BI-DIRECTIONAL ATTENTION FLOW FOR MACHINE COMPREHENSION

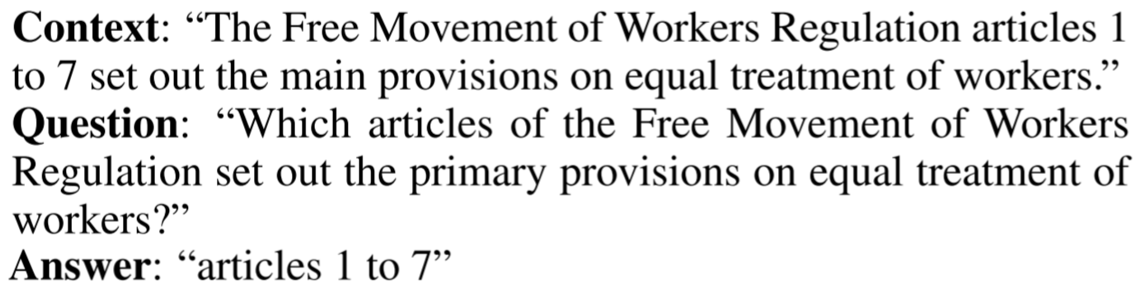
Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, Hannaneh Hajishirzi

Presenter: Wenda Qiu

04/01/2020

Machine Comprehension

- Question Answering:
  - Answer a query about a given context paragraph
- In this paper:
  - Bi-Directional Attention Flow (BIDAF) network
  - Query-aware context representation without early summarization
  - Achieve SOTA (when published) in SQuAD and CNN/DailyMail Cloze Test



Context: "The Free Movement of Workers Regulation articles 1 to 7 set out the main provisions on equal treatment of workers."
Question: "Which articles of the Free Movement of Workers Regulation set out the primary provisions on equal treatment of workers?"
Answer: "articles 1 to 7"

Previous works

Why are we using attention?

  - Allows model to focus on a small portion of the context, shown to be effective

- Dynamic attention (Bahdanau et al., 2015)
  - Attention weights updated dynamically, given the query, the context and previous attention
  - BIDAF uses a memory-less attention mechanism
- Attention calculated only once (Kadlec et al., 2016)
  - Summarize context and query with fixed-size vectors in the attention layer
  - BIDAF does not make a summarization
  - BIDAF lets the attention vectors flow into the modeling (RNN) layer
- Multi-hop attention (Sordoni et al., 2016; Dhingra et al., 2016)

Bi-Directional Attention Flow Model

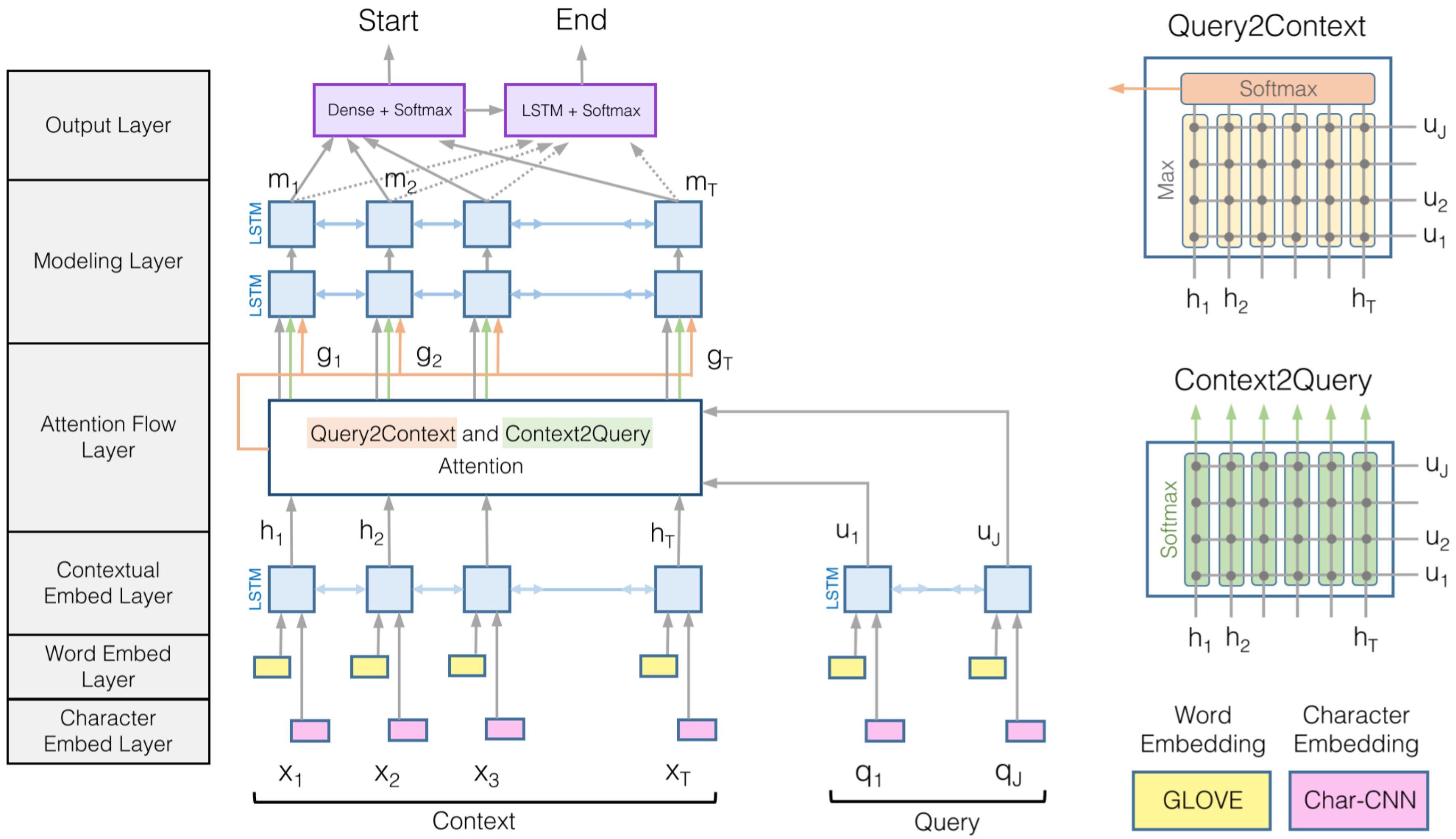
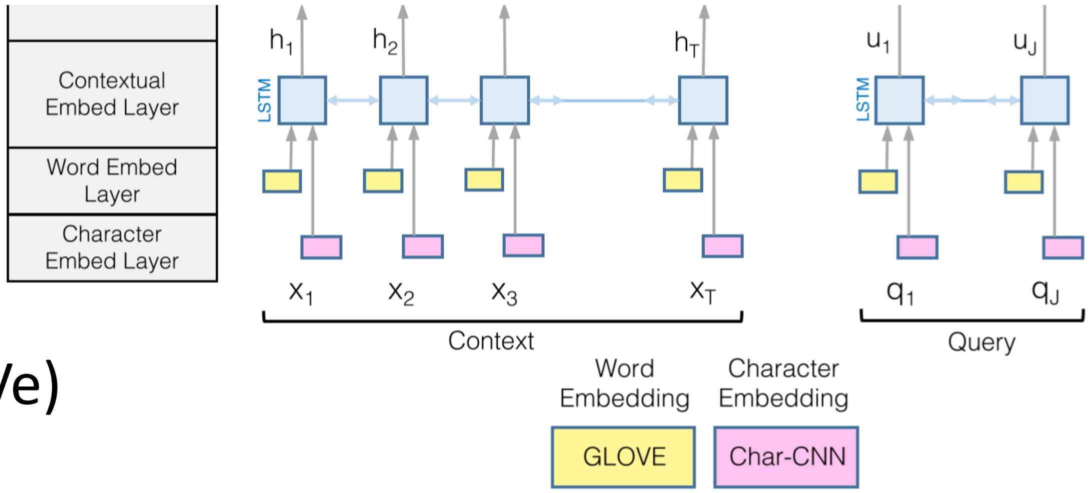


Figure 1: BiDirectional Attention Flow Model *(best viewed in color)*

1. - 3. Three Embedding Layers

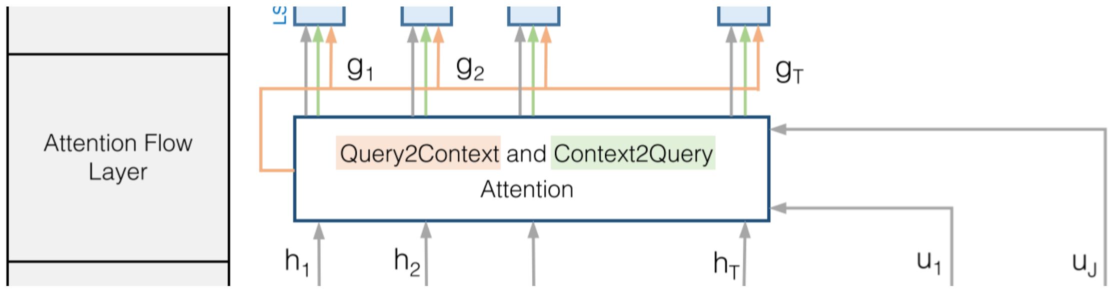
- Character Embedding Layer
  - Character-level CNNs
- Word Embedding Layer
  - Pre-trained word embedding model (GloVe)
- Contextual Embedding Layer
  - LSTMs in both directions



Applied to both context and query

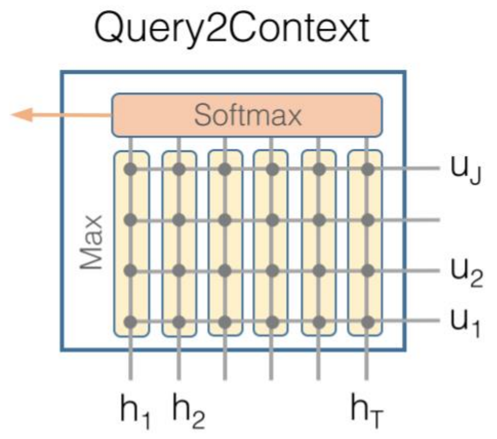
Computing features at different levels of granularity

4. Attention Flow Layer



Linking and fusing the information from the context and the query words

Attention flow: embeddings from previous layers are allowed to flow through



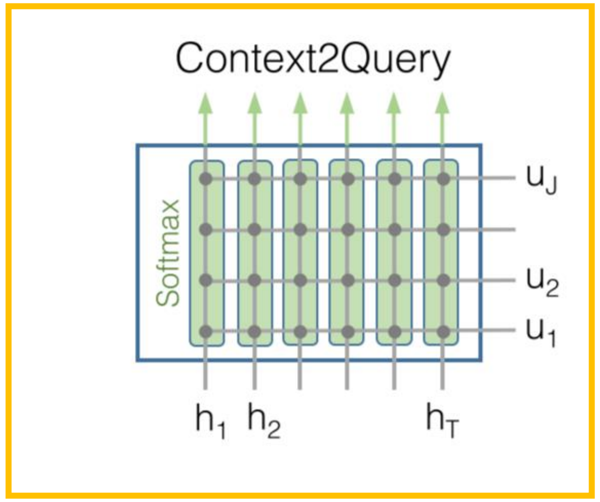
- Similarity Matrix $\mathbf{S}_{tj} = \alpha(\mathbf{H}_{:t}, \mathbf{U}_{:j}) \in \mathbb{R}^{T \times J}$

$$\alpha(\mathbf{h}, \mathbf{u}) = \mathbf{w}_{(\mathbf{S})}^{\top} [\boxed{\mathbf{h}; \mathbf{u};} \mathbf{h} \circ \mathbf{u}]$$

