Neural Symbolism: a few thoughts and trials

Zhengdong Lu

What is Neural Symbolism?

- Basically, Neural Symbolism means explicitly modeling symbolic behavior in neural models, and being proud of it

as opposed to

➊ Believing in "thought vector"

➋ Letting the data drive everything (probably in an end-to-end way) and gives up the interpretability {we were here}

➌ Using content-based addressing to access memory

➍ Thinking Markov Logic Net as the only/right way to soften logic

Example-1: semantic representation

Find representation for

"I met a pretty girl at bar last night, and her number is 555-503-2665"

- The soft/"continuous" part:
  - I met a girl at bar, she is pretty, we really hit it off, I got her number …
- The hard/symbolic part:
  - Her number is *555-503-2665* (you don't want this part to be fuzzy) , and you probably memorized it digit-by-digit (as symbols)

Example-II: "execute" a natural language query

Query: "Where was the Olympic game held after Beijing?"

To execute it against a table, we need to decompose it into the following (discrete) operations

❶ Find the row whose field_city = "`Beijing`"

❷ Find the value of field_year of that row, save it as YEAR

❸ Filter out all the rows whose field_year <= YEAR

❹ In the rows that are left, find the one whose field_year is the smallest

❺ Output the value of field_city of that row

Example-III: teach NN rules

Suppose we want to teach a neural dialogue system rules like the following

[Rule-0001] when the user says "hi/hello", you should say "hello, what can I do for you?"

[Rule-0002] when the user's first line includes "My name is X ", you should say "Nice to meet you ,X. What can I do for your"

…

[Rule-5238] when the user mentions "It", it refers to the closest entity-name for nonliving thing

…

It has the following characteristics

❶ It may contain variables

❷ It can be violated

❸ Rules can be combined

(Why we need) Neural Symbolism

- Symbolic behavior is ubiquitous in the information processing of human brain, and considered to be important to human intelligence
- It is however hard to be "implemented" in the current neural NLP models, which is based on distributed & continuous representation of everything

From two places we can seek the solution

- Our own brain (a tremendous neural net)
- Many old school models for NLP, e.g., statistical machine translation

Our surface-scratching explorations:

- Representation:
  - How to mix symbols with distributed representation
  - How to represent discrete structures in distributed fashion
- Operation, two possible paths
  - How to realize basic symbolic operation in a neural system
  - How to let a neural system coordinate symbolic operations
- Learning:
  - Effective way to learn the representation and operation

Some of our work/ideas in this view

Representation

- Hybrid representation of symbols and embedding
  - The hard way: Neural generative QA, Phrase-Net
  - The soft way: CopyNet
- Representing concepts (I don't have much an idea), but it is related to

Interaction with discrete data structures

- Neuralizing symbolic operations
  - Neural Enquirer
- Seeking representation with discrete structure
  - Neural Semantic Parser (a proposal)

Interaction with rules ("abstract" symbolic form)

- Taking rules in symbolic form (rules in)
  - Rule Assimilation (ongoing)
- Extracting rules from data (rules out)
  - Neural Inductive Programming (a proposal)

Several key questions

① Can symbolic behavior be effectively implemented with an entirely soft model (as opposed to containing some hard ingredients)?

- Generalization issues
- Efficiency issues

② If the answer to ① is YES, is BP enough to learn the model?

- Since you guys think our brain doesn't do it

③ If the answer to ① is NO, is sampling-based method unavoidable?

- We can struggle to make it differentiable, but is that worth it?

④ If the answer to ③ is YES, are the current RL methods (policy search, Q-learning, actor-critic, etc) enough?

- Could be facing exceedingly large action space and long-delay
- Some new hybrid learning paradigm?

