Scallop: A Language for Neurosymbolic Programming

Mayur Naik
University of Pennsylvania

Joint work with Ziyang Li and Jiani Huang
Two Prevalent Paradigms of Modern Programming

Deep Learning
[System 1]

Classical Algorithms
[System 2]
## Two Prevalent Paradigms of Modern Programming

<table>
<thead>
<tr>
<th>Deep Learning</th>
<th>Classical Algorithms</th>
</tr>
</thead>
<tbody>
<tr>
<td>[System 1]</td>
<td>[System 2]</td>
</tr>
</tbody>
</table>

- Sub-symbolic knowledge
- Open-domain knowledge
- Rapid reasoning
- Handling noise and naturalness
- In-context learning
Two Prevalent Paradigms of Modern Programming

Deep Learning

- Sub-symbolic knowledge
- Open-domain knowledge
- Rapid reasoning
- Handling noise and naturalness
- In-context learning

Classical Algorithms

- Domain-specific knowledge
- Complex reasoning
- Interpretability
- Compositional reasoning
- Generalizability
Neurosymbolic to Combine Both Worlds …

Deep Learning
[System 1]
- Sub-symbolic knowledge
- Open-domain knowledge
- Rapid reasoning
- Handling noise and naturalness
- In-context learning

Classical Algorithms
[System 2]
- Domain-specific knowledge
- Complex reasoning
- Interpretability
- Compositional reasoning
- Generalizability

neural + symbolic = neurosymbolic
Challenges With Combining Them

Deep Learning

Classical Algorithms

Neurosymbolic
Challenges With Combining Them

1. Choice of Symbolic Data Representation for $R$
2. Choice of Symbolic Reasoning Language for $P$
3. Automatic and Efficient Differentiable Reasoning Engine for learning $\frac{\delta Y}{\delta R}$ under algorithmic supervision
4. Ability to tailor learning $\frac{\delta Y}{\delta R}$ to individual applications' characteristics
5. Mechanism to leverage and integrate with existing training pipelines $\frac{\delta R}{\delta \theta}$ and neural models $M_{\theta}$
Our Approach: Scallop

- Relational Representation for $R$
- Datalog-based Language for $P$
- Provenance Semirings Framework for $\frac{\delta Y}{\delta R}$
- Integration with Pytorch for $\frac{\delta R}{\delta \theta}$ and $M_\theta$
A Motivating Example
PacMan-Maze

**State:** 200x200 colored image

**Action:** Up, Down, Left, Right

(Environments are 5x5 grids randomized for each session)
PacMan-Maze

State: 200x200 colored image
Action: Up, Down, Left, Right

(Environments are 5x5 grids randomized for each session)
type Action = UP | RIGHT | DOWN | LEFT

type actor(x: i32, y: i32), goal(x: i32, y: i32), enemy(x: i32, y: i32)

rel safe_cell(x, y) = grid_cell(x, y) and not enemy(x, y)
rel edge(x, y, x, y + 1, UP) = safe_cell(x, y) and safe_cell(x, y + 1)
// Rules for RIGHT, DOWN, and LEFT edges are omitted for brevity...

rel next_pos(p, q, a) = actor(x, y) and edge(x, y, p, q, a)
rel path(x, y, x, y) = next_pos(x, y, _)
rel path(x1, y1, x3, y3) = path(x1, y1, x2, y2) and edge(x2, y2, x3, y3, _)
rel next_action(a) = next_pos(p, q, a) and path(p, q, r, s) and goal(r, s)
PacMan-Maze

State: 200x200 colored image
Action: Up, Down, Left, Right
(Environments are 5x5 grids randomized for each session)

<table>
<thead>
<tr>
<th></th>
<th>Neurosymbolic (with Scallop)</th>
<th>DQN</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Success rate</strong></td>
<td>99.4%</td>
<td>84.9%</td>
</tr>
<tr>
<td>(reaches the goal within 50 steps)</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong># of Training episodes</strong></td>
<td>50</td>
<td>50K</td>
</tr>
<tr>
<td>(to achieve the success rate)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(Note: this is not entirely a fair comparison since our Scallop program encodes system dynamics and human knowledge)
Architecture of Scallop Compiler

```plaintext
rel constraint() =
for all (a, b:
    father(a, b) implies
    (son(b, a) or daughter(b, a))
)
```

