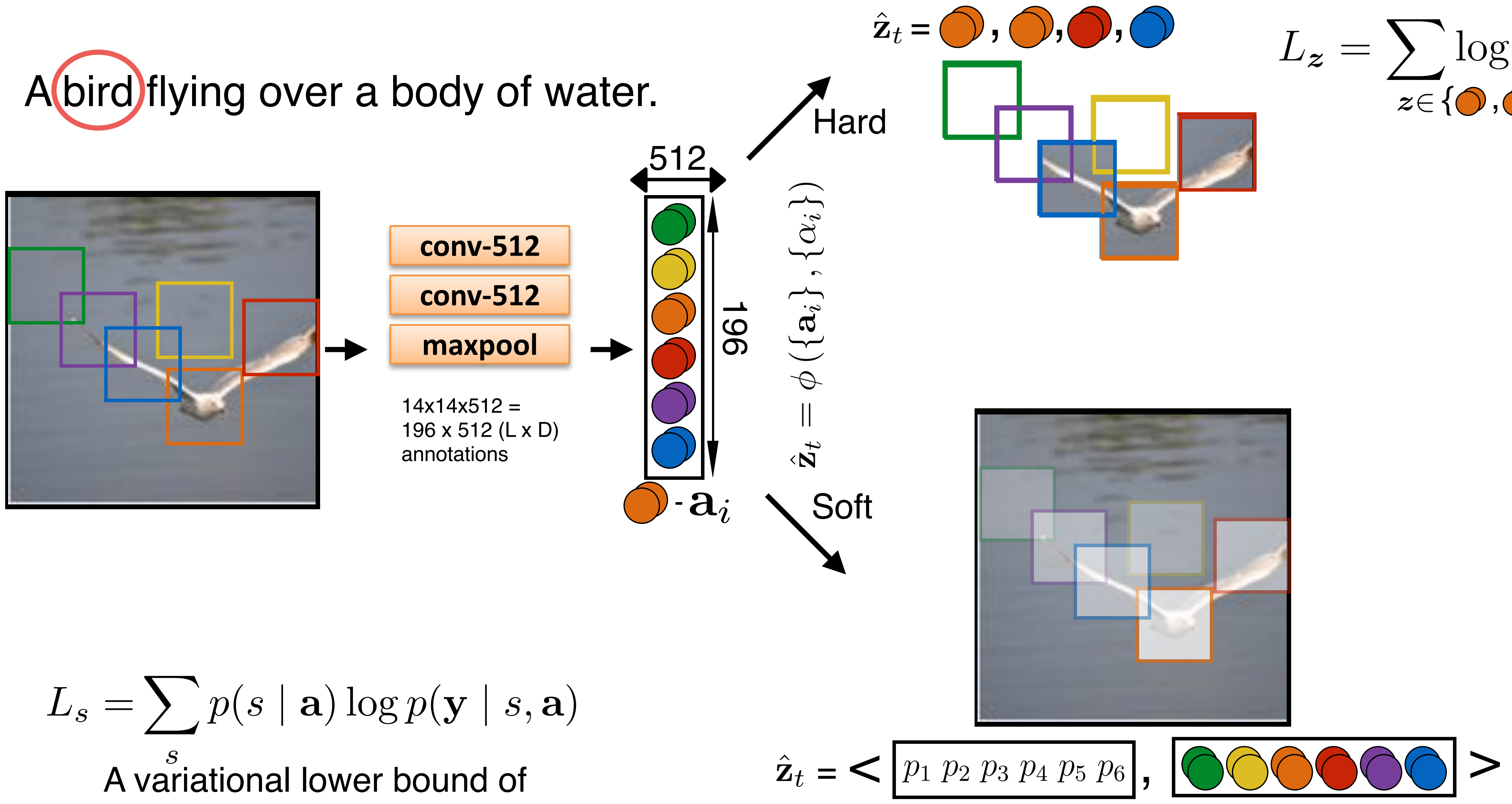
[1] K. Simonyan et. al. "Very Deep ConvNets for Large-Scale Image Recognition"

How soft/hard attention works



How soft/hard attention works

A bird flying over a body of water.



Sample regions of attention
$\hat{\mathbf{z}}_t$ = ●, ●, ●, ●
Hard
Soft
conv-512
conv-512
maxpool
14x14x512 = 196 x 512 (L x D) annotations
512
196
● - $\mathbf{a}_i$
$\hat{\mathbf{z}}_t = \phi(\{\mathbf{a}_i\}, \{\alpha_i\})$
$\hat{\mathbf{z}}_t$ = < $p_1\ p_2\ p_3\ p_4\ p_5\ p_6$, ●●●●●● >

$$L_z = \sum_{z \in \{●,●,●,●\}} \log p(\boldsymbol{y} \mid \boldsymbol{z})$$

Computes the expected attention

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

A variational lower bound of maximum likelihood

Training

- Adam for Flickr30k/MS COCO, RM-SProp on Flickr8k
- VGG to produce the annotations a_i pertained on ImageNet without fine-tuning (19 layers)
    - 14x14x512 feature map of the fourth convolutional layer
    - Flattened 196 x 512 (L x D) annotation (encoder)
    - small kernels (3x3) with stride 1 (no loss of information)

conv-512
conv-512
maxpool

- Mini-batches are built so that they data with captions of the same length are taken
- MS COCO + Soft attention on NVIDIA Titan Black <= 3 days of training
- Dropout + early stopping on BLEU scores
- Code in Theano

Qualitative results

Figure 2. Attention over time. As the model generates each word, its attention changes to reflect the relevant parts of the image. "soft" (top row) vs "hard" (bottom row) attention. (Note that both models generated the same captions in this example.)

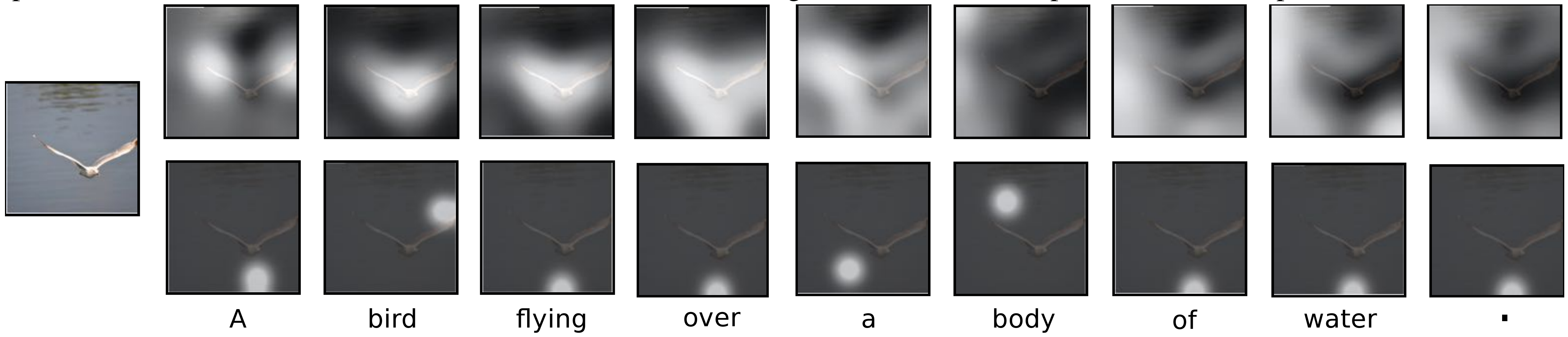
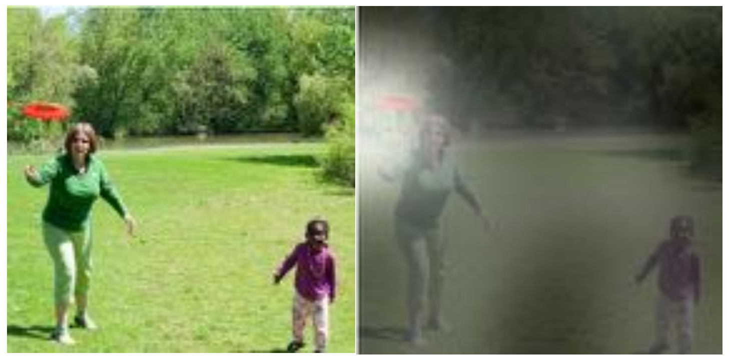
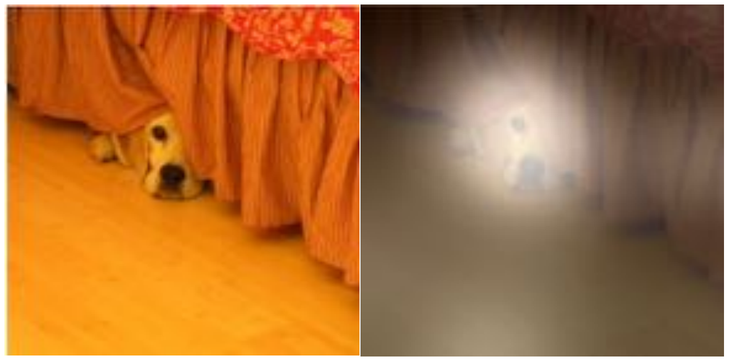
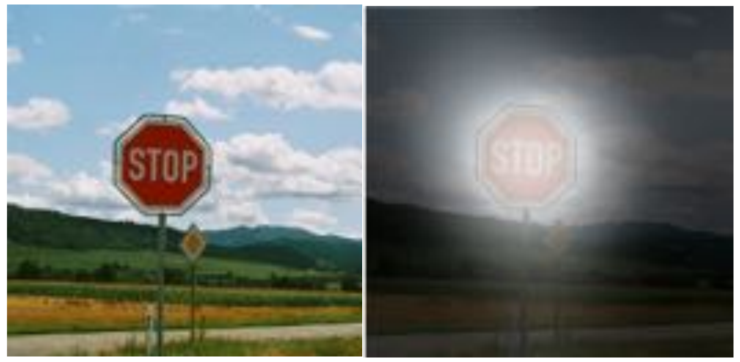
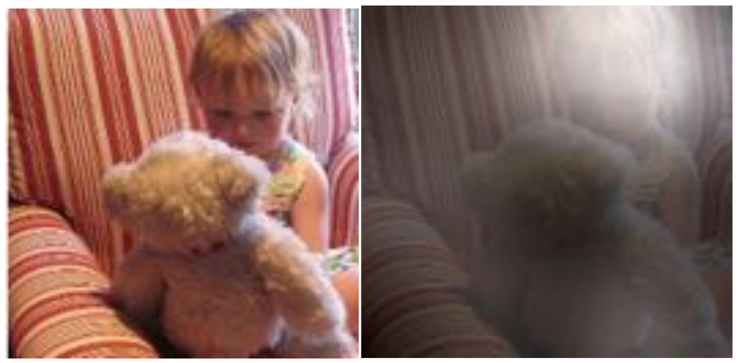


