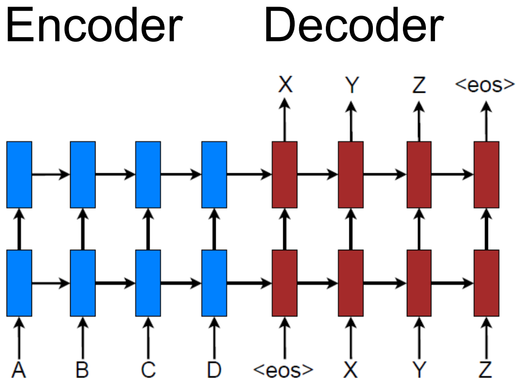
CS182/282A: Designing, Visualizing and Understanding Deep Neural Networks

John Canny

Spring 2019

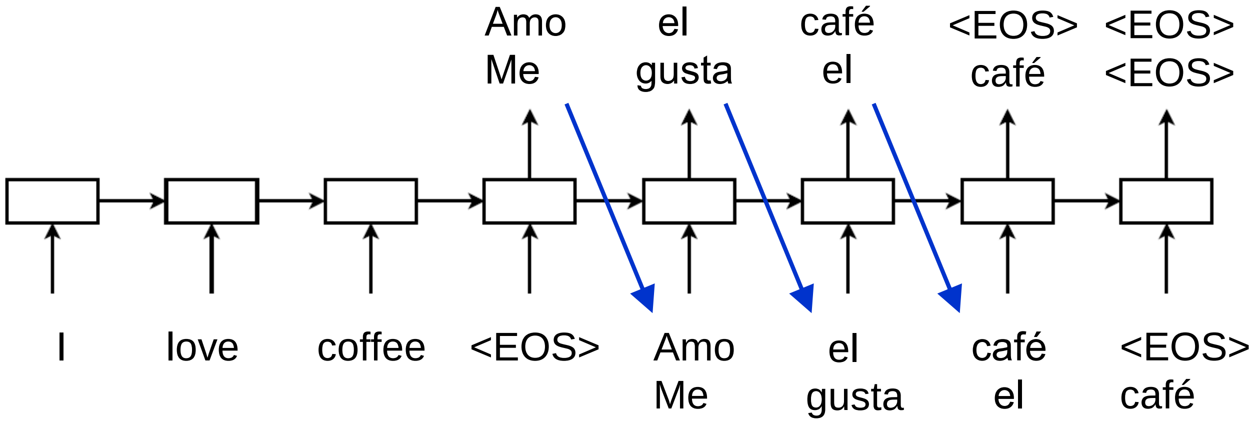
Lecture 14: Text Question and Answer Systems

Last Time: Sequence-To-Sequence Translation



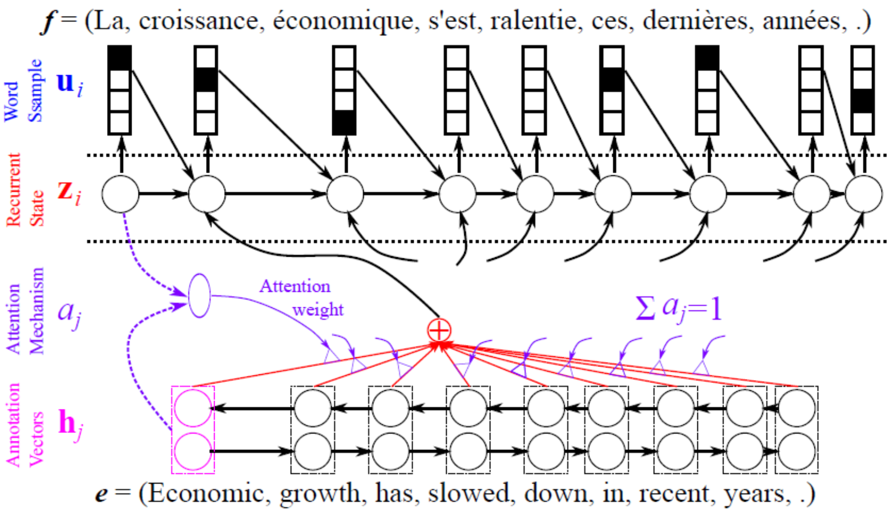
Encoder Decoder

A depth-2 model

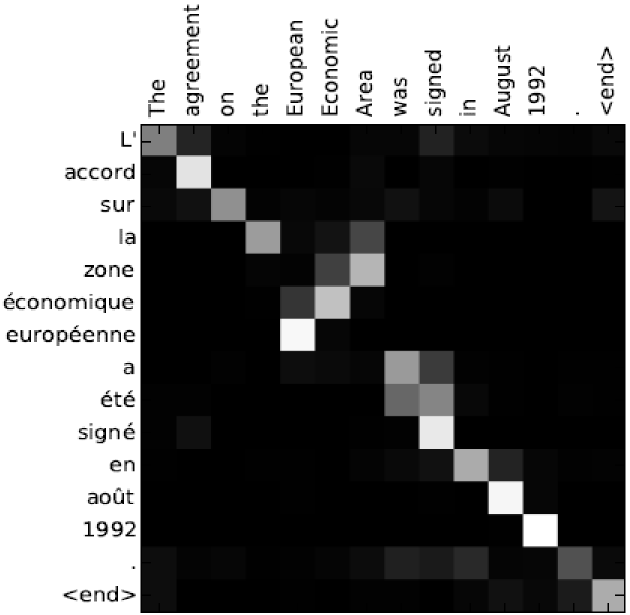


Beam search
width = 2

Last Time: Soft Attention for Translation



From Y. Bengio CVPR 2015 Tutorial

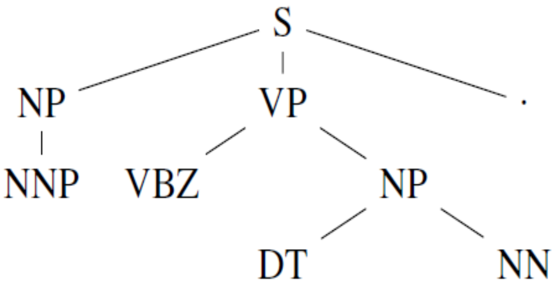


(a)

Last Time: Parsing as Translation

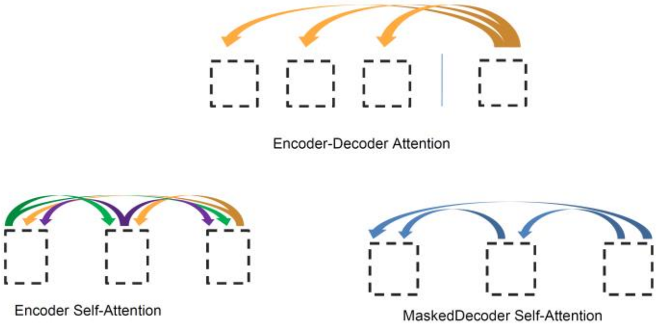
Sequence models generate linear structures, but these can easily encode trees by “closing parens” (prefix tree notation):

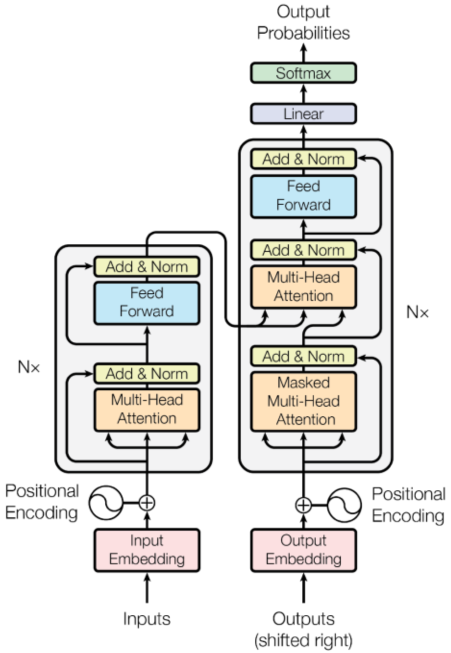
John has a dog . →

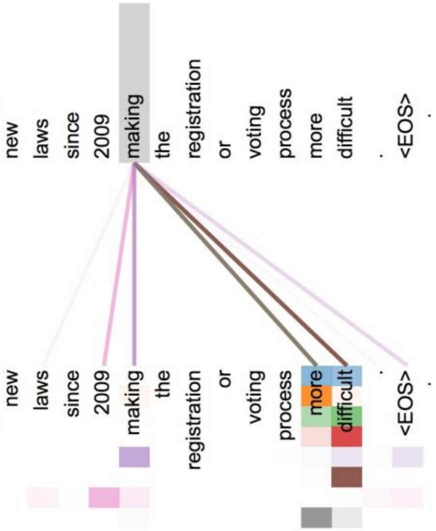


John has a dog . → (S (NP NNP $)_{NP}$ (VP VBZ (NP DT NN $)_{NP}$ $)_{VP}$. $)_{S}$

Last Time: Attention only Models: Transformer







Multi-headed self-attention

image from Lukas Kaiser, Stanford NLP seminar

This Time: Text Q&A Systems

"Hey Google, Remember that my next class
is on Monday"

"Hey Google, Remember that my next class
is about Text Q&A Systems"

"Hey Google, what is Monday's class about?" ☹

"Hey Google, what did I tell you about my next class?" ☺

Memory Networks

- Convolutional Networks: Activations (content) fully predictable from inputs.

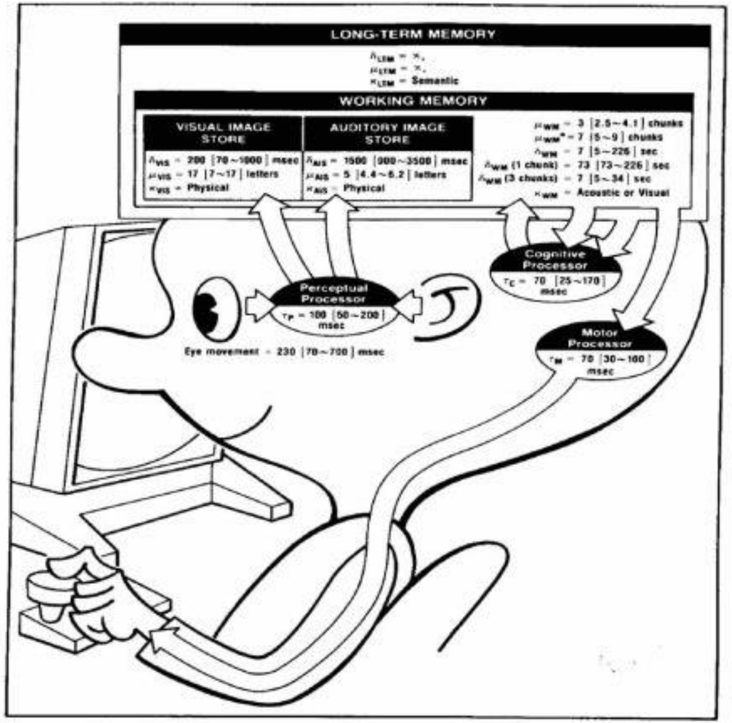
- Attention Models: Activations (content) depends mostly on the input, agent has also dynamic attention. You can think of attention as an analog of pointers or references in traditional programming languages.

- Memory Networks: Provide general purpose memory, pointers (via attention), and read/write capability. Critical for dynamic memory in conversational agents.

Human Memory

- Short-term or Working Memory: Dynamic, ephemeral, over a time scale of seconds.

- Long-Term Memory: Stores Events, Write-Once, Read-Many (WORM). Time frame is minutes to years.