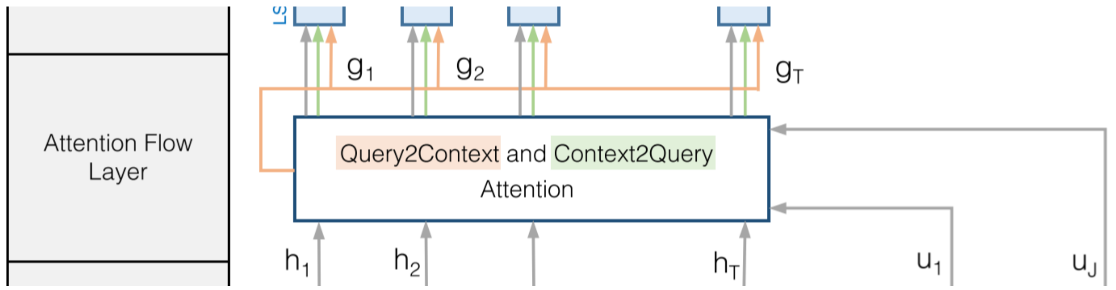
- Context-to-query Attention: signifies which query words are most relevant to each context word

$$\mathbf{a}_t = \mathrm{softmax}(\mathbf{S}_{t:}) \in \mathbb{R}^J$$

$$\tilde{\mathbf{U}}_{:t} = \sum_j \mathbf{a}_{tj} \mathbf{U}_{:j}$$



4. Attention Flow Layer



Linking and fusing the information from the context and the query words

Attention flow: embeddings from previous layers are allowed to flow through

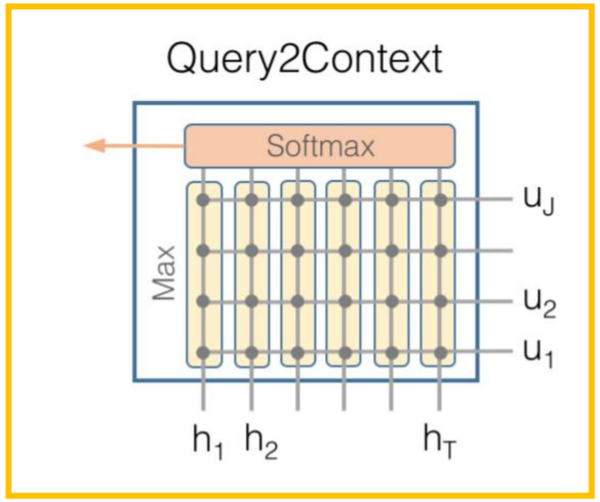
- Similarity Matrix $\mathbf{S}_{tj} = \alpha(\mathbf{H}_{:t}, \mathbf{U}_{:j}) \in \mathbb{R}^{T \times J}$

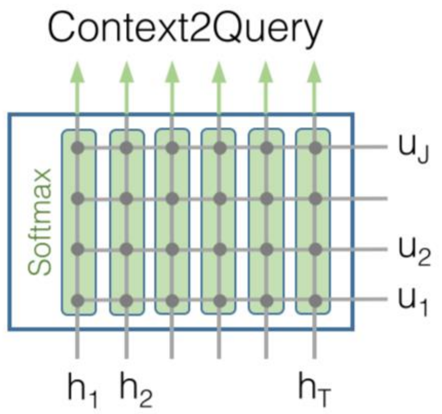
$$\alpha(\mathbf{h}, \mathbf{u}) = \mathbf{w}_{(\mathbf{S})}^{\top}[\mathbf{h}; \mathbf{u}; \mathbf{h} \circ \mathbf{u}]$$

- Query-to-context Attention: signifies which context words have the closest similarity to one of the query words

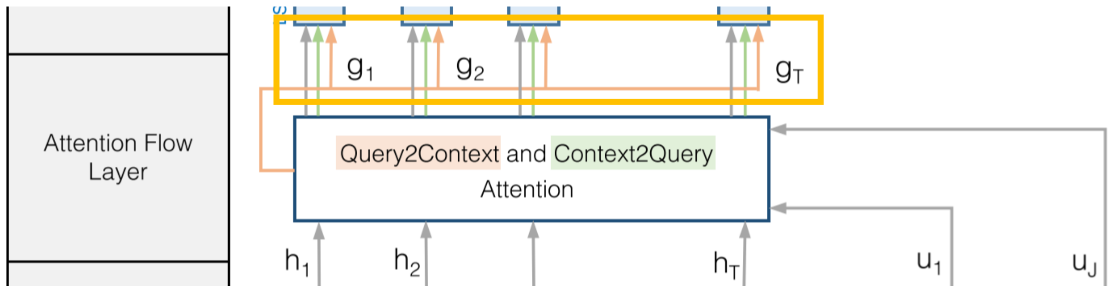
$$\mathbf{b} = \mathrm{softmax}(\max_{col}(\mathbf{S})) \in \mathbb{R}^{T}$$

$$\tilde{\mathbf{h}} = \sum_t \mathbf{b}_t \mathbf{H}_{:t} \in \mathbb{R}^{2d}$$





4. Attention Flow Layer

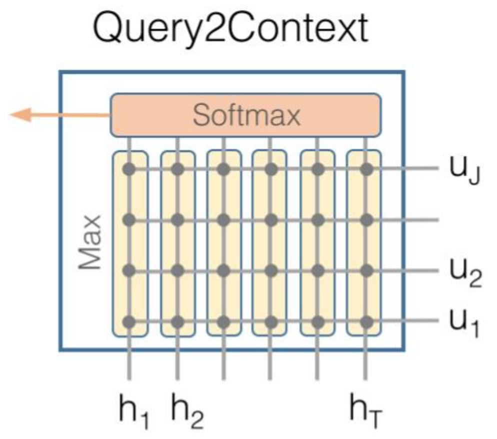


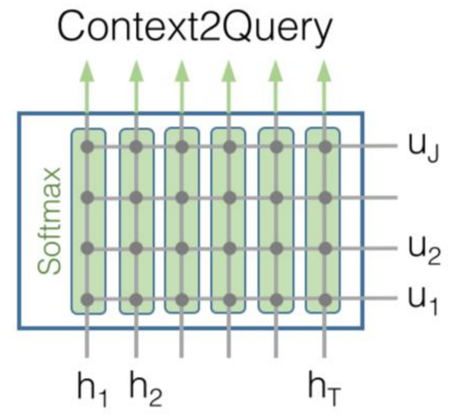
- Layer output: query-aware representation of each context word

$$\mathbf{G}_{:t} = \boldsymbol{\beta}(\mathbf{H}_{:t}, \tilde{\mathbf{U}}_{:t}, \tilde{\mathbf{H}}_{:t}) \in \mathbb{R}^{d_{\mathbf{G}}}$$

- Simple concatenation shows a good result

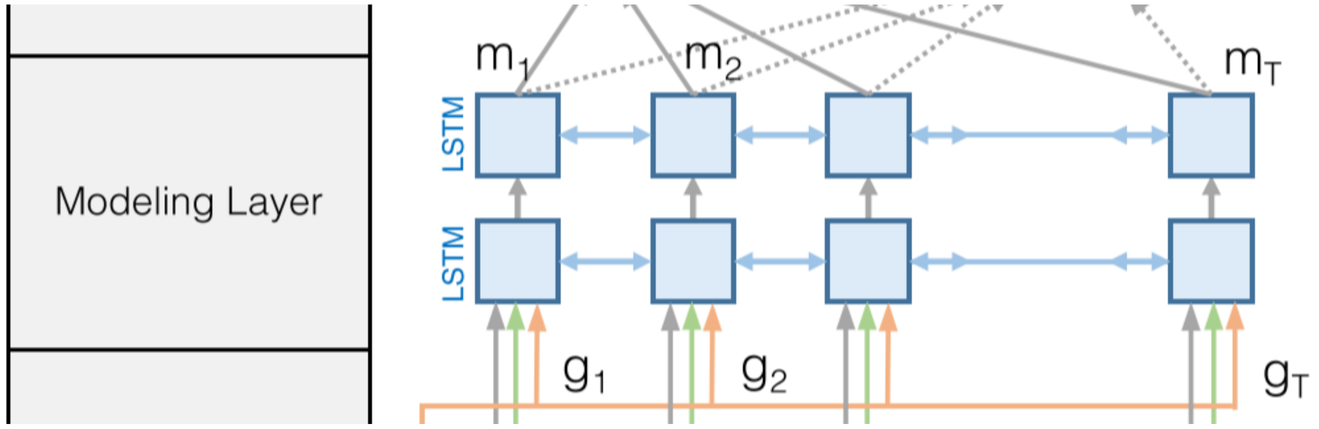
$$\boldsymbol{\beta}(\mathbf{h}, \tilde{\mathbf{u}}, \tilde{\mathbf{h}}) = [\mathbf{h}; \tilde{\mathbf{u}}; \mathbf{h} \circ \tilde{\mathbf{u}}; \mathbf{h} \circ \tilde{\mathbf{h}}] \in \mathbb{R}^{8d \times T}$$



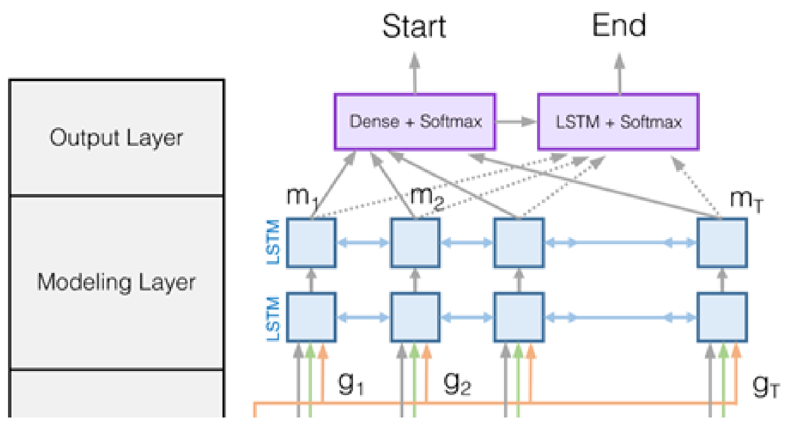


5. Modeling Layer

- Capture the interaction among the context words conditioned on the query
- 2-layer Bi-Directional LSTM



6. Output Layer



- Application-specific
- For QA: the answer phrase is derived by predicting the start and the end indices of the phrase in the paragraph
  - Probability distribution of the start index

$$\mathbf{p}^1 = \mathrm{softmax}(\mathbf{w}_{(\mathbf{p}^1)}^\top[\mathbf{G};\mathbf{M}])$$

  - Probability distribution of the end index: M2 obtained by another bidirectional LSTM layer

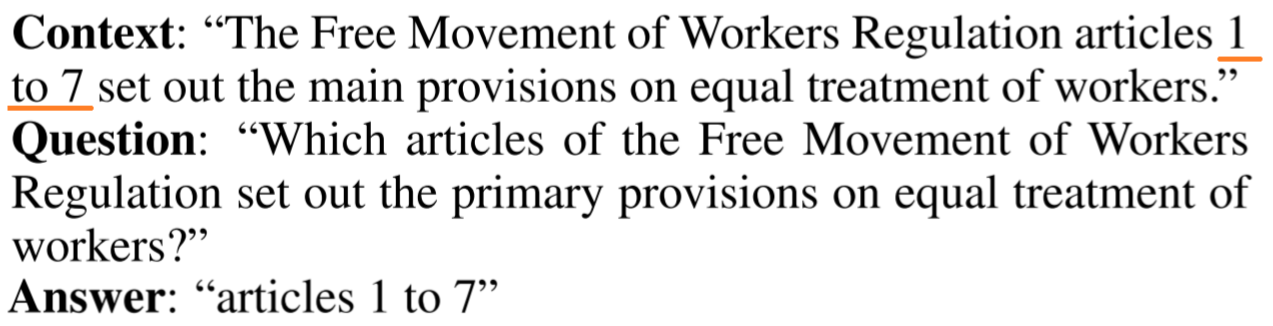
$$\mathbf{p}^2 = \mathrm{softmax}(\mathbf{w}_{(\mathbf{p}^2)}^\top[\mathbf{G};\mathbf{M}^2])$$