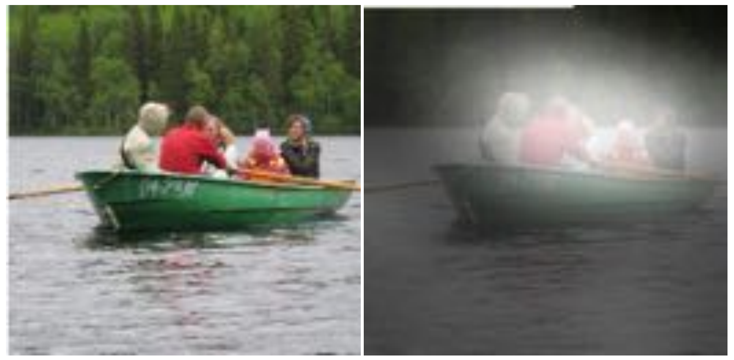
Figure 3. Examples of attending to the correct object (*white* indicates the attended regions, *underlines* indicated the corresponding word)

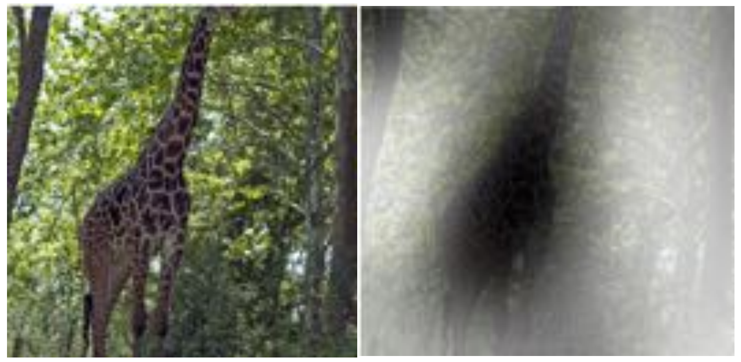

A woman is throwing a frisbee in a park.


A dog is standing on a hardwood floor.


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


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


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


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

Qualitative results

Figure 2. Attention over time. As the model generates each word, its attention changes to reflect the relevant parts of the image. “soft” (top row) vs “hard” (bottom row) attention. (Note that both models generated the same captions in this example.)

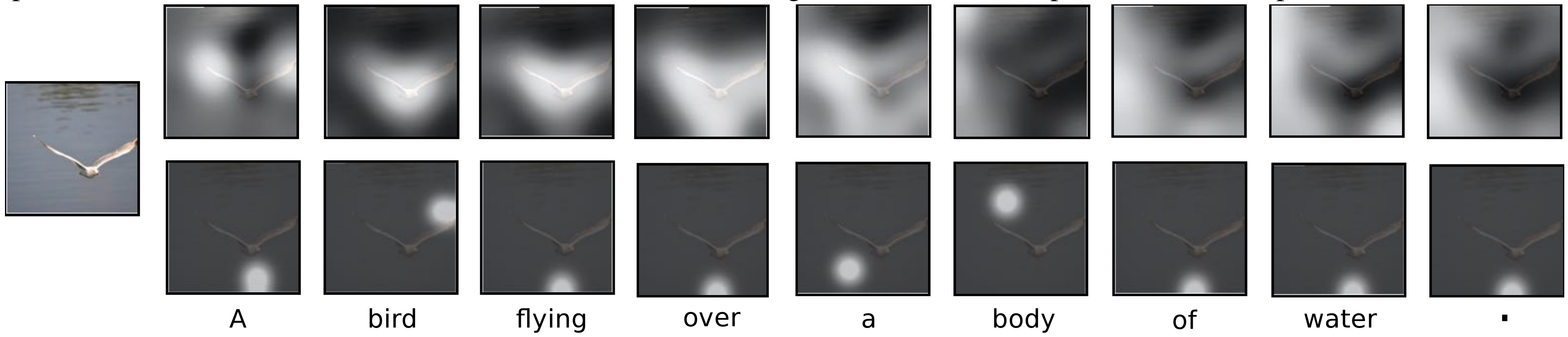
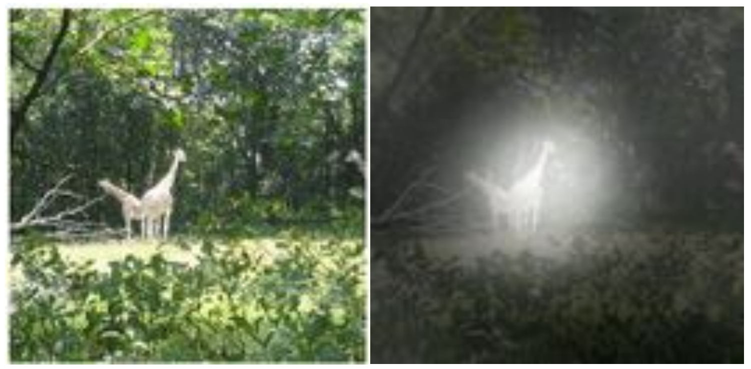


Figure 5. Examples of mistakes where we can use attention to gain intuition into what the model saw.


A large white bird standing in a forest.

A woman holding a clock in her hand.

A man wearing a hat and a hat on a skateboard.

A person is standing on a beach with a surfboard.

A woman is sitting at a table with a large pizza.

A man is talking on his cell phone while another man watches.

Quantitative results

Model	Human M1	Human M2	Automatic BLEU	Automatic CIDEr
Human	0.638	0.675	0.471	0.91
Google*	0.273	0.317	0.587	0.946
MSR$^\bullet$	0.268	0.322	0.567	0.925
Attention-based*	0.262	0.272	0.523	0.878
Captivator$^\circ$	0.250	0.301	0.601	0.937
Berkeley LRCN$^\diamond$	0.246	0.268	0.534	0.891