Scallop Program
.scl file written by user
Architecture of Scallop Compiler

```c
rel constraint() =
forall(a, b:
father(a, b) implies
(som(b, a) or daughter(b, a))
)
```

Scallop Program
.scl file written by user

Parser

Front-IR Program
Abstract Syntax Tree (AST)

- Type Inference
- Syntax De-sugaring
- Boundness Analysis
...
Architecture of Scallop Compiler

```plaintext
rel constraint() =
forall(a, b:
  father(a, b) implies
  (son(b, a) or daughter(b, a))
)
```

Scallop Program
.scl file written by user

Parser

Front-IR Program
Abstract Syntax Tree (AST)
- Type Inference
- Syntax De-sugaring
- Boundness Analysis
...

F2B

Back-IR Program
Extended Datalog
- Query Planning
- Constant Fold & Prop
- Magic-set Transformation
...
Architecture of Scallop Compiler

rel constraint() =
forall(a, b:
  father(a, b) implies
  (son(b, a) or daughter(b, a))
)
Architecture of Scallop Compiler

```
rel constraint() =
  forall(a, b:
    father(a, b) implies
    (son(b, a) or daughter(b, a))
  )
```

daugther_{1,0} ← \pi_{a,b}(daughter)
son_{1,0} ← \pi_{a,b}(son)
agg_body ← (father – daughter_{1,0}) – son_{1,0}
constraint ← \pi_{x}()(\sigma_{x.x=false}(\gamma exists(agg_body)))

Scallop Program
.scl file written by user

Parser
Abstract Syntax Tree (AST)

Front-IR Program
- Type Inference
- Syntax Desugar
- Boundness Analysis
...

F2B
Back-IR Program
- Query Planning
- Constant Fold & Prop
- Magic-set Transformation
...

B2R
SCLRAM Program
- Rule Deduplication
- Projection Cascade
...

Page 19
Semantics and Provenance Framework

- The formal semantics of SCLRAM is parameterized by a provenance structure inspired by the theory of Provenance Semirings [PODS’07]

- A Provenance Structure is an algebraic structure that specifies:
  - **Tag Space**: the space of additional information associated with each tuple
  - **Operations**: how tags propagate during execution

<table>
<thead>
<tr>
<th>Abstract Provenance</th>
<th>( \text{max-min-prob(mmp)} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Tag Space) ( t \in T )</td>
<td>([0, 1])</td>
</tr>
<tr>
<td>(False) ( 0 \in T )</td>
<td>0</td>
</tr>
<tr>
<td>(True) ( 1 \in T )</td>
<td>1</td>
</tr>
<tr>
<td>(Disjunction) ( \oplus : T \times T \rightarrow T )</td>
<td>( \text{max} )</td>
</tr>
<tr>
<td>(Conjunction) ( \otimes : T \times T \rightarrow T )</td>
<td>( \text{min} )</td>
</tr>
<tr>
<td>(Negation) ( \ominus : T \rightarrow T )</td>
<td>( \lambda p.(1 - p) )</td>
</tr>
<tr>
<td>(Saturation) ( \ominus : T \times T \rightarrow \text{Bool} )</td>
<td>( == )</td>
</tr>
</tbody>
</table>
Provenance Framework: An Example

Scallop program

\[ \text{rel } \text{safe}_\text{cell}(x, y) = \text{grid}_\text{cell}(x, y) \text{ and not } \text{enemy}(x, y) \]

SCLRam program

\[ \text{safe}_\text{cell} \leftarrow \text{grid}_\text{cell} \text{ - enemy} \]

Untagged Semantics
Provenance Framework: An Example

Scallop program
rel safe_cell(x, y) = grid_cell(x, y) and not enemy(x, y)

SclRam program
safe_cell ← grid_cell - enemy

---

Untagged Semantics

Tagged Semantics
Provenance Framework: An Example

Scallop program

\[ \text{rel safe_cell}(x, y) = \text{grid_cell}(x, y) \text{ and not enemy}(x, y) \]

SclRam program

\[ \text{safe_cell} \leftarrow \text{grid_cell} - \text{enemy} \]