- Loss function:
  - Negative log loss:

$$L(\theta) = -\frac{1}{N}\sum_i^N \log(\mathbf{p}^1_{y_i^1}) + \log(\mathbf{p}^2_{y_i^2})$$

Experiments – QA

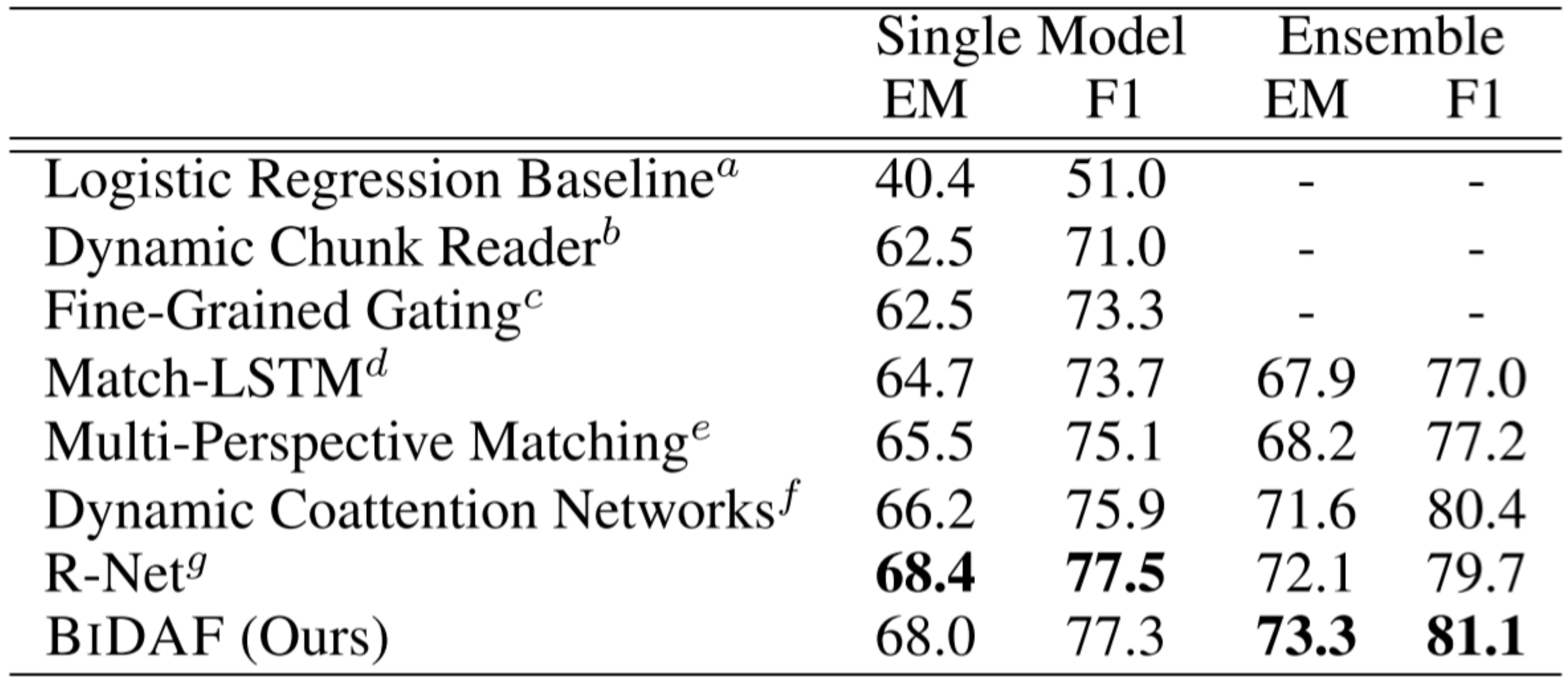
- Dataset: SQuAD
  - Wikipedia articles with more than 100,000 questions
  - The answer to each question is always a span in the context (i.e. start/end index pair)
- Evaluation metrics:
  - Exact Match (EM)
  - softer metric, character level F1 score



Context: “The Free Movement of Workers Regulation articles 1 to 7 set out the main provisions on equal treatment of workers.”
Question: “Which articles of the Free Movement of Workers Regulation set out the primary provisions on equal treatment of workers?”
Answer: “articles 1 to 7”

Experiments – QA Results

- Outperforms all previous methods when ensemble learning is applied

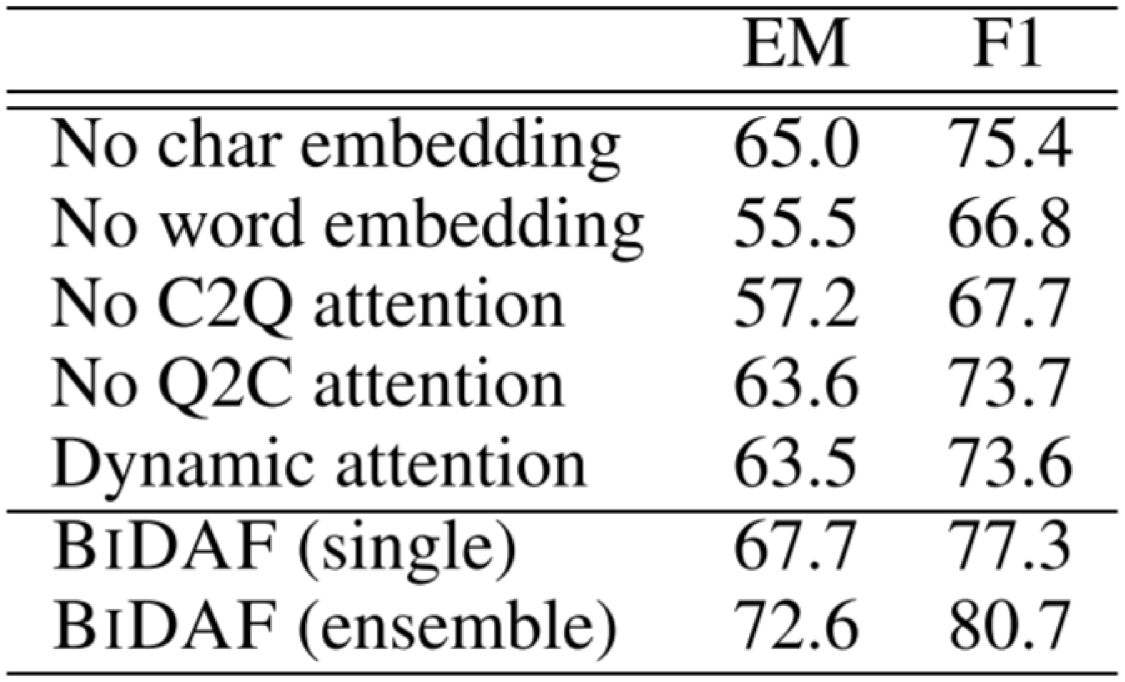


	Single Model EM	Single Model F1	Ensemble EM	Ensemble F1
Logistic Regression Baseline[a]	40.4	51.0	-	-
Dynamic Chunk Reader[b]	62.5	71.0	-	-
Fine-Grained Gating[c]	62.5	73.3	-	-
Match-LSTM[d]	64.7	73.7	67.9	77.0
Multi-Perspective Matching[e]	65.5	75.1	68.2	77.2
Dynamic Coattention Networks[f]	66.2	75.9	71.6	80.4
R-Net[g]	**68.4**	**77.5**	72.1	79.7
BIDAF (Ours)	68.0	77.3	**73.3**	**81.1**

(a) Results on the SQuAD test set

Experiments – QA Ablation Study

- Both char-level and word-level embeddings contribute
- Both directions of attention (C2Q & Q2C) are needed
- Attention flow introduced this paper is better than dynamic attention (previous works, attention is dynamically computed in modeling layer)

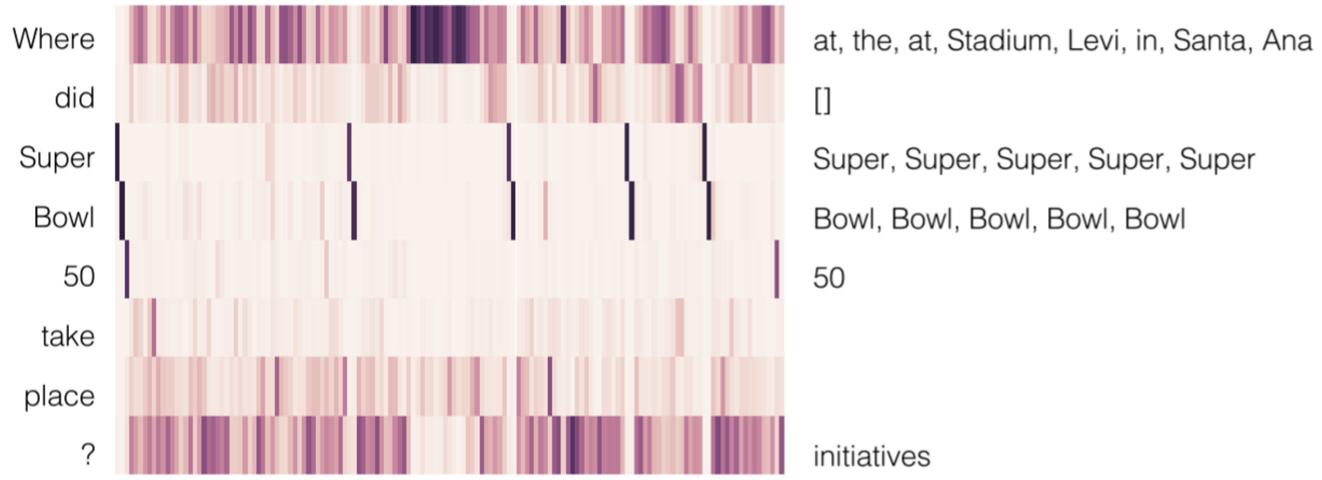


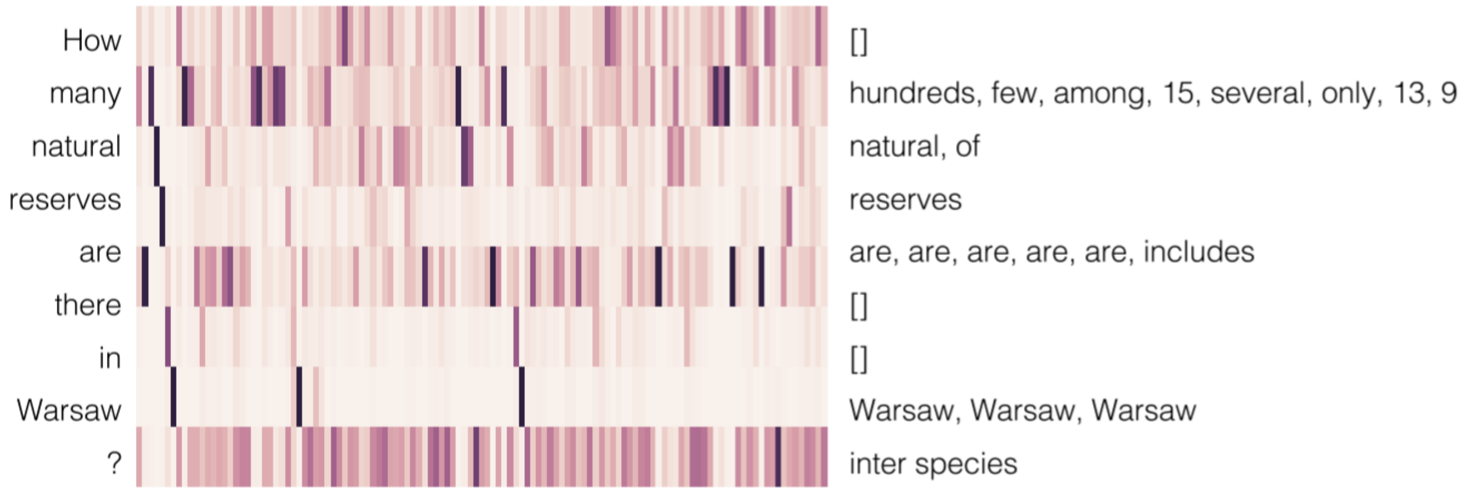
	EM	F1
No char embedding	65.0	75.4
No word embedding	55.5	66.8
No C2Q attention	57.2	67.7
No Q2C attention	63.6	73.7
Dynamic attention	63.5	73.6
BIDAF (single)	67.7	77.3
BIDAF (ensemble)	72.6	80.7

