Learning to Understand Questions and Organize Knowledge

Ni Lao

8.25.2021
Speaker Background

Research Scientist @Apple
- Web Answers, KG Answers

Chief Scientist @mosaix.ai
- Created the AI platform team for NLP/ML/quality infrastructures
- Research and publications

Research Scientist @Google
- Model based KG construction/cleaning
- Semantic parsing for KG QA
- Online/offline methods for Web QA

PhD (ML) @Carnegie Mellon U.
- Large scale inference on KG
- Relational/structure learning
- IR, NLP, QA

MS (CS) BS (EE) @Tsinghua U.
- Automatic system diagnosis based on IR and data mining
- Web search, Product search
- World champion of RoboCup simulation league (2001 & 2002)
Plan

• **Query understanding**
  o Weak supervision semantic parsing tasks
  o Neural Symbolic Machines
    ■ Symbolic representations for efficient inference
  o Memory Augmented Policy Optimization
    ■ Unbiased low-variance gradient estimation with experience replays
    ■ RL vs MML vs ML
  o Reranking for RL trained decoders
    ■ Sequence scorer, Stacked Learning, Scorer Ensembles

• **Document understanding**
  o Generalizable, yet accountable & scalable
  o The return of inverted lists
  o Experiment with n-gram machines
Weak Supervision Query Understanding

- Reducing the knowledge acquisition bottleneck with ML is challenging
- Both ML and data labeling/acquisition are central to system design

end to end examples
(QA pairs, preferences)

machine learning

intelligent systems
with knowledge
WebQuestionsSP

- 5,810 questions from Google Suggest API & Amazon MTurk
- Answerable through FreeBase, a large open knowledge graph (KG)
- 3,098 training examples, 1,639 testing examples remaining
- Open-domain and contains grammatical error
- Multiple entities as answer => macro-averaged F1

[Berant et al, 2013; Yih et al, 2016]

- What do Michelle Obama do for a living?  
  writer, lawyer
- What character did Natalie Portman play in Star Wars?  
  Padme Amidala
- What currency do you use in Costa Rica?  
  Costa Rican colon
- What did Obama study in school?  
  political science
- What killed Sammy Davis Jr?  
  throat cancer
**WikiTableQuestions**

**Breadth**
- No fixed schema: Tables at test time are not seen during training, needs to generalize based on column name.

**Depth**
- More compositional questions, thus require longer programs
- More operations like arithmetic operations and aggregation operations

---

<table>
<thead>
<tr>
<th>Year</th>
<th>City</th>
<th>Country</th>
<th>Nations</th>
</tr>
</thead>
<tbody>
<tr>
<td>1896</td>
<td>Athens</td>
<td>Greece</td>
<td>14</td>
</tr>
<tr>
<td>1900</td>
<td>Paris</td>
<td>France</td>
<td>24</td>
</tr>
<tr>
<td>1904</td>
<td>St. Louis</td>
<td>USA</td>
<td>12</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>2004</td>
<td>Athens</td>
<td>Greece</td>
<td>201</td>
</tr>
<tr>
<td>2008</td>
<td>Beijing</td>
<td>China</td>
<td>204</td>
</tr>
<tr>
<td>2012</td>
<td>London</td>
<td>UK</td>
<td>204</td>
</tr>
</tbody>
</table>

- $x_1$: "Greece held its last Summer Olympics in which year?"
  - $y_1$: \{2004\}
- $x_2$: "In which city’s the first time with at least 20 nations?"
  - $y_2$: \{Paris\}
- $x_3$: "Which years have the most participating countries?"
  - $y_3$: \{2008, 2012\}
- $x_4$: "How many events were in Athens, Greece?"
  - $y_4$: \{2\}
- $x_5$: "How many more participants were there in 1900 than in the first year?"
  - $y_5$: \{10\}
Domain Specific Languages

- An interpreter for LISP like syntax
  - Program => exp₁ exp₂ ... expₙ <END>
  - Exp => (f arg₁ arg₂ ... argₙ)

- Functions
  - 4 operations for WebQuestionsSP
    - hop, argmax, argmin, filter
  - 22 different operations for WikiTableQuestions
    - hop, argmax, argmin
    - filter₁, filter₂, filter₃, filter₄, filter₅, filter₆, filter₇, filter₈, filter₉, filter₁₀, filter₁₁, filter₁₂, filter₁₃, filter₁₄, filter₁₅, filter₁₆, filter₁₇, filter₁₈, filter₁₉, filter₂₀
    - first, last, previous, next
    - max, min, average, sum, mode, diff, same

(Liang+ 2017)
Semantics as A Foreign Language

Largest city in US?

GO
(Hop V1 CityIn)
(Argmax V2 Population)
RETURN

NYC

Paraphrase
Many ways to ask the same question, e.g.,
“What was the date that Minnesota became a state?”
“When was the state Minnesota created?”