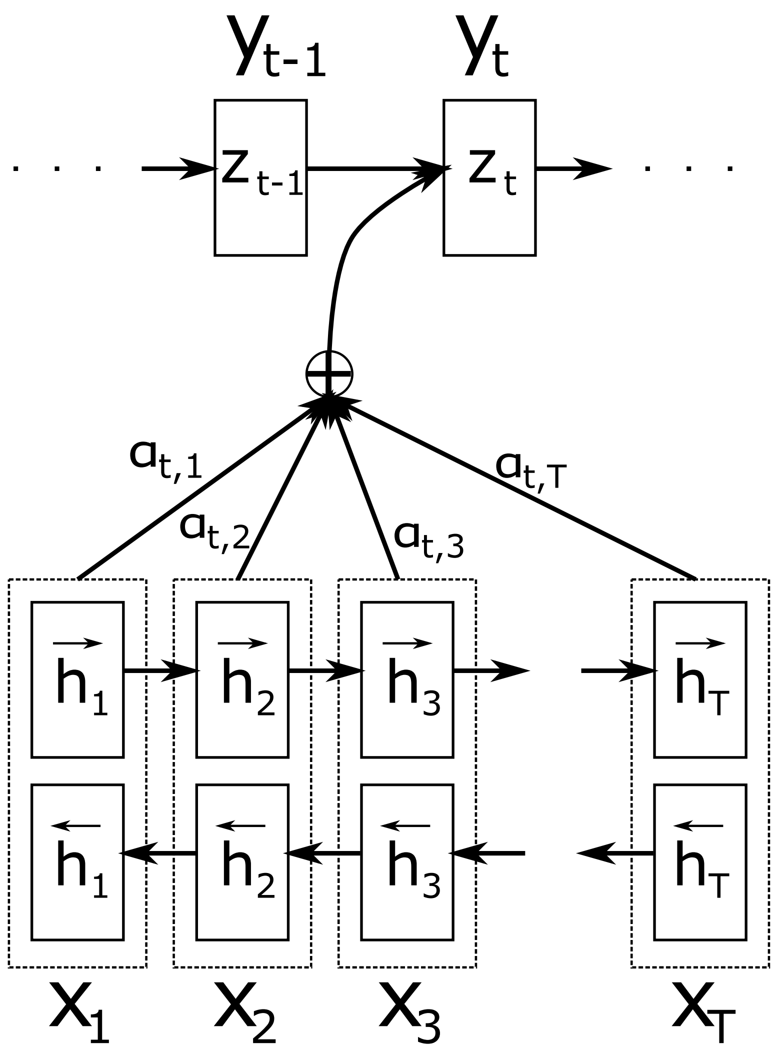
M1 - humans preferred (or equal) the method over human annotation
M2 - turing test

Other applications

Applications

Machine Learning Translation

- D. Bahdanu et. al. “Neural machine translation by jointly learning to align and translate”
- Make neural machine translation more robust to long sentences
- Bidirectional recurrent neural network (BiRNN) as encoder
- Context vector is a concatenation of the forward and backward networks $\mathbf{c}_t = \left[\overrightarrow{\mathbf{h}}_t; \overleftarrow{\mathbf{h}}_t\right]$
- BiRNN is crucial as the context information from the whole sentence is important
- Results comparable with the State-of-the-art SMT



y_{t-1}
y_t
z_{t-1}
z_t
$\oplus$
$a_{t,1}$
$a_{t,2}$
$a_{t,3}$
$a_{t,T}$
$\overrightarrow{h}_1$ $\overrightarrow{h}_2$ $\overrightarrow{h}_3$ $\overrightarrow{h}_T$
$\overleftarrow{h}_1$ $\overleftarrow{h}_2$ $\overleftarrow{h}_3$ $\overleftarrow{h}_T$
x_1 x_2 x_3 x_T

Model	BLEU	Rel. Improvement
Simple Enc–Dec	17.82	–
Attention-based Enc–Dec	28.45	+59.7%
Attention-based Enc–Dec (LV)	34.11	+90.7%
Attention-based Enc–Dec (LV)*	**37.19**	**+106.0%**
State-of-the-art SMT$^\circ$	37.03	–

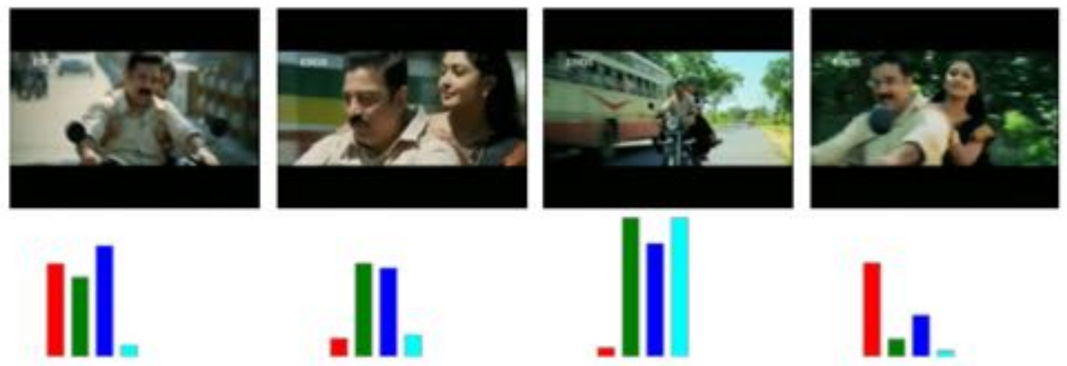
English-to-French translation task

Other applications

Applications

Video Description Generation

- L. Yao et. al. “Describing videos by exploiting temporal structure”
- Two encoders
  - Context set consists of per-frame context vectors, and attention mechanism that selects one of those vectors for each output symbol being decoded - capturing the global temporal structure across frames
  - 3-D conv-net that applies local filters across spation-temporal dimensions working on motion statistics
- Both encoders are complementary

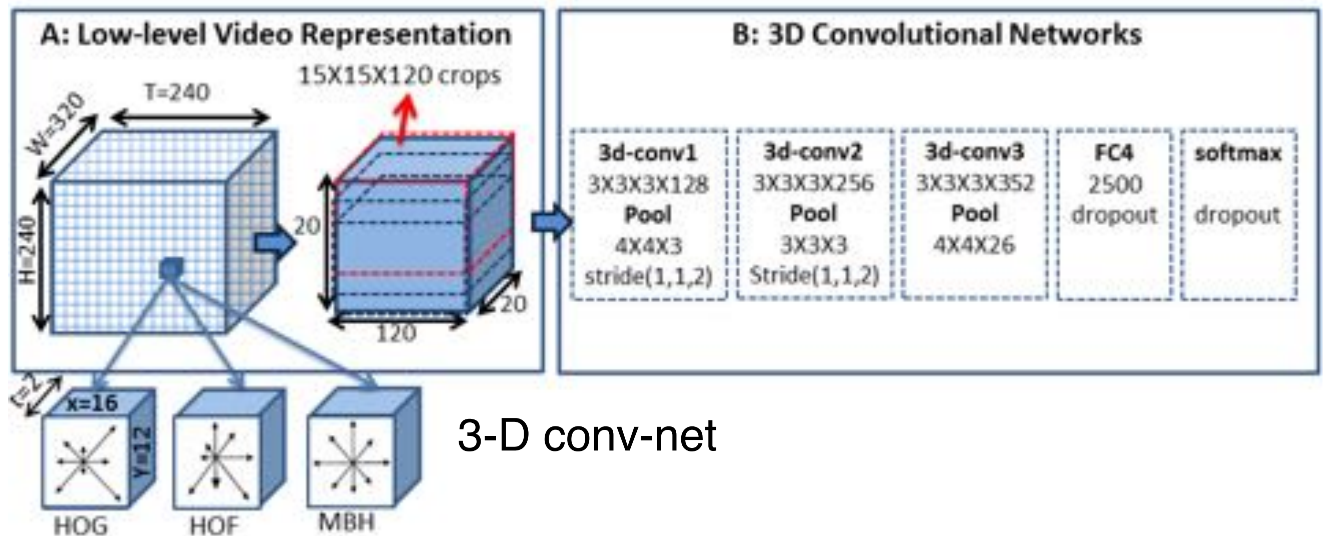


+Local+Global: A man and a woman are talking on the road

Ref: A man and a woman ride a motorcycle

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

Ref: A woman is frying food



3-D conv-net

THE PERFORMANCE OF THE VIDEO DESCRIPTION GENERATION MODELS ON YOUTUBE2TEXT AND MONTREAL DVS. (⋆) HIGHER THE BETTER. (○) LOWER THE BETTER.

Model	Youtube2Text		Montreal DVS	
	METEOR⋆	Perplexity○	METEOR	Perplexity
Enc-Dec	0.2868	33.09	0.044	88.28
+ 3-D CNN	0.2832	33.42	0.051	84.41
+ Per-frame CNN	0.2900	27.89	.040	66.63
+ Both	0.2960	27.55	0.057	65.44

