Deep Memory-based Architectures for Reasoning

Zhengdong Lu
Noah’s Ark Lab
http://luz-id.com/
Deep Memory-based Architecture

Executor-1

input
Deep Memory-based Architecture

Executor-1

Memory Layer-1

write

input
Deep Memory-based Architecture

Executor-1

Memory Layer-1

Executor-2

input

write

read
Deep Memory-based Architecture

Executor-1 → Memory Layer-1

Executor-2 → Memory Layer-2

input

read

write
Deep Memory-based Architecture

- Executor-1
  - Memory Layer-1
  - write
- Executor-2
  - Memory Layer-2
  - read
- Executor-3
  - input

Diagram shows the flow of data between executors and memory layers, with 'write' and 'read' operations indicated.
Deep Memory-based Architecture

Executor-1 \rightarrow Memory Layer-1

Executor-2 \rightarrow Memory Layer-2

Executor-3 \rightarrow Memory Layer-3

input

write

read
Deep Memory-based Architecture

prediction

↑

Executor-4

Executor-3

Executor-2

Executor-1

Memory Layer-3

Memory Layer-2

Memory Layer-1

input

read

write
Deep Memory-based Architecture

In a sense

- We can take anything as the input
  - Sentences, Tables, logic forms, etc
  - Symbolic form, “embedding”, or both

input

write

read

prediction

Executor-1 → Memory Layer-1

Executor-2 → Memory Layer-2

Executor-3 → Memory Layer-3

Executor-4
Deep Memory-based Architecture

In a sense

- We can take anything as the input
- We can do anything with the executor
  - N.T.M. style read-write
  - Anything problem-specific design
Deep Memory-based Architecture

In a sense
- We can take anything as the input
- We can do anything with the executor
- We can save anything in the memory
  - structured/unstructured, half-processed, hybrid forms, interpretable/uninterpretable
In a sense
• We can take anything as the input
• We can do anything with the executor
• We can save anything in the memory
Deep Memory-based Architecture

In a sense
• We can take anything as the input
• We can do anything with the executor
• We can save anything in the memory

as long as
• the dependency/transformation between memory layers is differentiable.

since
• any “discrete decision” often requires more inefficient optimization, such as RL
In a sense
- We can take anything as the input
- We can do anything with the executor
- We can save anything in the memory

- Semantic Parsing & Table Querying (arxiv:1512.00965)
- Reasoning (arXiv:1508.05008)
Neural Reasoner
Neural Reasoner

Input
- Varying number of facts \( \{F_1, F_2, \ldots, F_K\} \)
- A question \( Q \)
- Not necessarily the same form

Output
- Answer to the question

\[
\{ Q, \ F_1, \ F_2, \ldots, \ F_K \}
\]
First Layer: Embedding Q and F

Encoders

- **RNN** (e.g., GRU) to encode question and facts to a fixed length vectors
  
  \[
  \{Q, F_1, F_2, \ldots, F_K\} \rightarrow \{Q^{(0)}, F_1^{(0)}, F_2^{(0)}, \ldots, F_K^{(0)}\}
  \]

- Many other choices, e.g., different encoders for questions and facts
Second Layer: Reasoning

- Interaction between Q and F
  - Analogous to resolution and sometimes reverse resolution
  - Generate \{Q_1, F_1, Q_2, F_2, \ldots, Q_K, F_K\}

- Pooling
  - Summarization step
  - Can be extended to include temporal information and gating
  - Acts something similar to the selection in, say, SLD resolution
Interaction between Q and F

- Analogous to resolution, but could be richer
- Generate \( \{ Q_1^{(1)}, F_1^{(1)}, Q_2^{(1)}, F_2^{(1)}, \ldots, Q_K^{(1)}, F_K^{(1)} \} \)
Second Layer: Reasoning

Interaction between Q and F

- Analogous to resolution, but could be richer
- Generate \( \{ Q_1^{(1)}, F_1^{(1)}, Q_2^{(1)}, F_2^{(1)}, \ldots, Q_K^{(1)}, F_K^{(1)} \} \)