LONG-TERM MEMORY
$\delta_{LTM} = \infty$
$\mu_{LTM} = \infty$
κ_{LTM} = Semantic
WORKING MEMORY
VISUAL IMAGE STORE
δ_{VIS} = 200 [70~1000] msec
μ_{VIS} = 17 [7~17] letters
κ_{VIS} = Physical
AUDITORY IMAGE STORE
δ_{AIS} = 1500 [900~3500] msec
μ_{AIS} = 5 [4.4~6.2] letters
κ_{AIS} = Physical
μ_{WM} = 3 [2.5~4.1] chunks
μ_{WM}^* = 7 [5~9] chunks
δ_{WM} = 7 [5~226] sec
δ_{WM} (1 chunk) = 73 [73~226] sec
δ_{WM} (3 chunks) = 7 [5~34] sec
κ_{WM} = Acoustic or Visual
Perceptual Processor
τ_P = 100 [50~200] msec
Eye movement = 230 [70~700] msec
Cognitive Processor
τ_C = 70 [25~170] msec
Motor Processor
τ_M = 70 [30~100] msec

Card, Moran and Simon “*The Model Human Processor: An Engineering Model of Human Performance*” *1986*

Memory Networks: Basic Dialog Tasks

Task 1:
1 Mary moved to the bathroom.
2 John went to the hallway.
3 Where is Mary? bathroom 1
4 Daniel went back to the hallway.
5 Sandra moved to the garden.
6 Where is Daniel? hallway 4
7 John moved to the office.
8 Sandra journeyed to the bathroom.
9 Where is Daniel? hallway 4
10 Mary moved to the hallway.
11 Daniel travelled to the office.
12 Where is Daniel? office 11

babI dataset: Facebook research

Memory Networks: babI dataset

Task 3
5 Mary journeyed to the office.
17 Mary journeyed to the bathroom.
23 Mary dropped the football.
25 Where was the football before the bathroom? office 23 17 5

Task 14
2 Julie went to the school this morning.
4 Yesterday Julie went to the office.
5 Where was Julie before the school?
Office 2 4

Task 16
6 Julius is a swan.
7 Julius is green.
9 Greg is a swan.
10 What color is Greg? Green 9 6 7

Task 19
2 The kitchen is north of the office.
4 The office is west of the garden.
6 How do you go from the kitchen to the garden? s,e 2 4

Memory Networks

Support several classes of tasks:

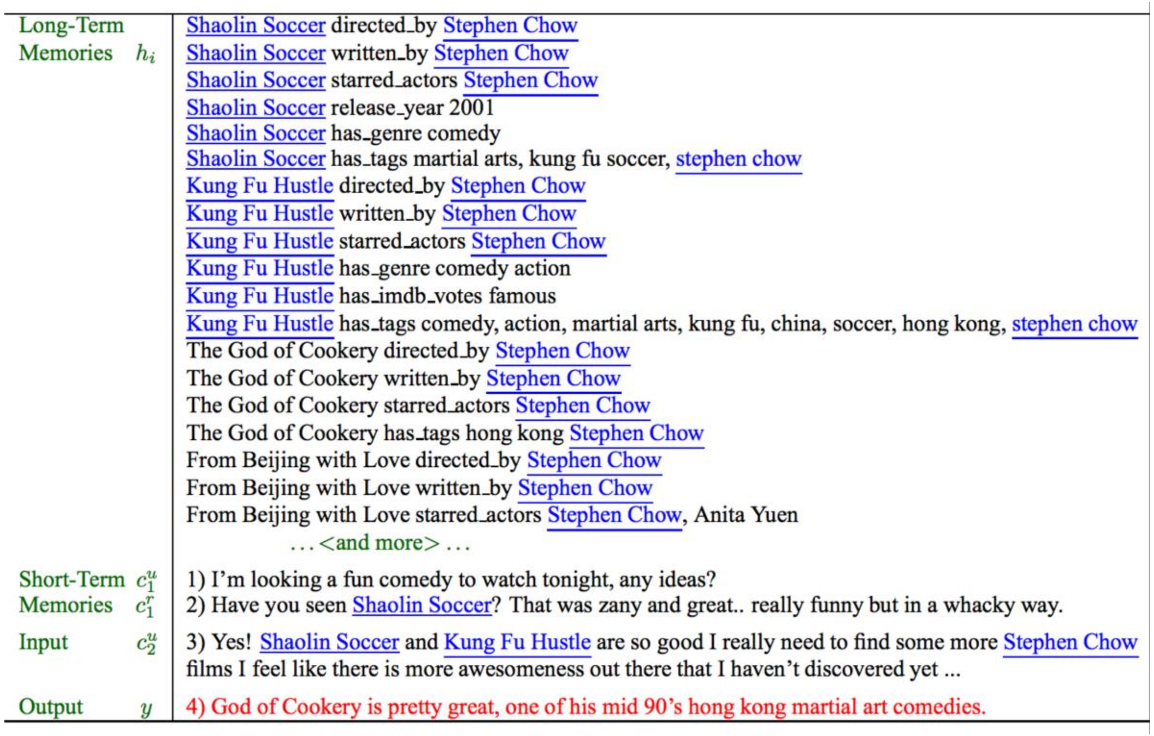
- Reading and Comprehension: Read a passage of text and answer questions about it.

- Dialog: To remember previous short- and long-term information during a conversation (what were we talking about?)

- Learning from Dialog: Learn new tasks from conversations with users

- Memory Networks support Reading with Attention over Memory (RAM).

Long- and Short-Term Memory



Long-Term Memories	h_i	Shaolin Soccer directed_by Stephen Chow
		Shaolin Soccer written_by Stephen Chow
		Shaolin Soccer starred_actors Stephen Chow
		Shaolin Soccer release_year 2001
		Shaolin Soccer has_genre comedy
		Shaolin Soccer has_tags martial arts, kung fu soccer, stephen chow
		Kung Fu Hustle directed_by Stephen Chow
		Kung Fu Hustle written_by Stephen Chow
		Kung Fu Hustle starred_actors Stephen Chow
		Kung Fu Hustle has_genre comedy action
		Kung Fu Hustle has_imdb_votes famous
		Kung Fu Hustle has_tags comedy, action, martial arts, kung fu, china, soccer, hong kong, stephen chow
		The God of Cookery directed_by Stephen Chow
		The God of Cookery written_by Stephen Chow
		The God of Cookery starred_actors Stephen Chow
		The God of Cookery has_tags hong kong Stephen Chow
		From Beijing with Love directed_by Stephen Chow
		From Beijing with Love written_by Stephen Chow
		From Beijing with Love starred_actors Stephen Chow, Anita Yuen
		. . . <and more> . . .
Short-Term Memories	c_1^u	1) I'm looking a fun comedy to watch tonight, any ideas?
	c_1^r	2) Have you seen Shaolin Soccer? That was zany and great.. really funny but in a whacky way.
Input	c_2^u	3) Yes! Shaolin Soccer and Kung Fu Hustle are so good I really need to find some more Stephen Chow films I feel like there is more awesomeness out there that I haven't discovered yet ...
Output	y	4) God of Cookery is pretty great, one of his mid 90's hong kong martial art comedies.

Note: Neural RAM long-term memory is typically "WORM" – Write Once, Read Many

Memory Network Framework

Four Components:

I: (input feature map) converts input data to internal feature representation.

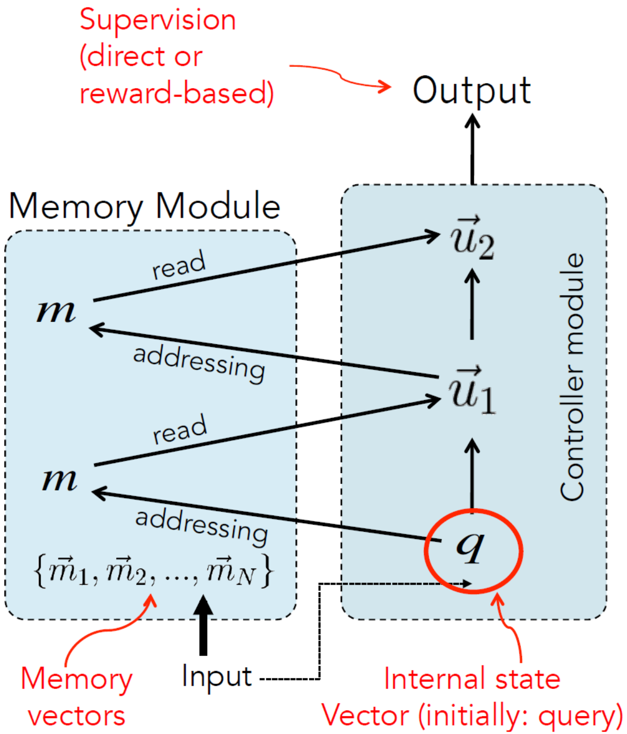
G: (generalization) update memories given new input.

O: produce new output (in feature representation space) given the memories.

R: (response) convert output O into a response seen by the outside world.

Igor borrowed from Jason Weston's 2016 ICML Tutorial

Memory Iteration



[Figure by Saina Sukhbaatar]

Q&A Memory Network

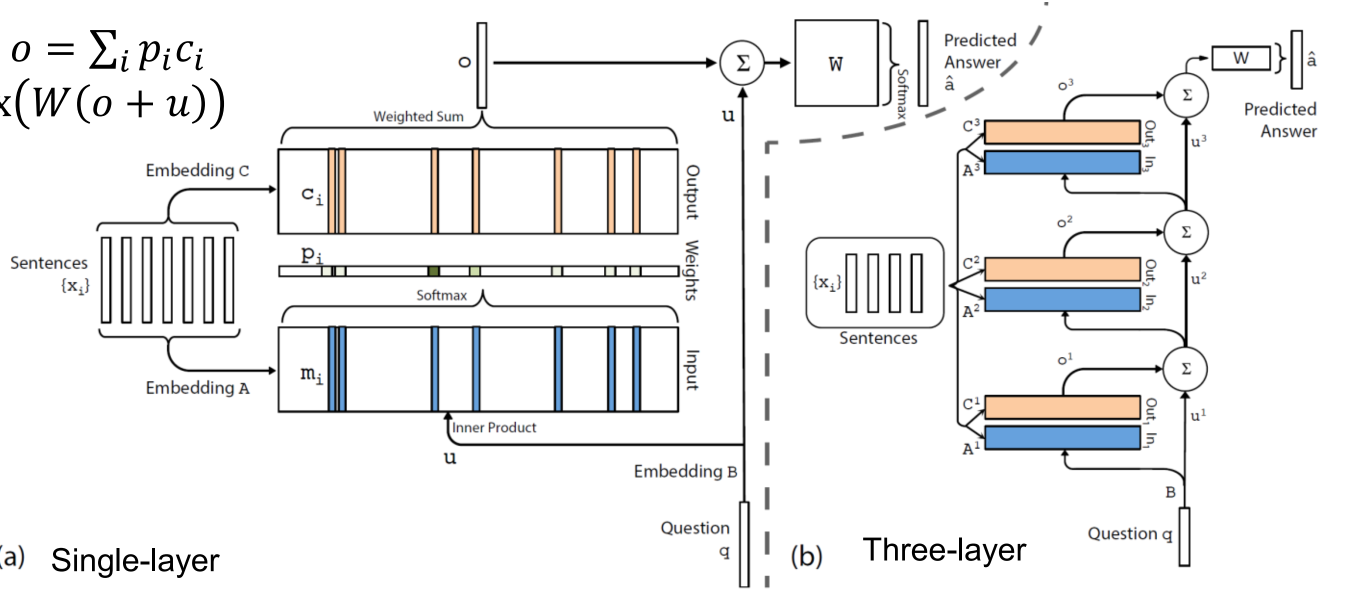
Sukhbaatar et al "End-to-End Memory Networks" 2015

Input data: $m_i = Ax_i,\ c_i = Cx_i$ a key-value store

Query embedding: $u = Bq$
Attention: $p_i = \mathrm{softmax}\left(u^T m_i\right)$

Output representation: $o = \sum_i p_i c_i$
Prediction: $\hat{a} = \mathrm{softmax}\left(W(o + u)\right)$

(no memory updating)