Compositionality

Large Search Space
E.g., Freebase:
23K predicates,
82M entities,
417M triplets

Optimization
Directly optimizing the metric (e.g., F1) with RL faces challenges
Can this KG pattern be learned?

Understanding Needs Reasoning

Marge  Homer
Gender  Gender

Parent

Bart Simpson

Bart's father is Homer
Reasoning May Incur A Lot of Computation

“impressionist painters during the 1920s”

impressionist <visual_artist> x.[/associated_periods_or_movements = /impressionism]

<artist> during the 1920s x.[/date_of_work < 1930; /date_of_work > 1920]
Symbolic Computations & Neural Nets

- Combine the **learnability** of neural nets with the **efficiency** of symbolic reasoning
Neural Symbolic Machines

Manager

Question

Answer

Programmer

Program

Results

Interpreter

Knowledge Base

Predefined Functions

Weak supervision

Neural Net

Symbolic

Low cost
User centric

Learning power

Abstract
Scalable
Precise
Non-differentiable

[Liang+ 2017]
- Equivalent to a linearised bottom-up derivation of the recursive program
- Aggressive pruning by code assists
Code Assistance with An Interpreter

Pen and paper

IDEs do a lot of computations!
Code Assistance: Syntactic Constraint

Decoder Vocab

Details

Last token is ‘(’, so has to output a function name next.

Softmax

Variables: <10

Functions: <10

Predicates: 23K

[Liakakos+ 2017]
Given definition of $\text{Hop}$, need to output a predicate that is connected to $R_2$ ($m \cdot \text{USA}$).

Decoder Vocab

- Variables: <10
- Functions: <10
- Predicates: 23K
- Valid Predicates: <100

[Details]

[Liang+ 2017]
Directly Optimizing The Expected Reward with RL

- **ML** optimizes the log likelihood of target sequences
  \[
  J_{ML}^{ML}(\theta) = \sum_q \log P(a^{best}_{0:T}(q)|q, \theta)
  \]

- **RL** optimizes the expected reward under a stochastic policy
  \[
  J_{RL}^{RL}(\theta) = \sum_q \mathbb{E}_{P(a_{0:T}|q,\theta)}[R(q,a_{0:T})]
  \]

[Sutton & Barto 1998, 2018]
Augmented REINFORCE

- Iterative ML training is not directly optimizing the F1 scores
- REINFORCE get stuck at local maxima
- Augmented REINFORCE obtains better performances
  - but the objective is biased

WebQuestionsSP Results

<table>
<thead>
<tr>
<th>Settings</th>
<th>Train Avg. F1@1</th>
<th>Valid Avg. F1@1</th>
</tr>
</thead>
<tbody>
<tr>
<td>iterative ML only</td>
<td>68.6</td>
<td>60.1</td>
</tr>
<tr>
<td>REINFORCE only</td>
<td>55.1</td>
<td>47.8</td>
</tr>
<tr>
<td>Augmented REINFORCE</td>
<td>83.0</td>
<td>67.2</td>
</tr>
</tbody>
</table>

[Liang+ 2017]
Augment REINFORCE

Linear combination of ML and RL objective:

- Converges fast to a reasonable policy
- The gradient is biased -- not robust against spurious programs

\[ \lambda \sum_{y \in \text{TopK}} \log p(y \mid x) + (1 - \lambda) \mathbb{E}_{\tilde{y} \sim p(y \mid x)} R(\tilde{y}) \]

- highest rewarded solutions in memory
- new samples from the current policy

[Liang+ 2016] [Abolafia+ 2017]
We can get the right answer with many wrong reasons

- **Correct program:**
  (argmax rows “Silver”)
  (hop v1 “Nation”)  

- **Many spurious programs:**
  (argmax rows “Gold”)
  (hop v1 “Nation”)
  
  (argmax rows “Bronze”)
  (hop v1 “Nation”)
  
  (argmin rows “Rank”)
  (hop v1 “Nation”)
...
The RL objective demotes spurious rewards

- **Reinforce** a *rewarded experience* only if the model (current policy) also thinks that it is the right thing to do

Correctness of the behavior

Experiences (programs)

Model’s prior (probabilities)

Reinforce (gradients)

[Details]

[Question 1]

[Question 2]

[Liang+ 2018]
RL models generate their training data on the fly

- Training sample management issue
  - Large search space & sparse reward lead to slow and unstable training
  - Spurious reward lead to biased solutions

[Liang+ 2018]
Memory Augmented Policy Optimization (MAPO)

[Image of a diagram showing the components of MAPO: Memory, Actor, Learner, Model checkpoint, Systematic Exploration, and a question mark indicating surprising return.]