(b) Ablations on the SQuAD dev set

Visualization – QA attention similarities

- "Where" matches locations
- "many" matches quantities and numerical symbols
- Entities in the question attend to the same entities in the context





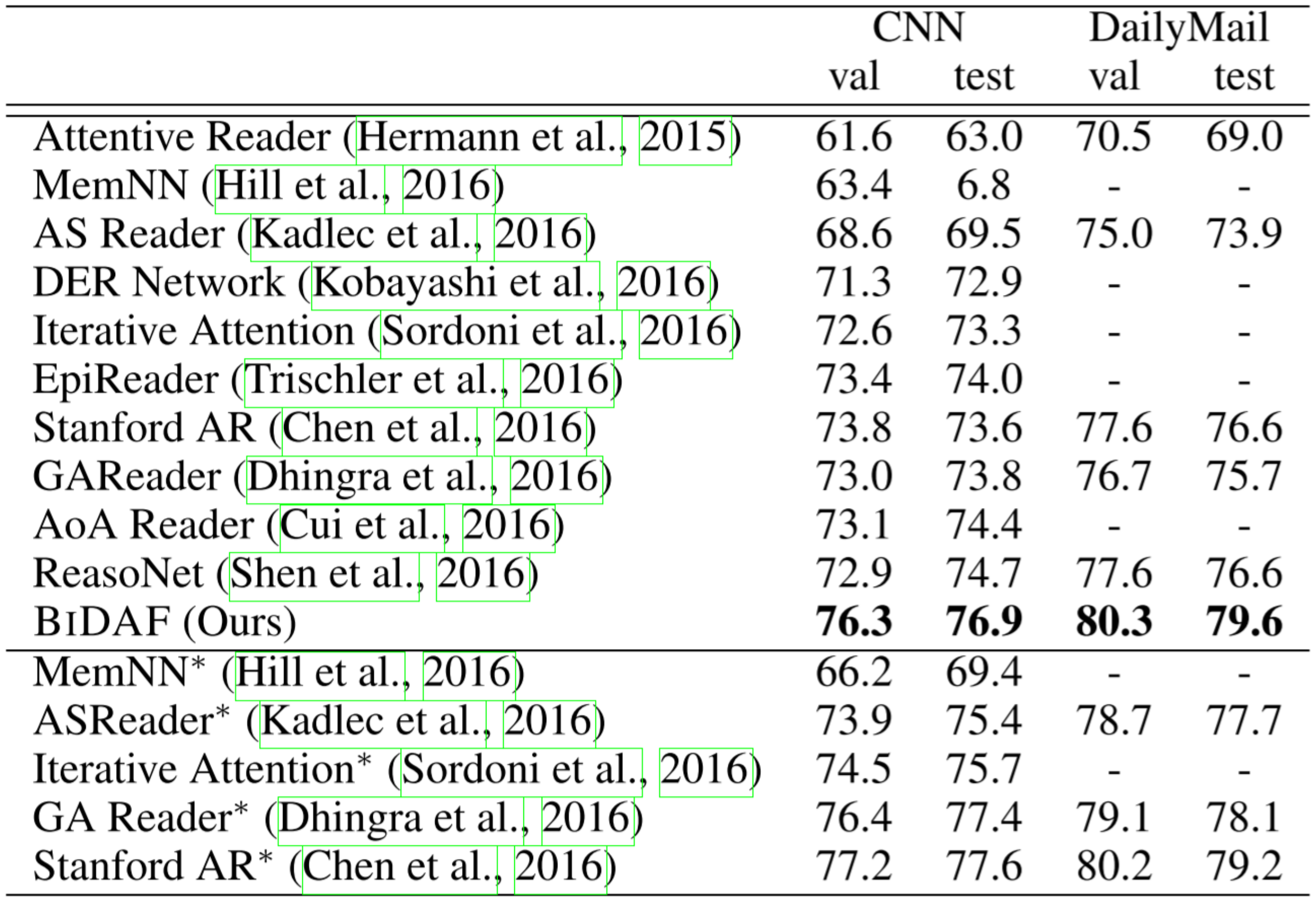
Experiments – Cloze Test

Cloze: fill in words that have been removed from a passage

- Dataset: CNN and DailyMail
  - Each example has a news article and an incomplete sentence extracted from the human-written summary of the article
- Output Layer:
  - The answer is always a single word, end index is not needed
  - $\mathbf{p}^2$ term is omitted in the loss function

Experiments – Cloze Test

- BIDAF outperforms previous single-run models on both datasets for both val and test data
- On DailyMail, BIDAF single-run model even outperforms the best ensemble method



	CNN		DailyMail	
	val	test	val	test
Attentive Reader (Hermann et al., 2015)	61.6	63.0	70.5	69.0
MemNN (Hill et al., 2016)	63.4	6.8	-	-
AS Reader (Kadlec et al., 2016)	68.6	69.5	75.0	73.9
DER Network (Kobayashi et al., 2016)	71.3	72.9	-	-
Iterative Attention (Sordoni et al., 2016)	72.6	73.3	-	-
EpiReader (Trischler et al., 2016)	73.4	74.0	-	-
Stanford AR (Chen et al., 2016)	73.8	73.6	77.6	76.6
GAReader (Dhingra et al., 2016)	73.0	73.8	76.7	75.7
AoA Reader (Cui et al., 2016)	73.1	74.4	-	-
ReasoNet (Shen et al., 2016)	72.9	74.7	77.6	76.6
BIDAF (Ours)	**76.3**	**76.9**	**80.3**	**79.6**
MemNN* (Hill et al., 2016)	66.2	69.4	-	-
ASReader* (Kadlec et al., 2016)	73.9	75.4	78.7	77.7
Iterative Attention* (Sordoni et al., 2016)	74.5	75.7	-	-
GA Reader* (Dhingra et al., 2016)	76.4	77.4	79.1	78.1
Stanford AR* (Chen et al., 2016)	77.2	77.6	80.2	79.2

Conclusion

- Bi-Directional Attention Flow (BIDAF) network is proposed
- BIDAF has:
  - Multi-stage hierarchical process
    - The 6 layers with different functions
  - Context representation at different levels of granularity
    - Character level, word level, contextualized level
  - Bi-directional attention flow mechanism
    - Context2Query, Query2Context
  - Query aware context representation without early summarization
    - Both attention and embeddings are fed into the modeling layer

Making Neural QA as Simple as Possible but not Simpler

DIRK WEISSENBORN, GEORG WIESE, AND LAURA SEIFFE 2017

PRESENTED BY KEVIN PEI

Outline

1. Motivation and Background

Current QA models are too complex

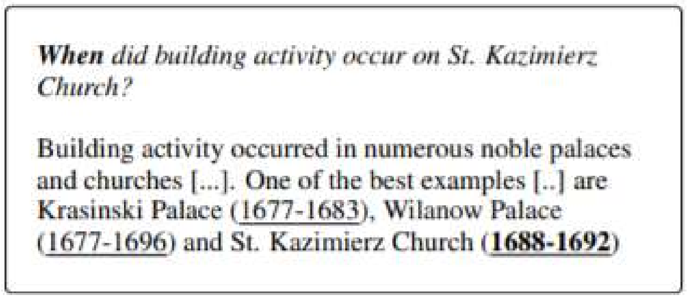
- Contain layers that measure word-to-word interactions
  - Much of the current work in neural QA focuses on this interaction layer (attention, co-attention, etc.)
- No good baseline for QA

Question type intuition

- The answer type should match the type specified by the question (e.g. time for "when")

Context intuition

- The words surrounding the answer should be words that appear in the question



***When** did building activity occur on St. Kazimierz Church?*

Building activity occurred in numerous noble palaces and churches [...]. One of the best examples [..] are Krasinski Palace (1677-1683), Wilanow Palace (1677-1696) and St. Kazimierz Church (**1688-1692**)

2. Baseline BoW Neural QA Model

Previous basic model baseline for SQuAD was logistic regression

This new baseline uses the Question type intuition and Context intuition in a neural model

Embeddings = word embeddings concatenated with character embeddings (Seo et. al 2017)

Question type intuition is captured by comparing the Lexical Answer Type (LAT) of the question to a candidate answer span

- LAT is expected type of the answer – Who, Where, When, etc. or noun phrase after What or Which
- Candidate answer spans are all word spans below a max length (10 in this paper)

When did building activity occur on St. Kazimierz Church?
What ***building activity*** occurred at St. Kazimierz Church on 1688?

2. Baseline BoW Neural QA Model

LAT encoding for question = concatenation of embeddings of first token, average embeddings of all tokens, and embeddings of last token

- LAT encoding is further transformed with fully connected layer and tanh non-linearity into $\tilde{\boldsymbol{z}} \in \mathbb{R}^n$

Span encoding = concatenation of average embeddings of context tokens to the left of the span, embeddings of first token, average embeddings of all tokens, embeddings of last token, and average embeddings of context tokens to the right of the span

- Window size = 5
- Span encoding is further transformed with fully connected layer and tanh non-linearity into $\tilde{\boldsymbol{x}}_{s,e} \in \mathbb{R}^n$

Final type score is derived from feeding $[\tilde{\boldsymbol{z}}; \tilde{\boldsymbol{x}}_{s,e}; \tilde{\boldsymbol{z}} \odot \tilde{\boldsymbol{x}}_{s,e}]$ into a feedforward network with one hidden layer

When did building activity occur on St. Kazimierz Church?	… Krasinski Palace (1677-1683), Wilanow Palace (1677-1696) and St. Kazimierz Church (1688-1692).
What ***building activity*** occurred at St. Kazimierz Church on 1688?	

2. Baseline BoW Neural QA Model

Context matching intuition is captured by word-in-question (wiq) features

- Binary wiq (wiq^b) – Only checks for presence of question words q in the context x

$$\text{wiq}^b_j = \mathbb{I}(\exists i : x_j = q_i)$$

- Weighted wiq (wiq^w) – Allows for matching of synonyms and different morphological forms while also emphasizing rare tokens in the context (rare tokens are probably more informative)

$$sim_{i,j} = \boldsymbol{v}_{wiq}(\boldsymbol{x}_j \odot \boldsymbol{q}_i) \quad , \boldsymbol{v}_{wiq} \in \mathbb{R}^n$$
$$\text{wiq}^w_j = \sum_i \text{softmax}(sim_{i,\cdot})_j$$

- Softmax emphasizes rare tokens because tokens in the context more similar to a question token and less similar to all other tokens in the context get higher softmax scores

wiq features are calculated for the left and right contexts for spans 5, 10, and 20 (12 total features)

A scalar weight for each wiq feature is learned (no method specified) and they're summed to obtain a context-matching score for a candidate answer span

Final score for a span is sum of type and context matching score

When did building activity occur on St. Kazimierz Church?	… Krasinski Palace (1677-1683), Wilanow Palace (1677-1696) and St. Kazimierz Church (1688-1692).
What ***building activity*** occurred at St. Kazimierz Church on 1688?	

