Visualization Question Answering Using Introspective Program Synthesis

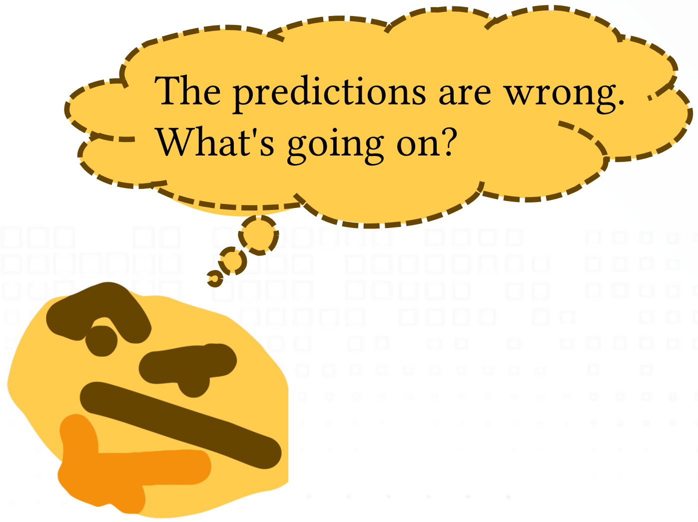
Yanju Chen, Xifeng Yan, Yu Feng
University of California, Santa Barbara
UCSB

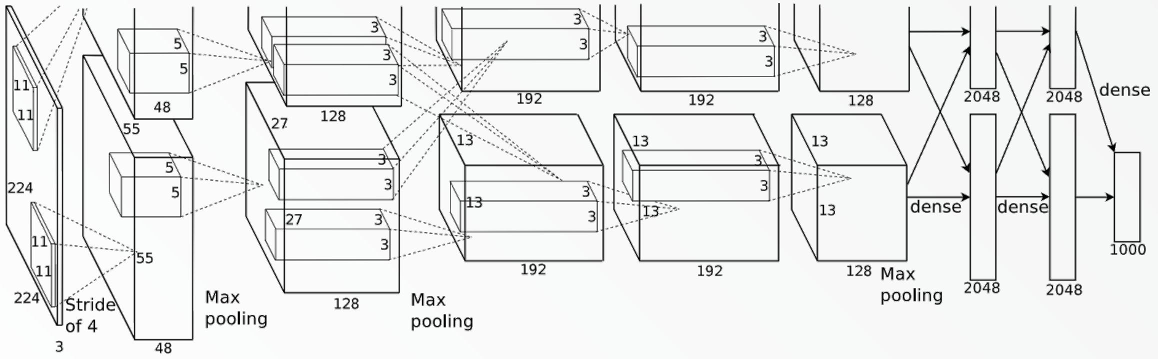
- Motivations -

Deep Learning Models

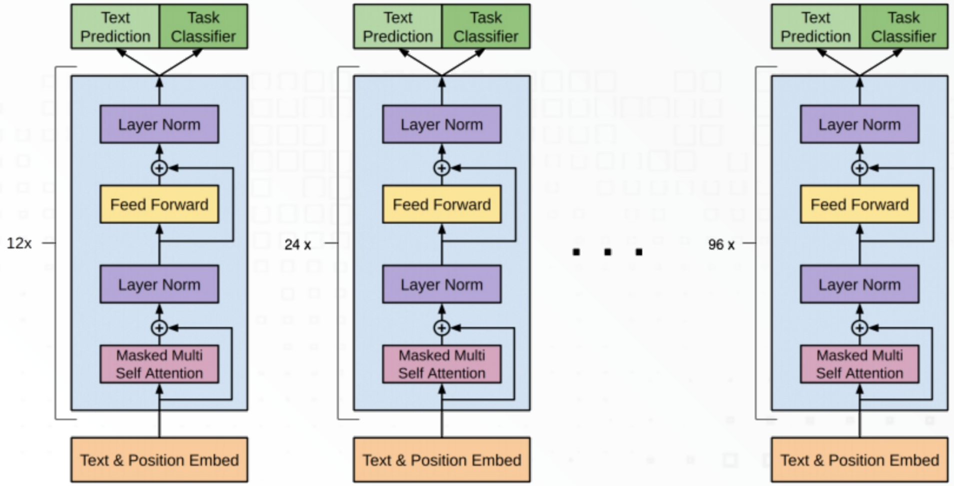
- Data-Driven/-Hungry
- Deep & Large-Scale

