Tracking the World State with Recurrent Entity Networks

Mikael Henaff, Jason Weston, Arthur Szlam, Antoine Bordes, Yann LeCun

Task

At each timestep, get information (in the form of a sentence) about the state of the world. Then answer a question.

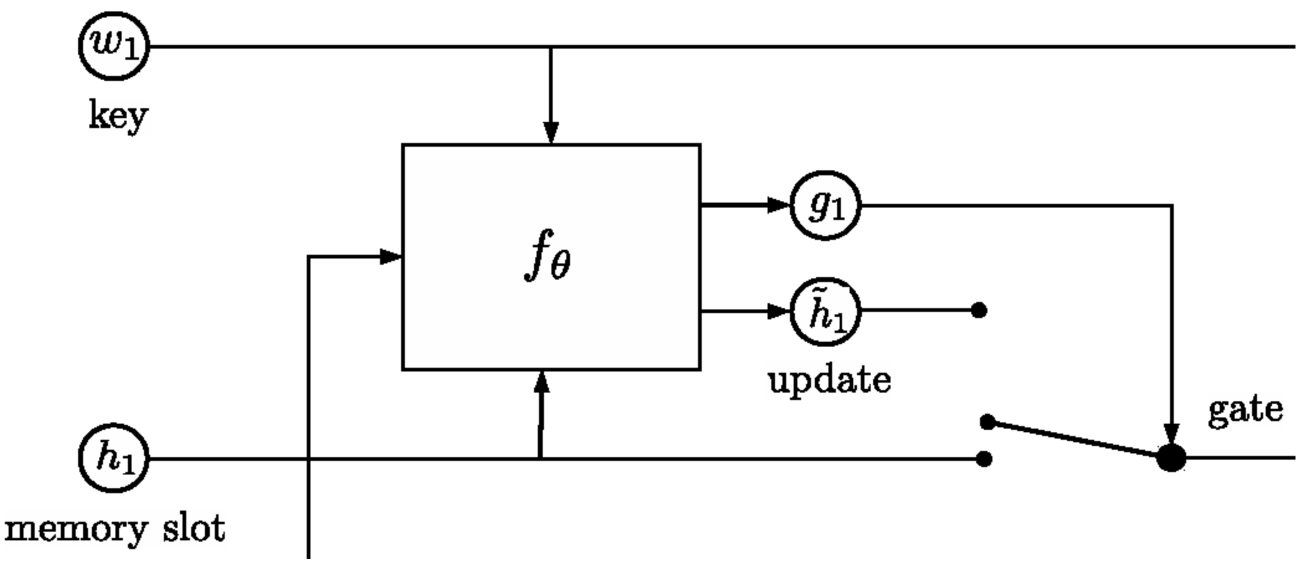
When we get new information, we should update our representation of the world state. The world state can be decomposed into the state of each entity in the world, so we only need to update one entity.

Architecture

The memory model:

- input: a sequence of vectors $s_1, \cdots s_T$
- output: a set of entity representations $h_1, \cdots h_k$

The world is a collection of entities. Information about each entity is stored in a single cell. Each cell comes with a key and a memory slot.



$\tilde{h}$ and g depends on h, w, s
standard gating mechanism:

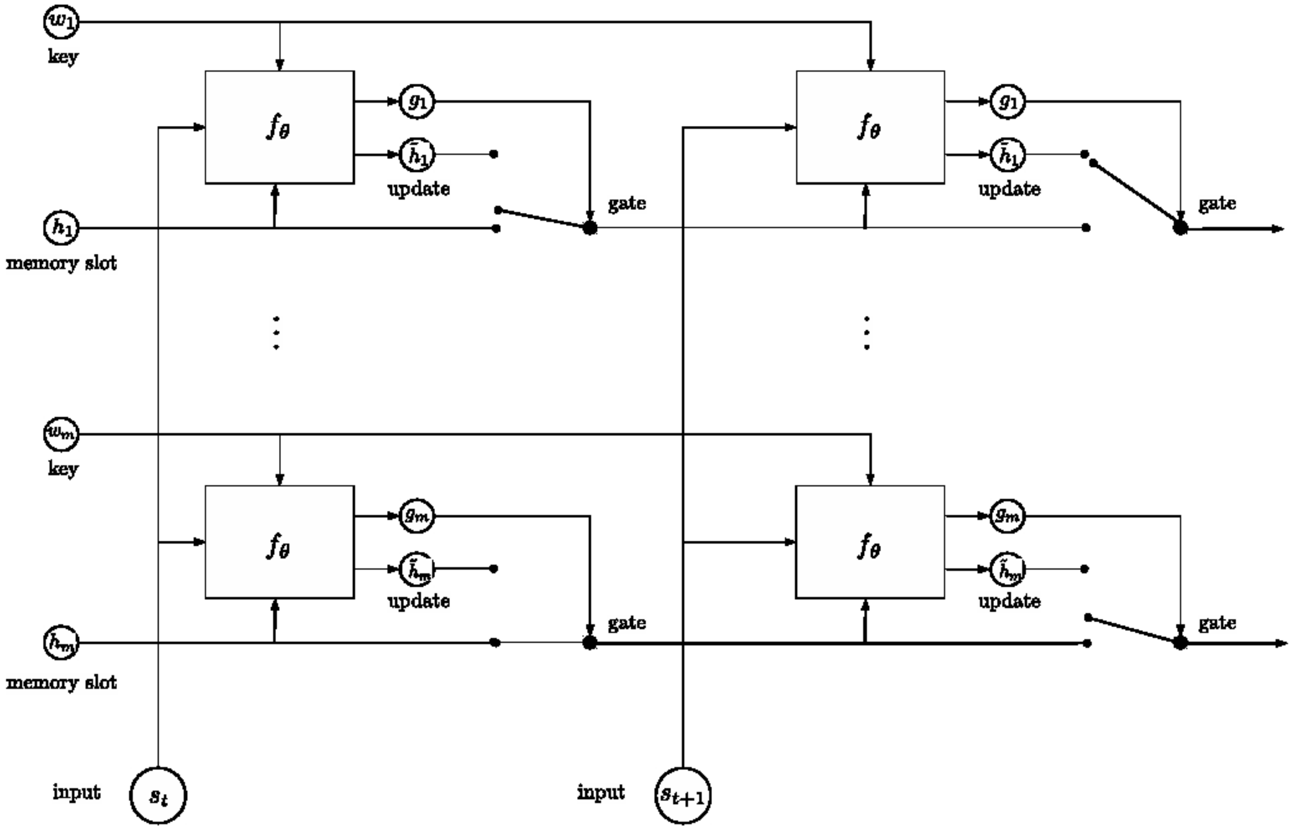
$$g_j \leftarrow \sigma(s_t^T h_j + s_t^T w_j)$$

$$\tilde{h_j} \leftarrow \phi(U h_j + V w_j + W s_t)$$

$$h_j \leftarrow h_j + g_j \odot \tilde{h_j}$$

$$h_j \leftarrow \frac{h_j}{||h_j||}$$

multiple-cells at multiple timesteps:



Input Encoder:

- input: a sequence of sentences.
- output: an encoding of each sentence as a fixed sized vector

$$s_t = \sum_i f_i \odot e_i$$

e_i are pretrained embeddings

Output Module:

- input: a query vector q and the outputs of the memory model
- output: arbitrary vector (log probabilities over words)

$$p_j = \text{Softmax}(q^T h_j)$$

$$u = \sum_j p_j h_j$$

$$y = R\phi(q + Hu)$$

Look at only one entity and drop the query:

$y = R\phi(Hh_j) = R\phi_j$

$y_i = R_i\phi_j$

Key vectors

- the model should identify entities by keys, which are trainable

$$g_j \leftarrow \sigma(s_t^T h_j + s_t^T w_j)$$

Key tying:

- Use parser to identify entities.
- One memory cell for each entity.
- Freeze key vector to be word embedding of an entity.

Related work

LSTM/GRU	RENN

scalar memory cell with full interaction	separate memory cells
just sigmoid layer of input and hidden state	content-based term between input and hidden state

LSTM:

- Forget gate layer:

$$f_t = \sigma_g(W_f x_t + U_f h_{t-1} + b_f)$$

- Input gate layer & tanh(hyperbolic tangent) layer:

$$i_t = \sigma_g(W_i x_t + U_i h_{t-1} + b_i)$$

$$c_t = f_t \circ c_{t-1} + i_t \circ \sigma_c(W_c x_t + U_c h_{t-1} + b_c)$$

Memory Network	**RENN**
store the entire input sequence in (dynamic long-term) memory	a fixed number of blocks (a window of words) of hidden states as memories
sequentially update a controller's hidden state via a softmax gating over the memories	update each block with an independent gated RNN

Gated graph network	**RENN**
inter-network communication with edges	parallel/independent recurrent models

Compared to RENN, CommNN, Interaction Network, Neural Physics Engine use parallel recurrent models without gating mechanism.