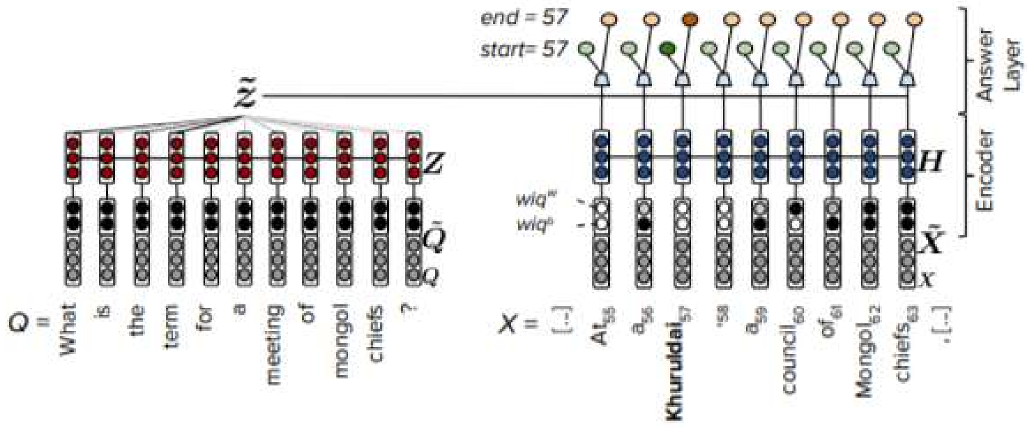
3. FastQA

BoW is insufficient – RNN-based networks can better capture syntax and semantics

FastQA consists of an embedding, encoding, and answer layer

Embeddings are handled similar to Seo et. al (2017) – word and character embeddings are jointly projected to n-dimensional representation and transformed by a single-layer highway network

$$\boldsymbol{x}' = P\boldsymbol{x} \quad , P \in \mathbb{R}^{n \times d}$$
$$\boldsymbol{g}_e = \boldsymbol{\sigma}(\mathrm{FC}(\boldsymbol{x}'))\,,\ \boldsymbol{x}'' = \tanh\left(\mathrm{FC}\left(\boldsymbol{x}'\right)\right)$$
$$\tilde{\boldsymbol{x}} = \boldsymbol{g}_e\boldsymbol{x}' + (1 - \boldsymbol{g}_e)\boldsymbol{x}''$$



end = 57
start= 57
z̃
Z
Q̃
Q
H
X̃
X
wiq^w
wiq^b
Answer Layer
Encoder
Q = What is the term for a meeting of mongol chiefs ?
X = [...] At_55 a_56 Khuruldai_57 ,_58 a_59 council_60 of_61 Mongol_62 chiefs_63 , [...]

3. FastQA

Encoding layer input consists of concatenation of embeddings and wiq features

$$\tilde{\boldsymbol{X}} = ([\tilde{\boldsymbol{x}}_1; \mathrm{wiq}_1^b; \mathrm{wiq}_1^w], ..., [\tilde{\boldsymbol{x}}_{L_X}; \mathrm{wiq}_{L_X}^b; \mathrm{wiq}_{L_X}^w])$$

The encoding layer is a BiLSTM

$$\boldsymbol{H'} = \text{Bi-LSTM}(\tilde{\boldsymbol{X}}) \quad , \boldsymbol{H'} \in \mathbb{R}^{2n \times L_X}$$

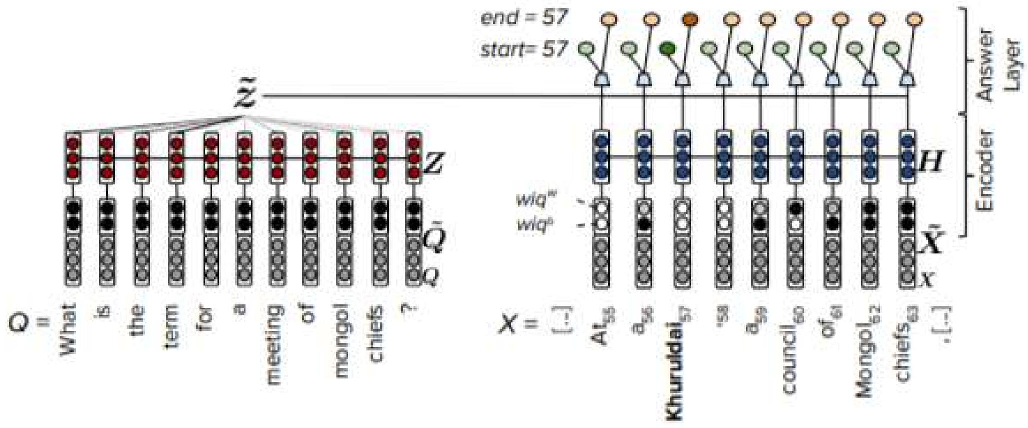
$$\boldsymbol{H} = \tanh(B\boldsymbol{H'}^\top) \quad , B \in \mathbb{R}^{n \times 2n}$$

The same encoder parameters are used for context and question words, except with different B and wiq features of 1 for question words

$$\boldsymbol{H} = [\boldsymbol{h}_1, ..., \boldsymbol{h}_{L_X}]$$

$$\boldsymbol{Z} = [\boldsymbol{z}_1, ..., \boldsymbol{z}_{L_Q}]$$

H is encoded context, Z is encoded question


end = 57
start= 57
$\tilde{z}$
Z
$\tilde{Q}$
Q
H
$\tilde{X}$
X
wiq^w
wiq^b
Answer Layer
Encoder
Q = What is the term for a meeting of mongol chiefs ?
X = [...] At$_{55}$ a$_{56}$ Khuruldai$_{57}$,$_{58}$ a$_{59}$ council$_{60}$ of$_{61}$ Mongol$_{62}$ chiefs$_{63}$,[...]

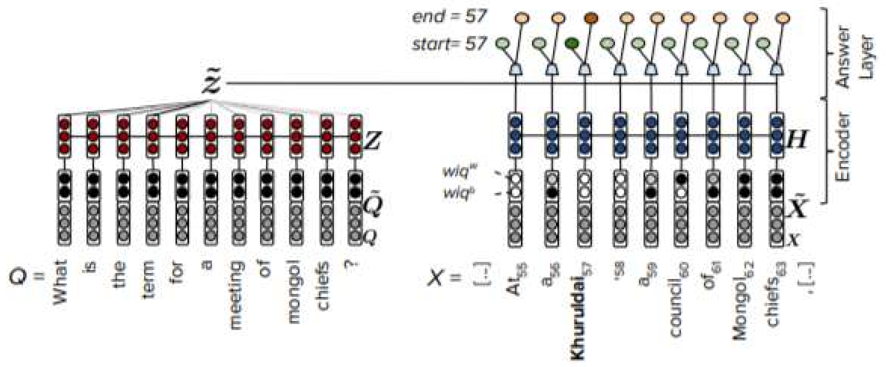
3. FastQA

The answer layer calculates probability distributions for the start and end locations of the answer span

$$\alpha = \text{softmax}(\boldsymbol{v}_q \boldsymbol{Z}) \quad , \boldsymbol{v}_q \in \mathbb{R}^n$$
$$\tilde{\boldsymbol{z}} = \sum_i \alpha_i \boldsymbol{z}_i$$

$$\boldsymbol{s}_j = \text{ReLU}\left(\text{FC}\left([\boldsymbol{h}_j; \tilde{\boldsymbol{z}}; \boldsymbol{h}_j \odot \tilde{\boldsymbol{z}}]\right)\right)$$
$$p_s(j) \propto \exp(\boldsymbol{v}_s \boldsymbol{s}_j) \quad , \boldsymbol{v}_s \in \mathbb{R}^n$$

p_s is the probability distribution of the start location


end = 57
start= 57
$\tilde{z}$
Z
H
$\tilde{Q}$
Q
$\tilde{X}$
X
wiq^w
wiq^b
Answer Layer
Encoder
Q = What is the term for a meeting of mongol chiefs ?
X = [...] At55 a56 Khurultai57 '58 a59 council60 of61 Mongol62 chiefs63 .[...]

$$\boldsymbol{e}_j = \text{ReLU}\left(\text{FC}\left([\boldsymbol{h}_j; \boldsymbol{h}_s; \tilde{\boldsymbol{z}}; \boldsymbol{h}_j \odot \tilde{\boldsymbol{z}}; \boldsymbol{h}_j \odot \boldsymbol{h}_s]\right)\right)$$
$$p_e(j|s) \propto \exp(\boldsymbol{v}_e \boldsymbol{e}_j) \quad , \boldsymbol{v}_e \in \mathbb{R}^n \qquad (9)$$

p_e is the probability distribution of the end location – it’s conditioned on the start location

Overall probability of predicting an answer span with start location s and end location e
$p(s, e) = p_s(s) * p_e(e|s)$

Beam-search is used to find the answer span with highest probability

4. FastQA Extended

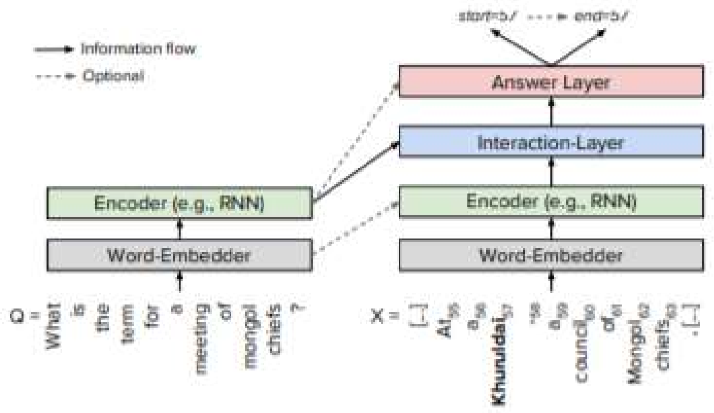
FastQA omits the interaction layer typical of neural QA systems

- Previous works have used attention, co-attention, bi-directional attention flow, multi-perspective context matching, or fine-grained gating

FastQA is extended with representation fusion

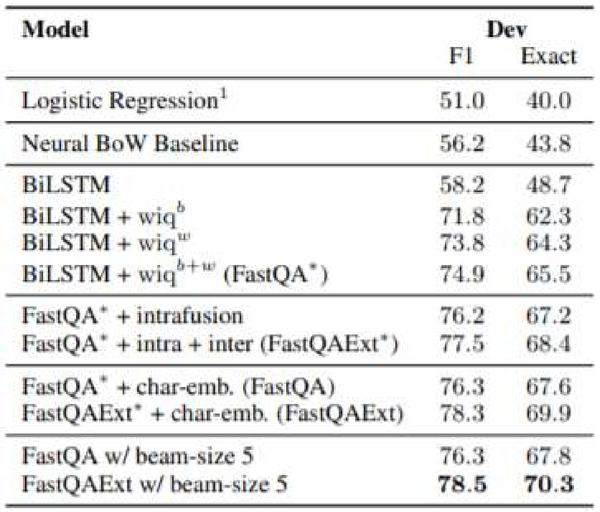
Each state representation has a weighted sum with co-representations retrieved via attention