[Liang+ 2018]
Most of the past experience are not helpful for improving the current model
Optimal Sample Allocation

- The optimal strategy (low variance in gradient estimations) is to allocate the same number of samples to reward vs no reward experiences (0-1 reward)

- and this is independent of the model's current performance

Image source: Guy Harris, 2018
How to Give Feedback in a Non-Threatening Way
Unbiased gradient estimation with low variances

Given a memory buffer of high return sequences \( \mathcal{B} \equiv \left\{ (y^{(i)}, r^{(i)}) \right\}_{i=1}^{n} \), re-express expected return as,

\[
p(\mathcal{B}) \underbrace{\mathbb{E}_{p(\tilde{y}) | \tilde{y} \in \mathcal{B}} R(\tilde{y})}_{\text{inside the buffer}} + (1 - p(\mathcal{B})) \underbrace{\mathbb{E}_{p(\tilde{y}) | \tilde{y} \notin \mathcal{B}} R(\tilde{y})}_{\text{outside the buffer}}
\]

- For each query
  - Sampling 1 solution from inside the buffer according to model
  - Rejection sampling 1 solution from outside the buffer according to model

[Liange+ 2018]
Comparison

- REINFORCE does not work at all
- MAPO is slower but less biased than max marginal likelihood and hard EM

The shaded area represents the standard deviation of the dev accuracy
The Issues with Sequence Probabilities

- Cannot consider information in the future; or global statistics
- The label bias problem of sequential models
  - states with limited choices effectively ignore their observations

MAPO probability per token for programs in beam. The score sequences have the same length (10) because of padding
### Leverage Global Discriminative features

[Biloki+ 2019]

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Type</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>$q^{\text{tok}}$</td>
<td>binary</td>
<td>The program token matches any of the question tokens.</td>
</tr>
<tr>
<td>$q^{\text{attn}}$</td>
<td>float vector</td>
<td>Softmax attention over query tokens per program token</td>
</tr>
<tr>
<td>$p^{\text{prob}}$</td>
<td>float</td>
<td>Program probability according to the search policy $\pi_\phi$</td>
</tr>
<tr>
<td>$t^{\text{prob}}$</td>
<td>float</td>
<td>Program token probability according to the search policy $\pi_\phi$</td>
</tr>
<tr>
<td>$t^{\text{agree}}$</td>
<td>count</td>
<td>Number of candidate programs having token $a_t$ at position $t$</td>
</tr>
</tbody>
</table>

NPP scores per token for a set of candidate programs in beam
Sum of token scores

Which programming is played the most?

Beam:

- (mode all_rows r.location-string) <END>  
- (mode all_rows r.programming-string) <END>  
- (mode all_rows r.psip-string) <END>  
- (argmax all_rows r.rf-number) (hop v8 r.location-string)<END>

Candidate program \( a \)

- Token level scoring helps with understandability
- Bi-LSTM considers information forward and backward in time
- ConvNet considers spans equal to the size of lisp clauses

[Biloki+ 2019]
Reranking & Stacked Learning

- **Reranking**
  - train a reranking model (B) to improve the output of a generator model (A)
  - Syntax parsing [Collins 2000; Charniak&Johnson 2005; Huang 2008]
  - Segmentation, POS tagging [Sun 2012]
  - Entity linking [He+ 2013]

- **Stacked learning** to correct the training/test mismatch
  - Create a k-fold cross-validation split on the original training data
  - Train k copies of model (A)
  - Generate the training data for model (B) with these k models
  - Train model (B)
Discriminative Training Objective

- Rank a set of candidates in beam

\[ O_{NPP}(\omega) = \sum_l \sum_{1 \leq i \neq j \leq |s_\phi(x^l)|} 1[r^l,i > r^l,j] \log \sigma(v^l,i - v^l,j) \]

\[ \sigma(v) = 1/(1 + e^{-v}) \]

- The score of program \( a^{l,i} \) considers query \( x^l \) and beam \( s_\phi(x^l) \)

\[ v^l,i = v_\omega(a^{l,i}, x^l, s_\phi(x^l)) \]
Ensemble the Scorers from Beams

- How to combine the scores of programs from K beam searches?
- The score of program \( a \) under context \( x \) given base models \( \Phi = \{\phi^k\}_{k=1}^K \)

\[ v_{\omega,\Phi}(a; x) = \sum_k [v'_\omega(a; x, s^k_\phi(x)) - \bar{v}_\omega(x)]. \]

where \( \bar{v}_\omega(x) \) is the average score for programs in beam \( s^k_\phi(x) \)

\[ \bar{v}_\omega(x) = \frac{1}{|s^k_\phi(x)|} \sum_{a \in s^k_\phi(x)} v_\omega(a; x, s^k_\phi(x)) \]

and \( v'_\omega \) backs-off \( v_\omega \) to \( \bar{v}_\omega(x) \) whenever \( a \) is not in beam

\[ v'_\omega(a; x, s^k_\phi(x)) = \begin{cases} 
  v_\omega(a; x, s^k_\phi(x)), & \text{if } a \in s^k_\phi(x) \\
  \bar{v}_\omega(x), & \text{else}
\end{cases} \]
## Results