... if they work





Model Architecture of AlexNet, ~**60M** Params



Model Architecture of GPT3, ~**175B** Params

- Motivations -

VQA: A Motivating Example
(Visualization Question Answering)

- Given a stacked bar chart that represents opinions for future economic growth for different countries, a user describes her query based on the visualization in natural language:

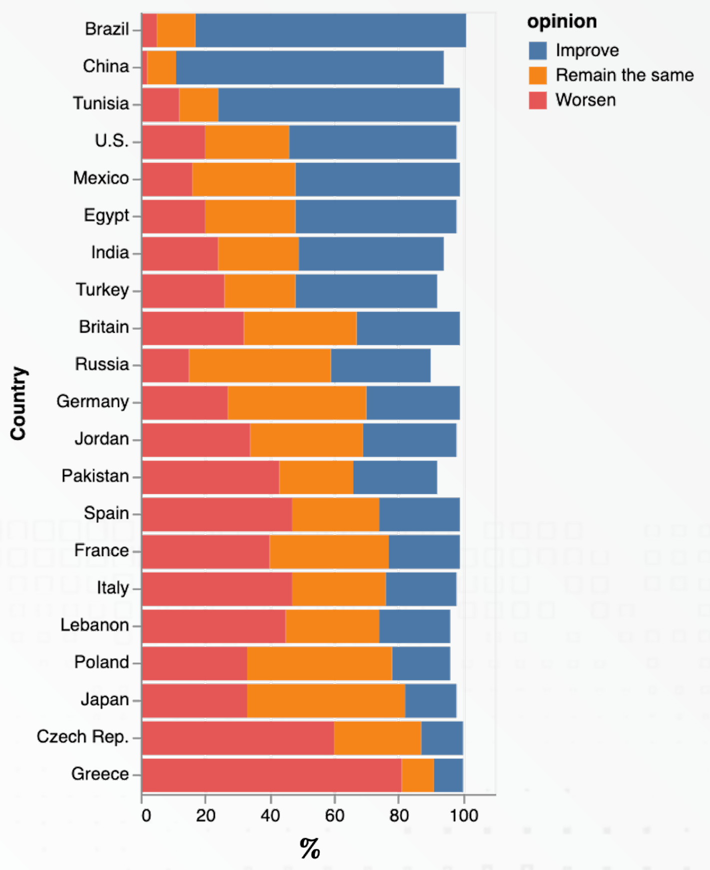
Query

Which ***country***'s economy will get ***most worse*** over next 12 months?

A Motivating Query

- A **Visualization Question Answering (VQA)** task is to design an algorithm that automatically finds the answer to a natural language query based on a given visualization.

Visualization



A Motivating Visualization

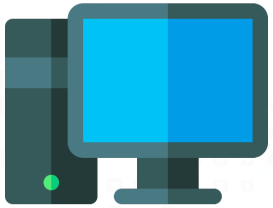
- Motivations -

Existing Approaches & Challenges

- Fully Supervised Machine Learning: SmBoP[1], NL2code[2]
  - Requires manual annotated logic forms / programs as supervised training data

- Weakly Supervised Machine Learning: TaPas[3]
  - Requires only question-answer pairs for training

+



+

Fully Supervised
Expensive and Hard to Get

Weakly Supervised
Cheap but Noisy

[1] **SmBoP: Semi-autoregressive Bottom-up Semantic Parsing**. Rubin, O. et al. *NAACL 2021.*
[2] **A Syntactic Neural Model for General-Purpose Code Generation**. Yin, P. et al. *ACL 2017.*
[3] **TaPas: Weakly Supervised Table Parsing via Pre-training**. Herzig, J. et al. *ACL 2020.*

- Motivations -

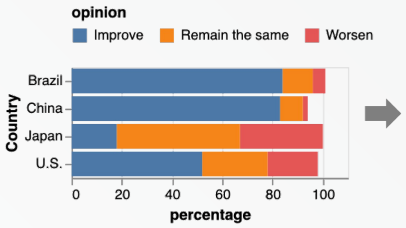
Observations

- For mainstream weakly supervised approaches that directly output VQA answers, they are:
  - non-trivial for human beings to understand, and
  - hard to fix if there's error in model reasoning/answer.