- Intra-fusion for other passages in the context
- Inter-fusion for the question



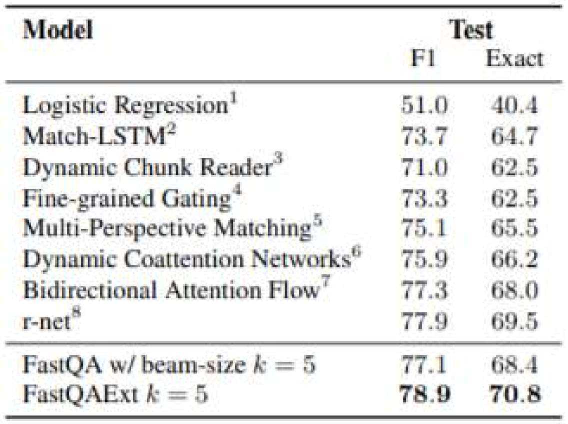
5. Results

SQuAD Ablation Studies



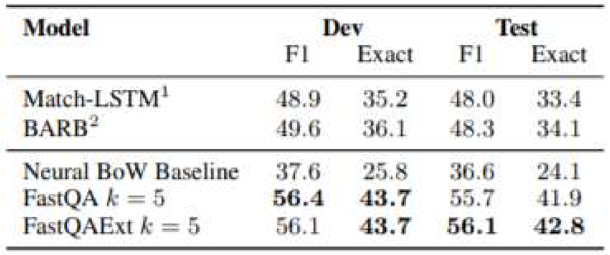
Model	Dev F1	Dev Exact
Logistic Regression[1]	51.0	40.0
Neural BoW Baseline	56.2	43.8
BiLSTM	58.2	48.7
BiLSTM + wiq^b	71.8	62.3
BiLSTM + wiq^w	73.8	64.3
BiLSTM + wiq^{b+w} (FastQA*)	74.9	65.5
FastQA* + intrafusion	76.2	67.2
FastQA* + intra + inter (FastQAExt*)	77.5	68.4
FastQA* + char-emb. (FastQA)	76.3	67.6
FastQAExt* + char-emb. (FastQAExt)	78.3	69.9
FastQA w/ beam-size 5	76.3	67.8
FastQAExt w/ beam-size 5	**78.5**	**70.3**

SQuAD (12/29/2016)



Model	Test F1	Test Exact
Logistic Regression[1]	51.0	40.4
Match-LSTM[2]	73.7	64.7
Dynamic Chunk Reader[3]	71.0	62.5
Fine-grained Gating[4]	73.3	62.5
Multi-Perspective Matching[5]	75.1	65.5
Dynamic Coattention Networks[6]	75.9	66.2
Bidirectional Attention Flow[7]	77.3	68.0
r-net[8]	77.9	69.5
FastQA w/ beam-size $k = 5$	77.1	68.4
FastQAExt $k = 5$	**78.9**	**70.8**

NewsQA Dataset



Model	Dev F1	Dev Exact	Test F1	Test Exact
Match-LSTM[1]	48.9	35.2	48.0	33.4
BARB[2]	49.6	36.1	48.3	34.1
Neural BoW Baseline	37.6	25.8	36.6	24.1
FastQA $k = 5$	**56.4**	**43.7**	55.7	41.9
FastQAExt $k = 5$	56.1	**43.7**	**56.1**	**42.8**

FastQAExt achieves state of the art performance (as of 12/29/16)

FastQAExt takes 2x as long to run and 2x more memory than FastQA

5. Results

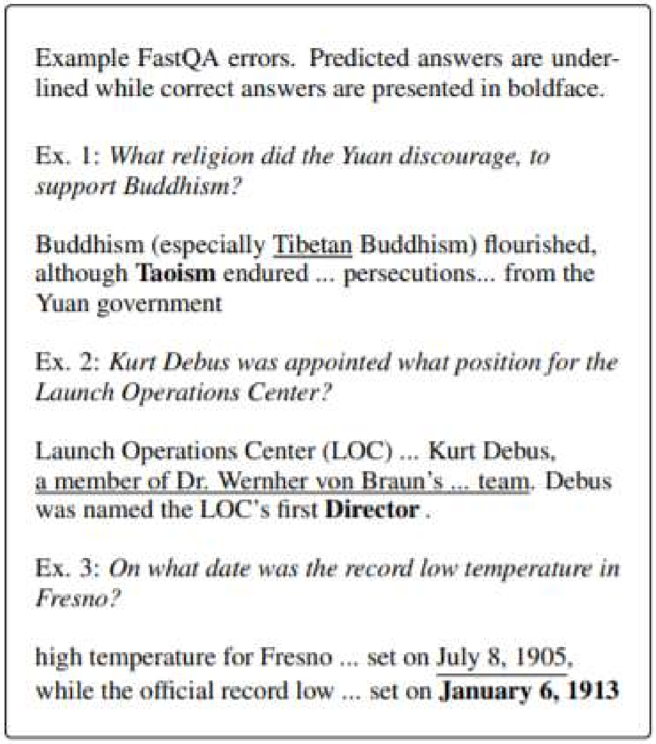
Ex. 1 Failure: Lack of fine-grained understanding of answer types

Ex. 2 Failure: Lack of co-reference resolution

Ex. 3 Failure: Nested syntactic structures, ignoring punctuations and conjunctions

Manual examination

- 35/55 mistakes can be attributed to the context and type matching heuristics
- 44/50 correct answers can be solved using the heuristics
- FastQA extended is not systematically better than FastQA, the questions it can answer that FastQA can't are varied
- Similarly, compared to the Dynamic Coattention Network (Xiong et. al 2017), DCN has slightly better performance but not in any specific way



Example FastQA errors. Predicted answers are underlined while correct answers are presented in boldface.

Ex. 1: *What religion did the Yuan discourage, to support Buddhism?*

Buddhism (especially Tibetan Buddhism) flourished, although **Taoism** endured ... persecutions... from the Yuan government

Ex. 2: *Kurt Debus was appointed what position for the Launch Operations Center?*

Launch Operations Center (LOC) ... Kurt Debus, a member of Dr. Wernher von Braun's ... team. Debus was named the LOC's first **Director** .

Ex. 3: *On what date was the record low temperature in Fresno?*

high temperature for Fresno ... set on July 8, 1905, while the official record low ... set on **January 6, 1913**

6. Discussion

This paper claims that previous work is created top-down

- Interaction layer’s complexity is justified post-hoc

This paper is built on intuitions about the problem

Features used resemble attention, and attention has same goals as intuitions

- Features are more transparent attention mechanism

More in-depth study of time and space requirements would be appreciated

- Make the tradeoff between models more clear

7. Conclusion

This paper introduces two new baseline neural QA models based on intuitions about QA

- Neural BoW model
- FastQA

Compared to more complex previous methods, FastQA is relatively simple

- Extending FastQA with a complex interaction layer similar to previous work gives it state-of-the-art performance

This paper identifies which parts of neural QA systems lead to the most gain

- Question awareness
- More complex models than BoW

Questions?

Gated Self-Matching Networks for Reading Comprehension and Question Answering

Zhouxiang Cai

Introduction

— **Reading comprehension style question answering**

- Passage P and question Q are given
- Predict an answer A (WHO, WHEN WHY) to question Q based on P.

— **Main Contirbution**

- Gate-attention: add an additional gate to the model, to account for the words in a passage are of different importance to answer a particular question.

- Self-matching: effectively aggregate evidence from the whole passage to infer the answer

Model Structure:

- Question and Passage Encoder
- Gated Attention-based Recurrent Networks
- Self-Matching Attention
- Output Layer

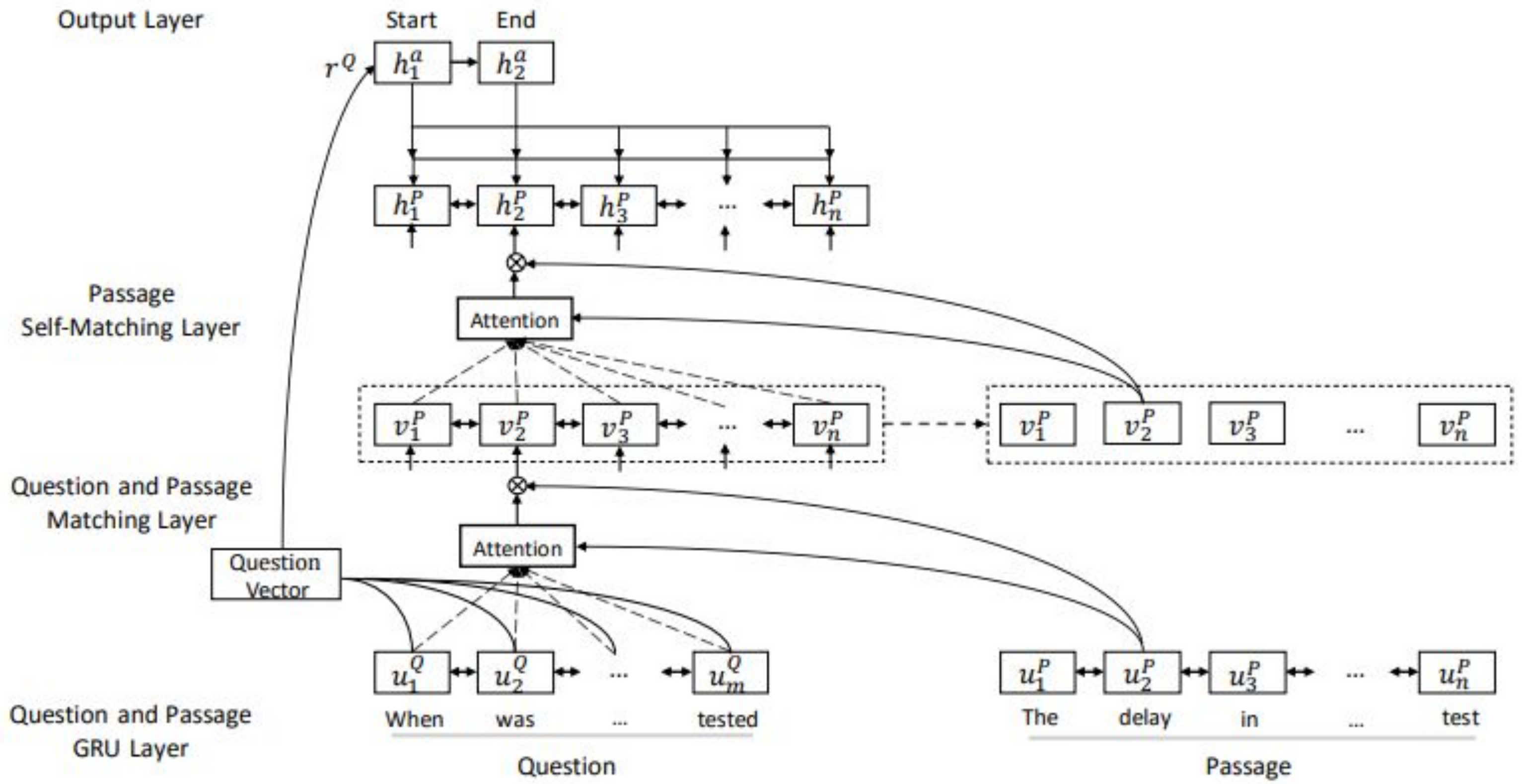
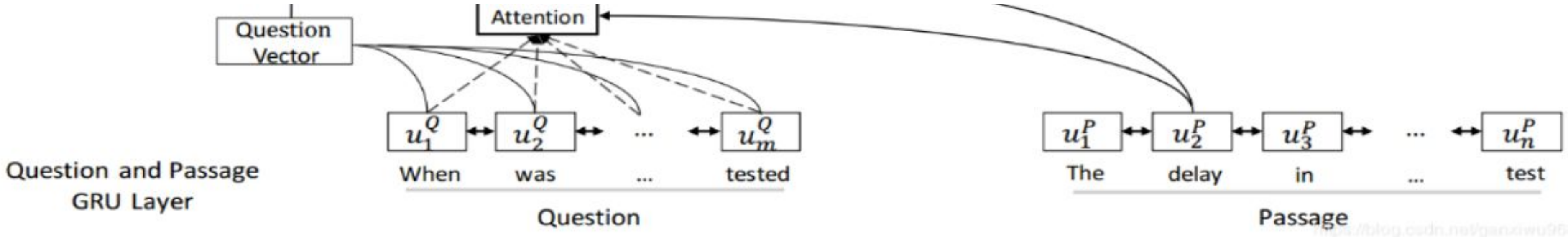


Figure 1: Gated Self-Matching Networks structure overview.