- The impact of NPP, stacked learning (LOO) and ensemble

<table>
<thead>
<tr>
<th>Setting</th>
<th>Model</th>
<th>Dev (std)</th>
<th>$\Delta^\dagger$</th>
<th>Test (std)</th>
<th>$\Delta^\dagger$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean of MAPOs trained on a single train/dev split</td>
<td>MAPO</td>
<td>41.9(0.3)</td>
<td>-</td>
<td>43.1(0.5)</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>MAPO + NPP</td>
<td>42.4(0.7)</td>
<td>0.8</td>
<td>43.7(0.6)</td>
<td>0.5</td>
</tr>
<tr>
<td>Mean of MAPOs trained on LOO splits</td>
<td>MAPO</td>
<td>41.7(1.1)</td>
<td>-</td>
<td>42.8(0.5)</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>MAPO + NPP*</td>
<td>43.0(0.2)$^+$</td>
<td>1.3</td>
<td>43.9(0.2)</td>
<td>1.1</td>
</tr>
<tr>
<td>Ensemble of 5 MAPOs trained on LOO splits</td>
<td>MAPO</td>
<td>-</td>
<td>-</td>
<td>45.5</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>MAPO + NPP*</td>
<td>-</td>
<td>-</td>
<td>46.6</td>
<td>1.1</td>
</tr>
<tr>
<td>Ensemble of 10 MAPOs trained on LOO splits</td>
<td>MAPO</td>
<td>-</td>
<td>-</td>
<td>46.3(−)</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>MAPO + NPP*</td>
<td>-</td>
<td>-</td>
<td>47.2(−)</td>
<td>0.9</td>
</tr>
</tbody>
</table>

Table 4: Main results. $^\dagger$Improvements compared to MAPO. *Stacked learning with Leave-One-Out (LOO) data splits. $^+$NPP uses 67%-33% train-dev splits from the stacked learning data.
Plan

● **Query understanding**
  ○ Weak supervision semantic parsing tasks
  ○ Neural Symbolic Machines
    ■ Symbolic representations for efficient inference
  ○ Memory Augmented Policy Optimization
    ■ Unbiased low-variance gradient estimation with experience replays
    ■ RL vs MML vs ML
  ○ Reranking for RL trained decoders
    ■ Sequence scorer, Stacked Learning, Scorer Ensembles

● **Document understanding**
  ○ Generalizable, yet accountable & scalable
  ○ The return of inverted lists
  ○ Experiment with n-gram machines
A shared external memory

- The progress of civilizations depends on their shared memories

"AS WE MAY THINK"
(1945)

Consider a future device for individual use, which is a sort of mechanized private file and library. It needs a name, and to coin one at random, memex will do. A memex is a device in which an individual stores all his books, records, and communications, and which is mechanized so that it may be consulted with exceeding speed and flexibility. It is an enlarged intimate supplement to his memory.
Modern search engines

- **Scalable**
  - Can respond to user's requests within a fraction of a second

- **Accountable**
  - Reference, authority, engagement

- **but less generalizable**
  - work very well for popular queries through associations with engagements
  - distillation helps to the extent of available unlabeled queries
  - less generalizable to novel situations

[Brin & Page 1998]
Generalizable, yet accountable & scalable

- **Generalizable**
  - personal/contextual
  - compositional
  - cross lingual
  - multi-modality

- **Accountable & Controllable**
  - understand why a system produces certain result and promote trust
  - control model behavior through the update of its data

- **Scalable**
  - respond to user's requests within a fraction of a second
Transfer learning is promising for generalization

- ImageNet has thousands of labeled examples per class, while
- Children often only need one training example to learn a new concept
Transfer learning favors big models

- the effective data transferred grows with the **model size** $N$ and fine-tuning data size $D_F$

$$D_T = \text{effective data transferred} = k (D_F)^\alpha (N)^\beta$$

- and, in low data region

$$D_E = D_F + D_T \approx D_T$$

<table>
<thead>
<tr>
<th>Transfer from</th>
<th>$k$</th>
<th>$\alpha$</th>
<th>$\beta$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Text (\rightarrow) Python</td>
<td>1.9e4</td>
<td>0.18</td>
<td>0.38</td>
</tr>
<tr>
<td>50% Text and 50% non-python code (\rightarrow) Python</td>
<td>2.1e5</td>
<td>0.096</td>
<td>0.38</td>
</tr>
</tbody>
</table>
Lack of reasoning, accountability and efficiency