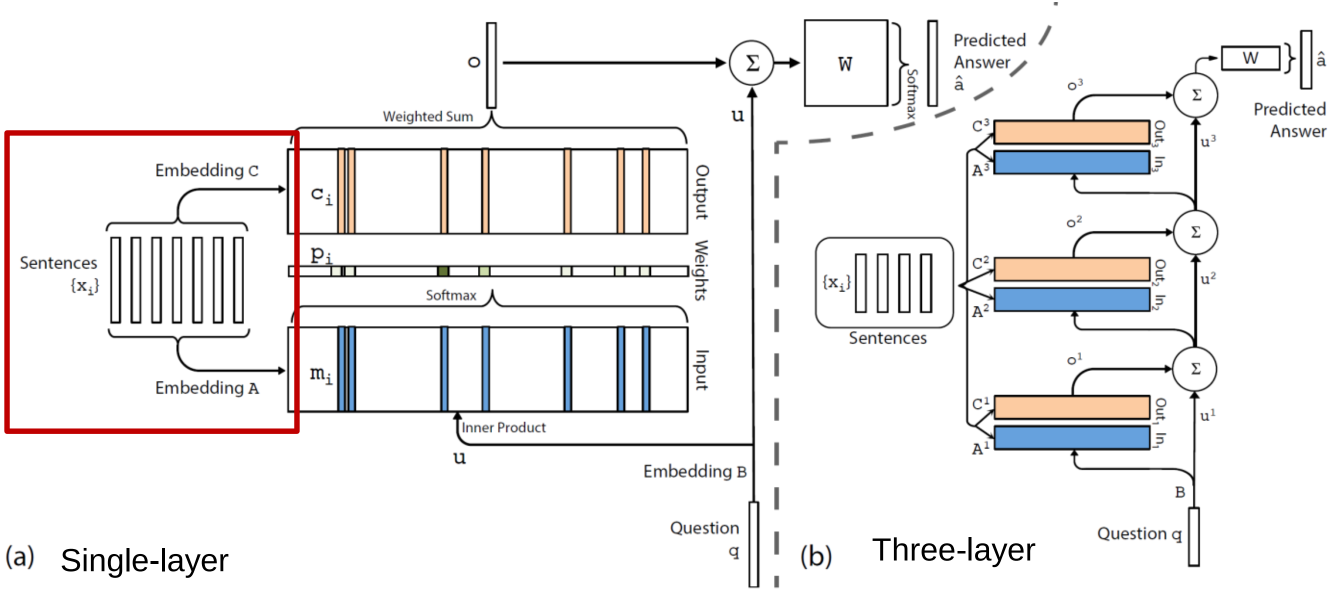
(a) Single-layer

(b) Three-layer

Q&A Memory Network

Sukbhaatar et al “End-to-End Memory Networks” 2015

Input data: $m_i = Ax_i, \, c_i = Cx_i$ a key-value store



(a) Single-layer

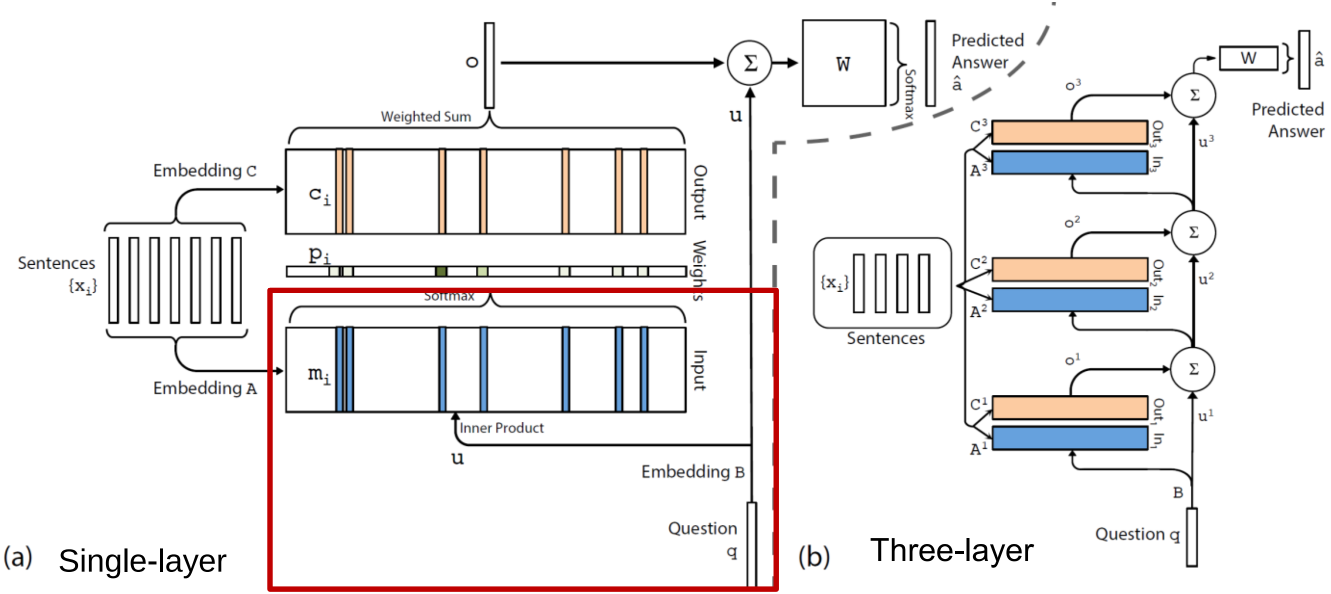
(b) Three-layer

Q&A Memory Network

Sukhbaatar et al “End-to-End Memory Networks” 2015

Query embedding: $u = Bq$

Attention: $p_i = \mathrm{softmax}(u^T m_i)$



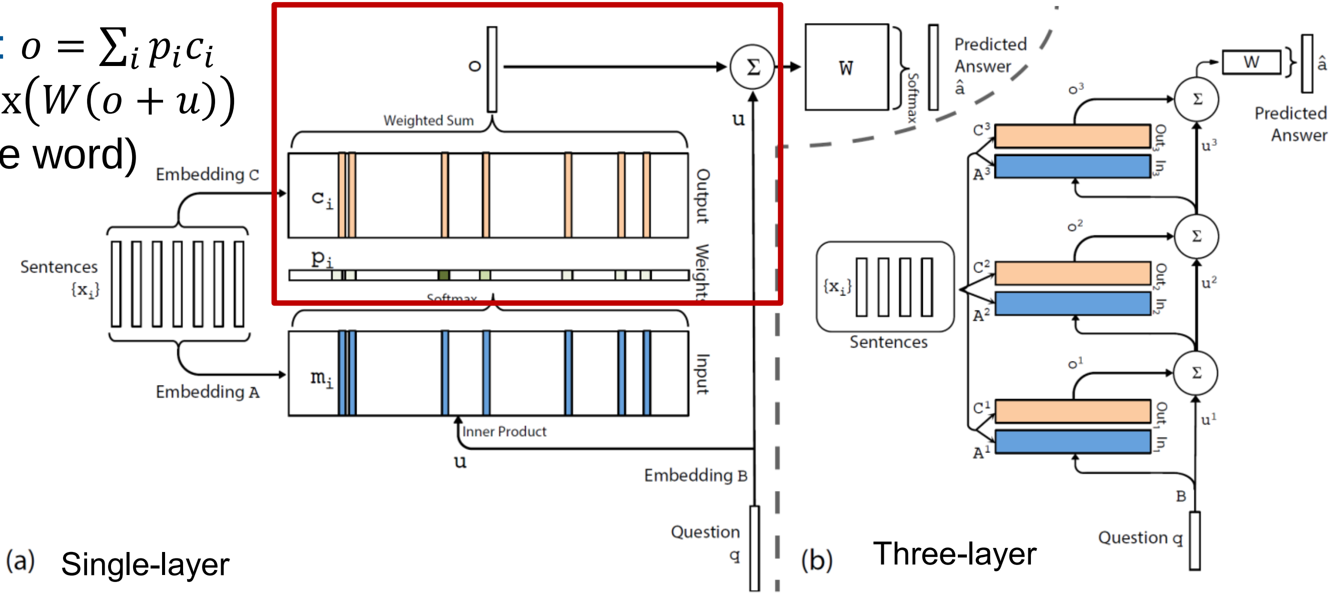
Q&A Memory Network

Sukbhaatar et al "End-to-End Memory Networks" 2015

Output representation: $o = \sum_i p_i c_i$

Prediction: $\hat{a} = \text{softmax}\big(W(o+u)\big)$

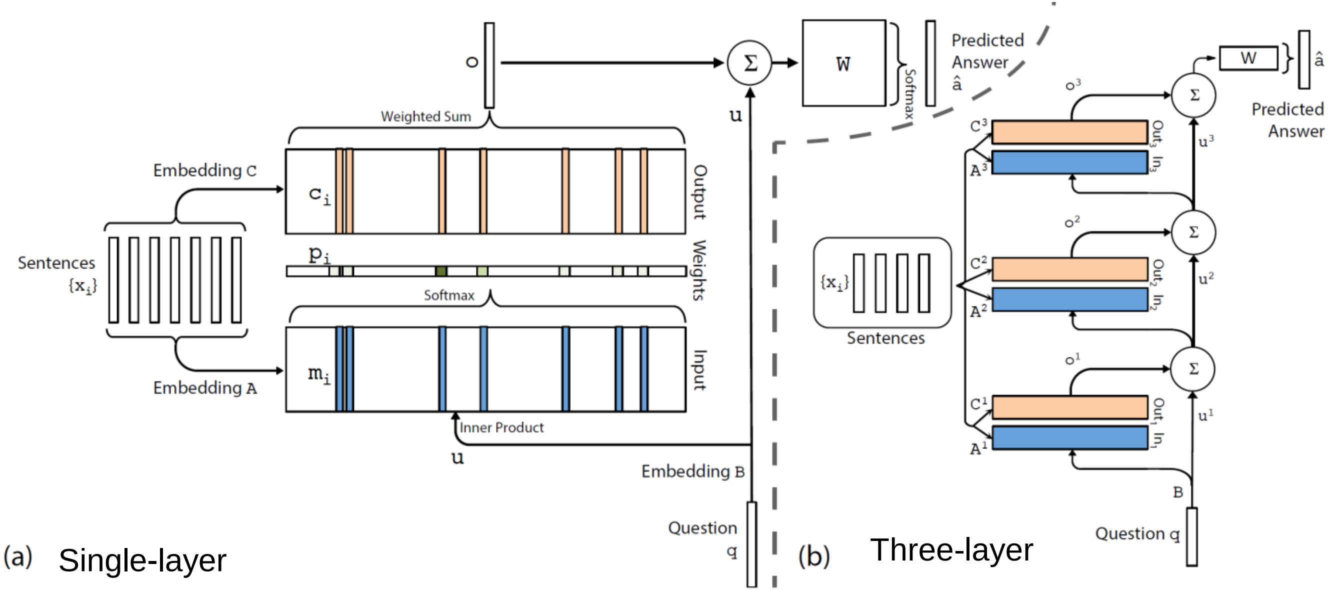
(The answer is a single word)



Q&A Memory Network

Sukbhaatar et al “End-to-End Memory Networks” 2015

Model is trained end-to-end on a series of Assertions and Questions, learns A, B, C and W.



(a) Single-layer

(b) Three-layer

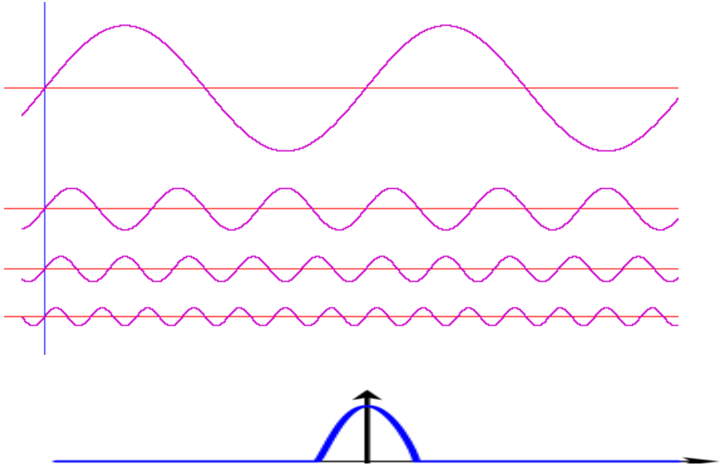
Aside: Position Encoding

Many models, including Transformer and memory networks, use position encoding so that embedded words carry information about their location in the input.

Memory networks multiply input words by a linear function of position.

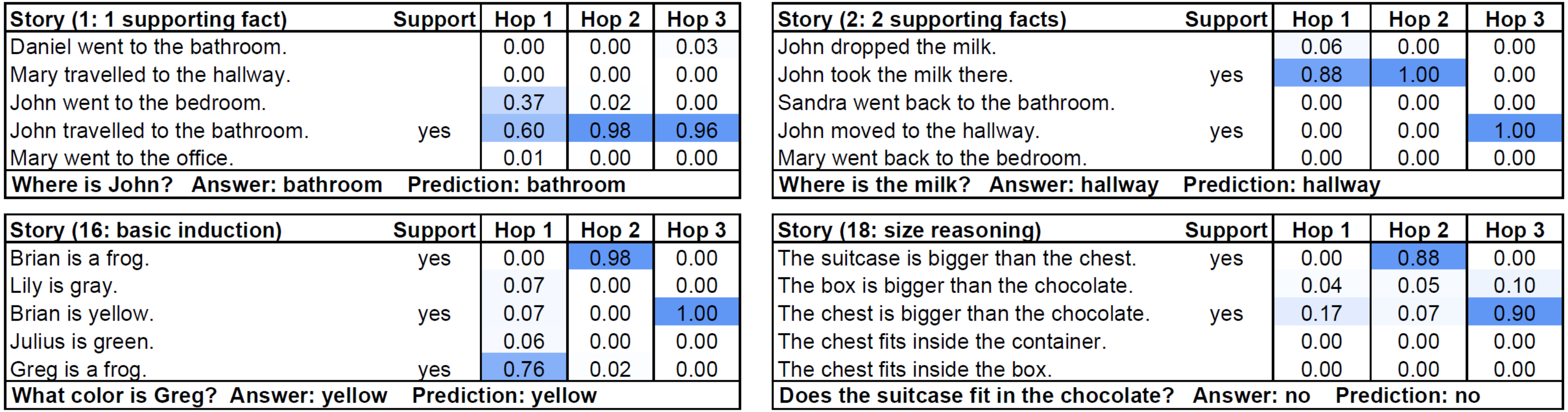
The Transformer uses a vector of sinusoids which is appended to the input vector.

The advantage of this representation is that the model can learn a linear combination of the sinusoids that is strongest at any particular word position, or a range of positions.