Pooling

- Summarization step \( \{ Q_1^{(1)}, Q_2^{(1)}, \ldots, Q_K^{(1)} \} \rightarrow Q^{(1)} \)
- Can be extended to include temporal information and gating
- Acts something similar to the selection in, say, SLD resolution
Same components as the second layer, but different parameters

- We can have more reasoning layers
- Not entirely sure whether we should share parameters across layers
more layers if you want
Generating answers

- After pooling, $Q^{(L)}$ gives us the final status of the reasoning
- Can use $Q^{(L)}$ to generate answers of different forms, including NL sentences
The whole architecture
Learning

- End-to-end training, with auxiliary reconstruction tasks for the embedding layer
bAbi data

- **Task:** reasoning with multiple support facts

**Task I: path finding**

1. The office is east of the hallway.
2. The kitchen is north of the office.
3. The garden is west of the bedroom.
4. The office is west of the garden.
5. The bathroom is north of the garden.

How do you go from the kitchen to the garden? **s,e**, relies on 2 and 4
How do you go from the office to the bathroom? **e,n**, relies on 4 and 5

**Task II: positional reasoning**

1. The triangle is above the pink rectangle.
2. The blue square is to the left of the triangle.

Is the pink rectangle to the right of the blue square? **Yes**, relies on 1 and 2
Is the blue square below the pink rectangle? **No**, relies on 1 and 2
Results on bAbi

- Ours is way better than other neural models with end-to-end learning
- Actually ours is even better than them with strong supervision, when there are enough training data

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step-by-step Supervision</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MEMORY NET-step</td>
<td>65.0%</td>
<td>75.4%</td>
</tr>
<tr>
<td>DYNAMIC MEMORY NET</td>
<td>59.6%</td>
<td>-</td>
</tr>
<tr>
<td><strong>End-to-End</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MEMORY NET-N2N</td>
<td>59.6%</td>
<td>60.3%</td>
</tr>
<tr>
<td>NEURAL REASONER</td>
<td><strong>66.4%</strong></td>
<td><strong>97.9%</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Path Finding (1K)</th>
<th>Path Finding (10K)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step-by-step Supervision</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MEMORY NET-step</td>
<td><strong>36.0%</strong></td>
<td>68.1%</td>
</tr>
<tr>
<td>DYNAMIC MEMORY NET</td>
<td>34.5%</td>
<td>-</td>
</tr>
<tr>
<td><strong>End-to-End</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MEMORY NET-N2N</td>
<td>17.2%</td>
<td>33.4%</td>
</tr>
<tr>
<td>NEURAL REASONER</td>
<td><strong>17.3%</strong></td>
<td><strong>87.0%</strong></td>
</tr>
</tbody>
</table>
Trying …

Given:

\[ \exists x \text{ IsPig}(x) \rightarrow \exists y \text{ IsFat}(y) \]
\[ \forall x (x \text{ CanFly}(x) \rightarrow \neg \text{IsCat}(x)) \]
\[ \forall x (x \text{ IsBat}(x) \rightarrow \text{CanFly}(x)) \]
\[ \text{IsBat}(\text{Jack}) \]

Find out:

\[ \text{CanFly}(\text{Jack}) \]

**Neural Reasoner** can get about 95% cases correct, but too early for any conclusion.
Remaining Questions

• **Does Neural Reasoner do reasoning like Prolog?**
  – Probably NOT if trained in the end-to-end fashion, but more likely so if tutored in a curriculum learning setting

• **Can Neural Reasoner handle variables?**
  – Only to some level, and pessimistic about that in this DNN-like structure

• **Symbolic behavior?**
  – Some at least, e.g. in a related model, the system can learn to treat entity-names almost as symbols

• **Scalability?**
  – Suppose we have 10K facts, but only 5 of them are relevant
Thank You