- Tested birthday fact questions on GPT-3 (2020, davinci)
- Given its source an answer could have been verified by the users or a cross encoder (2020 EfficientQA Competition)

<table>
<thead>
<tr>
<th>Q &amp; A</th>
<th>Fact / Fiction</th>
</tr>
</thead>
</table>
| Q: what is barack obama's birthday?  
A: Barack Obama's birthday is **August 4, 1961**. | Fact |
| Q: what is barack obama's wife's birthday?  
A: Michelle Obama's birthday is **January 17, 1964**. | Fact |
| Q: what is barack obama's father's birthday?  
A: Barack Obama's father's birthday is **August 4, 1961**. | Fiction |
| Q: who are barack obama's daughters  
A: Sasha and Malia | Fact |
| Q: what are barack obama's daughters birthday?  
A: barack obama's daughters birthday is **on august 4** | Fiction |
| Q: what are Sasha and Malia obama's birthdays  
A: Sasha and Malia Obama's birthdays are **on the same day, July 4, 1981**. | Fiction |

Barack Hussein Obama Sr.,  
Born: June 18th, 1934  

Sasha Obama, Born: June 10, 2001  
Mammalian memory

- Very rapid adaptation (in just one or few trials) is necessary for survival
  - E.g., associating smell of food to sickness

- However, good representation is learnt gradually
  - E.g., learning during sleeps to prevent interference with established associations
Complementary Learning Theory

Connections within and among neocortical areas (green) support gradual acquisition of structured knowledge through interleaved learning.

Bidirectional connections (blue) link neocortical representations to the hippocampus/MTL for storage, retrieval, and replay.

Encode

Episodic Memory

Rapid learning in connections within hippocampus (red) supports initial learning of arbitrary new information.

Record & Replay

Observations

Primary sensory and motor cortices

[McClelland+ 1995]
[Kumaran+ 2016]
Explicit memory and control with retrieval-based models

- **Generalizable**
  - achieve generalizability with a small model + a big memory

- **Accountable & Controllable**
  - understand from which piece of memory certain result is deduced
  - the memory can be updated independent of the model

- **Scalable**
  - achieve low latency with an index structure to the memory
The return of inverted lists

- Classical IR systems
  - rely on exact lexical matches, which can carry out search efficiently with inverted list index
  - fall short of matching related terms (**vocabulary mismatch**) or modeling context of the terms (**semantic mismatch**).

- Dense Retrievers
  - lack of lexical matches
  - huge indices (100x) and large latencies (10x) especially for multi-vector representations like ColBERT, DensePhrase

- The two can be combined to get the best of both worlds

[Gao+ 2021]
[Lee+ 2021]
Related work

- kNN-LM (Khandelwal+ 2020)
  - 2.9 perplexity improvement simply by linearly interpolating an LM's token prediction with the next token counts of k-nearest neighbors in the decoder state on training data
  - too expensive to perform retrieval during training

- REALM (Guu+ 2020), RAG (Lewis+ 2020)
  - augment sequence models with a latent knowledge (document) retriever for better interpretability and controllability
  - e2e training by backpropagating through a dense doc retrieval step

- This presentation
  - can we use a single explicit knowledge representation, i.e., short sequences of words, and achieve both efficiency and semantic matching?
Question answering as a simple test bed

- A good representation should also support reasoning & scalability
End-to-End Question Answering

- A hard optimization problem
N-Gram Machines

[Yang+ 2018]
Seq2Seq components

- A知识编码器 defines a distribution over knowledge tuples given sentences, and the distribution over knowledge store
  \[ P(\Gamma_i | s_i, s_{i-1}; \theta_{\text{enc}}) \]
  \[ P(\Gamma | s; \theta_{\text{enc}}) = \prod_{\Gamma_i \in \Gamma} P(\Gamma_i | s_i, s_{i-1}; \theta_{\text{enc}}) \]

- A知识解码器 defines a distribution over sentences given tuples
  \[ P(s_i | \Gamma_i, s_{i-1}; \theta_{\text{dec}}) \]

- A程序员 defines a distribution over programs given a question
  \[ P(C | q, \Gamma; \theta_{\text{prog}}) \]
Given an example \((s, q, a)\)
- maximize the **expected reward** (QA) + **sentence reconstruction** (AE)
  (VAE or contrastive loss might produce better latent space distributions)

\[
O_{QA}^{\text{enc}, \text{prog}} = \sum_{\Gamma} \sum_{C} P(\Gamma|s; \theta_{\text{enc}}) P(C|q, \Gamma; \theta_{\text{prog}}) R(\Gamma, C, a),
\]

\[
O_{AE}^{\text{enc}, \text{dec}} = \mathbb{E}_{p(z|x;\theta_{\text{enc}})} \left[ \log p(x|z; \theta_{\text{dec}}) \right] + \sum_{z \in Z^N(x)} \log p(x|z; \theta_{\text{dec}}),
\]