Multi-Hop Inference

Attention weights during various hops:



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 (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 (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

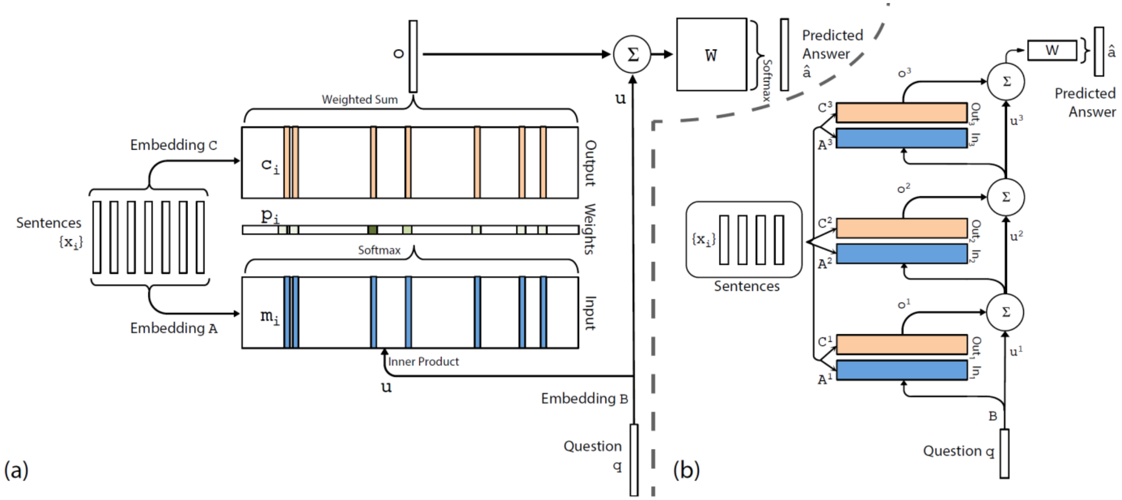
Story (18: size reasoning)	Support	Hop 1	Hop 2	Hop 3
The suitcase is bigger than the chest.	yes	0.00	0.88	0.00
The box is bigger than the chocolate.		0.04	0.05	0.10
The chest is bigger than the chocolate.	yes	0.17	0.07	0.90
The chest fits inside the container.		0.00	0.00	0.00
The chest fits inside the box.		0.00	0.00	0.00

Does the suitcase fit in the chocolate? Answer: no Prediction: no

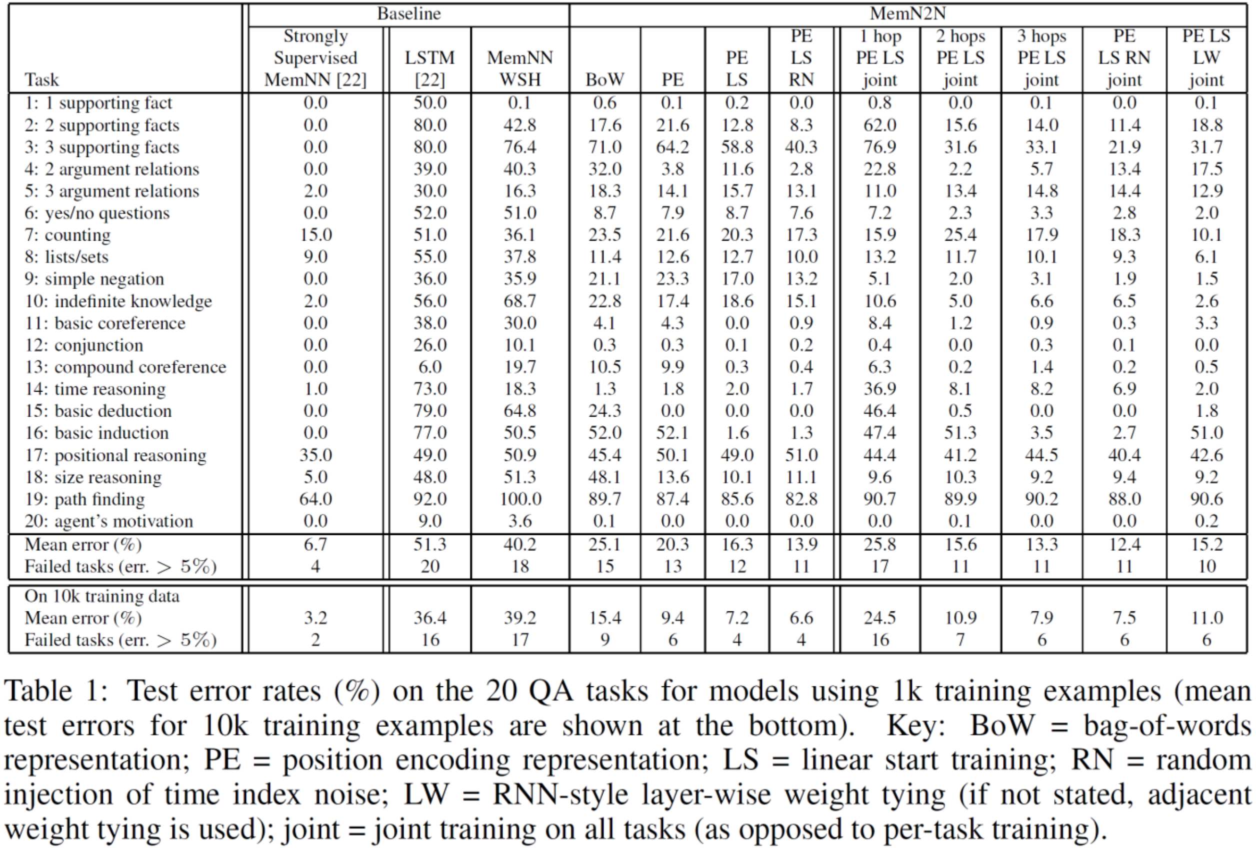
Note: Answers are single-word, predicted by the output softmax

Differences between Memory Nets and Xformer?

??



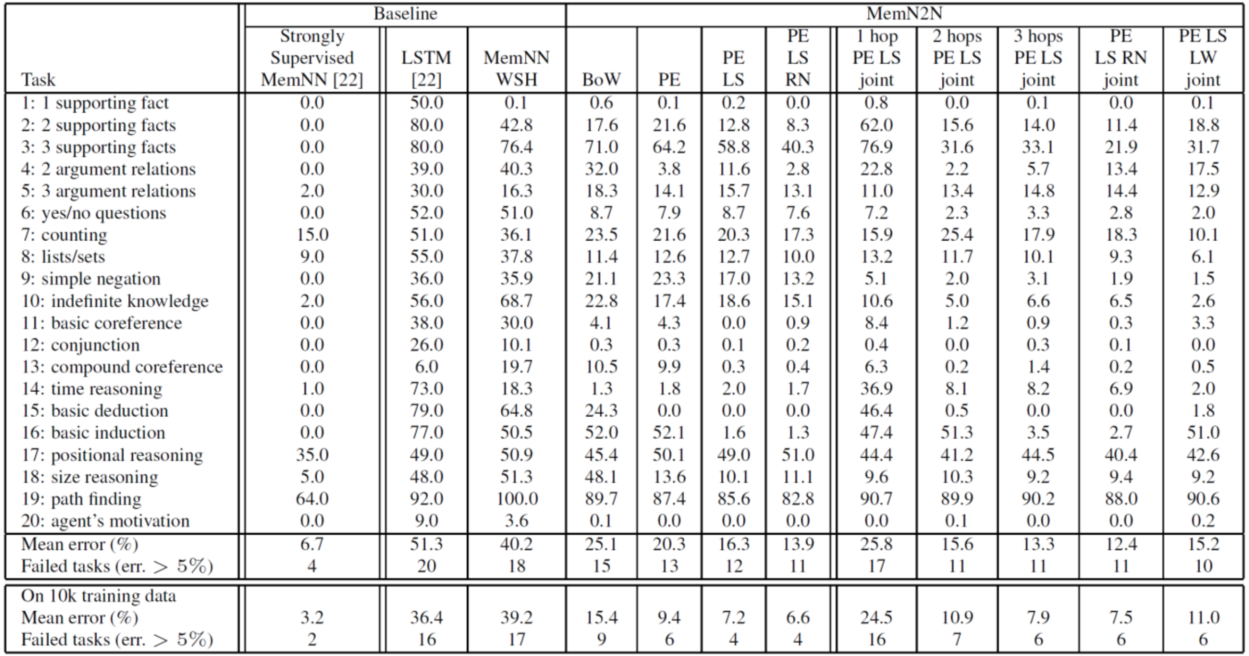
Performance on Q&A (babI) tasks



	Baseline			MemN2N								
Task	Strongly Supervised MemNN [22]	LSTM [22]	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).

Performance on Q&A (babI) tasks

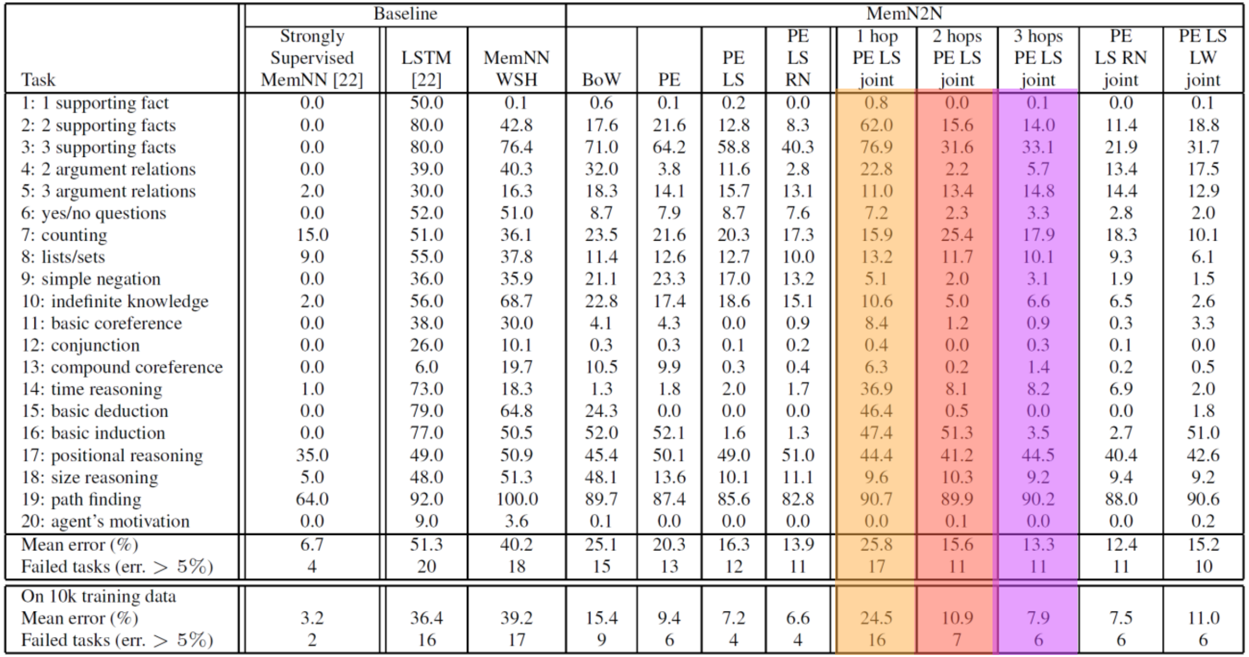


	Baseline			MemN2N								
Task	Strongly Supervised MemNN [22]	LSTM [22]	MemNN WSH	BoW	PE	PE LS	PE LS RN	1 hop PE LS joint	2 hops PE LS joint	3 hops PE LS joint	PE LS RN joint	PE LS LW joint
1: 1 supporting fact	0.0	50.0	0.1	0.6	0.1	0.2	0.0	0.8	0.0	0.1	0.0	0.1
2: 2 supporting facts	0.0	80.0	42.8	17.6	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

Per-Task Training

Trained on All Tasks

Performance Improves with Number of Hops



	Baseline			MemN2N									
Task	Strongly Supervised MemNN [22]	LSTM [22]	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).

Memory Networks and Natural Language

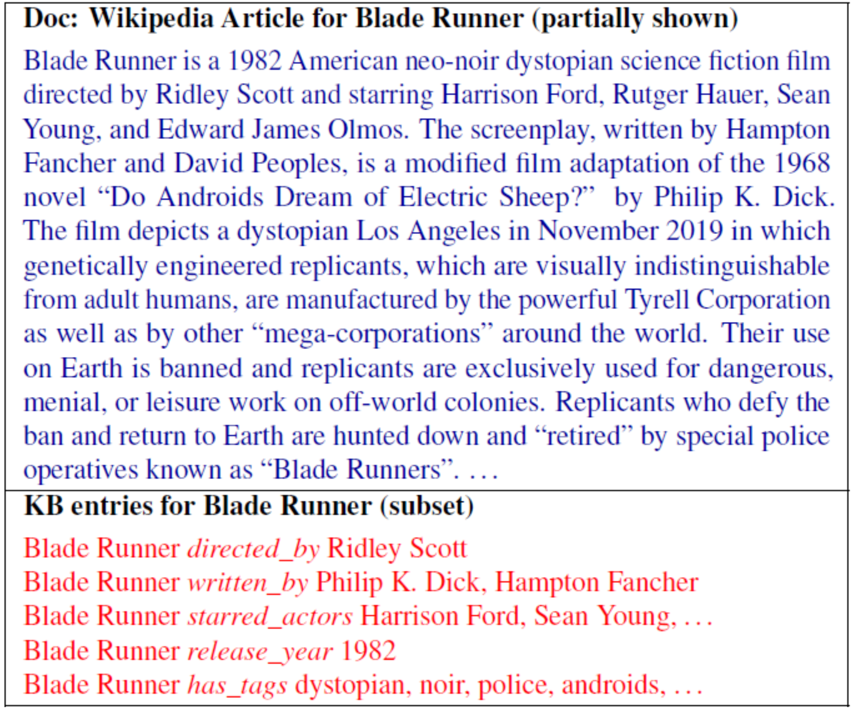
The memory network considered so far is designed to work with structured text documents (a knowledge base or KB).

It can be extended to deal with Natural language text, and performance on the two types of data source can be compared.

The domain is movie knowledge.