1: Question and Passage Encoder

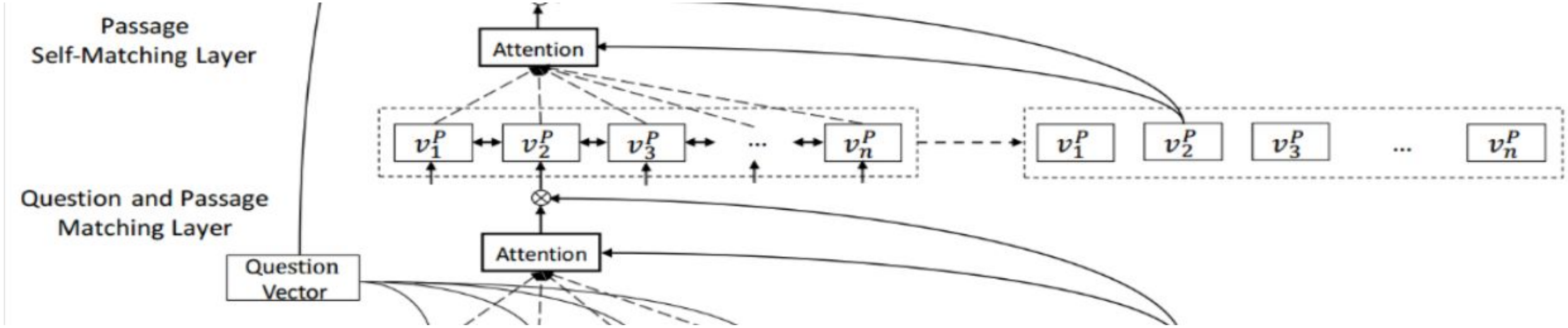
- Convert the words to word-level embeddings character-level embeddings.
- Use a bi-directional RNN to produce new representation

$$u_t^Q = \mathrm{BiRNN}_Q(u_{t-1}^Q, [e_t^Q, c_t^Q])$$
$$u_t^P = \mathrm{BiRNN}_P(u_{t-1}^P, [e_t^P, c_t^P])$$



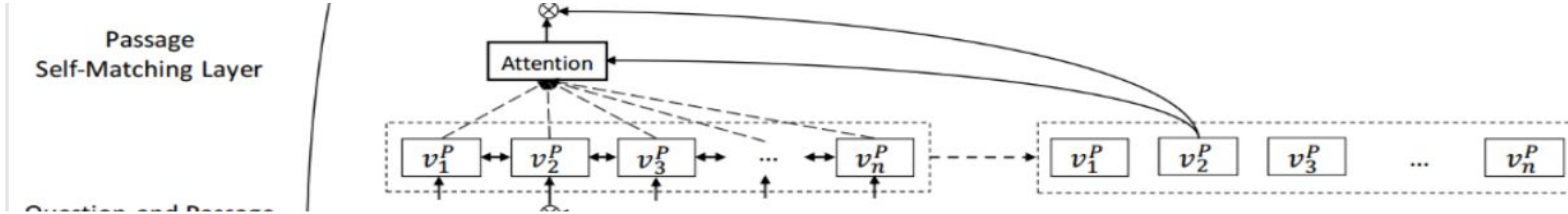
2: Gated Attention-based Recurrent Networks

- Incorporate question information into passage representation
- Addtional gate



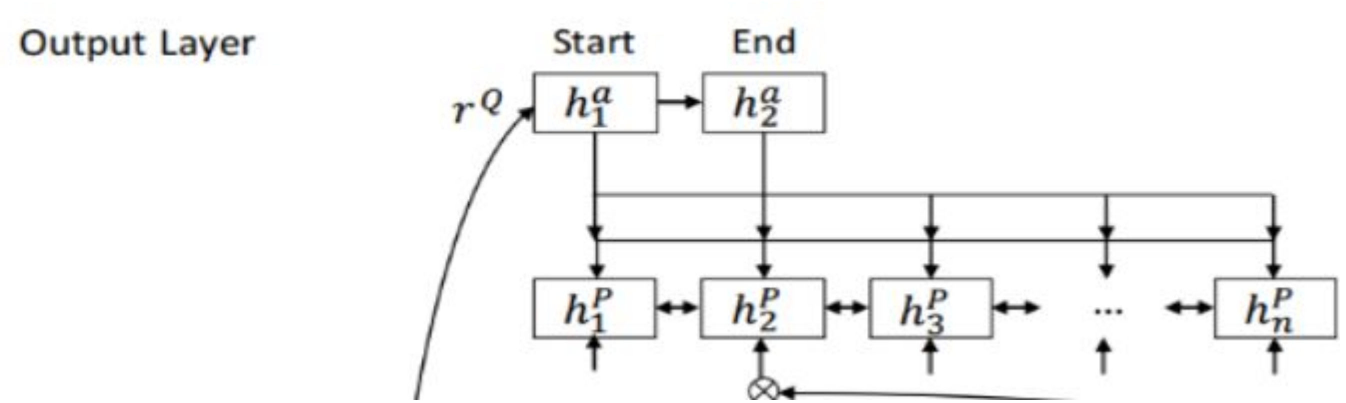
3: Self-Matching Attention

- One problem with attention representation is that it has very limited knowledge of context.
- One answer candidate is often oblivious in the passage outside its surrounding window
- Add a self-matching layer to solve this problem



4: Output Layer

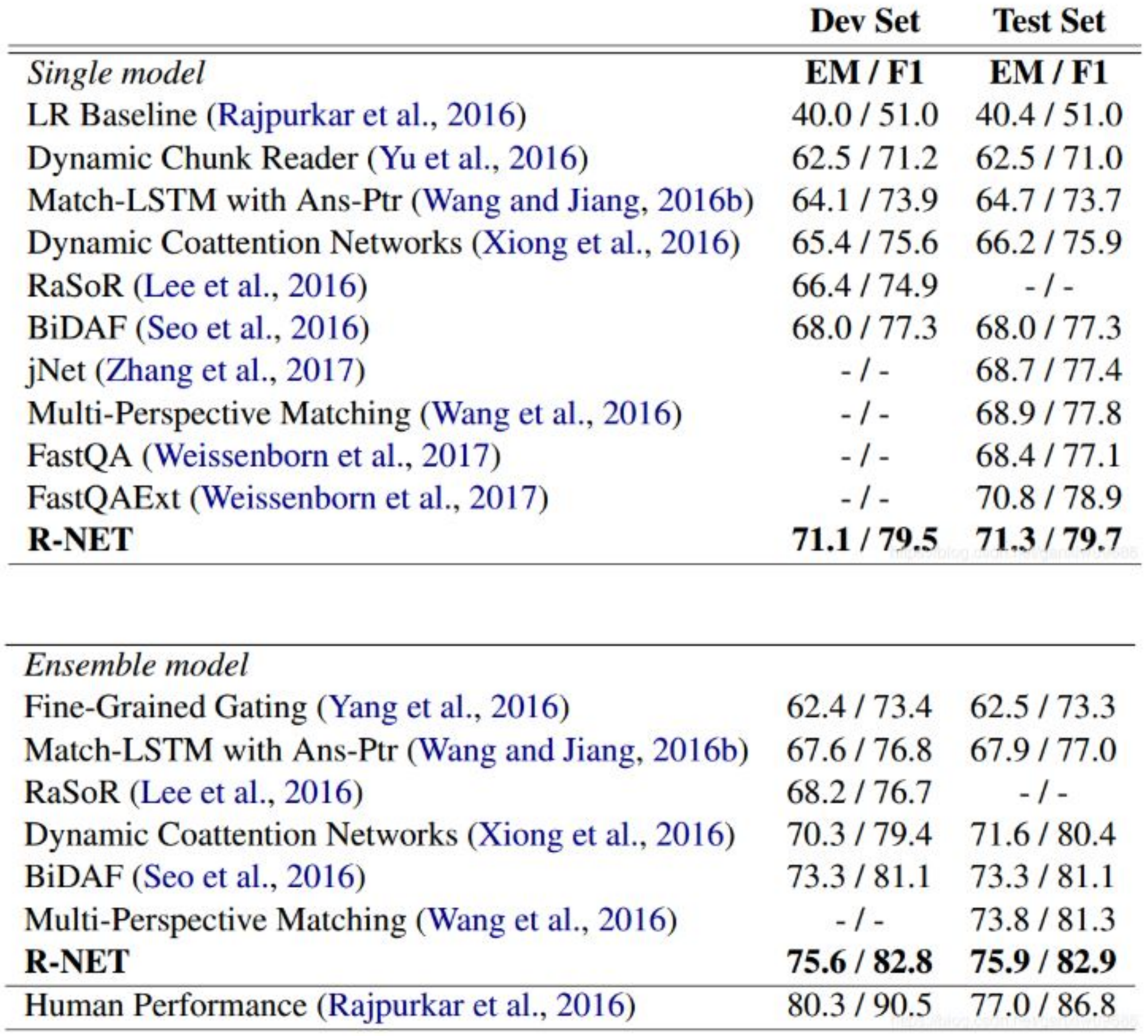
- Predict the start and end position of the answer.
- Use an attention-pooling over the question representation to generate the initial hidden vector for the pointer network



Result

Database: Stanford Question Answering Dataset (SQuAD) (Rajpurkar et al., 2016)

a large scale dataset for reading comprehension and question answering which is manually created through crowdsourcing.

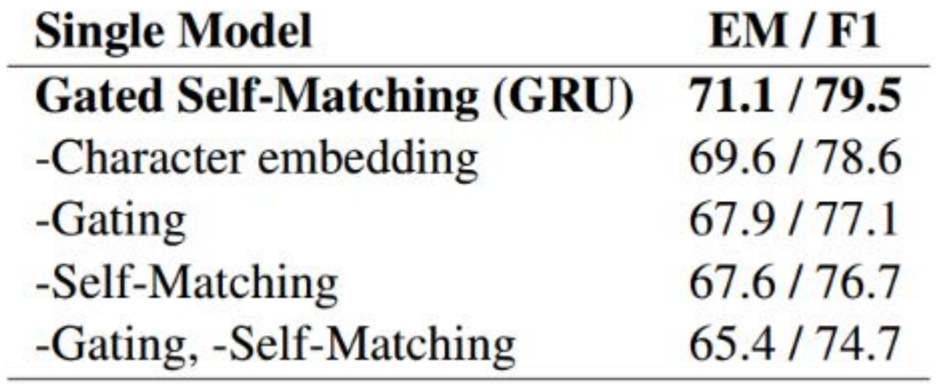


	Dev Set	Test Set
Single model	**EM / F1**	**EM / F1**
LR Baseline (Rajpurkar et al., 2016)	40.0 / 51.0	40.4 / 51.0
Dynamic Chunk Reader (Yu et al., 2016)	62.5 / 71.2	62.5 / 71.0
Match-LSTM with Ans-Ptr (Wang and Jiang, 2016b)	64.1 / 73.9	64.7 / 73.7
Dynamic Coattention Networks (Xiong et al., 2016)	65.4 / 75.6	66.2 / 75.9
RaSoR (Lee et al., 2016)	66.4 / 74.9	- / -
BiDAF (Seo et al., 2016)	68.0 / 77.3	68.0 / 77.3
jNet (Zhang et al., 2017)	- / -	68.7 / 77.4
Multi-Perspective Matching (Wang et al., 2016)	- / -	68.9 / 77.8
FastQA (Weissenborn et al., 2017)	- / -	68.4 / 77.1
FastQAExt (Weissenborn et al., 2017)	- / -	70.8 / 78.9
R-NET	**71.1 / 79.5**	**71.3 / 79.7**
Ensemble model		
Fine-Grained Gating (Yang et al., 2016)	62.4 / 73.4	62.5 / 73.3
Match-LSTM with Ans-Ptr (Wang and Jiang, 2016b)	67.6 / 76.8	67.9 / 77.0
RaSoR (Lee et al., 2016)	68.2 / 76.7	- / -
Dynamic Coattention Networks (Xiong et al., 2016)	70.3 / 79.4	71.6 / 80.4
BiDAF (Seo et al., 2016)	73.3 / 81.1	73.3 / 81.1
Multi-Perspective Matching (Wang et al., 2016)	- / -	73.8 / 81.3
R-NET	**75.6 / 82.8**	**75.9 / 82.9**
Human Performance (Rajpurkar et al., 2016)	80.3 / 90.5	77.0 / 86.8

Result

Database: Stanford Question Answering Dataset (SQuAD) (Rajpurkar et al., 2016)

a large scale dataset for reading comprehension and question answering which is manually created through crowdsourcing.