- **Gradient estimation**
  - beam search for its low variances

- **Coordinate ascent**
  - updates three components in alternation with **REINFORCE**
Facebook bAbI Tasks

- Simulated question answering tasks to test the ability to "understand"
- We introduce a special version ("life-long bAbI"), which has stories of up to 10 million sentences

<table>
<thead>
<tr>
<th>Sam walks into the kitchen.</th>
<th>Brian is a lion.</th>
<th>Mary journeyed to the den.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sam picks up an apple.</td>
<td>Julius is a lion.</td>
<td>Mary went back to the kitchen.</td>
</tr>
<tr>
<td>Sam walks into the bedroom.</td>
<td>Julius is white.</td>
<td>John journeyed to the bedroom.</td>
</tr>
<tr>
<td>Sam drops the apple.</td>
<td>Bernhard is green.</td>
<td>Mary discarded the milk.</td>
</tr>
<tr>
<td>Q: Where is the apple?</td>
<td>Q: What color is Brian?</td>
<td>Q: Where was the milk before the den?</td>
</tr>
<tr>
<td>A. Bedroom</td>
<td>A. White</td>
<td>A. Hallway</td>
</tr>
</tbody>
</table>

[Weston+ 2015]
# Example Knowledge Store & Program

Table 6: Task 2 Two Supporting Facts

<table>
<thead>
<tr>
<th>Story</th>
<th>Knowledge Storage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sandra journeyed to the hallway.</td>
<td>Sandra journeyed hallway</td>
</tr>
<tr>
<td>John journeyed to the bathroom.</td>
<td>John journeyed bathroom</td>
</tr>
<tr>
<td>Sandra grabbed the football.</td>
<td>Sandra got football</td>
</tr>
<tr>
<td>Daniel travelled to the bedroom.</td>
<td>Daniel journeyed bedroom</td>
</tr>
<tr>
<td>John got the milk.</td>
<td>John got milk</td>
</tr>
<tr>
<td>John dropped the milk.</td>
<td>John got milk</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Question</th>
<th>Program</th>
</tr>
</thead>
<tbody>
<tr>
<td>Where is the milk?</td>
<td>ArgmaxFR milk got</td>
</tr>
<tr>
<td></td>
<td>Argmax V1 journeyed</td>
</tr>
</tbody>
</table>
Scalability

- A lot more scalable than commonly used deep model

[Yang+ 2018]
Thanks!
Ranking models on MS MARCO (Tesla V100)
Table 3: Performance and latency of COIL systems with different representation dimensions. Results not applicable are denoted ‘−’ and no available ‘n.a.’. Here $n_c$ denotes COIL CLS dimension and $n_t$ token vector dimension. *: ColBERT use approximate search and quantization. We exclude I/O time from measurements.

<table>
<thead>
<tr>
<th>Model</th>
<th>Dev Retrieval</th>
<th></th>
<th>DL2019 Retrieval</th>
<th></th>
<th>Latency/ms</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MRR@10</td>
<td>Recall@1K</td>
<td>NDCG@10</td>
<td>MRR</td>
<td>CPU</td>
</tr>
<tr>
<td>BM25</td>
<td>0.184</td>
<td>0.853</td>
<td>0.506</td>
<td>0.825</td>
<td>36</td>
</tr>
<tr>
<td>Dense</td>
<td>0.304</td>
<td>0.932</td>
<td>0.635</td>
<td>0.898</td>
<td>293</td>
</tr>
<tr>
<td>ColBERT</td>
<td>0.360</td>
<td>0.968</td>
<td>n.a.</td>
<td>n.a.</td>
<td>458*</td>
</tr>
<tr>
<td>COIL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$n_c$</td>
<td>$n_t$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>768</td>
<td>32</td>
<td>0.355</td>
<td>0.963</td>
<td>0.704</td>
<td>380</td>
</tr>
<tr>
<td>128</td>
<td>32</td>
<td>0.350</td>
<td>0.953</td>
<td>0.692</td>
<td>125</td>
</tr>
<tr>
<td>128</td>
<td>8</td>
<td>0.347</td>
<td>0.956</td>
<td>0.694</td>
<td>113</td>
</tr>
<tr>
<td>0</td>
<td>32</td>
<td>0.341</td>
<td>0.949</td>
<td>0.660</td>
<td>67</td>
</tr>
<tr>
<td>0</td>
<td>8</td>
<td>0.336</td>
<td>0.940</td>
<td>0.678</td>
<td>55</td>
</tr>
</tbody>
</table>
Typical Cognitive Architectures

- The design of mammalian brains is inspiring to NLP systems
  - Since they are solving similar problems

- The design has not changed much since 30 years ago
  - "We've totally solved it already"
    -- Nate Derbinsky, Northeastern U.