---

Untagged Semantics

(1,2)  
(2,3)  

\[ [\text{grid cell}]_F \]

\[ [\text{grid cell} - \text{enemy}]_F \]

(1,2)

---

Tagged Semantics with mmp

\[ [\text{grid cell}]_{F_{\text{mmp}}} \]

\[ [\text{enemy}]_{F_{\text{mmp}}} \]

\[ 0.9 :: (1,2) \]
\[ 0.9 :: (2,3) \]
\[ 0.2 :: (2,3) \]

\[ \min(0.9, 1 - 0.2) = 0.8 :: (2,3) \]
## Built-in Library of Provenance Structures

<table>
<thead>
<tr>
<th>Kind</th>
<th>Provenance</th>
<th>$T$</th>
<th>0</th>
<th>1</th>
<th>$\oplus$</th>
<th>$\otimes$</th>
<th>$\ominus$</th>
<th>$\ominus$</th>
<th>$\tau$</th>
<th>$\rho$</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Discrete</strong></td>
<td>unit</td>
<td>${()}$</td>
<td>()</td>
<td>()</td>
<td>$\lambda t_1, t_2.$</td>
<td>$\lambda t_1, t_2.$</td>
<td>$\lambda a.\text{FAIL}$</td>
<td>$\equiv$</td>
<td>$\lambda i.$</td>
<td>$\lambda t.$</td>
</tr>
<tr>
<td></td>
<td>bool</td>
<td>${\top, \bot}$</td>
<td>$\bot$</td>
<td>$\top$</td>
<td>$\lor$</td>
<td>$\land$</td>
<td>$\neg$</td>
<td>$\equiv$</td>
<td>$\equiv$</td>
<td>$\equiv$</td>
</tr>
<tr>
<td></td>
<td>natural</td>
<td>$\mathbb{N}$</td>
<td>0</td>
<td>1</td>
<td>$+$</td>
<td>$\times$</td>
<td>$\lambda n.{ n &gt; 0 }$</td>
<td>$\equiv$</td>
<td>$\equiv$</td>
<td>$\equiv$</td>
</tr>
<tr>
<td><strong>Probabilistic</strong></td>
<td>max-min-prob</td>
<td>$[0, 1]$</td>
<td>0</td>
<td>1</td>
<td>$\text{max}$</td>
<td>$\text{min}$</td>
<td>$\lambda t_1 - t$</td>
<td>$=\equiv$</td>
<td>$\equiv$</td>
<td>$\equiv$</td>
</tr>
<tr>
<td></td>
<td>add-mult-prob</td>
<td>$[0, 1]$</td>
<td>0</td>
<td>1</td>
<td>$\lambda t_1, t_2.\text{clamp}(t_1 + t_2)$</td>
<td>$\lambda t_1, t_2.\cdot(t_1 \cdot t_2)$</td>
<td>$\lambda t_1, t_2.\cdot(t_1 \cdot t_2)$</td>
<td>$\equiv$</td>
<td>$\equiv$</td>
<td>$\equiv$</td>
</tr>
<tr>
<td></td>
<td>rand-min-prob</td>
<td>$[0, 1]$</td>
<td>0</td>
<td>1</td>
<td>$\lambda t_1, t_2.\cdot(1 - t_1)(1 - t_2)$</td>
<td>$\lambda t_1, t_2.\cdot(1 - t_1)(1 - t_2)$</td>
<td>$\lambda t_1, t_2.\cdot(1 - t_1)(1 - t_2)$</td>
<td>$\equiv$</td>
<td>$\equiv$</td>
<td>$\equiv$</td>
</tr>
<tr>
<td></td>
<td>rand-mult-prob</td>
<td>$[0, 1]$</td>
<td>0</td>
<td>1</td>
<td>$\lambda t_1, t_2.\cdot(1 - t_1)(1 - t_2)$</td>
<td>$\lambda t_1, t_2.\cdot(1 - t_1)(1 - t_2)$</td>
<td>$\lambda t_1, t_2.\cdot(1 - t_1)(1 - t_2)$</td>
<td>$\equiv$</td>
<td>$\equiv$</td>
<td>$\equiv$</td>
</tr>
<tr>
<td></td>
<td>top-k-proofs</td>
<td>$\Phi$</td>
<td>$\emptyset$</td>
<td>${\emptyset}$</td>