Miller et al. “Key-Value Memory Networks for Directly Reading Documents” 2016

Memory Networks and Natural Language



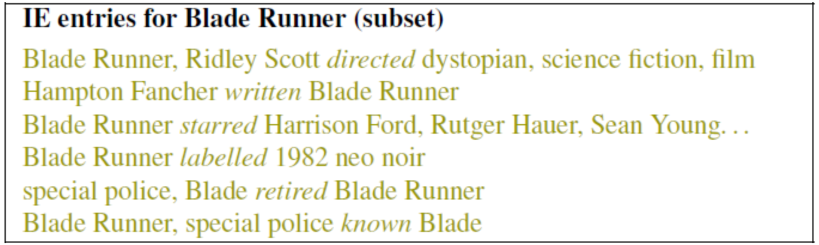
Doc: Wikipedia Article for Blade Runner (partially shown)

Blade Runner is a 1982 American neo-noir dystopian science fiction film directed by Ridley Scott and starring Harrison Ford, Rutger Hauer, Sean Young, and Edward James Olmos. The screenplay, written by Hampton Fancher and David Peoples, is a modified film adaptation of the 1968 novel “Do Androids Dream of Electric Sheep?” by Philip K. Dick. The film depicts a dystopian Los Angeles in November 2019 in which genetically engineered replicants, which are visually indistinguishable from adult humans, are manufactured by the powerful Tyrell Corporation as well as by other “mega-corporations” around the world. Their use on Earth is banned and replicants are exclusively used for dangerous, menial, or leisure work on off-world colonies. Replicants who defy the ban and return to Earth are hunted down and “retired” by special police operatives known as “Blade Runners”. . . .

KB entries for Blade Runner (subset)

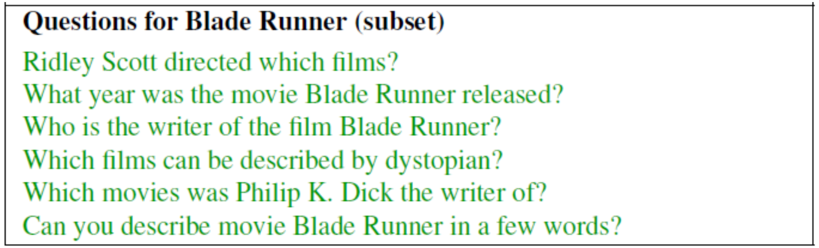
Blade Runner *directed_by* Ridley Scott
Blade Runner *written_by* Philip K. Dick, Hampton Fancher
Blade Runner *starred_actors* Harrison Ford, Sean Young, . . .
Blade Runner *release_year* 1982
Blade Runner *has_tags* dystopian, noir, police, androids, . . .

After running an IE (Information Extraction) pipeline on the full text we build this table:



IE entries for Blade Runner (subset)

Blade Runner, Ridley Scott *directed* dystopian, science fiction, film
Hampton Fancher *written* Blade Runner
Blade Runner *starred* Harrison Ford, Rutger Hauer, Sean Young. . .
Blade Runner *labelled* 1982 neo noir
special police, Blade *retired* Blade Runner
Blade Runner, special police *known* Blade



Questions for Blade Runner (subset)

Ridley Scott directed which films?
What year was the movie Blade Runner released?
Who is the writer of the film Blade Runner?
Which films can be described by dystopian?
Which movies was Philip K. Dick the writer of?
Can you describe movie Blade Runner in a few words?

Miller et al. “Key-Value Memory Networks for Directly Reading Documents” 2016

Memory Network Key-Value Store

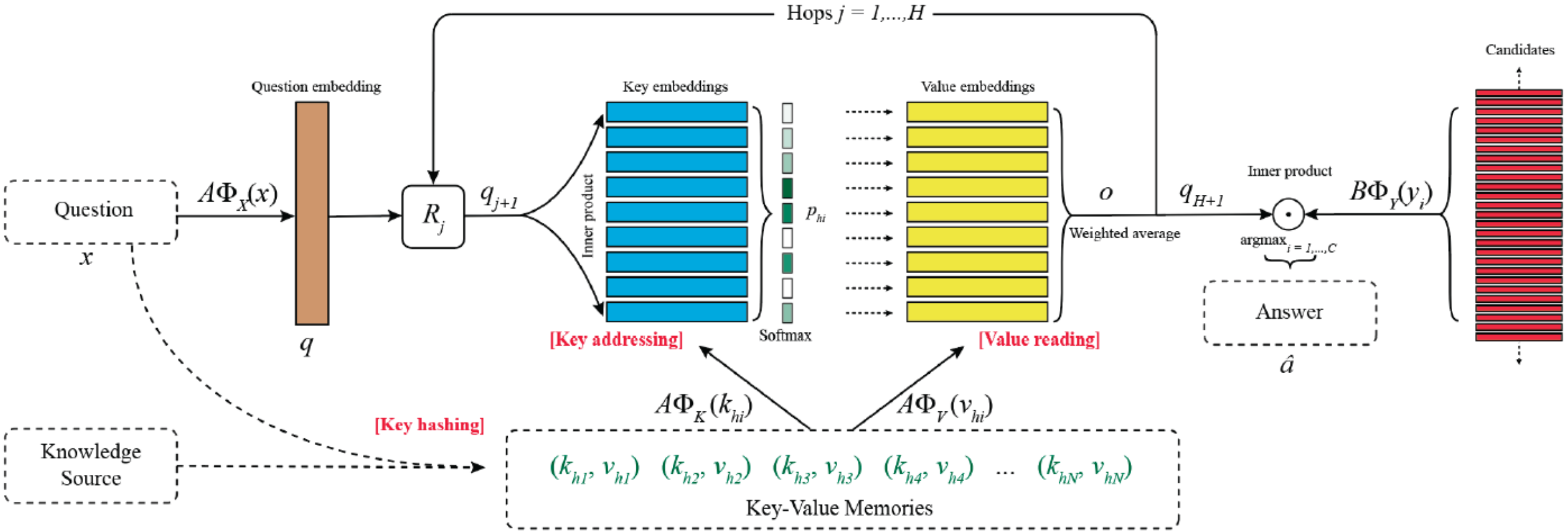
Input feature maps: $\Phi_{\mathrm{K}}(k_i)$ and $\Phi_{\mathrm{V}}(v_i)$ - key, value embeddings (k_i, v_i different this time)

Query: x

Attention: $p_i = \mathrm{softmax}(A\Phi_X(x) \cdot A\Phi_K(k_{h_i}))$

Output representation: $o = \sum_i p_{h_i} A\Phi\_V\left(v_{h_i}\right)$

Prediction: $\hat{a} = \mathrm{argmax}_{i=1,\ldots,C} \mathrm{Softmax}(q_{H+1}^\top B\Phi_Y(y_i))$



Memory Networks: Using the Key-Value Store

Sentence-Level Encoding: Free-text input is broken into sentences, and each is encoded in BoW as key and value – equivalent to standard MemNN.

Window-Level: Encode a window of W words in BOW as the key. Use the center word as the value.

KB-Triple: Typically have the form "subject relation object," key is subject-relation pair, value is the object.
For better retrieval relations are typically encoded twice, e.g.:

Blade Runner directed_by Ridley Scott
Ridley Scott !directed_by Blade Runner

Question Standardization

Original natural language questions were standardized using the SimpleQuestions dataset.

“What movies did Harrison Ford star in?”
➔
Instance of the pattern “What movies did [@actor] star in?”

Created 100k training pairs.

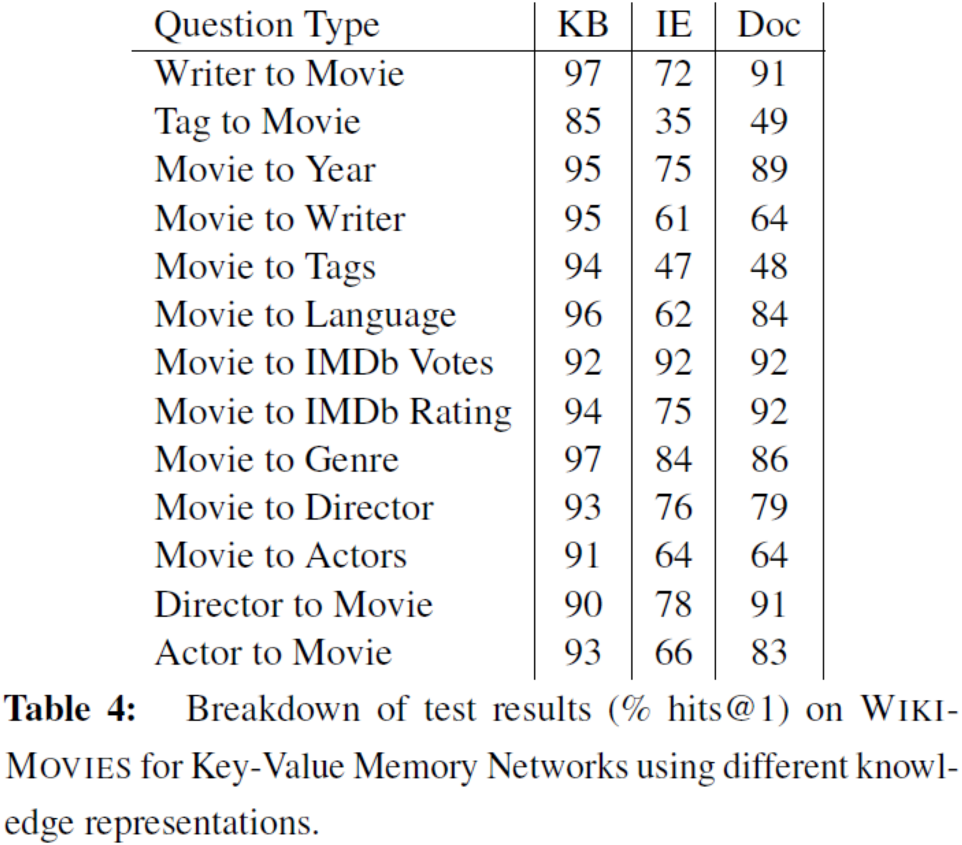
Scaling Up

Its impractical to test queries against the entire database.

Instead the query text is used to perform full-text search across the database.

Only document that are similar enough to the query (e.g. contain at least one query word) are actually considered.

Results!



Question Type	KB	IE	Doc
Writer to Movie	97	72	91
Tag to Movie	85	35	49
Movie to Year	95	75	89
Movie to Writer	95	61	64
Movie to Tags	94	47	48
Movie to Language	96	62	84
Movie to IMDb Votes	92	92	92
Movie to IMDb Rating	94	75	92
Movie to Genre	97	84	86
Movie to Director	93	76	79
Movie to Actors	91	64	64
Director to Movie	90	78	91
Actor to Movie	93	66	83

Table 4: Breakdown of test results (% hits@1) on WIKIMOVIES for Key-Value Memory Networks using different knowledge representations.

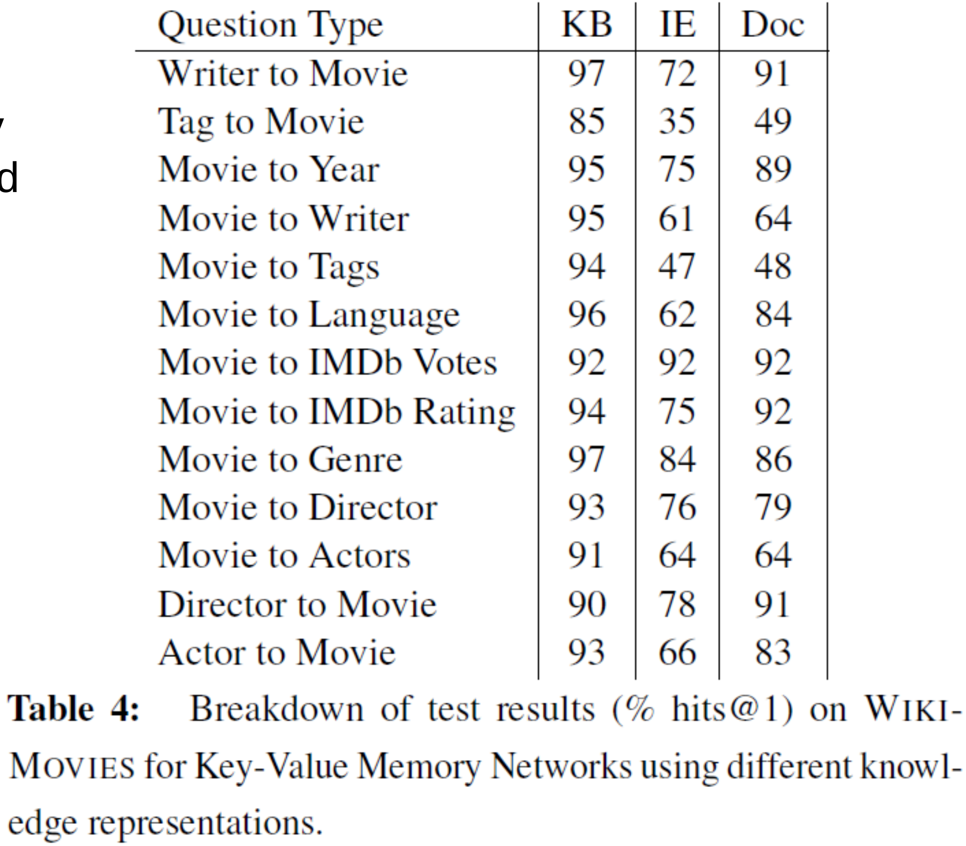
Results!

Reading raw docs (Doc column) usually does much better than the doc-extracted KB (IE column).

Structed KBs (KB column) often better than the document generated answer.

The experiment was done on a subset of questions that were in the KB. Many possible questions are not.

Moral: Use KB when possible, fall back on free text.