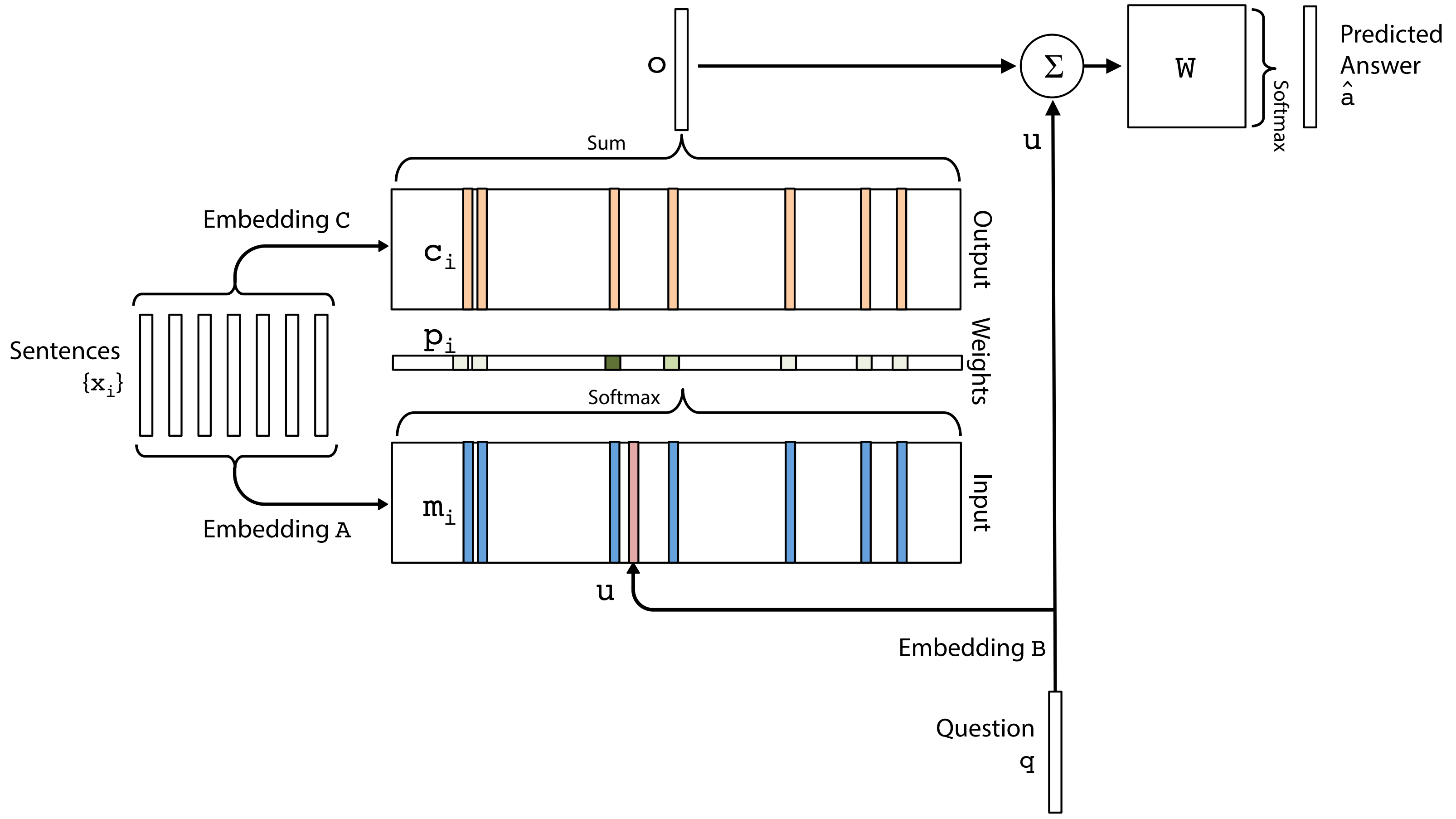
Other applications

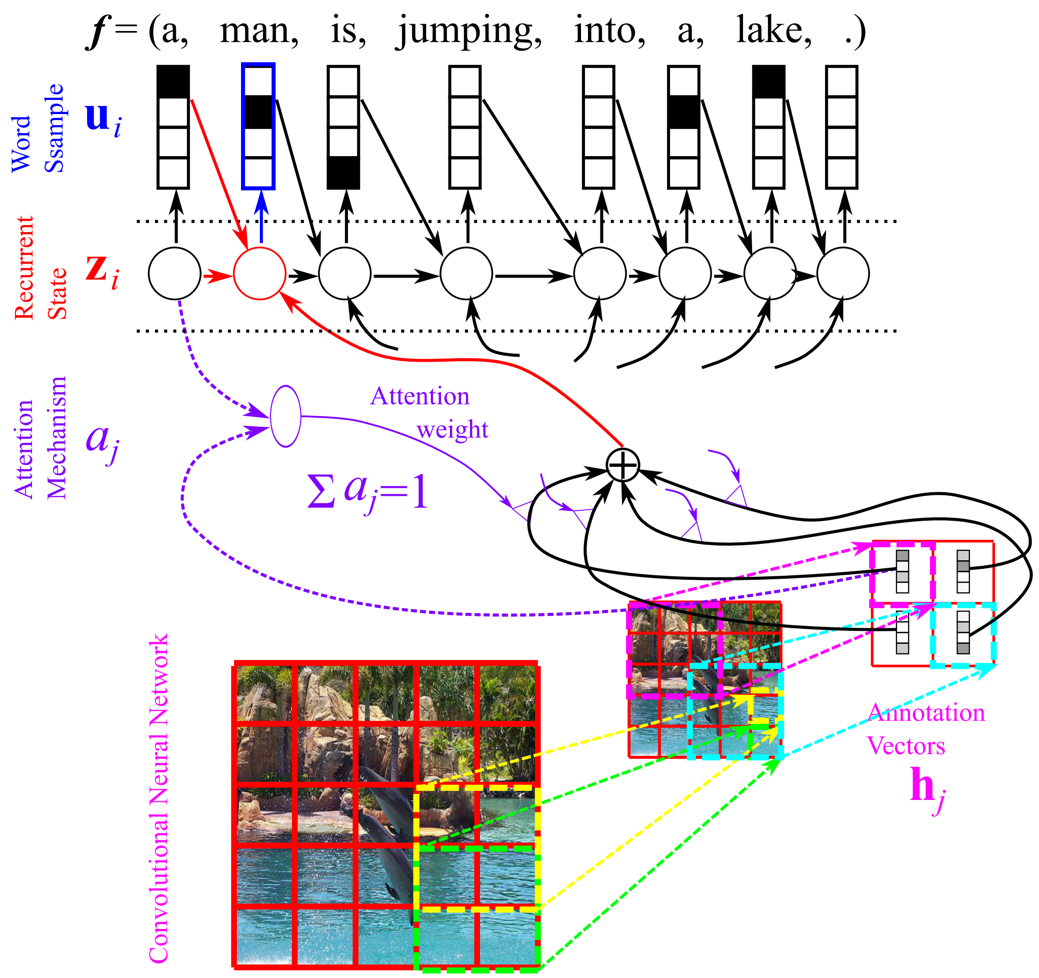
Applications

- Parsing-Grammar
  - Machine Translation with a parsing-tree as a ‘target sentence’
  - Learnt parsing algorithm performance matches state-of-the-art (domain-specific) parsers
  - O. Vinyals et. al. “Grammar as a foreign language”
- (Approximately) Solving combinatorial problems
  - Decoder predicts which one of the source symbols/nodes should be chosen at each time step
  - TSP
    - Context set = cities in the input graph
    - The attention mechanism choses cities
    - Generalizes to any discrete optimization problem whose solution is a subset of the input symbols
  - O. Vinyals et. al. “Pointer networks”
- Speech Recognition
  - Traditional approaches use Deep Nets for the acoustic part to establish a relationship between audio (wavelength) and phonemes followed by HMM to map those into sentences
  - J. Chorowski et. al. “Attention-based models for speech recognition”
    - Encoder is a stacked BiRNN, which reads the input sequence of speech frames
    - Context set is the concatenated hidden states of the top-level BiRNN
    - Peculiarities (in contrast to the machine translation task)
      - Significant difference in the input speech frames and output sequence of words
      - Alignment between the input and output symbols is monotonic
  - W. Chan et. al. “Listen, Attend and Spell”
    - Listener - pyramidal RNN encoder that accepts filter bank spectra as input
    - Speller - attention-based RNN decoder that emits characters as outputs

So far …

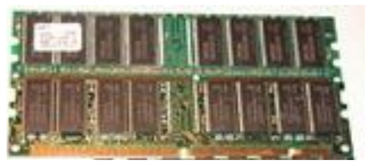
- Attention mechanism in Memory Networks
  - Distribution over different data points
  - Task is Question Answering about textual story
- Attention mechanism in Show, Attend, and Tell …
  - Visual attention as a normalized time-dependent linear map

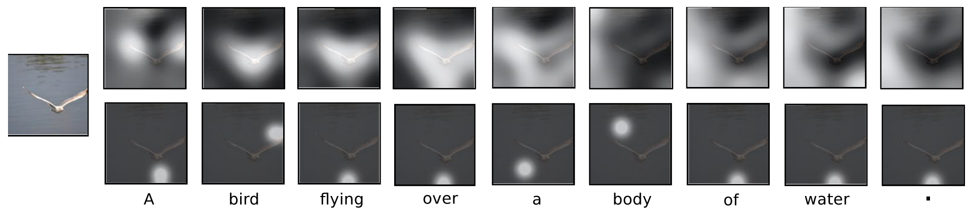


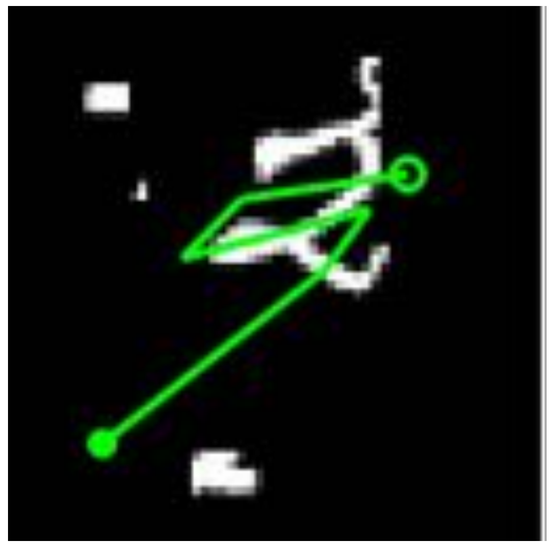


f = (a, man, is, jumping, into, a, lake, .)
Word Ssample $\mathbf{u}_i$
Recurrent State $\mathbf{z}_i$
Attention Mechanism a_j
Attention weight
$\sum a_j=1$
Annotation Vectors $\mathbf{h}_j$
Convolutional Neural Network