Can we use program synthesis techniques to get "background explanation" programs based on the model's predictions?

input: visualization → **output: model prediction**

- In this work, we investigate such a slightly different problem setting where:
  - not all the predictions are correct (usually only ***one*** of them is correct, or even sometimes ***none***), and
  - predictions may conflict with each other



Country	Improve	Worsen	Remain the same
Brazil	84	5	12
China	83	2	9
Japan	18	33	49
U.S.	52	20	26

A Simple Visualization and Its Table

Query

Which country has highest Improve value?

A Simple Query

"Brazil", "Japan", "China", "U.S.", ...

Model Predictions

```
project(aggregate(I, null, max ◇), ["Country"])
```

⇨ feasible for "Brazil", "Japan"

```
project(aggregate(I, null, ◇, ◇), ["Country"])
```

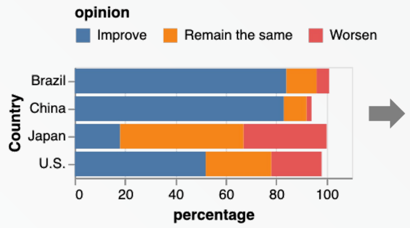
⇨ feasible for all: "Brazil", "Japan", "China", "U.S."

- Motivations -

A Straw-Man Proposal

Can we use program synthesis techniques to get "background explanation" programs based on the model's predictions?

input: visualization → **output: model prediction**



Country	Improve	Worsen	Remain the same
Brazil	84	5	12
China	83	2	9
Japan	18	33	49
U.S.	52	20	26

A Simple Visualization and Its Table

Query

Which country has highest Improve value?

A Simple Query

"Brazil", "Japan", "China", "U.S.", ...