Case study: Neural Enquirer

Neural Enquirer (Yin et al., 2015)

- A end-to-end model that learn to execute a natural language query against a structured knowledge-base (Table)
- Fully neural model that does symbolic stuff

[Yin et al., 2015] Neural Enquirer: Learning to Query Tables with Natural Language. Pengcheng Yin, Zhengdong Lu, Hang Li, Ben Kao

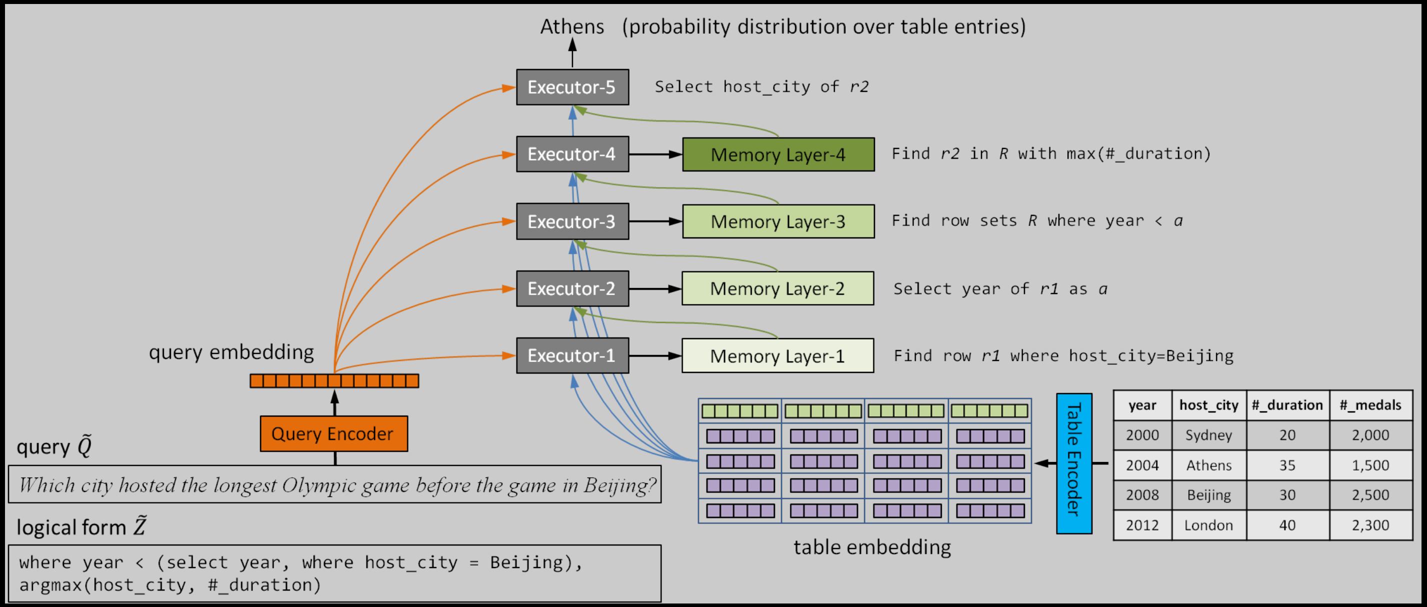
Neural Overview

Natural language query to find answers from a knowledge-base (one or more KB tables)

① Fully neuralized (representation, execution etc), everything (tables, query sentences, etc) is embedded