Question Type	KB	IE	Doc
Writer to Movie	97	72	91
Tag to Movie	85	35	49
Movie to Year	95	75	89
Movie to Writer	95	61	64
Movie to Tags	94	47	48
Movie to Language	96	62	84
Movie to IMDb Votes	92	92	92
Movie to IMDb Rating	94	75	92
Movie to Genre	97	84	86
Movie to Director	93	76	79
Movie to Actors	91	64	64
Director to Movie	90	78	91
Actor to Movie	93	66	83

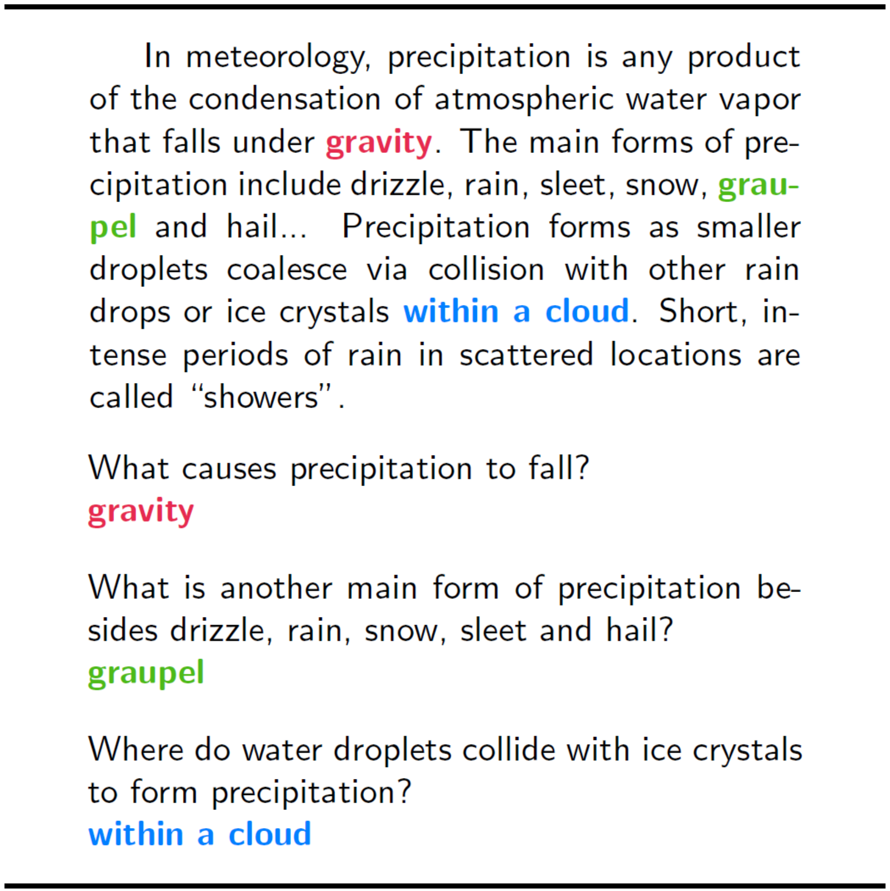
Table 4: Breakdown of test results (% hits@1) on WIKI-MOVIES for Key-Value Memory Networks using different knowledge representations.

SQuAD 1.0 – Stanford Question Answering Dataset

100,000+ questions posed by crowd workers on a set of Wikipedia articles.

The answer is a segment of text from the article:

This is an example of ***extractive*** question answering.



In meteorology, precipitation is any product of the condensation of atmospheric water vapor that falls under **gravity**. The main forms of precipitation include drizzle, rain, sleet, snow, **graupel** and hail... Precipitation forms as smaller droplets coalesce via collision with other rain drops or ice crystals **within a cloud**. Short, intense periods of rain in scattered locations are called "showers".

What causes precipitation to fall?
gravity

What is another main form of precipitation besides drizzle, rain, snow, sleet and hail?
graupel

Where do water droplets collide with ice crystals to form precipitation?
within a cloud

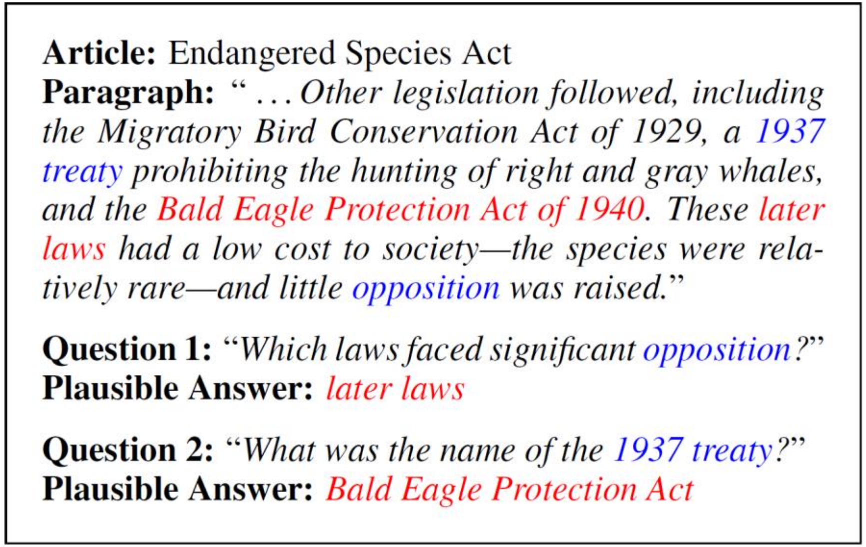
SQuAD 2.0 – SQuAD 1.0 + Adversarial Examples

On SQuAD 1, rather than really finding an answer, systems could cheat by providing an answer of appropriate type.

SQuAD 2.0 Includes the original SQuAD 1 dataset plus crowdsourced adversarial examples of unanswerable questions.

SQuAD 2.0 prevents “educated guesses” on SQuAD 1 questions such as those at the right.

SQuAD 2.0 requires systems to check that the passage entails the answer, rather than the answer only being relevant to the query.



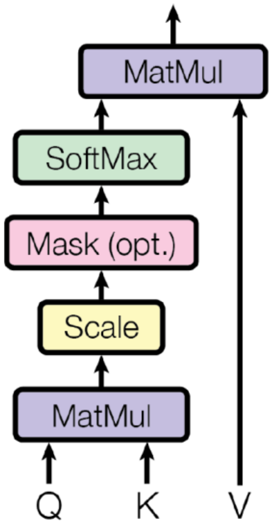
Article: Endangered Species Act
Paragraph: “ . . . *Other legislation followed, including the Migratory Bird Conservation Act of 1929, a 1937 treaty prohibiting the hunting of right and gray whales, and the Bald Eagle Protection Act of 1940. These later laws had a low cost to society—the species were relatively rare—and little opposition was raised.*”

Question 1: “*Which laws faced significant opposition?*”
Plausible Answer: *later laws*

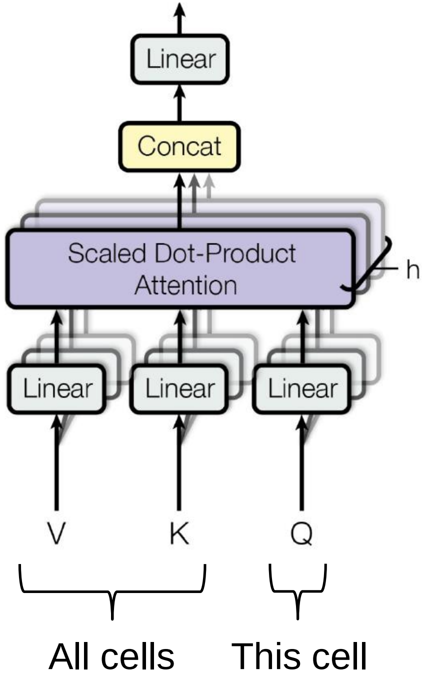
Question 2: “*What was the name of the 1937 treaty?*”
Plausible Answer: *Bald Eagle Protection Act*

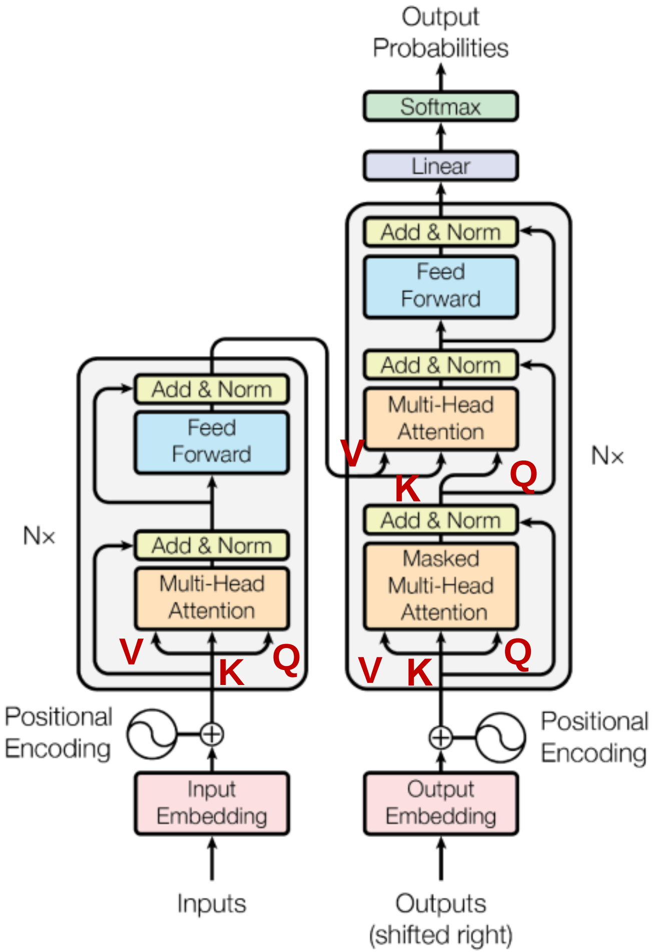
Review: The Transformer

Scaled Dot-Product Attention



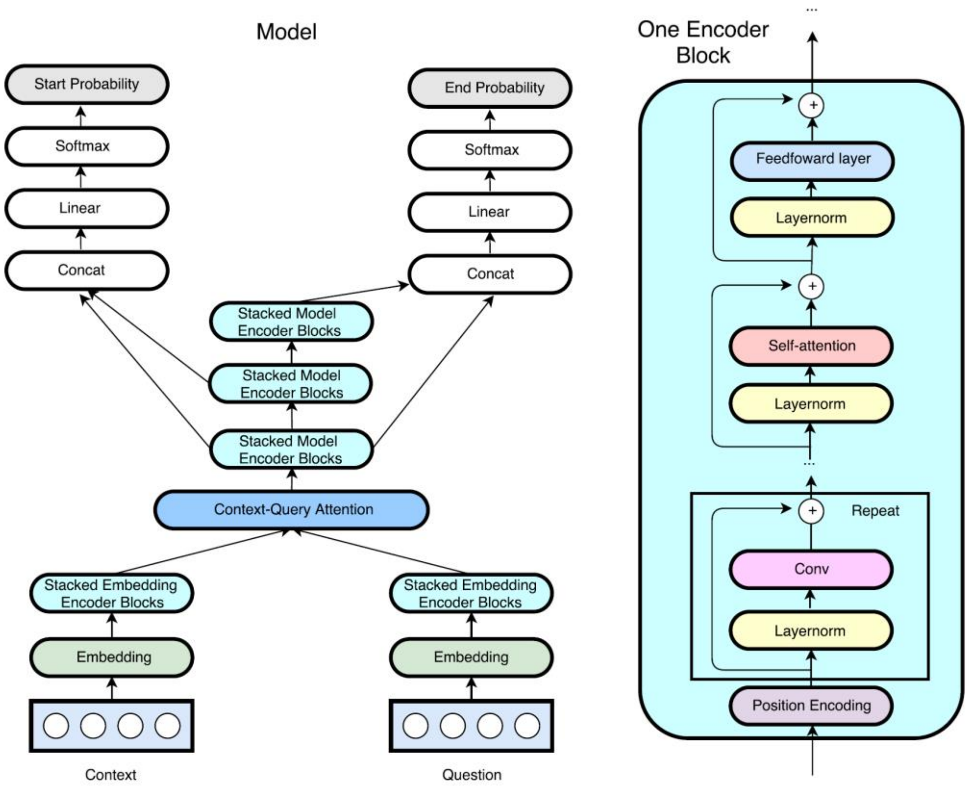
Multi-headed attention





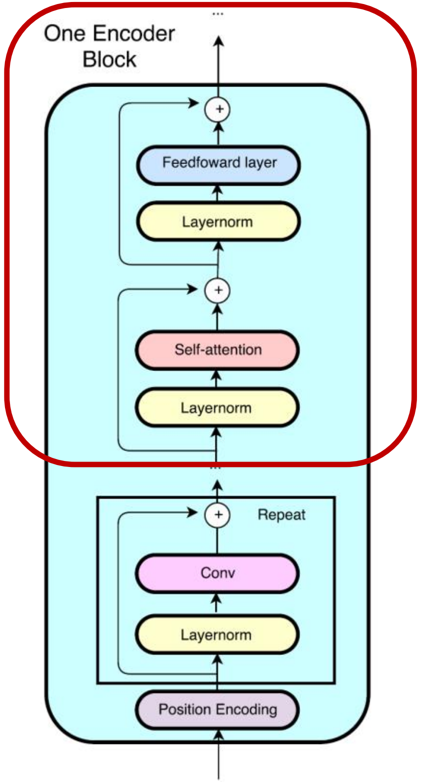
QANet – Convolution + Attention for Q&A.