- Lack of real applications, but now
  - internet economy and data
  - computation and machine learning

https://soar.eecs.umich.edu
Optimization

- For training stability and tweaking, we augment the training objective with **experience replays**

\[ \nabla_{\theta_{\text{dec}}} O'(\theta) = \sum_{s_i \in s} \sum_{\Gamma_i} [\beta(\Gamma_i) + P(\Gamma_i | s_i, s_{i-1}; \theta_{\text{enc}})] \nabla_{\theta_{\text{dec}}} \log P(s_i | \Gamma, s_{i-1}; \theta_{\text{dec}}), \]

\[ \beta(\Gamma_i) = 1 \text{ if } \Gamma_i \text{ only contains tokens from } s_i \text{ and } 0 \text{ otherwise} \]

\[ \nabla_{\theta_{\text{enc}}} O'(\theta) = \sum_{s_i \in s} \sum_{\Gamma_i} [P(\Gamma_i | s_i, s_{i-1}; \theta_{\text{enc}}) \log P(s_i | \Gamma_i, s_{i-1}; \theta_{\text{dec}}) + \mathcal{R}(G'(\Gamma_i)) + \mathcal{R}(G(\Gamma_i))] \nabla_{\theta_{\text{enc}}} \log P(\Gamma_i | s_i, s_{i-1}; \theta_{\text{enc}}), \]

where \( \mathcal{R}(G) = \sum_{\Gamma \in G} \sum_{C} P(\Gamma | s; \theta_{\text{enc}}) P(C | q, \Gamma; \theta_{\text{prog}}) R(\Gamma, C, a) \) is the total expected reward for a set of valid knowledge stores \( G, G(\Gamma_i) \) is the set of knowledge stores which contains the tuple \( \Gamma_i \), and \( G'(\Gamma_i) \) is the set of knowledge stores which contains the tuple \( \Gamma_i \) through tweaking.

\[ \nabla_{\theta_{\text{prog}}} O'(\theta) = \sum_{\Gamma} \sum_{C} [\alpha I[C \in C^*(s, q)] + P(C | q, \Gamma; \theta_{\text{prog}})] \cdot P(\Gamma | s; \theta_{\text{enc}}) R(\Gamma, C, a) \nabla_{\theta_{\text{prog}}} \log P(C | q, \Gamma; \theta_{\text{prog}}), \]

where \( C^*(s, q) \) is the experience replay buffer for \((s, q)\), \( \alpha = 0.1 \) is a constant. During training, the program with the highest weighted reward (i.e. \( P(\Gamma | s; \theta_{\text{enc}}) R(\Gamma, C, a) \)) is added to the replay buffer.

- **optimize by coordinate ascent** – updating three components in alternation with **REINFORCE**
### WikiTableQuestions: example solutions

<table>
<thead>
<tr>
<th>Superlative</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>nt-13901: the most points were scored by which player?</strong></td>
<td>Sort all rows by column ‘points’ and take the first row.</td>
</tr>
<tr>
<td>(argmax all_rows r.points-num)</td>
<td>Output the value of column ‘player’ for the rows in v0.</td>
</tr>
<tr>
<td>(hop v0 r.player-str)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Difference</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>nt-457: how many more passengers flew to los angeles than to saskatoon?</strong></td>
<td>Find the row with ‘saskatoon’ matched in column ‘city’.</td>
</tr>
<tr>
<td>(filter&lt;sub&gt;in&lt;/sub&gt; all_rows ['saskatoon'] r.city-str)</td>
<td>Find the row with ‘los angeles’ matched in column ‘city’.</td>
</tr>
<tr>
<td>(filter&lt;sub&gt;in&lt;/sub&gt; all_rows ['los angeles'] r.city-str)</td>
<td>Calculate the difference of the values</td>
</tr>
<tr>
<td>(diff v1 v0 r.passengers-num)</td>
<td>in the column ‘passenger’ of v0 and v1.</td>
</tr>
</tbody>
</table>
# More examples

## Before / After

<table>
<thead>
<tr>
<th>Example</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>nt-10832: which nation is before peru?</strong></td>
<td>(filter ( \text{in} ) all_rows ‘peru’ r.nation-str) Find the row with ‘peru’ matched in ‘nation’ column. (previous v0) Find the row before v0. (hop v1 r.nation-str) Output the value of column ‘nation’ of v1.</td>
</tr>
</tbody>
</table>