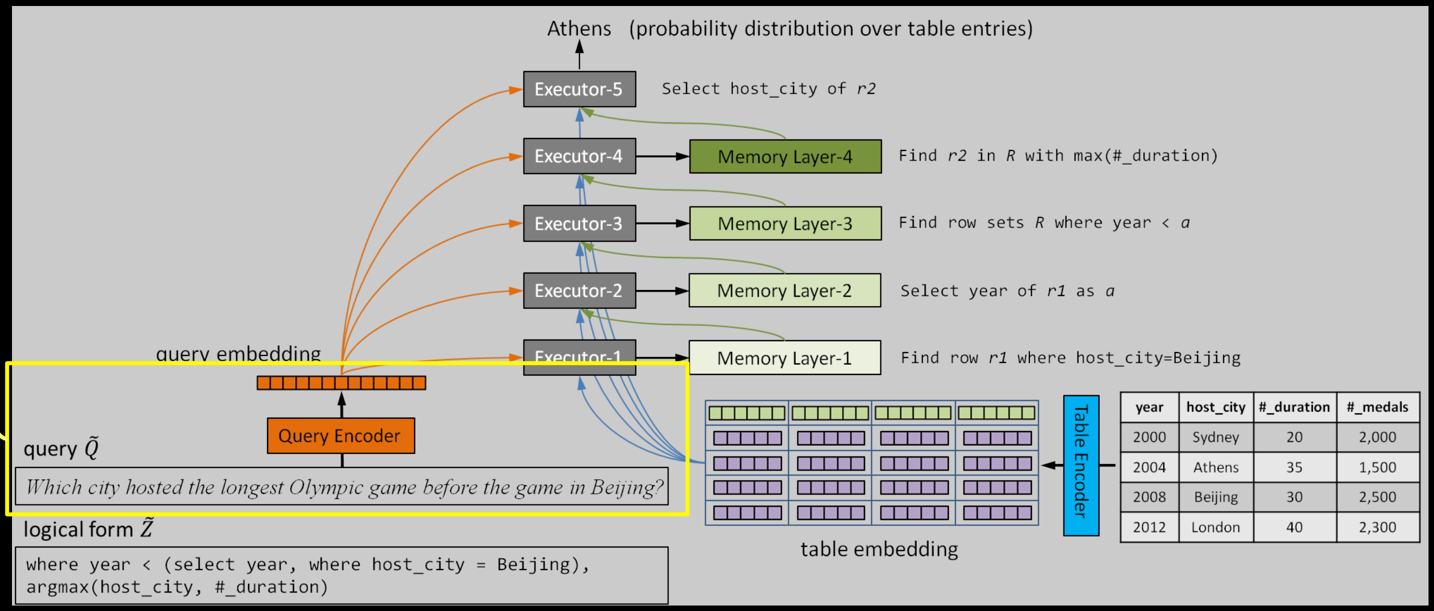
② End-to-end training, although extra step-by-step supervision does help

③ It can take out-of-vocabulary entity names (eg., "Istanbul") without specific design, an interesting emerging symbolic behavior