Recurrent Models of Visual Attention







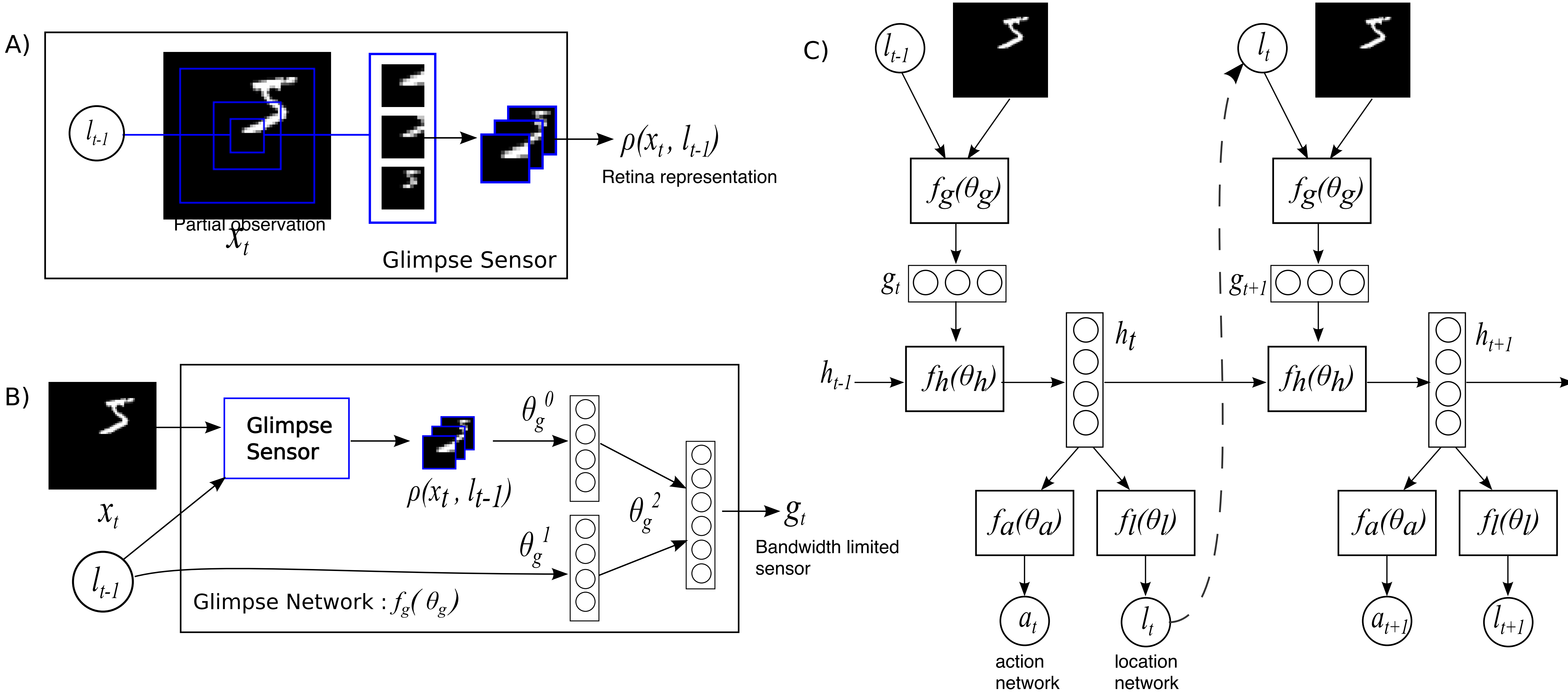
Glimpse-driven mechanism

Motivation

- Applying CNN is expensive
- Framework that
  - Selects a sequence of regions
  - Scales up independently of the image size
  - 4 x fewer floating point operations than CNN
- Model is non-differentiable
  - Reinforcement learning as a rescue

Model

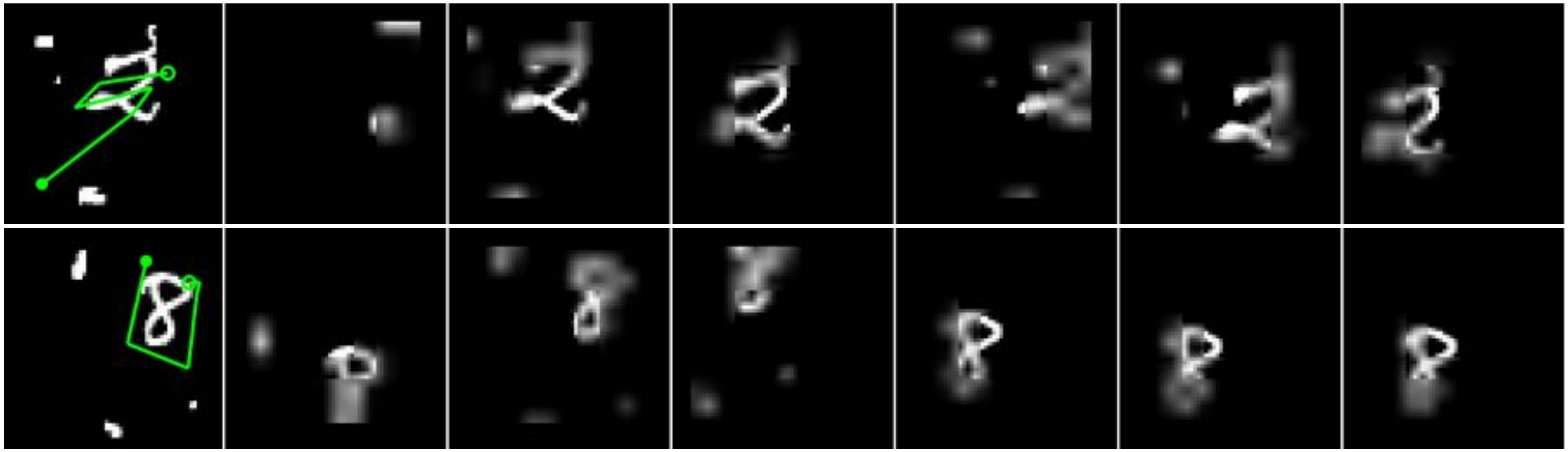
- Recurrent Attention Model (RAM)



The network (agent) can actively control how to deploy its sensor resources (choose the sensor location)