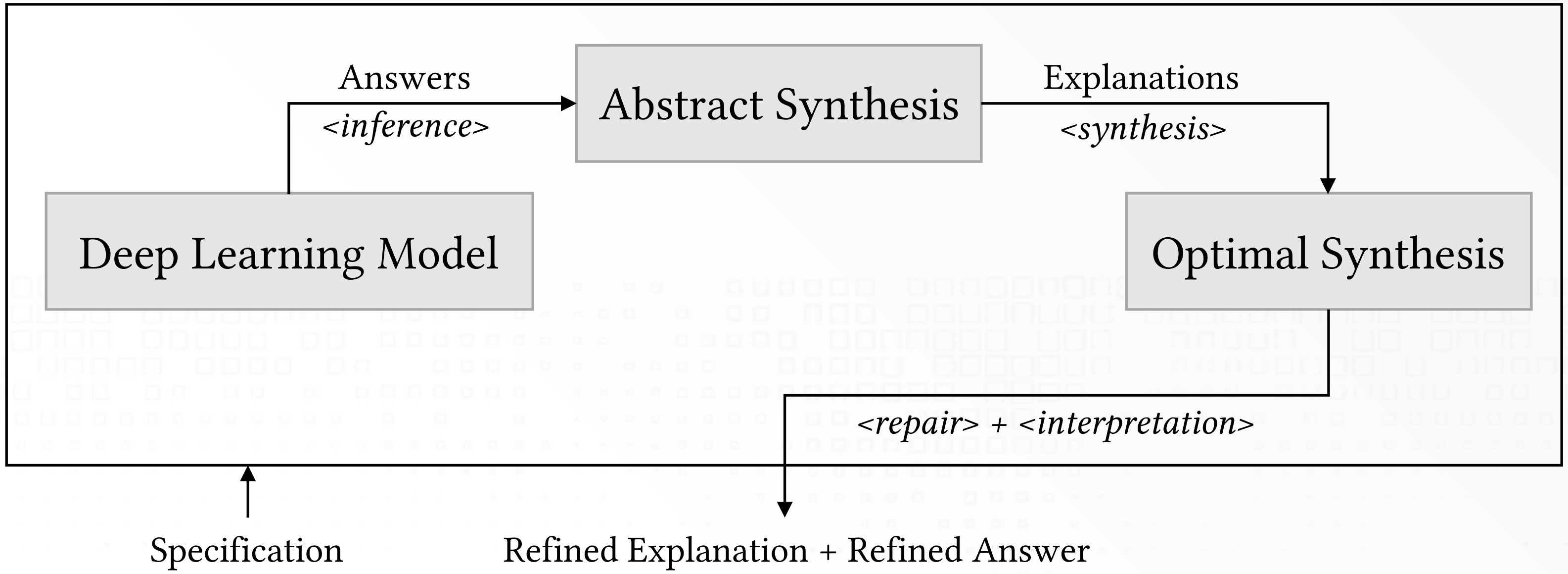
Model Predictions

- The Straw-Man Proposal
  1. For every single prediction of the deep learning model, we trigger an off-the-shelf synthesizer to solve for program(s)
  2. Then we end up with a bunch of programs and (maybe) ask the user to pick the "best" one

- There are potential issues:
  - ***Not scalable***: For cases where large number of predictions are produced, this won't scale well
  - ***Model dependent***: If predictions do not contain the correct answer, synthesis done will be meaningless
  - ***Unclear of best-fitting definition***: There's no formal definition of best-fitting program; need to connect 3 parties: *elements in visualization*, *language units in query* and *production rules in explanation programs*

Overview: POE

- Fixing Deep Learning Model's (Noisy) Outputs via **Introspective Program Synthesis**
  - Search Space Induction via **Abstract Program Synthesis**
  - Finding Best Consistent Programs via **Optimal Program Synthesis**



In the context of this work, we use *programs* and *explanations* interchangeably.

- Synthesis Using POE -

A Walkthrough of POE

Illustration of A Walkthrough of POE

- Synthesis Using POE -

A Walkthrough of POE

- Original TAPAS Outputs:

(0.78, Brazil), (0.67, Japan), (0.55, Greece), ...

- POE's Abstract Program Synthesis Outputs:

```
project(select(pivot(T, ◇, ◇), ◇, ◇, ◇), ◇)
project(select(T, ◇, ◇, ◇), ◇)
...
```

- POE's Optimal Program Synthesis Outputs:

```
project(select(pivot(
  T, "opinion", "%"), "Improve", eqmax, null), ["Country"])

project(select(pivot(
  T, "opinion", "%"), "Worsen", eqmax, null), ["Country"])
```

$$
\begin{array}{lcl}
\langle \mathit{Table} \rangle & ::= & \text{project(} \langle \mathit{Table} \rangle, \langle \mathit{ColList} \rangle \text{)} \\
 & \mid & \text{select(} \langle \mathit{Table} \rangle, \langle \mathit{BoolOp} \rangle, \langle \mathit{ColInt} \rangle, \langle \mathit{ConstVal} \rangle \text{)} \\
 & \mid & \text{pivot(} \langle \mathit{Table} \rangle, \langle \mathit{ColInt} \rangle, \langle \mathit{ColInt} \rangle \text{)} \\
 & \mid & \text{aggregate(} \langle \mathit{Table} \rangle, \langle \mathit{ColList} \rangle, \langle \mathit{AggrOp} \rangle, \langle \mathit{ColInt} \rangle \text{)} \\
\langle \mathit{AggrOp} \rangle & ::= & \text{count} \mid \text{min} \mid \text{max} \mid \text{sum} \mid \text{mean} \\
\langle \mathit{BoolOp} \rangle & ::= & < \mid <= \mid == \mid >= \mid > \mid != \mid \text{eqmax} \mid \text{eqmin}
\end{array}
$$

$$\langle \mathit{Table} \rangle \in \textbf{tables}, \quad \langle \mathit{ConstVal} \rangle \in \textbf{constants}$$
$$\langle \mathit{ColInt} \rangle \in \textbf{columns}, \quad \langle \mathit{ColList} \rangle \in \textbf{columns}^n$$

Syntax of A Motivating Toy DSL

Query

Which ***country***'s economy will get ***most worse*** over next 12 months?

