Learning to Query, Reason, and Answer Questions On Ambiguous Texts

Xiaoxiao Guo, Tim Klinger, Clemens Rosenbaum, Joseph P. Bigus, Murray Campbell, Ban Kawas, Kartik Talamadupula, Gerald Tesauro, Satinder Singh

University of Michigan, IBM Research, UMass Amherst

ICLR 2017
Presenter: Jack Lanchantin
Outline

1 Introduction
   • Background
   • This paper: QRAQ

2 QRAQ

3 Model
   • Control Loop
   • baseRL
   • impRL
   • Policy Gradient & Reward Function

4 Data, Training, and Results
Outline

1 Introduction
   • Background
     • This paper: QRAQ

2 QRAQ

3 Model
   • Control Loop
   • baseRL
   • impRL
   • Policy Gradient & Reward Function

4 Data, Training, and Results
Motivation

- Human conversation is incomplete, ambiguous and full of extraneous detail
- Conversational agents must be able to reason in the presence of missing or unclear info
Previous Datasets

- bAbI & children’s book: answer questions about short stories
- Task-oriented dialog systems: answer questions to find restaurants or movies (slot filling)
Outline

1 Introduction
   • Background
   • This paper: QRAQ

2 QRAQ

3 Model
   • Control Loop
   • baseRL
   • impRL
   • Policy Gradient & Reward Function

4 Data, Training, and Results
Simulator provides a story and a question to the agent, with some of the entities replaced by variables.
1. Simulator provides a story and a question to the agent, with some of the entities replaced by variables.

2. The agent must be able to decide whether it has enough info to answer the question.
QRAQ (Query, Reason, and Answer Questions) Dataset

1. Simulator provides a story and a question to the agent, with some of the entities replaced by variables.

2. The agent must be able to decide whether it has enough info to answer the question.

3. If the agent cannot answer the question by reasoning alone, it must learn to query the simulator for a variable value.
QRAQ Problem

C1. Hannah is in the garden.
C2. $u$ is Emma.
C3. $u$ is in the garden.
C4. The gift is in the garden.
C5. John is in the kitchen.
C6. The ball is in the kitchen.
C7. The skateboard is in the kitchen
E1. Hannah picks up the gift.
E2. John picks up $x$.
E3. $v$ goes from the garden to the kitchen.
E4. $w$ walks from the kitchen to the patio.
E5. Having left the garden, $u$ goes to the patio.

Q. Where is the gift?
GT. $v = \text{Hannah}; w = \text{Hannah}; \text{Answer} = \text{Patio}$

$C_1, C_2, \ldots$ : Context
$E_1, E_2, \ldots$ : Events
Q: Question
GT: Ground Truth
QRAQ Query Graph

C1. Hannah is in the garden.
C2. $u$ is Emma.
C3. $u$ is in the garden.
C4. The gift is in the garden.
C5. John is in the kitchen.
C6. The ball is in the kitchen.
C7. The skateboard is in the kitchen
E1. Hannah picks up the gift.
E2. John picks up $x$.
E3. $v$ goes from the garden to the kitchen.
E4. $w$ walks from the kitchen to the patio.
E5. Having left the garden, $u$ goes to the patio.
Q. Where is the gift?
GT. $v = \text{Hannah}; w = \text{Hannah}; \text{Answer} = \text{Patio}$

(a) for Example 1

Legend: Nodes represent Agent state = ($\mathcal{R}$, $\mathcal{A}$). 
$\mathcal{R}$ = relevant variables
$\mathcal{A}$ = possible answers to the challenge question.
Each arrow represents a variable query action (solid part) and observed outcomes (dashed part).
Multiple variables on a query-edge means that they have the same outcome set.
Double-bordered nodes have a unique challenge question answer.
A thick line denotes a ground-truth-path.
C1. Joe is in the kitchen.
C2. Bob is in the kitchen.
C3. Hannah is in the patio.
E1. $v$ goes from the kitchen to the garden.
E2. $w$ goes from the garden to the patio.
E3. $x$ goes from the patio to the basement.
Q. Where is Joe?
GT. $v = \text{Joe}; w = \text{Joe}; x = \text{Hannah}; \text{answer} = \text{Patio}$