Glimpse - a multi-resolution crop of the input image

Results



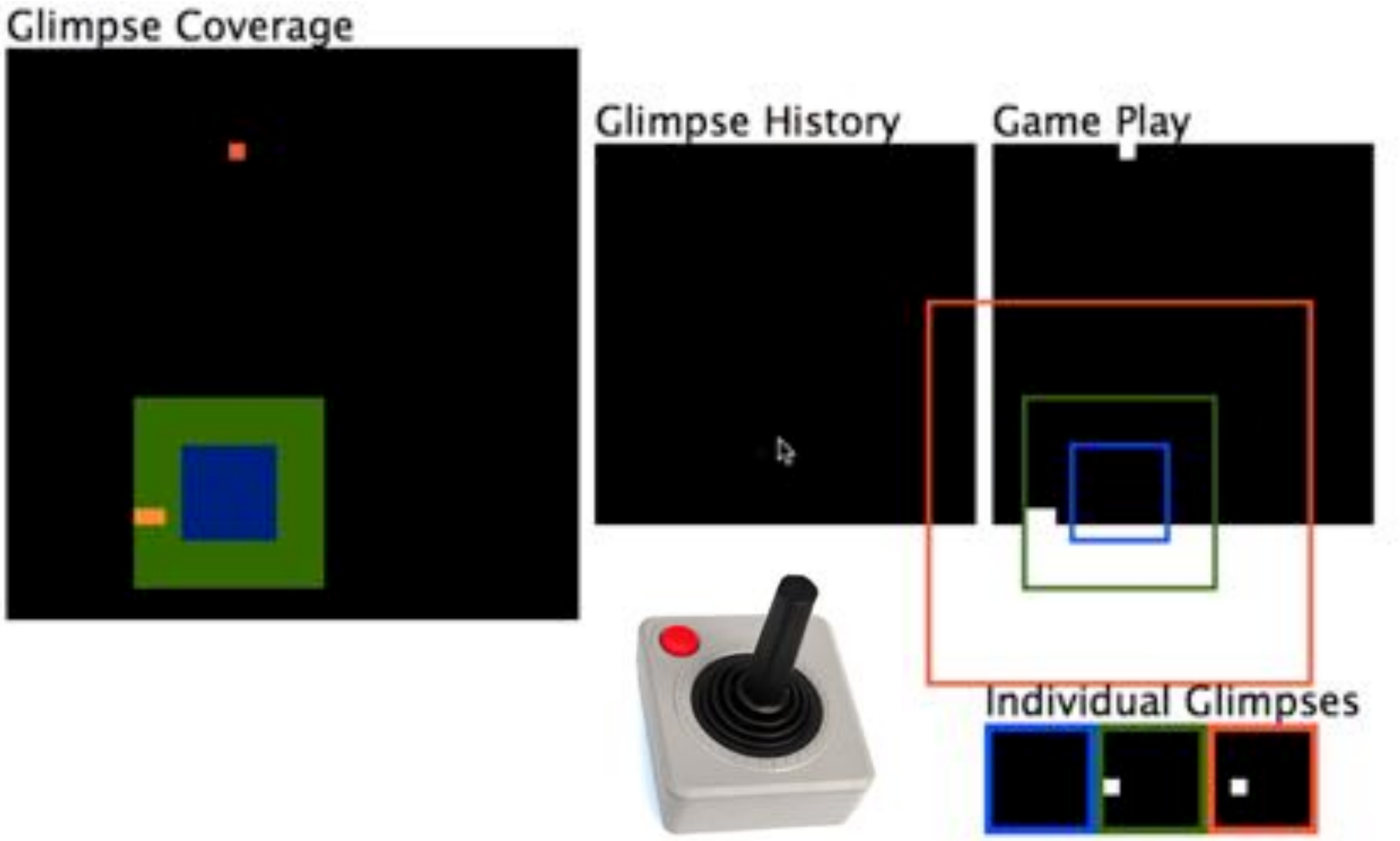
Glimpses deployed - 1st column shows the sequence of deployed glimpses, other columns show glimpses deployed

Dynamic environment

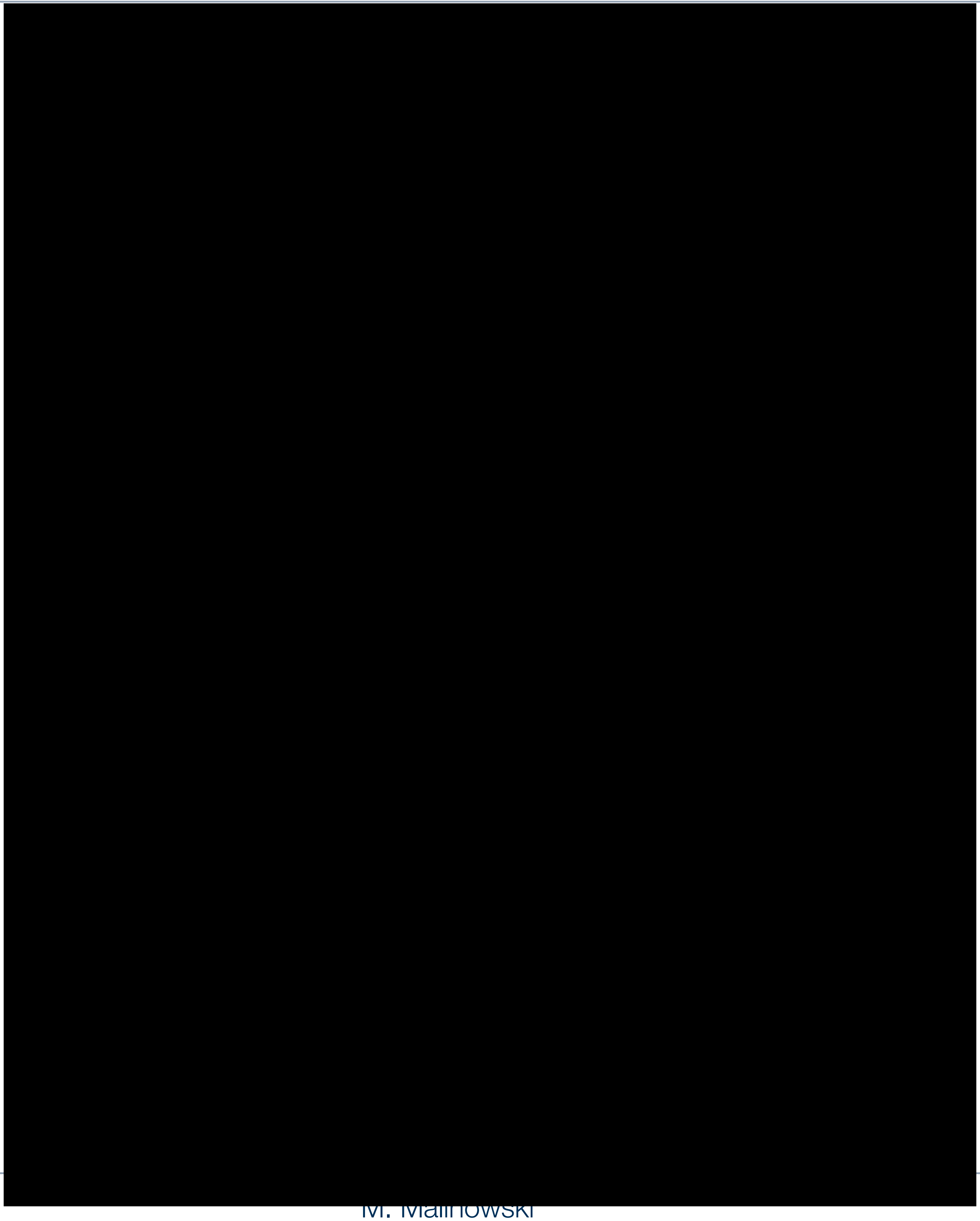
Sensor - agent receives a (partial) observation of the environment through bandwidth limited sensor

Actions - deploy sensor via the sensor control, and perform an environment action

Reward - $R = \sum_{t=1}^{T} r_t$ e.g. $r_T = 1$ if the object is classified correctly for detection



Dynamic environment - more games



Model - objective

- Maximize expected reward under the policy $\pi_\theta := p\left((l_j, a_j)_{j=0}^{T}\right)$

$$J(\theta) := E_{\pi_\theta}\left[\sum_{t=0}^{T} r_t\right] \qquad J(\theta) = \sum_{t=0}^{T}\sum_{a_t,\, l_t} r(a_t) p\left((l_t,\ a_t) \mid (l,\ a)_{0:(t-1)}\right)$$

- Gradient with sampling (REINFORCE rule [1])

$$\nabla J(\theta) = \sum_{t}\sum_{a_t, l_t} \underbrace{[r_t(l_t,\ a_t)]}_{\text{Use samples}} \underbrace{\nabla \pi_\theta\left((l_t,\ a_t) \mid (l_j,\ a_j)_{0:(t-1)}\right)}_{\text{How to sample from gradient?}}$$

$$\nabla J(\theta) = \sum_{t}\sum_{a_t,\, l_t} \left\{[r_t(l_t, a_t)] \underbrace{\nabla \log\left(\pi_\theta\right)}_{\text{Backprop}}\right\} \underbrace{\pi_\theta}_{\text{Sampling}}$$

Because of $(\log\ x)' = \frac{x'}{x}$

- Variance reduction techniques (bias normalization) [3]
- ‘Natural supervision’ - best actions are unknown and training signal comes only through the reward function
- Explore (samples), exploit (backprop)

[1] R.J, Williams “Simple statistical gradient-following algorithm for connectionist reinforcement learning”
[2] N. de Freitas “Deep Learning Lecture 15”
[3] R. S. Sutton et. al. “Policy gradient methods for reinforcement learning with function approximation”

Recurrent Models of Visual Attention - Results

Translated MNIST

Cluttered Translated MNIST

(a) 28x28 MNIST

Model	Error
FC, 2 layers (256 hiddens each)	**1.35**%
1 Random Glimpse, 8×8, 1 scale	42.85%
RAM, 2 glimpses, 8×8, 1 scale	6.27%
RAM, 3 glimpses, 8×8, 1 scale	2.7%
RAM, 4 glimpses, 8×8, 1 scale	1.73%
RAM, 5 glimpses, 8×8, 1 scale	1.55%
RAM, 6 glimpses, 8×8, 1 scale	**1.29**%
RAM, 7 glimpses, 8×8, 1 scale	1.47%