Single Model	EM / F1
Gated Self-Matching (GRU)	**71.1 / 79.5**
-Character embedding	69.6 / 78.6
-Gating	67.9 / 77.1
-Self-Matching	67.6 / 76.7
-Gating, -Self-Matching	65.4 / 74.7

Disscuss

Advatange:

Use gate to link question to the message

Use self-matching to find evidence from the message

Shortcoming:

Perform bad in long question

Perform bad in why question

Future Woek

As for future work, authors are applying the gated self-matching networks to other reading comprehension and question answering datasets, such as the MS MARCO dataset.

Thank You

Memory Networks

Jason Weston, Sumit Chopra & Antoine Bordes

Facebook AI Research

Presented by Xiaoyan Wang (xiaoyan5@Illinois.edu)

Motivation

- Previous models tend to have small memory
- RNNs can process a sequence but cannot accurately remembering things from the past (i.e. without compressing information into dense vectors)
- Therefore:
  - The author introduces memory network, which has a long-term memory component that can be read and written to
  - The network is good on tasks that required long sequence memorization (e.g. question answering with long text)

Outline

- Motivation
- What is a Memory Network
- MemNN: A special type Memory Network for text-based input
- Experiments

Memory Networks

- A memory network is a type of network consists of
  - $\boldsymbol{m}$: the memory
    - Represented as an array of “objects”
  - I: the input feature map
    - Converts the input to internal feature representation
  - G: generalization
    - Updates old memories given the new input
  - O: the output feature map
    - Produces the new output (in the feature representation space)
  - R: response
    - Converts the output into the response format

Memory Networks

- Given an input x, the flow of a memory network is as follows:
  1. Convert x to an internal feature representation $I(x)$
  2. Updates each entry of $\boldsymbol{m}$, given the new input: $m_i := G(m_i, I(x), \boldsymbol{m}) \forall i$
  3. Compute output feature o given the new input and the memory: $o = O(I(x), \boldsymbol{m})$
  4. Decode o, the output features, to produce the final response $r = R(o)$

Memory Networks

- **The $\boldsymbol{I}$ component**: Preprocessing (parsing/entity resolution), embeddings, etc.
- **The $\boldsymbol{G}$ component**: Decide which memories to update. The simplest form of G could be implemented as $m_{H(x)} = I(x)$, where H selects an index for a given input.
- **The $\boldsymbol{O}$ component**: Read from memory and perform inference (e.g. by retrieving relevant memories from the list)
- **The $\boldsymbol{R}$ component**: Produce the final response given O (e.g., generating answers based on the retrieved texts).

Memory Neural Networks (MemNN) for text

- A special type of memory network where the components are neural networks
- The basic version:
  - Assumes that the inputs are sentences
  - The text is stored in its original form in the next available memory slot
  - The *O* module retrieves *k* supporting memories:

$$o_1 = O_1(x, \mathbf{m}) = \underset{i=1,\ldots,N}{\arg\max}\; s_O(x, \mathbf{m}_i)$$

$$o_2 = O_2(x, \mathbf{m}) = \underset{i=1,\ldots,N}{\arg\max}\; s_O([x, \mathbf{m}_{o_1}], \mathbf{m}_i)$$

Memory Neural Networks (MemNN) for text

- The basic version:
  - Assumes that the inputs are sentences
  - The text is stored in its original form in the next available memory slot
  - The *O* module retrieves *k* supporting memories
  - The *R* module produce the textual response, e.g. if the answer is a single word:

$$r = \operatorname{argmax}_{w \in W} s_R([x, \mathbf{m}_{o_1}, \mathbf{m}_{o_2}], w)$$

$$s(x, y) = \Phi_x(x)^\top U^\top U \Phi_y(y).$$

where $\phi(\cdot)$ computes the mapping from the original text to the feature space, and U is a $n \times D$ matrix (n is the embedding dimension and D is the number of features)

Memory Neural Networks (MemNN) for text

- The basic version:
  - Assumes that the inputs are sentences
  - The text is stored in its original form in the next available memory slot
  - The *O* module retrieves *k* supporting memories
  - The *R* module produce the textual response
  - During training time, try minimize the loss:

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

where $\bar{f}$, $\bar{f}'$ and $\bar{r}$ are all other choices than the correct labels

Memory Neural Networks (MemNN) for text

- For word sequences as input: learn a segmentation function

$$seg(c) = W_{seg}^{\top} U_S \Phi_{seg}(c)$$

- To take into account *when* a memory slot was written to:

$$s_{O_t}(x, y, y') = \Phi_x(x)^{\top} {U_{O_t}}^{\top} U_{O_t} \Big(\Phi_y(y) - \Phi_y(y') + \Phi_t(x, y, y') \Big)$$

  - the dimensionality of $\phi_t(x, y, y')$ is extended by 3 to include three binary features: whether x is older than y, x is older than y', and y older than y'

- To handle previously unseen words: for each word we see, we store a bag of words it has co-occurred with, and do the matching by

$$\Phi_x(x)^{\top} U^{\top} U \Phi_y(y) + \lambda \Phi_x(x)^{\top} \Phi_y(y)$$

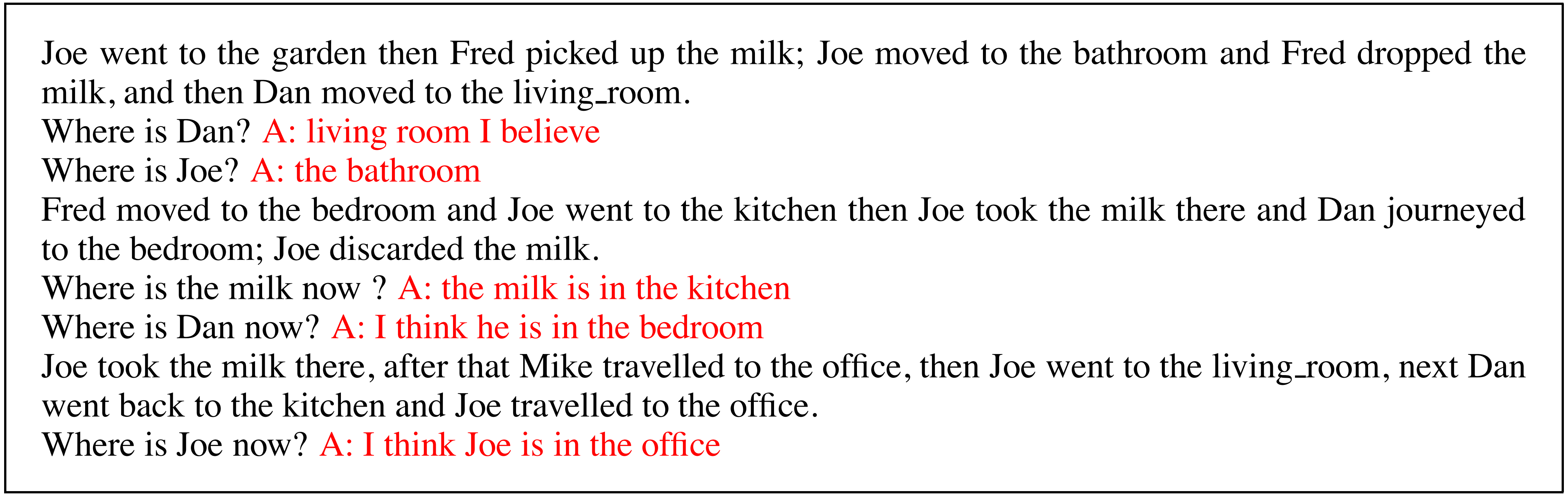
Experiments: Large-scale QA task (Fader et al, 2013)

- 14M Statements stored as (subject, relation, object) triples
- A MemNN with k = 1 supporting memory is used

Method	F1
(Fader et al., 2013)	0.54
(Bordes et al., 2014b)	0.73
MemNN (embedding only)	0.72
MemNN (with BoW features)	0.82

Experiments: Simulated World QA

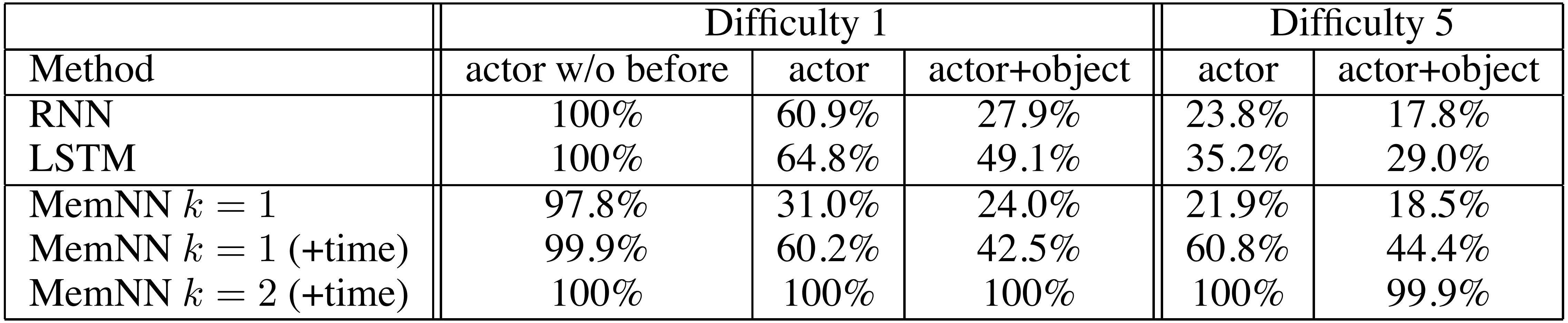
- A simple simulation of 4 characters, 3 objects and 5 rooms, with characters moving around, picking up and dropping objects
- Sample predictions on test set:



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

Experiments: Simulated World QA

- Compare to baselines:



	Difficulty 1			Difficulty 5	
Method	actor w/o before	actor	actor+object	actor	actor+object
RNN	100%	60.9%	27.9%	23.8%	17.8%
LSTM	100%	64.8%	49.1%	35.2%	29.0%
MemNN $k = 1$	97.8%	31.0%	24.0%	21.9%	18.5%
MemNN $k = 1$ (+time)	99.9%	60.2%	42.5%	60.8%	44.4%
MemNN $k = 2$ (+time)	100%	100%	100%	100%	99.9%

- The difficulty is controlled by the number of time steps in the past the entity being asked the question about was last mentioned.

Experiments: Simulated World QA

- MemNN is able to deal with previously unseen words at the test time by discovering simple linguistic patterns such as (X, dropped, Y), (X, took, Y) or (X, journeyed to, Y)

Bilbo travelled to the cave. Gollum dropped the ring there. Bilbo took the ring.
Bilbo went back to the Shire. Bilbo left the ring there. Frodo got the ring.
Frodo journeyed to Mount-Doom. Frodo dropped the ring there. Sauron died.
Frodo went back to the Shire. Bilbo travelled to the Grey-havens. The End.
Where is the ring? A: Mount-Doom
Where is Bilbo now? A: Grey-havens
Where is Frodo now? A: Shire

Experiments: Simulated World QA

- Taken a model trained on the Large Scale QA dataset and a model trained on the simulated world dataset and ensemble the two
- The combined model shows knowledge about the general world

Fred went to the kitchen. Fred picked up the milk. Fred travelled to the office.
Where is the milk ? A: office
Where does milk come from ? A: milk come from cow
What is a cow a type of ? A: cow be female of cattle
Where are cattle found ? A: cattle farm become widespread in brazil
What does milk taste like ? A: milk taste like milk
What does milk go well with ? A: milk go with coffee
Where was Fred before the office ? A: kitchen

Thank you