Convolutional Input + Transformer



QANet – Convolution + Attention for Q&A.

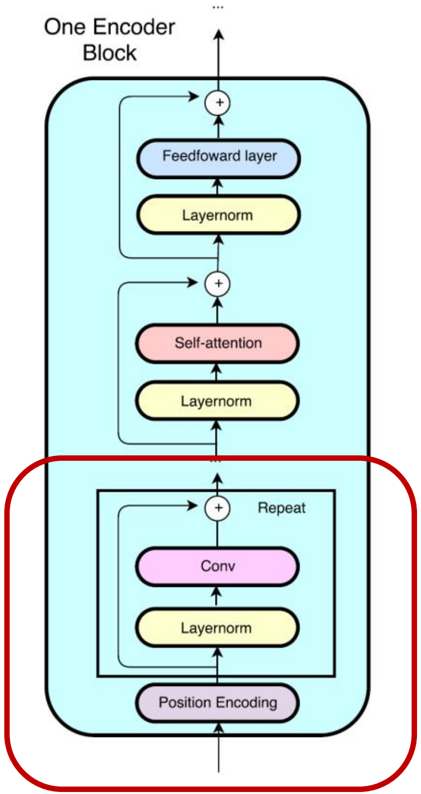
Convolutional Input + Transformer



Transformer Encoder Block
(modified normalization)

QANet – Convolution + Attention for Q&A.

Convolutional Input + Transformer



Convolutional Layers (1D):

This stack of layers is almost linear: there are no activations between convolutions.

Aside: Layer Norm

Like batch norm, tries to reduce covariate shift by scaling and biasing activations.

But statistics are computed across all units with each layer:

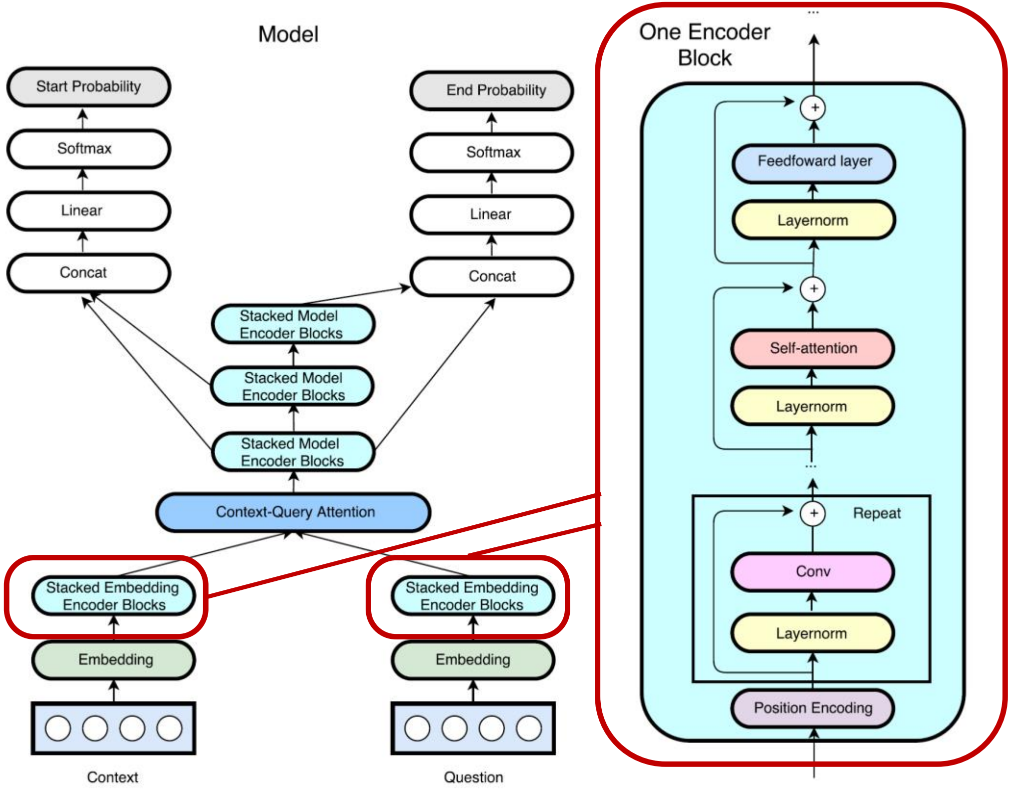
$$\mu^l = \frac{1}{H}\sum_{i=1}^{H} a_i^l \qquad \sigma^l = \sqrt{\frac{1}{H}\sum_{i=1}^{H}\left(a_i^l - \mu^l\right)^2}$$

The activation for the i^{th} neuron in layer l is then

$$\bar{a}_i^l = \frac{1}{\sigma^l}(a_i^l - \mu^l)$$

QANet – Convolution + Attention for Q&A.

Convolutional Input + Transformer



4x convolution layers, kernel=7

QANet – Context-Query Attention Block

The Context-Query block is similar to many other Q&A systems, query q, context c:

Similarity Function: $f(q, c) = W(q, c, q \odot c)$

where $q \odot c$ is element-wise product of q and c, W is a weight matrix.

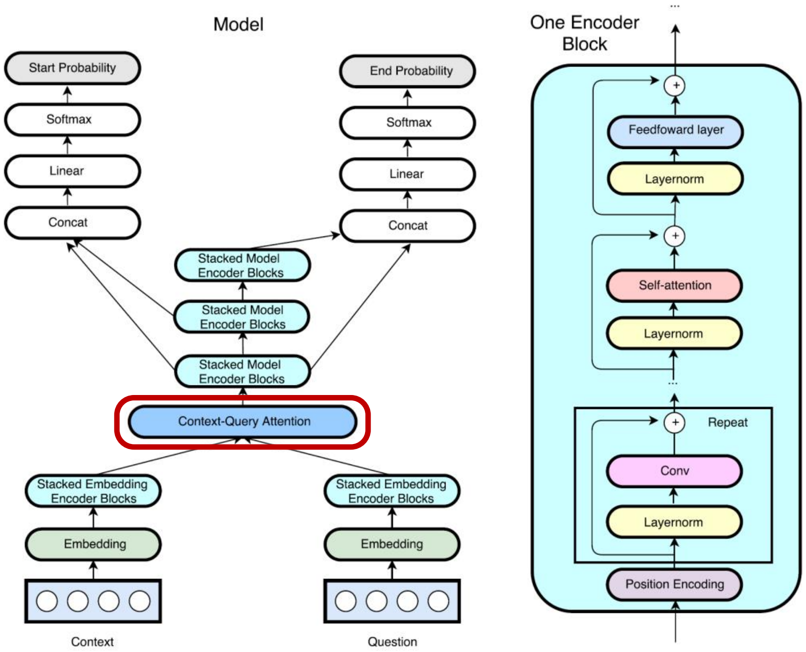
which defines a similarity matrix S where

$$S_{ij} = f(q_j, c_i)$$

then apply a row-wise softmax (over queries) σ giving $\bar{S}$

$$\bar{S} = \sigma(S)$$

The Context-Query attention is $\boxed{A = \bar{S}Q^T}$



QANet – Query-Context Attention Block

The Context-Query block is similar to many other Q&A systems, query q, context c:

Similarity Function: $f(q, c) = W(q, c, q \odot c)$

where $q \odot c$ is element-wise product of q and c, W is a weight matrix.

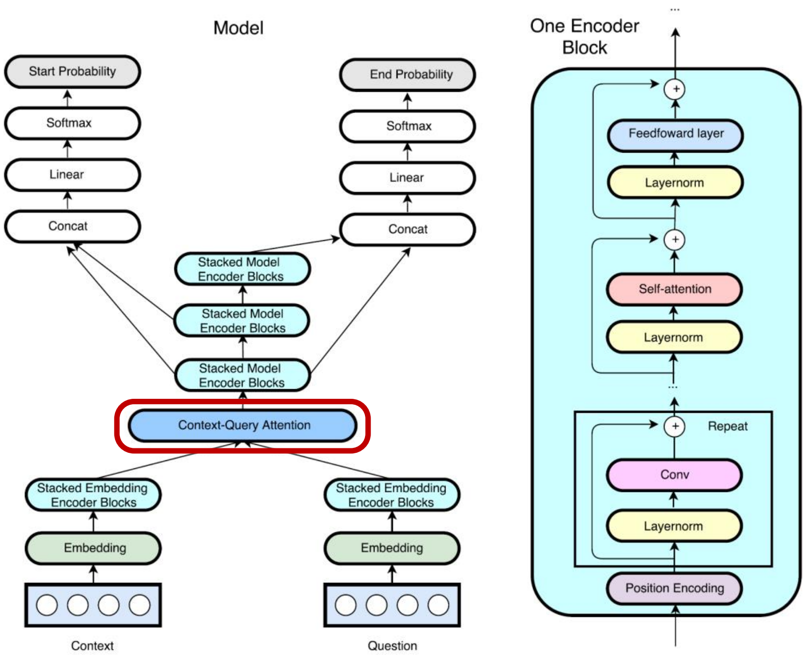
which defines a similarity matrix S where

$$S_{ij} = f(q_j, c_i)$$

then apply a column-wise softmax (over contexts) σ' giving $\bar{\bar{S}}$ where

$$\bar{\bar{S}} = \sigma'(S)$$

The Query-Context attention is $\boxed{B = \bar{S}\bar{\bar{S}}C^T}$



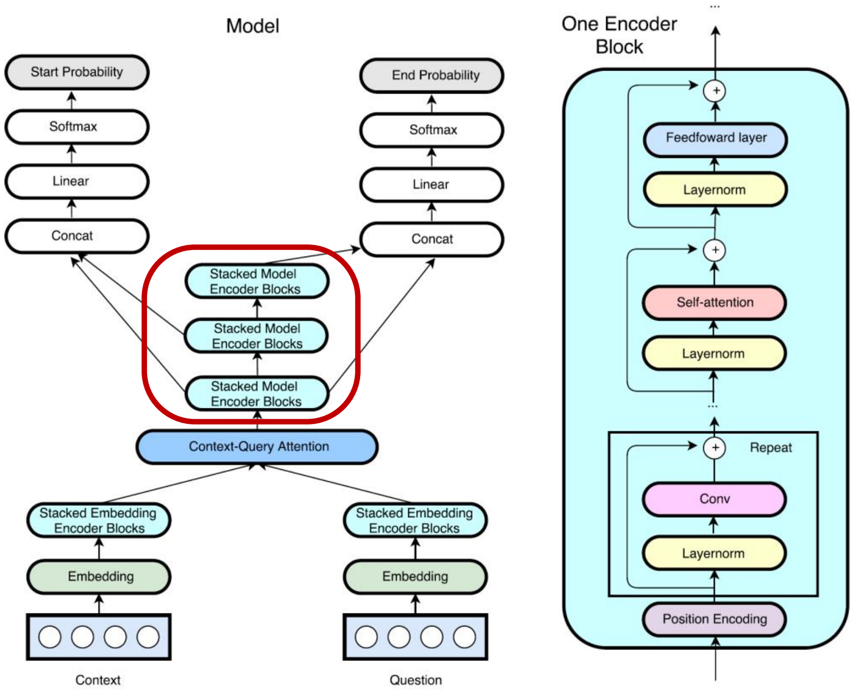
QANet – Model Encoder Block

The Model encoder blocks take as input:

$$[c, a, c \odot a, c \odot b]$$

where a and b are respectively rows of attention matrices A and B.

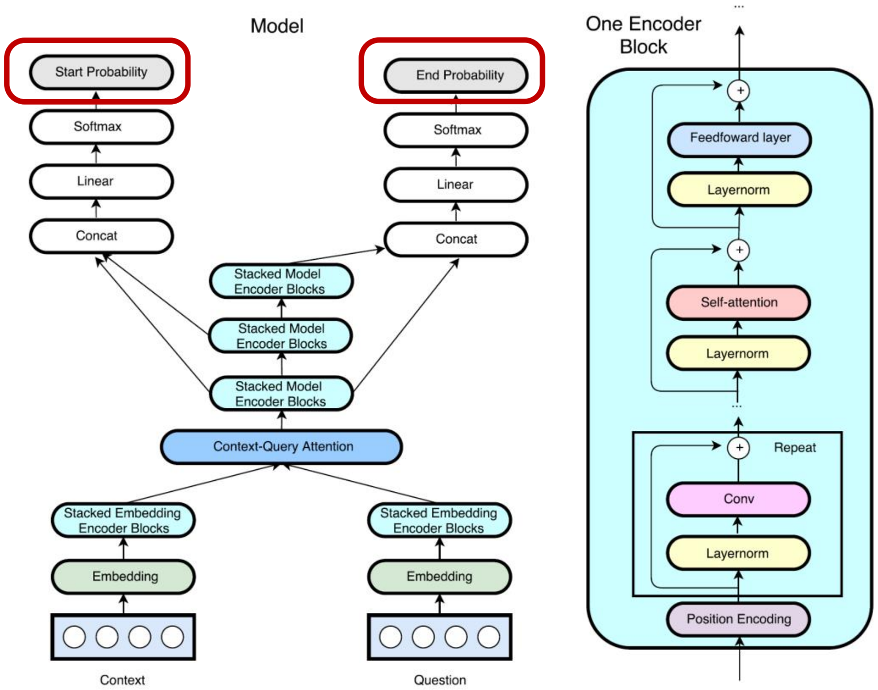
Inside they are the same as the encoder blocks, except that they are more deeply stacked (8 blocks vs. 1 block).