(b) 60x60 Translated MNIST

Model	Error
FC, 2 layers (64 hiddens each)	7.56%
FC, 2 layers (256 hiddens each)	3.7%
Convolutional, 2 layers	2.31%
RAM, 4 glimpses, 12×12, 3 scales	2.29%
RAM, 6 glimpses, 12×12, 3 scales	**1.86**%
RAM, 8 glimpses, 12×12, 3 scales	**1.84**%

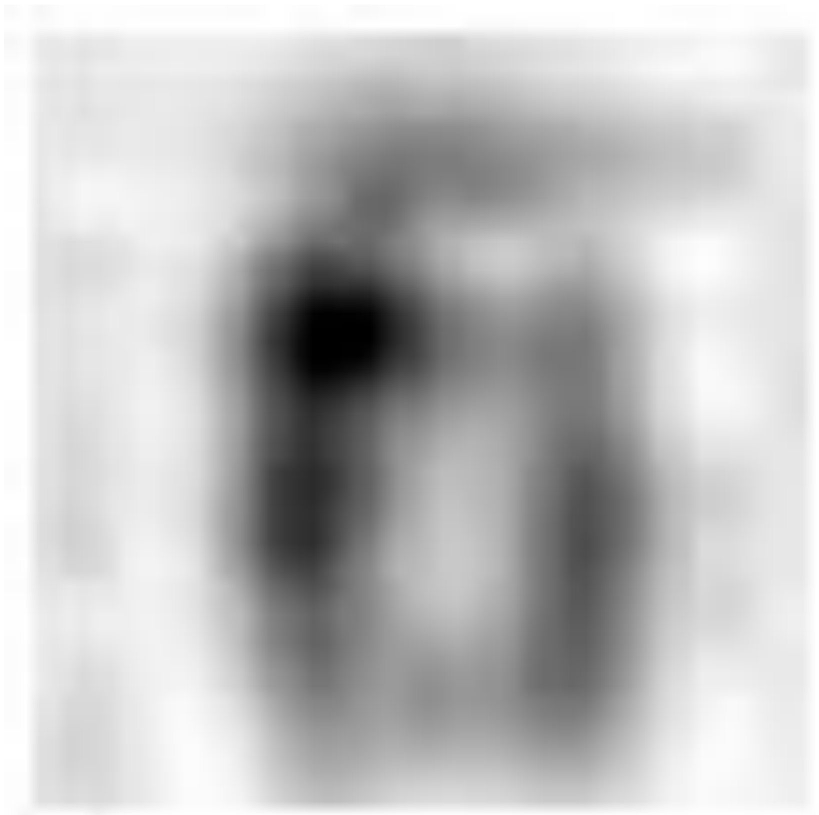
(a) 60x60 Cluttered Translated MNIST

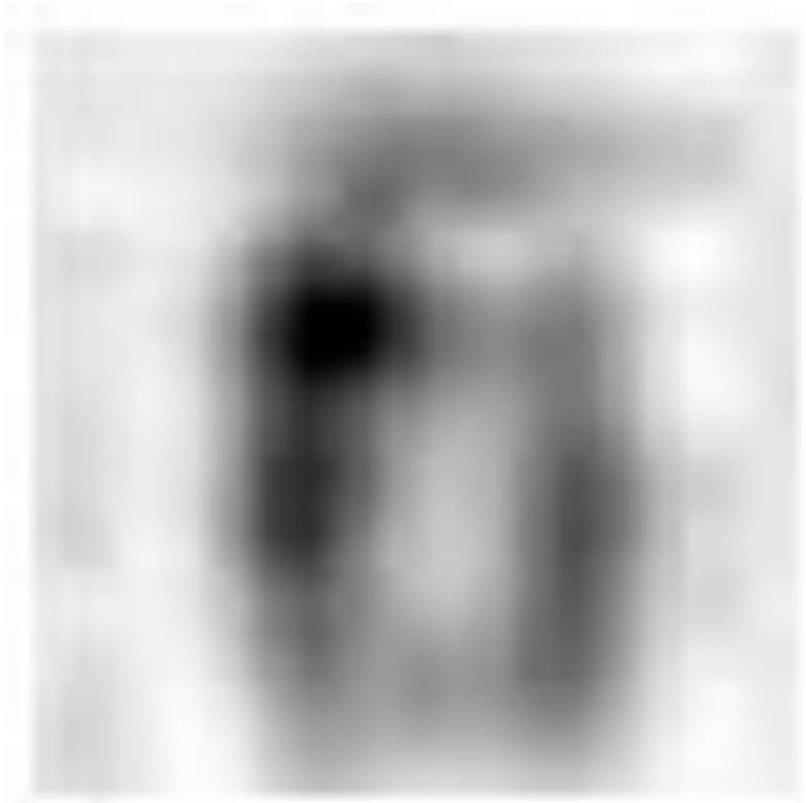
Model	Error
FC, 2 layers (64 hiddens each)	28.96%
FC, 2 layers (256 hiddens each)	13.2%
Convolutional, 2 layers	7.83%
RAM, 4 glimpses, 12×12, 3 scales	7.1%
RAM, 6 glimpses, 12×12, 3 scales	5.88%
RAM, 8 glimpses, 12×12, 3 scales	5.23%

(b) 100x100 Cluttered Translated MNIST

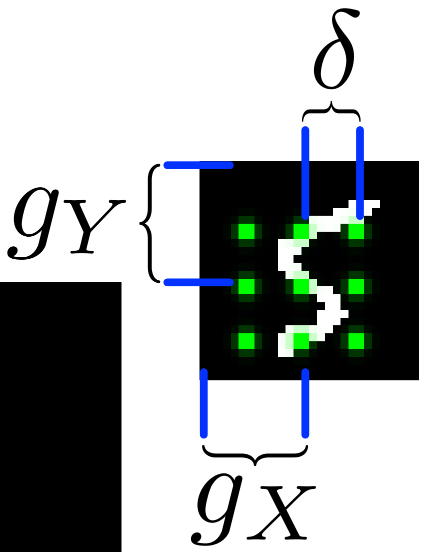
Model	Error
Convolutional, 2 layers	16.51%
RAM, 4 glimpses, 12×12, 4 scales	14.95%
RAM, 6 glimpses, 12×12, 4 scales	11.58%
RAM, 8 glimpses, 12×12, 4 scales	10.83%

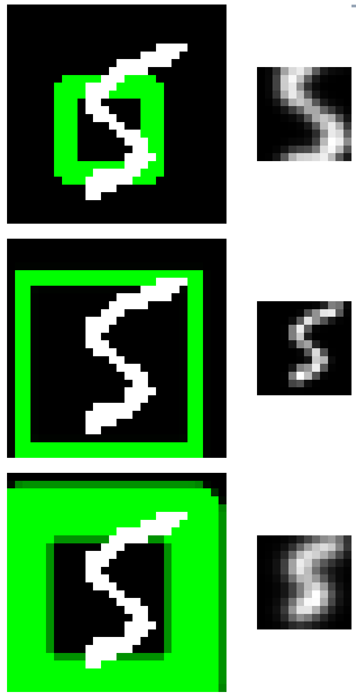
Multiple Object Recognition with Visual Attention