A Motivating Query

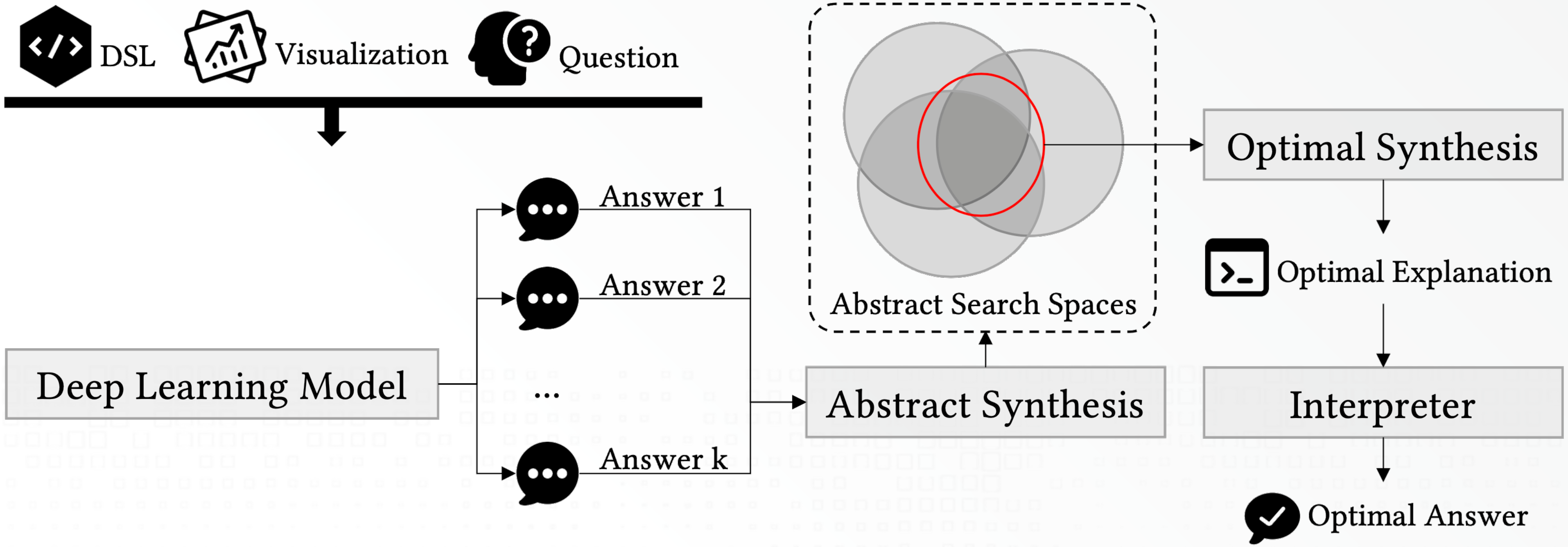
Data

Country	opinion	%	color
Brazil	Improve	84	blue
Brazil	Remain the same	12	orange
Brazil	Worsen	5	red
China	Improve	83	blue
China	Remain the same	9	orange
China	Worsen	2	red
Tunisia	Improve	75	blue
...			
Japan	Improve	16	blue
Japan	Remain the same	49	orange
Japan	Worsen	33	red
Czech Rep.	Improve	13	blue
Czech Rep.	Remain the same	27	orange
Czech Rep.	Worsen	60	red
Greece	Improve	9	blue
Greece	Remain the same	10	orange
Greece	Worsen	81	red

Converted Table of the Visualization

- Synthesis Using POE -

System Workflow in POE

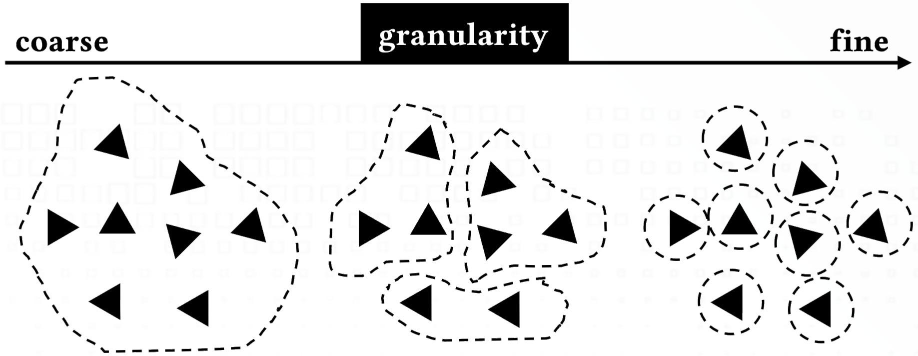


A System Workflow in POE

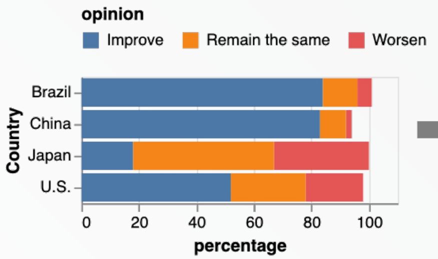
- Synthesis Using POE -

Abstract Program Synthesis with Noisy Specification

- Intuition: Narrow down program search space to such a sweet spot that:
  - respects the model outputs, and
  - promote synthesis efficiency.