## Compare & Count

<table>
<thead>
<tr>
<th>Example</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>nt-647: in how many games did sri lanka score at least 2 goals?</strong></td>
<td>(filter ( \geq ) all_rows [2] r.score-num) Select the rows whose value in the ‘score’ column ( \geq 2 ). (count v0) Count the number of rows in v0.</td>
</tr>
</tbody>
</table>

## Exclusion

<table>
<thead>
<tr>
<th>Example</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>nt-1133: other than william stuart price, which other businessman was born in tulsa?</strong></td>
<td>(filter ( \text{in} ) all_rows ‘tulsa’ r.hometown-str) Find rows with ‘tulsa’ matched in column ‘hometown’. (filter ( \text{not in} ) v0 ‘william stuart price’ r.name-str) Drop rows with ‘william stuart price’ matched in the value of column ‘name’. (hop v1 r.name-str) Output the value of column ‘name’ of v1.</td>
</tr>
</tbody>
</table>
Results on WebQuestionsSP

- First end-to-end seq2seq to achieve SOTA on semantic parsing with weak supervision over large knowledge base
- The performance approached SOTA with full supervision

<table>
<thead>
<tr>
<th>Model</th>
<th>Avg. Prec. @1</th>
<th>Avg. Rec. @1</th>
<th>Avg. F1 @1</th>
</tr>
</thead>
<tbody>
<tr>
<td>STAGG</td>
<td>67.3</td>
<td>73.1</td>
<td>66.8</td>
</tr>
<tr>
<td>NSM – our model</td>
<td>70.8</td>
<td>76.0</td>
<td>69.0</td>
</tr>
<tr>
<td>STAGG (full supervision)</td>
<td>70.9</td>
<td>80.3</td>
<td>71.7</td>
</tr>
</tbody>
</table>
Comparison of model update strategies

Correctness of the behavior

Experiences & Reward

Model’s preference

On-policy optimization (REINFORCE)

Iterative Maximum Likelihood (IML)

Maximum Marginal Likelihood (MML)

MAPO

Question 1

Question 2

[Liang+ 2018]

63
Clipping Mechanism

- Training becomes less biased over time

[Liang+ 2018]
## Results with weak supervision

<table>
<thead>
<tr>
<th>Model</th>
<th>E.S.</th>
<th>Dev.</th>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pasupat &amp; Liang (2015)[28]</td>
<td>-</td>
<td>37.0</td>
<td>37.1</td>
</tr>
<tr>
<td>Neelakantan et al. (2017)[26]</td>
<td>1</td>
<td>34.1</td>
<td>34.2</td>
</tr>
<tr>
<td>Neelakantan et al. (2017)[26]</td>
<td>15</td>
<td>37.5</td>
<td>37.7</td>
</tr>
<tr>
<td>Haug et al. (2017)[15]</td>
<td>1</td>
<td>-</td>
<td>34.8</td>
</tr>
<tr>
<td>Haug et al. (2017)[15]</td>
<td>15</td>
<td>-</td>
<td>38.7</td>
</tr>
<tr>
<td>Zhang et al. (2017)[51]</td>
<td>-</td>
<td>40.4</td>
<td>43.7</td>
</tr>
<tr>
<td>MAPO</td>
<td>1</td>
<td>42.7</td>
<td>43.8</td>
</tr>
<tr>
<td>MAPO (ensembled)</td>
<td>5</td>
<td>-</td>
<td>46.2</td>
</tr>
</tbody>
</table>

Table 3: Results on **WikiTableQuestions**. E.S. is the number of ensembles (if applicable).

<table>
<thead>
<tr>
<th>Model</th>
<th>Dev.</th>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zhong et al. (2017)[52]*</td>
<td>60.8</td>
<td>59.4</td>
</tr>
<tr>
<td>Wang et al. (2017)[40]*</td>
<td>67.1</td>
<td>66.8</td>
</tr>
<tr>
<td>Xu et al. (2017)[46]*</td>
<td>69.8</td>
<td>68.0</td>
</tr>
<tr>
<td>Huang et al. (2018)[18]*</td>
<td>68.3</td>
<td>68.0</td>
</tr>
<tr>
<td>Yu et al. (2018)[48]*</td>
<td>74.5</td>
<td>73.5</td>
</tr>
<tr>
<td>Sun et al. (2018)[38]*</td>
<td>75.1</td>
<td>74.6</td>
</tr>
<tr>
<td>Dong &amp; Lapata (2018)[12]*</td>
<td>79.0</td>
<td>78.5</td>
</tr>
<tr>
<td>MAPO</td>
<td>72.4</td>
<td>72.6</td>
</tr>
<tr>
<td>MAPO (ensemble of 5)</td>
<td>-</td>
<td>74.9</td>
</tr>
</tbody>
</table>

Table 4: Results on **WikiSQL**. *All other methods use question-program pairs as strong supervision, while MAPO only uses question-answer pairs as weak supervision.*