(b) For Example 2

Answer abbreviations:
K: Kitchen, P: Patio, B: Basement
Introduction
- Background
- This paper: QRAQ

QRAQ

Model
- Control Loop
  - baseRL
  - impRL
  - Policy Gradient & Reward Function

Data, Training, and Results
Control Loop

1. **Initialization**: Question vector $c$ and memory matrix $S_0$ are initialized.
Control Loop

1. **Initialization**: Question vector $c$ and memory matrix $S_0$ are initialized.

2. **Action Selection**: Policy $\pi(a|S_t, c)$ maps memory matrix $S_t$ and question $c$, into a distribution over actions. Action $a$ could be either a query for a variable or a final answer.
Initialization: Question vector $c$ and memory matrix $S_0$ are initialized.

Action Selection: Policy $\pi(a|S_t, c)$ maps memory matrix $S_t$ and question $c$, into a distribution over actions. Action $a$ could be either a query for a variable or a final answer.

Variable Query and Memory Update: If action is a query, simulator provides the true value $v_t$ for the variable in action $a_t$. All occurrences of variable in action $a_t$ in the memory $S_t$ are replaced with the true value $v_t: S_{t+1} = S_t[a_t \rightarrow v_t]$.
Control Loop

1. **Initialization**: Question vector $c$ and memory matrix $S_0$ are initialized.

2. **Action Selection**: Policy $\pi(a|S_t, c)$ maps memory matrix $S_t$ and question $c$, into a distribution over actions. Action $a$ could be either a query for a variable or a final answer.

3. **Variable Query and Memory Update**: If action is a query, simulator provides the true value $v_t$ for the variable in action $a_t$. All occurrences of variable in action $a_t$ in the memory $S_t$ are replaced with the true value $v_t$: $S_{t+1} = S_t[a_t \rightarrow v_t]$

4. **Final Answer Generation and Termination**: If the action is an answer, task terminates and a reward is generated based on correctness.
Outline

1 Introduction
   • Background
   • This paper: QRAQ

2 QRAQ

3 Model
   • Control Loop
   • baseRL
   • impRL
   • Policy Gradient & Reward Function

4 Data, Training, and Results
Maps the memory matrix, $S$, and the challenge question representation, $c$, into an action distribution.

$S^{ij}$ is dictionary index of $j^{th}$ word in the $i^{th}$ sentence.

c$_i$ is the dictionary index of the $i^{th}$ word in the question.

$$m_i = \sum_j l_j \circ A[S^{ij}]$$  \hspace{1cm} (1)

$$q = \sum_j l_j \circ A[c_j]$$  \hspace{1cm} (2)

$A \in \mathbb{R}^{d \times N}$, $A[k]$ returns the $k$-th column vector (sentence), $d$ is embedding dimension, $N$ is the dictionary size, $l^k_j = (1 - j/J)(k/d)(1 - 2j/J)$, with $J$ being the number of words in the sentences.
\[ m_i = \sum_j l_j \circ A[S^{ij}] \] (3)

\[ q = \sum_j l_j \circ A[c_j] \] (4)

Output vector from reading \( \{m_i\} \) after the \( k^{th} \) hop is \( u_k \) (\( u_0 = q \)):

\[ u_k = \tanh(H(o_k + u_{k-1})) \] (5)

\[ o_k = \sum_i p_i^k m_i \] (6)

\[ p_i^k = \text{softmax}(u_{k-1}^T m_i) \] (7)

\[ (8) \]
Query Network Output. Since the problems have at most one variable per sentence, the distribution can be converted into the distribution over sentences:

$$\pi^i_Q = \text{softmax}(u_K^T m_i)$$  \hspace{1cm} (9)

Answer Network Output. The final output of the policy module is a distribution over potential answers:

$$\pi_A = \text{softmax}(Wu_K + b)$$  \hspace{1cm} (10)
baseRL $\rightarrow$ impRL