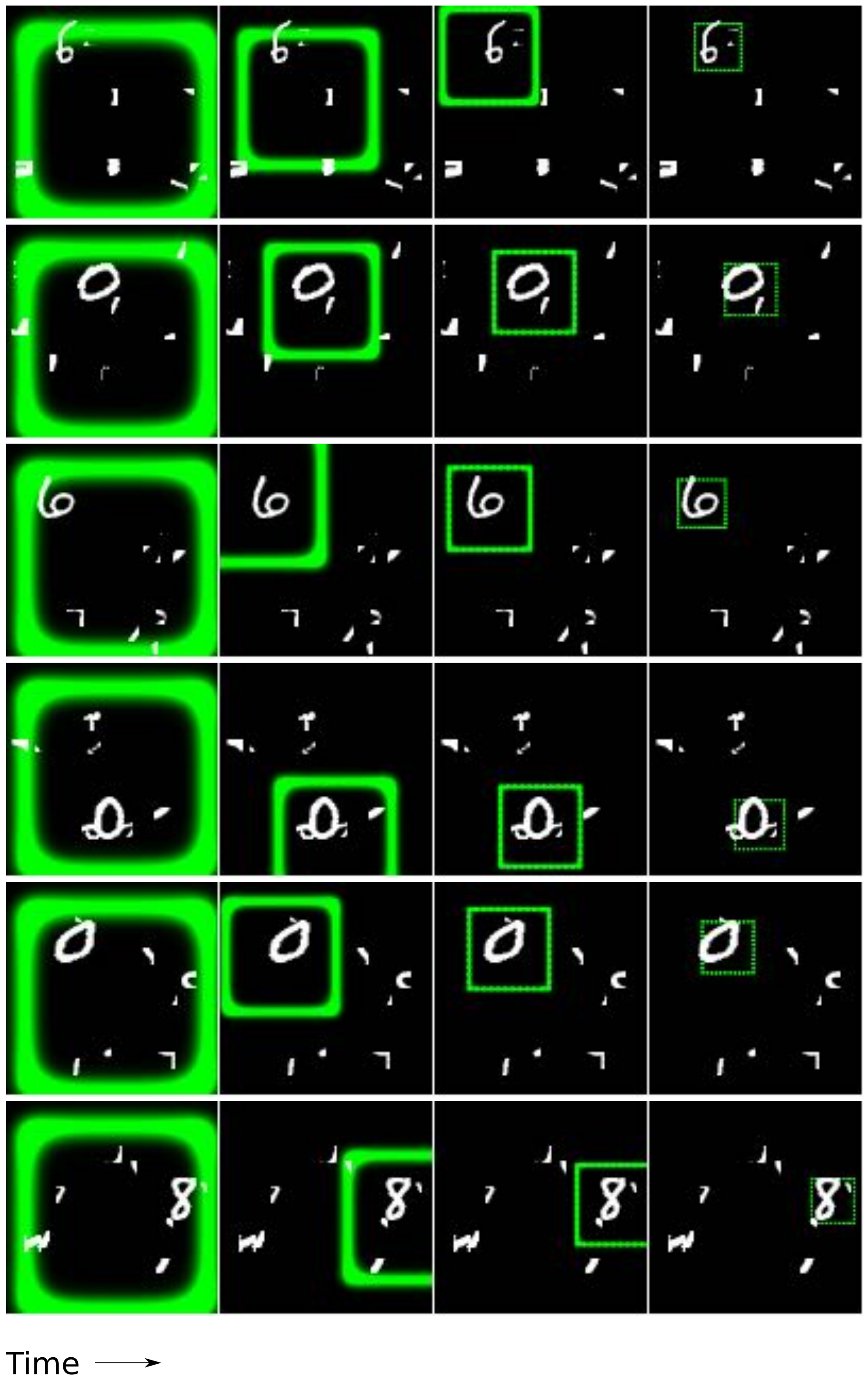


DRAW - Generative model with visual attention

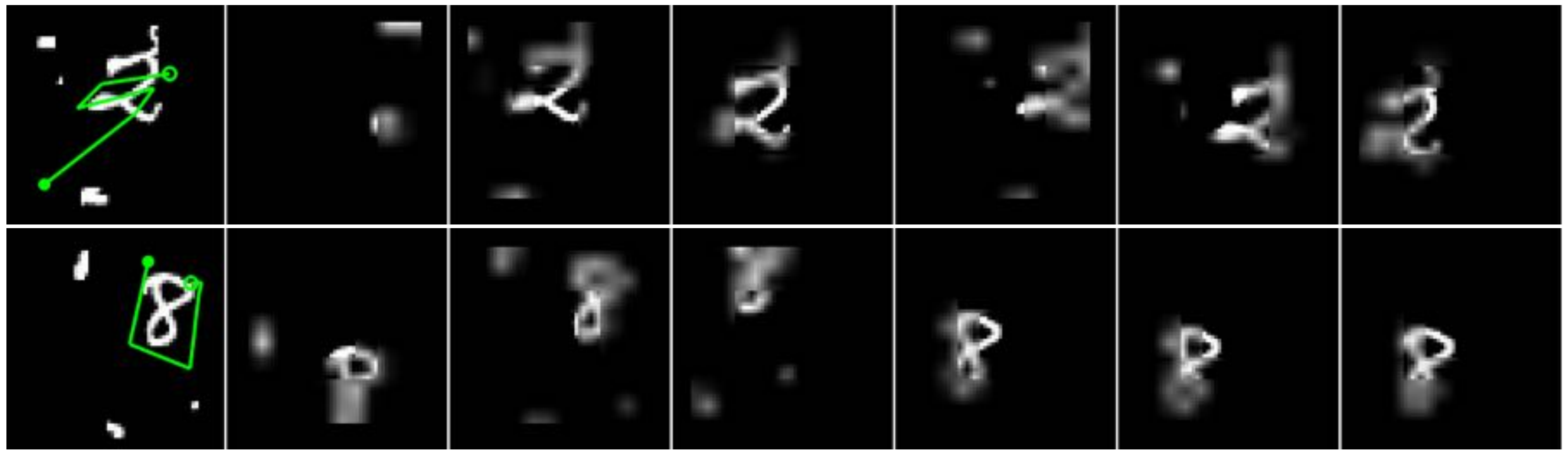




DRAW / Recognition

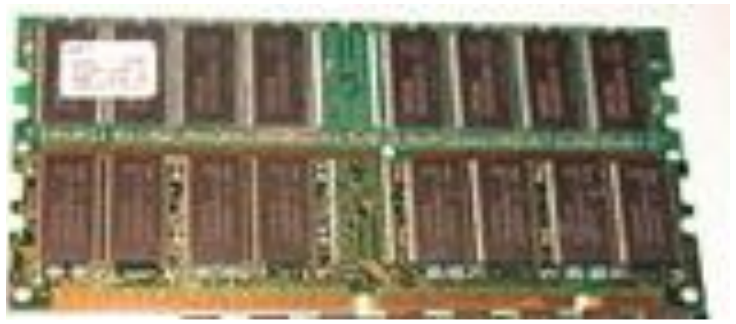


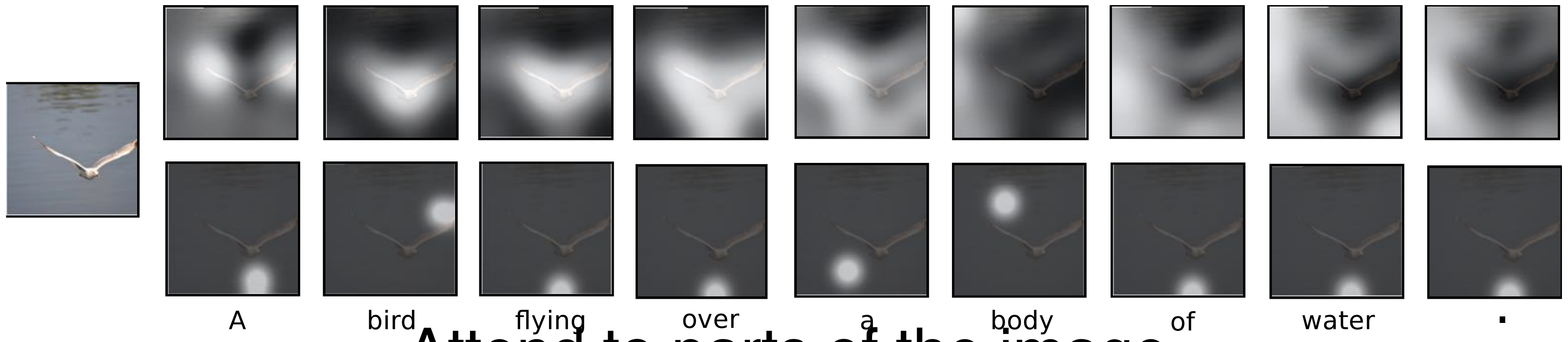
Draw - Continuous transitions (smooth pursuit?)



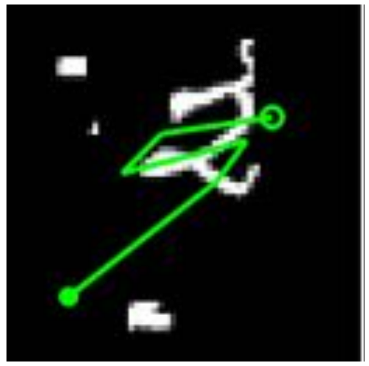
Recognition - Rapid jumps (saccades?)

Summary


Attend to memory cells



Attend to parts of the image



Glimpse-driven mechanism

Literature

- “Memory Networks” Weston et. al.
- “End-to-End Memory Networks” Sukhbaatar et. al.
- “Neural Turing Machines” Graves et. al.
- “Show, attend and tell: Neural Image Caption Generation with Visual Attention” Xu et. al.
- “Describing Multimedia Content using Attention-based Encoder-Decoder Networks” Cho et. al.
- "Recurrent Models of Visual Attention” Mnih et. al.
- “Multiple Object Recognition with Visual Attention” Ba et. al.
- “Describing videos by exploiting temporal structure” L. Yao et. al.

Literature

- "Neural Machine Translation by Jointly Learning to Align and Translate" D. Bahdanu et. al.
- "Grammar as a Foreign Language" O. Vinyals et. al.
- "Pointer Networks" O. Vinyals et. al.
- "Attention-based Models for Speech Recognition" J. Chorowski et. al.
- "Listen, Attend and Spell" W. Chan et. al.
- "DRAW: A Recurrent Neural Network for Image Generation" Gregor et. al.
- "Human-level Control through Deep Reinforcement Learning" (the Atari Games paper) Mnih et. al.
- Machine Learning: 2014-2015 at Oxford, N. de Freitas et. al.
  - https://www.cs.ox.ac.uk/people/nando.defreitas/machinelearning/