Abstract Synthesis Granularities



Country	Improve	Worsen	Remain the same
Brazil	84	5	12
China	83	2	9
Japan	18	33	49
U.S.	52	20	26

A Simple Visualization and Its Table

Model Outputs: “Brazil”, “Japan”, “China”, “U.S.”

Abstract Synthesis Breakdown:

$\diamond$
⇨ feasible for all examples

project($\diamond$)
⇨ feasible for all examples

project(aggregate(I, null, $\diamond_0$, $\diamond_1$), ["Country"])
⇨ feasible for "Brazil", "Japan", "China"

project(aggregate(I, null, max, $\diamond_1$), ["Country"])
⇨ feasible for "Brazil", "Japan"

- Synthesis Using POE -

Optimal Program Synthesis for Explanation Refinement

- Intuition: Maximize consistency between explanation, visualization and query.
- Hard Constraints (Syntactic Correctness)
- Soft Constraints (Semantic Approximation)
  - **NSYN**: Near-Synonym Linguistic Engine
  - A linguistic engine that determines whether two linguistic units are near-synonyms (semantically similar)

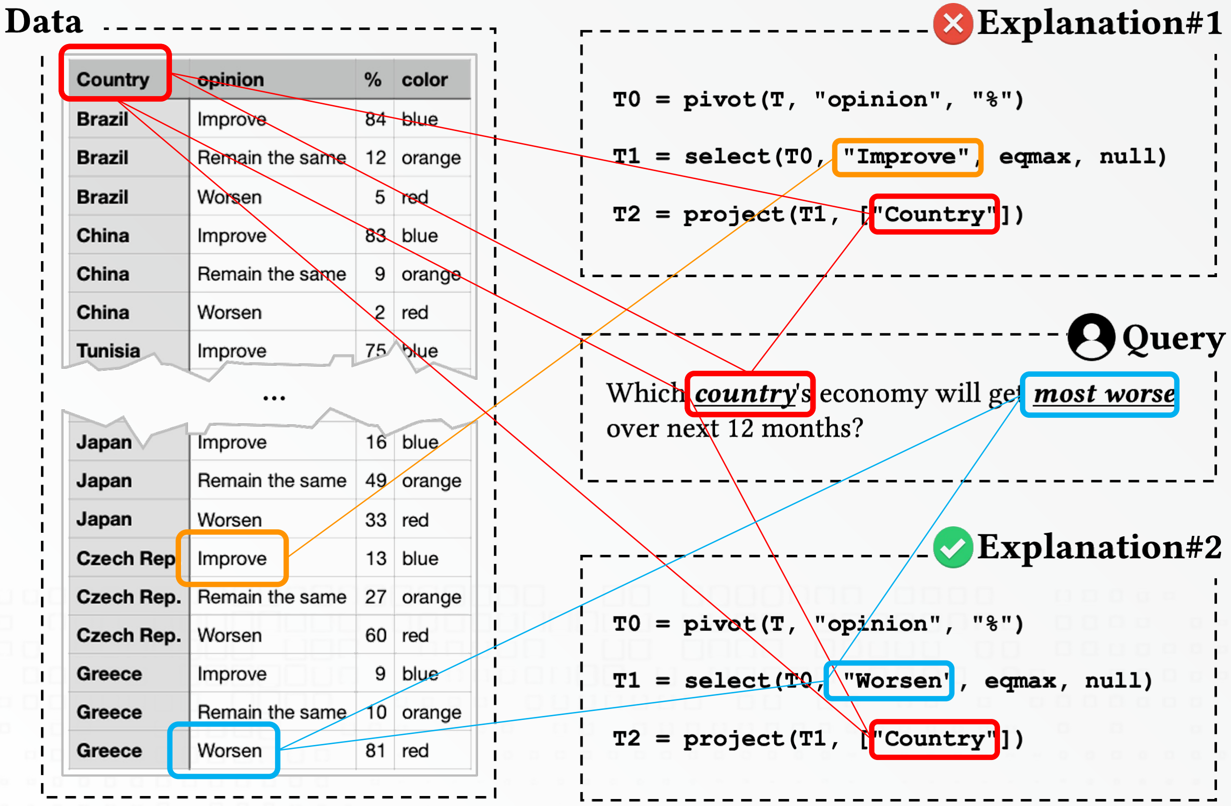
$$\textsc{NSyn}(\text{“high”}, \text{“highest”}) > \textsc{NSyn}(\text{“high”}, \text{“low”})$$