- **baseRL**: Final action-distribution output only conditions on the last hop output
- **impRL**: computes the final action-distribution-output over all memory hop outputs
Outline

1 Introduction
   • Background
   • This paper: QRAQ

2 QRAQ

3 Model
   • Control Loop
   • baseRL
   • impRL
   • Policy Gradient & Reward Function

4 Data, Training, and Results
**impRL: Improved End-to-End Memory Net Based Policy Learner**

\[
\pi_i^Q = \text{softmax}(u^T m_i) \tag{11}
\]
\[
\pi_A = \text{softmax}(Wu + b) \tag{12}
\]
\[
u = \sum_j z_j u_j \tag{13}
\]
\[
z_j = \text{softmax}(q^T u_j) \tag{14}
\]
impRL: Improved End-to-End Memory Net Based Policy Learner

(b) the memory logic

challenge question
e.g. “Where is Joe?”

sentences $s_t$:
context + events
e.g. “Joe is in the garden.”

“$v$ goes from …”

policy $\pi$

$W$

$\Sigma$

$z_i$

$u_0, u_1, u_2, u_3$

$\sigma$

$\{m_i\}$

softmax $p^1$

softmax $p^2$

softmax $p^3$

softmax $p^4$
Outline

1 Introduction
   • Background
   • This paper: QRAQ

2 QRAQ

3 Model
   • Control Loop
   • baseRL
   • impRL
     • Policy Gradient & Reward Function

4 Data, Training, and Results
Policy Gradient & Reward Function

- Reward when the action is the correct answer
- Reward when the action is a wrong answer
- Reward when the action is a query to a variable

Objective function is to optimize the expected cumulative reward over $M$ training problem instances

$$
\sum_{m=1}^{M} \mathbb{E}\left\{ \sum_{t} r^m_t \right\} 
$$

(15)

GPOMDP (Weaver & Tao (2001)) is used to calculate the policy gradient
Data

- 4 Types
- 107,000 QRAQ problems in each type
  - 100,000 training
  - 2,000 testing
  - 5,000 validation
Data

- **(Loc)**: Context and events describe locations and movements of people in rooms. Questions are about the location of a specific person.
- **(+obj)**: Adds objects to (Loc).
- **(+alias)**: Adds aliases to (Loc). Some of them are defined in the context.
- **(+par)**: Substitutes sentences with semantically equivalent paraphrases in (Loc).
Curriculum Learning

- First encourage the agent to query variables by assigning positive rewards for querying any variable.
- After convergence under this initial reward function, switch to the true reward function that assigns a negative reward for querying a variable to reduce the number of unnecessary queries.
- +1 for correct final answers, -5 for wrong final answers, query reward +/-0.005.
Supervised Learning (SL) Baseline

- Upper-bound on achievable reinforcement learning performance
- Relevant variables and (when appropriate) the correct final answers are provided at each turn, and the cross entropy is used to optimize
Accuracy Metrics

Trajectory: a sequence of variable queries followed by an answer

1. **Answer-accuracy**: proportion of trajectories in which the agent answered correctly

2. **Trajectory-accuracy**: proportion of trajectories in which the agent queried only relevant variables then answered correctly

3. **Trajectory-completeness**: proportion of trajectories in which the agent queried all and only relevant variables before answering correctly

4. **Query accuracy**: the proportion of correct queries among all queries made in any trajectory
Table 1: Datasets. The first 7 rows give statistics on the datasets themselves. The last 8 rows show results for answer accuracy (AnsAcc), trajectory accuracy (TrajAcc), trajectory completeness (TrajCmpl) and query accuracy (QryAcc) for the impRL and baseRL agents on the respective datasets. The middle 8 rows show results for the supervised learning agents.