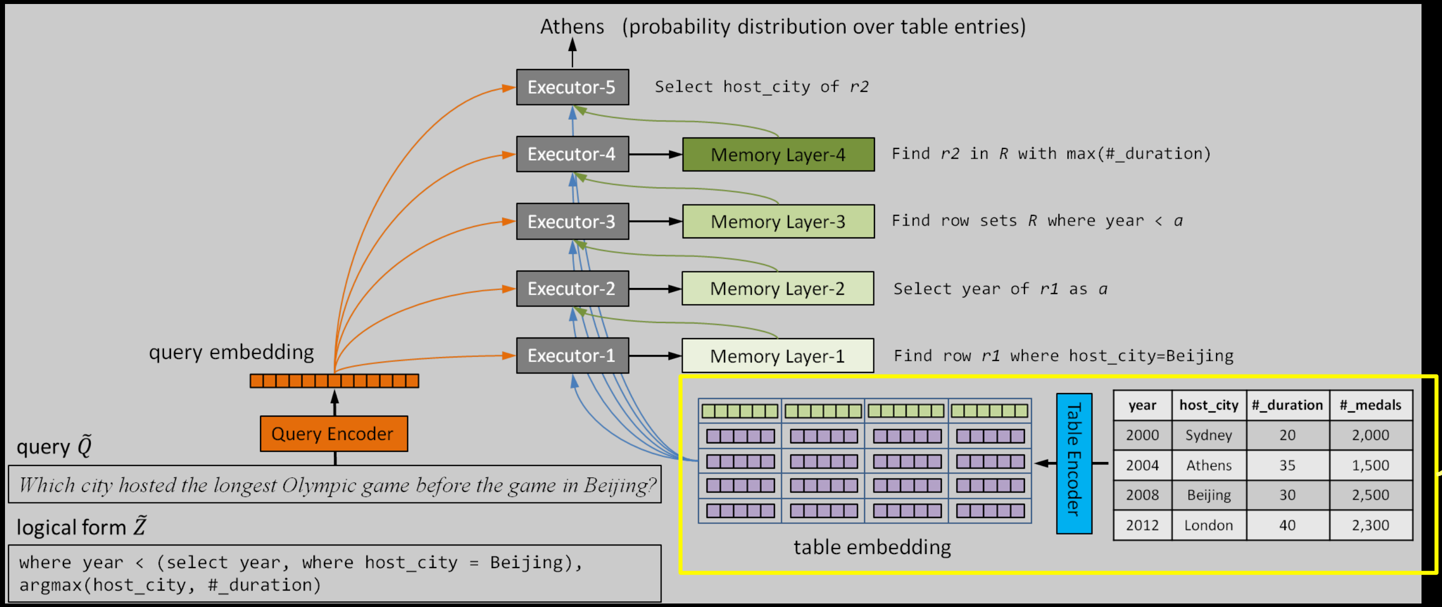
Neural Enquirer: Architecture

Query Encoder

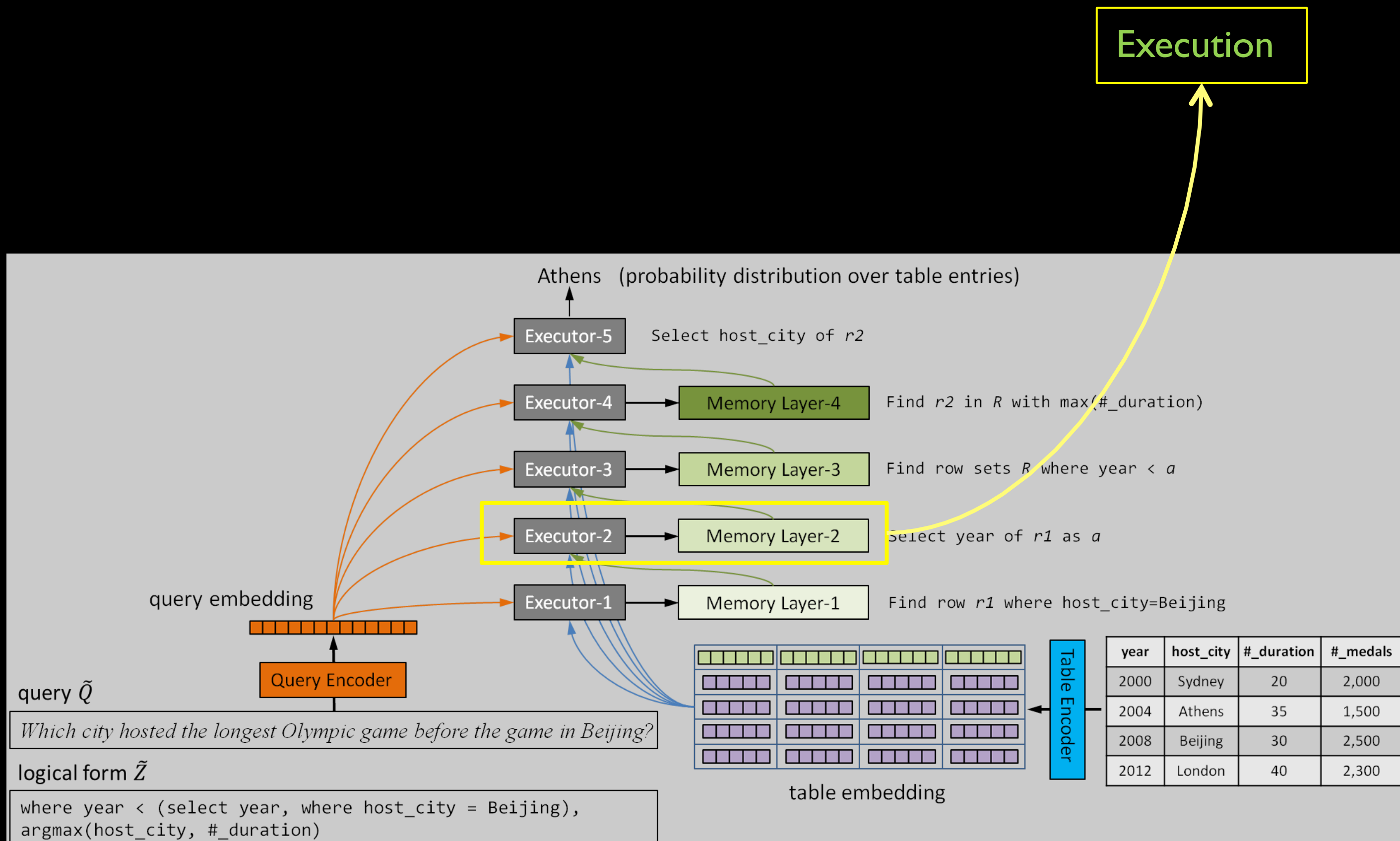


Neural Enquirer: Architecture

Table Encoder



Neural Enquirer: Architecture



Neural Enquirer: Execution

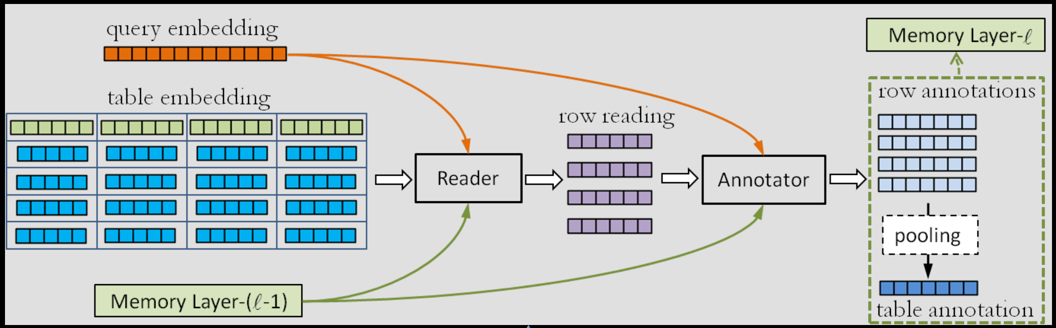
Each execution consists of two steps

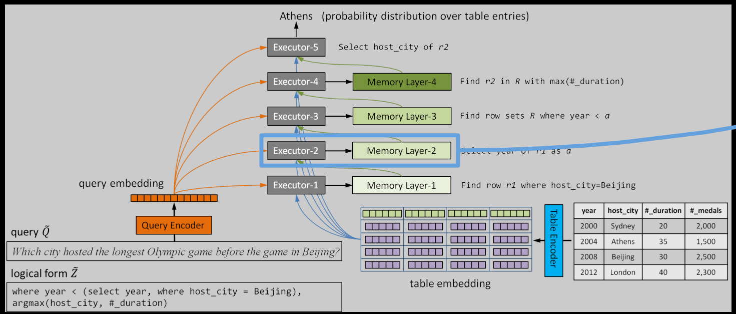
① **Reading :** for each row, find the value of the "right" field through attentive reading

② **Annotation:**

- ① each row reading will be transform into row annotation, with the information in previous layer of memory
- ② Pool the row annotation into table annotation