- Objective Function

$$\boxed{\sum_{w \in V_w} \sum_{t \in V_t} (1 - \textsc{NSyn}(w, t)) \cdot x_w^t} + \boxed{\sum_{p \in V_P} \mathrm{PPL}(P) \cdot u^P}$$

Maximize consistency matching.

More common abstract programs are preferred.



Triangle Alignments between Three Parties

Evaluation

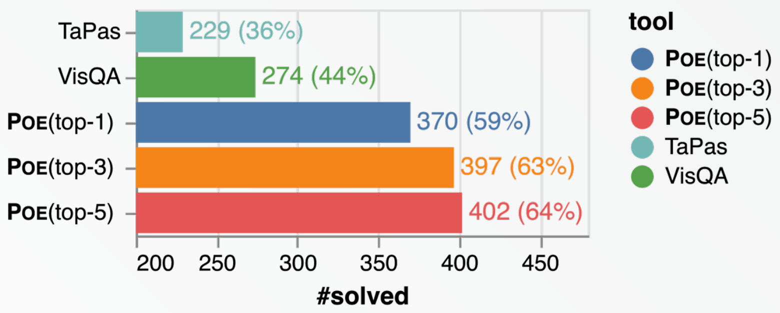
- Research Questions
  - **RQ1**. **Performance**: How does POE compare against state-of-the-art tools on visualization queries?
  - **RQ2**. **Effectiveness**: Can POE rectify wrong answers proposed by other tools?
  - **RQ3**. **Explainability**: Does POE synthesize explanations that well capture the question intentions and make sense to human end-users?
  - **RQ4**. **Ablation**: How significant is the benefit of abstract synthesis and optimal alignment?
- Benchmarks
  - **629** Visualization Question Answering Tasks from VisQA[1]
    - Real-World Data Sources
    - Non-Trivial Questions from Real Users
    - Wide Coverage of Question Types

[1] **Answering Questions about Charts and Generating Visual Explanations**. Kim, D.H. et al. *CHI 2020.*

- Evaluation -

Performance

- Comparison against TAPAS[1] and VisQA[2]
  - VisQA: +8%
  - POE (top-1): +23%
  - POE (top-3): +27%



Performance Comparison

POE can greatly boost performance of weakly supervised models.

- Stats of Different Questions Types
  - Retrieval
  - Comparison
  - Aggregation
  - Other
  - Total

question type	total	VisQA (baseline)	TAPAS	POE (top-1)
retrieval	183 (29%)	101 (55%)	98 (54%)	**123 (67%)**
comparison	87 (14%)	50 (57%)	0 (0%)	**71 (82%)**
aggregation	253 (40%)	92 (36%)	119 (47%)	**161 (64%)**
other	106 (17%)	**31 (29%)**	12 (11%)	15 (14%)
total	629 (100%)	274 (44%)	229 (36%)	**370 (59%)**

Performance Comparison on Different Question Types

POE is effective across different types of benchmarks.

[1] **TaPas: Weakly Supervised Table Parsing via Pre-training**. Herzig, J. et al. *ACL 2020*.
[2] **Answering Questions about Charts and Generating Visual Explanations**. Kim, D.H. et al. *CHI 2020*.

Discussions & Conclusions

- Discussions
  - Incomprehensive Questions
  - Limitation of NLP Modules
- Conclusions: A Tool for Visualization Question Answering via Introspective Program Synthesis
  - helps understand deep learning model's VQA predictions
  - fixes potentially wrong predictions by refinement

https://github.com/chyanju/Poe

POE is open-sourced and publicly available.