Experiments

Synthetic world model task

Task details:

- Two agents randomly placed in a 10x10 grid
- Answer the locations of the agents based on up to T-2 supporting facts

Details:

- 5 memory slots
- 20D per cell

Model	$T = 10$	$T = 20$	$T = 40$
MemN2N	0.09	0.633	0.896
LSTM	0	0.157	0.226
EntNet	0	0	0

(a)

bAbI

Details:

- 20 memory cells
- 100D embedding
- $U = V = 0, W = I, \phi =$ identity matrix

Story

```
mary got the milk there
john moved to the bedroom
sandra went back to the kitchen
mary travelled to the hallway
john got the football there
john went to the hallway
john put down the football
mary went to the garden
john went to the kitchen
sandra travelled to the hallway
daniel went to the hallway
mary discarded the milk
where is the milk ?
answer:  garden
```

Task	NTM	D-NTM	MemN2N	DNC	DMN+	EntNet
1: 1 supporting fact	31.5	4.4	0	0	0	0
2: 2 supporting facts	54.5	27.5	0.3	0.4	0.3	0.1
3: 3 supporting facts	43.9	71.3	2.1	1.8	1.1	4.1
4: 2 argument relations	0	0	0	0	0	0
5: 3 argument relations	0.8	1.7	0.8	0.8	0.5	0.3
6: yes/no questions	17.1	1.5	0.1	0	0	0.2
7: counting	17.8	6.0	2.0	0.6	2.4	0
8: lists/sets	13.8	1.7	0.9	0.3	0.0	0.5
9: simple negation	16.4	0.6	0.3	0.2	0.0	0.1
10: indefinite knowledge	16.6	19.8	0	0.2	0	0.6
11: basic coreference	15.2	0	0.0	0	0.0	0.3
12: conjunction	8.9	6.2	0	0	0.2	0
13: compound coreference	7.4	7.5	0	0	0	1.3
14: time reasoning	24.2	17.5	0.2	0.4	0.2	0
15: basic deduction	47.0	0	0	0	0	0
16: basic induction	53.6	49.6	51.8	55.1	45.3	0.2
17: positional reasoning	25.5	1.2	18.6	12.0	4.2	0.5
18: size reasoning	2.2	0.2	5.3	0.8	2.1	0.3
19: path finding	4.3	39.5	2.3	3.9	0.0	2.3
20: agent's motivation	1.5	0	0	0	0	0
Failed Tasks ($> 5\%$ error):	16	9	3	2	1	**0**
Mean Error:	20.1	12.8	4.2	3.8	2.8	**0.5**

Interpreting representations

Recall that

$$y_i = R_i \phi_j$$

Find closest R_i for each entity ϕ_j:

Key	1-NN	2-NN
football	hallway (0.135)	dropped (0.056)
milk	garden (0.111)	took (0.011)
john	kitchen (0.501)	dropped (0.027)
mary	garden (0.442)	took (0.034)
sandra	hallway (0.394)	kitchen (0.121)
daniel	hallway (0.689)	to (0.076)
bedroom	hallway (0.367)	dropped (0.075)
kitchen	kitchen (0.483)	daniel (0.029)
garden	garden (0.281)	where (0.026)
hallway	hallway (0.475)	left (0.060)

Story

```
mary got the milk there
john moved to the bedroom
sandra went back to the kitchen
mary travelled to the hallway
john got the football there
john went to the hallway
john put down the football
mary went to the garden
john went to the kitchen
sandra travelled to the hallway
daniel went to the hallway
mary discarded the milk
where is the milk ?
answer:  garden
```

CBT

Input:

1. 20 sentences
2. 21st sentence with missing word
3. list of candidate words

Details:

- Tied keys to candidate words
- Dropout
- $U = V = 0, W = I, \phi = \mathrm{id}$
- No normalization

	Model	Named Entities	Common Nouns
Single Pass	Kneser-Ney Language Model + cache	0.439	0.577
	LSTMs (context + query)	0.418	0.560
	Window LSTM	0.436	0.582
	EntNet (general)	0.484	0.540
	EntNet (simple)	**0.616**	**0.588**
Multi Pass	MemNN	0.493	0.554
	MemNN + self-sup.	0.666	0.630
	Attention Sum Reader (Kadlec et al., 2016)	0.686	0.634
	Gated-Attention Reader (Bhuwan Dhingra & Salakhutdinov, 2016)	0.690	0.639
	EpiReader (Trischler et al., 2016)	0.697	0.674
	AoA Reader (Cui et al., 2016)	0.720	0.694
	NSE Adaptive Computation (Munkhdalai & Yu, 2016)	**0.732**	**0.714**