<table>
<thead>
<tr>
<th>Data Set</th>
<th>(Loc)</th>
<th>(+obj)</th>
<th>(+alias)</th>
<th>(+par)</th>
</tr>
</thead>
<tbody>
<tr>
<td>#names in vocab</td>
<td>5</td>
<td>20</td>
<td>10</td>
<td>20</td>
</tr>
<tr>
<td>#var in vocab</td>
<td>5</td>
<td>20</td>
<td>10</td>
<td>20</td>
</tr>
<tr>
<td>#sentence/prob.</td>
<td>5-6</td>
<td>5-6</td>
<td>7-10</td>
<td>15-20</td>
</tr>
<tr>
<td>#var/prob.</td>
<td>0-2</td>
<td>0-2</td>
<td>0-2</td>
<td>0-3</td>
</tr>
<tr>
<td>depth</td>
<td>0-2</td>
<td>0-2</td>
<td>0-2</td>
<td>0-2</td>
</tr>
<tr>
<td>avg. depth</td>
<td>0.817</td>
<td>0.872</td>
<td>0.558</td>
<td>0.459</td>
</tr>
<tr>
<td>sum(depth) / sum(#var)</td>
<td>0.734</td>
<td>0.748</td>
<td>0.313</td>
<td>0.204</td>
</tr>
<tr>
<td>AnsAcc in %; impSL</td>
<td>99.9</td>
<td>99.5</td>
<td>92.1</td>
<td>95.3</td>
</tr>
<tr>
<td>AnsAcc in %; baseSL</td>
<td>99.9</td>
<td>99.2</td>
<td>92.3</td>
<td>92.4</td>
</tr>
<tr>
<td>TrajAcc in %; impSL</td>
<td>99.6</td>
<td>98.9</td>
<td>90.2</td>
<td>88.4</td>
</tr>
<tr>
<td>TrajAcc in %; baseSL</td>
<td>98.9</td>
<td>98.7</td>
<td>90.3</td>
<td>86.5</td>
</tr>
<tr>
<td>TrajCmpl in %; impSL</td>
<td>99.5</td>
<td>98.8</td>
<td>89.9</td>
<td>85.6</td>
</tr>
<tr>
<td>TrajCmpl in %; baseSL</td>
<td>98.7</td>
<td>98.7</td>
<td>90.0</td>
<td>83.5</td>
</tr>
<tr>
<td>QryAcc in %; impSL</td>
<td>99.5</td>
<td>99.2</td>
<td>96.4</td>
<td>84.6</td>
</tr>
<tr>
<td>QryAcc in %; baseSL</td>
<td>98.7</td>
<td>99.3</td>
<td>96.3</td>
<td>85.5</td>
</tr>
<tr>
<td>AnsAcc in %; impRL</td>
<td>99.1</td>
<td>94.4</td>
<td>86.5</td>
<td>89.0</td>
</tr>
<tr>
<td>AnsAcc in %; baseRL</td>
<td>98.4</td>
<td>95.0</td>
<td>88.4</td>
<td>88.2</td>
</tr>
<tr>
<td>TrajAcc in %; impRL</td>
<td>94.5</td>
<td>90.9</td>
<td>61.9</td>
<td>52.0</td>
</tr>
<tr>
<td>TrajAcc in %; baseRL</td>
<td>94.8</td>
<td>90.4</td>
<td>63.6</td>
<td>52.5</td>
</tr>
<tr>
<td>TrajCmpl in %; impRL</td>
<td>94.5</td>
<td>88.7</td>
<td>55.8</td>
<td>46.9</td>
</tr>
<tr>
<td>TrajCmpl in %; baseRL</td>
<td>94.6</td>
<td>89.5</td>
<td>59.9</td>
<td>47.4</td>
</tr>
<tr>
<td>QryAcc in %; impRL</td>
<td>94.3</td>
<td>95.4</td>
<td>49.2</td>
<td>32.1</td>
</tr>
<tr>
<td>QryAcc in %; baseRL</td>
<td>95.5</td>
<td>94.1</td>
<td>54.6</td>
<td>32.0</td>
</tr>
</tbody>
</table>
Conclusion

- New dataset, QRAQ, for reasoning under insufficient information
- First to formulate these types of QA problems in the RL format