QANet – Predictions

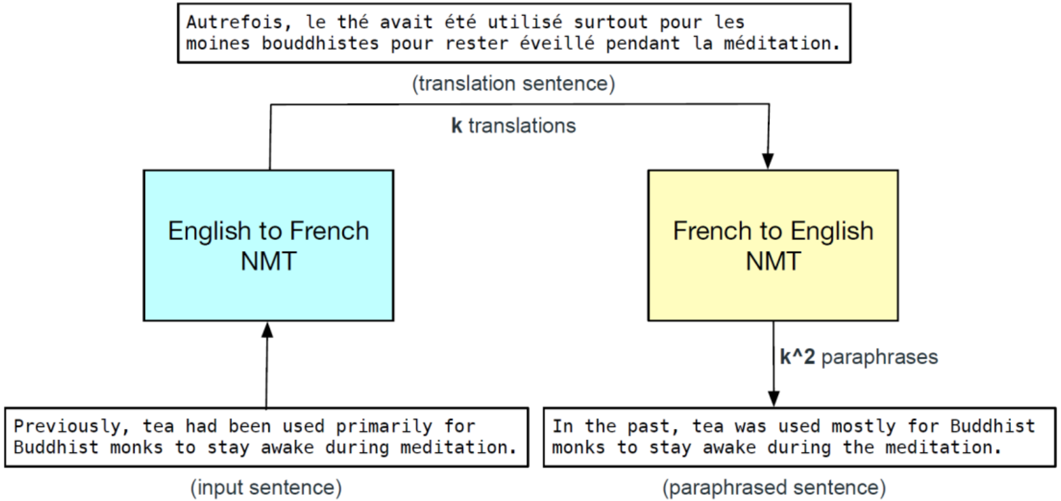
Remember that SQuAD is an extractive Q&A dataset, i.e. the answer is a span of text from the context data.

QANet outputs a softmax over word positions for both the start and end of the answer span.



QANet – Backtranslation

QANet’s design is non-recurrent, so training time does not grow with input length.
Since it is considerably faster to train than recurrent designs, it can be trained on larger datasets in the same time.
Backtranslation provides more diversity in the training data:



Autrefois, le thé avait été utilisé surtout pour les moines bouddhistes pour rester éveillé pendant la méditation.

(translation sentence)

k translations

English to French NMT

French to English NMT

k^2 paraphrases

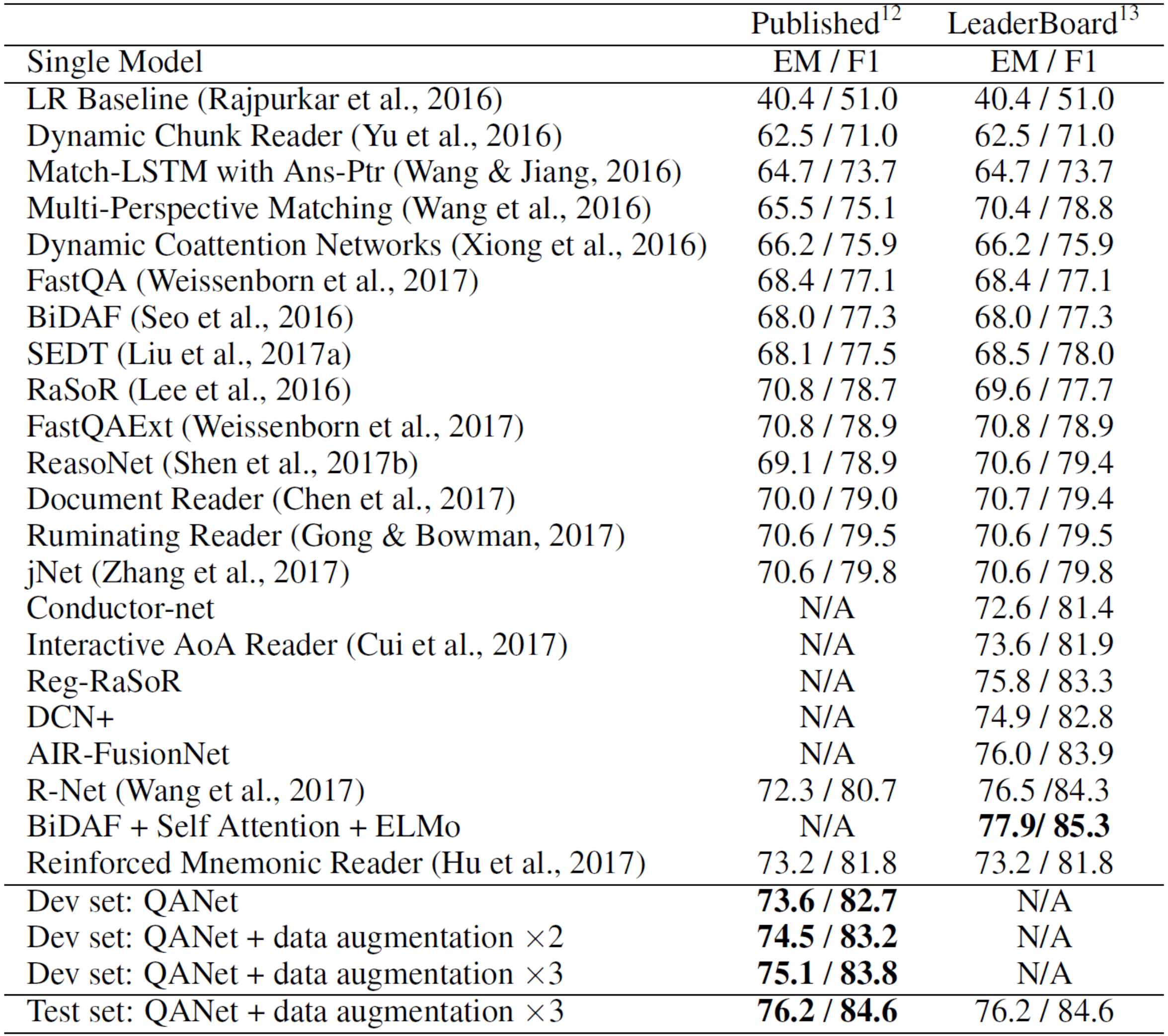
Previously, tea had been used primarily for Buddhist monks to stay awake during meditation.

(input sentence)

In the past, tea was used mostly for Buddhist monks to stay awake during the meditation.

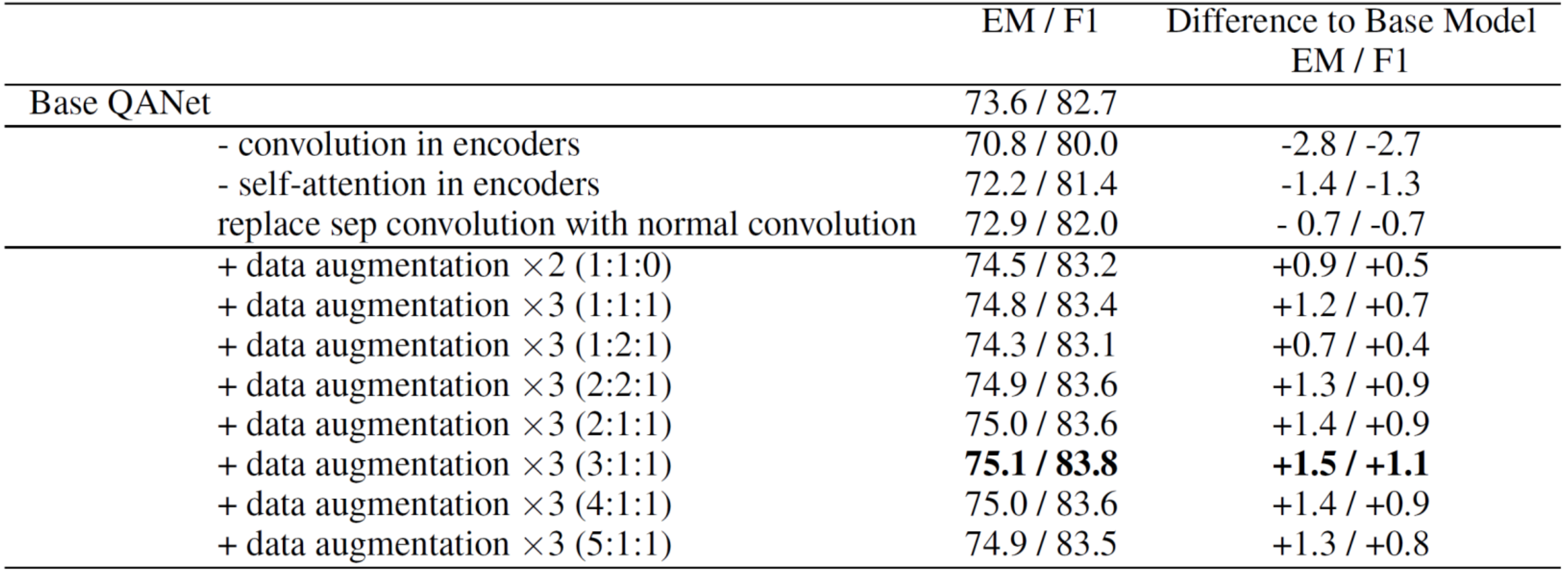
(paraphrased sentence)

QANet – Performance



	Published[12]	LeaderBoard[13]
Single Model	EM / F1	EM / F1
LR Baseline (Rajpurkar et al., 2016)	40.4 / 51.0	40.4 / 51.0
Dynamic Chunk Reader (Yu et al., 2016)	62.5 / 71.0	62.5 / 71.0
Match-LSTM with Ans-Ptr (Wang & Jiang, 2016)	64.7 / 73.7	64.7 / 73.7
Multi-Perspective Matching (Wang et al., 2016)	65.5 / 75.1	70.4 / 78.8
Dynamic Coattention Networks (Xiong et al., 2016)	66.2 / 75.9	66.2 / 75.9
FastQA (Weissenborn et al., 2017)	68.4 / 77.1	68.4 / 77.1
BiDAF (Seo et al., 2016)	68.0 / 77.3	68.0 / 77.3
SEDT (Liu et al., 2017a)	68.1 / 77.5	68.5 / 78.0
RaSoR (Lee et al., 2016)	70.8 / 78.7	69.6 / 77.7
FastQAExt (Weissenborn et al., 2017)	70.8 / 78.9	70.8 / 78.9
ReasoNet (Shen et al., 2017b)	69.1 / 78.9	70.6 / 79.4
Document Reader (Chen et al., 2017)	70.0 / 79.0	70.7 / 79.4
Ruminating Reader (Gong & Bowman, 2017)	70.6 / 79.5	70.6 / 79.5
jNet (Zhang et al., 2017)	70.6 / 79.8	70.6 / 79.8
Conductor-net	N/A	72.6 / 81.4
Interactive AoA Reader (Cui et al., 2017)	N/A	73.6 / 81.9
Reg-RaSoR	N/A	75.8 / 83.3
DCN+	N/A	74.9 / 82.8
AIR-FusionNet	N/A	76.0 / 83.9
R-Net (Wang et al., 2017)	72.3 / 80.7	76.5 /84.3
BiDAF + Self Attention + ELMo	N/A	**77.9/ 85.3**
Reinforced Mnemonic Reader (Hu et al., 2017)	73.2 / 81.8	73.2 / 81.8
Dev set: QANet	**73.6 / 82.7**	N/A
Dev set: QANet + data augmentation ×2	**74.5 / 83.2**	N/A
Dev set: QANet + data augmentation ×3	**75.1 / 83.8**	N/A
Test set: QANet + data augmentation ×3	**76.2 / 84.6**	76.2 / 84.6

QANet – Effects of Data Augmentation



	EM / F1	Difference to Base Model
		EM / F1
Base QANet	73.6 / 82.7	
- convolution in encoders	70.8 / 80.0	-2.8 / -2.7
- self-attention in encoders	72.2 / 81.4	-1.4 / -1.3
replace sep convolution with normal convolution	72.9 / 82.0	- 0.7 / -0.7
+ data augmentation ×2 (1:1:0)	74.5 / 83.2	+0.9 / +0.5
+ data augmentation ×3 (1:1:1)	74.8 / 83.4	+1.2 / +0.7
+ data augmentation ×3 (1:2:1)	74.3 / 83.1	+0.7 / +0.4
+ data augmentation ×3 (2:2:1)	74.9 / 83.6	+1.3 / +0.9
+ data augmentation ×3 (2:1:1)	75.0 / 83.6	+1.4 / +0.9
+ data augmentation ×3 (3:1:1)	**75.1 / 83.8**	**+1.5 / +1.1**
+ data augmentation ×3 (4:1:1)	75.0 / 83.6	+1.4 / +0.9
+ data augmentation ×3 (5:1:1)	74.9 / 83.5	+1.3 / +0.8

Text Q&A Take-aways

- Memory nets include short-term and long-term memory with an indexing (attention) mechanism over long-term (WORM) memory.

- MemNN is like an associative key-value story. Its supports multiple hops to follow inference chains or get up-to-date results.

- The MemNet design (2015) included position encoding to preserve word position in later inference (borrowed in the transformer design).

- QANet (2018) used transformer-style self-attention units, which added multiple attention heads. Stacked transformer layers support multi-hop inference.

- Like almost all Q&A systems, QANet uses a bilinear combination of context and query.