Save row and table annotation in this layer of memory



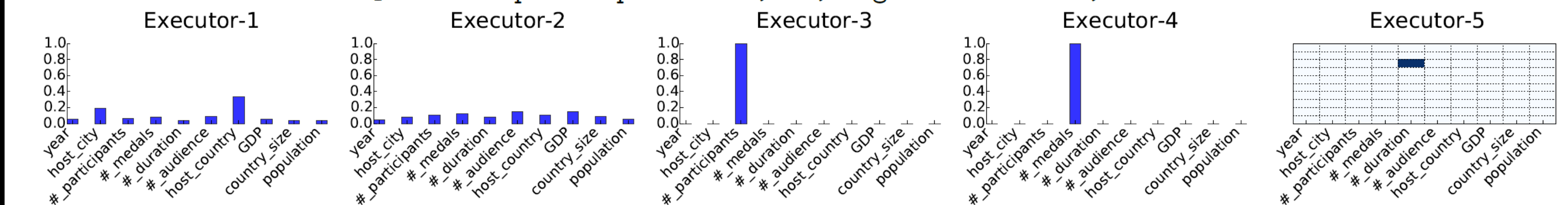


Some case study

- A relatively simple query

Q_1: *How long is the game with the most medals that has fewer than 3,000 participants?*

Z_1: `where #_participants < 3,000, argmax(#_duration, #_medals)`



- A relatively complex query

Q_2: *Which country hosted the longest game before the game in Athens?*

Z_2: `where year<(select year,where host_city=Athens),argmax(host_country,#_duration)`

Quantitative Results

- Kinda toyish data, but big and hard enough

	MIXTURED-25K				MIXTURED-100K		
	SEMPRE	N2N	SbS	N2N-OOV	N2N	SbS	N2N-OOV
SELECT_WHERE	93.8%	96.2%	99.7%	90.3%	99.3%	100.0%	97.6%
SUPERLATIVE	97.8%	98.9%	99.5%	98.2%	99.9%	100.0%	99.7%
WHERE_SUPERLATIVE	34.8%	80.4%	94.3%	79.1%	98.5%	99.8%	98.0%
NEST	34.4%	60.5%	92.1%	57.7%	64.7%	99.7%	63.9%
Overall Acc.	65.2%	84.0%	96.4%	81.3%	90.6%	99.9%	89.8%

Another way of doing similar thing

Neural Programmer (Neelakantan et al., 2015)

- Predefined symbolic operations with distributed embedding
- A controlling neural network that learns to activate a certain operation at a particular time
- Can be made differentiable by using weighted sum of the result of symbolic operations as intermediate representation
- In testing, it can take only the operation with maximum likelihood (kinda like an action), therefore fairly efficient

[Neelakantan et al. 2015] Neural Programmer: Inducing Latent Programs with Gradient Descent. Arvind Neelakantan, Quoc V. Le, and Ilya Sutskever.

Both models suck

① What is difficult for Neural Enquirer? (i.e. having a fully NN-based model)

- Generalization problem

"what state ranks 3rd in area in U.S.A.?" is still very difficult after we solve "what state rank 1st in area in U.S.A. ?" and "what state ranks 2nd in area in U.S.A.?"

② What is difficult for Neural Programmer? (i.e. a differentiable data-structure with symbolic atomic operations)

- Finding a path of representation for the back-propagation when the operations involve comparing multiple instances. e. g. , argmax( )

  "what state ranks 1st in area in U.S.A.?"

Both models suck

① What is difficult for Neural Enquirer? (i.e. having a fully NN-based model)

- Generalization problem

`“what state ranks 3rd in area in U.S.A.?”` is still very difficult after we solve “what state rank 1st in area in U.S.A. ?” and `“what state ranks 2nd in area in U.S.A.?”`

② What is difficult for Neural Programmer? (i.e. a differentiable data-structure with symbolic atomic operations)

- Finding a path of representation for the back-propagation when the operations involve comparing multiple instances. e. g. , argmax( )

  `“what state ranks 1st in area in U.S.A.?”`

Shall we give up BP and just do RL? Probably not entirely

Two-view Neural Symbolic Models (Rescue?)

Ongoing, actually just started

① We are sort of dropping our love of end-to-end model

② We try to exploit the correspondence between the symbolic view and the distributed view of the execution, just like in multi-view learning

③ It has to combine unsupervised learning, reinforcement learning, and supervised learning

Thanks