<td>$\lor_{\text{top-k}}$</td>
<td>$\land_{\text{top-k}}$</td>
<td>$\neg_{\text{top-k}}$</td>
<td>$\equiv$</td>
<td>$\equiv$</td>
<td>$\equiv$</td>
</tr>
<tr>
<td></td>
<td>sample-k-proofs</td>
<td>$\Phi$</td>
<td>$\emptyset$</td>
<td>${\emptyset}$</td>
<td>$\lor_{\text{sample-k}}$</td>
<td>$\land_{\text{sample-k}}$</td>
<td>$\neg_{\text{sample-k}}$</td>
<td>$\equiv$</td>
<td>$\equiv$</td>
<td>$\equiv$</td>
</tr>
<tr>
<td><strong>Differentiable</strong></td>
<td>diff-max-min-prob</td>
<td>$D$</td>
<td>0</td>
<td>1</td>
<td>$\text{max}$</td>
<td>$\text{min}$</td>
<td>$\lambda i. - i$</td>
<td>$=\equiv$</td>
<td>$\equiv$</td>
<td>$\equiv$</td>
</tr>
<tr>
<td></td>
<td>diff-add-mult-prob</td>
<td>$D$</td>
<td>0</td>
<td>1</td>
<td>$\lambda i_1, i_2.\text{clamp}(i_1 + i_2)$</td>
<td>$\lambda i_1, i_2, i_1 \cdot i_2$</td>
<td>$\lambda i_1, i_2, i_1 \cdot i_2$</td>
<td>$\equiv$</td>
<td>$\equiv$</td>
<td>$\equiv$</td>
</tr>
<tr>
<td></td>
<td>diff-rand-min-prob</td>
<td>$[\hat{0}, 1]$</td>
<td>$\hat{0}$</td>
<td>$\hat{1}$</td>
<td>$\lambda i_1, i_2, - (\hat{1} - i_1)(\hat{1} - i_2)$</td>
<td>$\lambda i_1, i_2.\cdot(1 - i_1)(1 - i_2)$</td>
<td>$\lambda i_1, i_2.\cdot(1 - i_1)(1 - i_2)$</td>
<td>$\equiv$</td>
<td>$\equiv$</td>
<td>$\equiv$</td>
</tr>
<tr>
<td></td>
<td>diff-rand-mult-prob</td>
<td>$[\hat{0}, 1]$</td>
<td>$\hat{0}$</td>
<td>$\hat{1}$</td>
<td>$\lambda i_1, i_2, - (\hat{1} - i_1)(\hat{1} - i_2)$</td>
<td>$\lambda i_1, i_2.\cdot(1 - i_1)(1 - i_2)$</td>
<td>$\lambda i_1, i_2.\cdot(1 - i_1)(1 - i_2)$</td>
<td>$\equiv$</td>
<td>$\equiv$</td>
<td>$\equiv$</td>
</tr>
<tr>
<td></td>
<td>diff-top-k-proofs</td>
<td>$\Phi$</td>
<td>$\emptyset$</td>
<td>${\emptyset}$</td>
<td>$\lor_{\text{top-k}}$</td>
<td>$\land_{\text{top-k}}$</td>
<td>$\neg_{\text{top-k}}$</td>
<td>$\equiv$</td>
<td>$\equiv$</td>
<td>$\equiv$</td>
</tr>
<tr>
<td></td>
<td>diff-sample-k-proofs</td>
<td>$\Phi$</td>
<td>$\emptyset$</td>
<td>${\emptyset}$</td>
<td>$\lor_{\text{sample-k}}$</td>
<td>$\land_{\text{sample-k}}$</td>
<td>$\neg_{\text{sample-k}}$</td>
<td>$\equiv$</td>
<td>$\equiv$</td>
<td>$\equiv$</td>
</tr>
</tbody>
</table>
Syntax and semantics of Scallop programs remains familiar to users. The provenance framework allows to customize learning performance and scalability via a rich and extensible library.
**Evaluation**

**MNIST-R** 60K
- \( \text{sum2}(3, 3) \rightarrow 5 \)
- \( \text{sum3}(3, 4, 7) \rightarrow 12 \)
- \( \text{sum4}(3, 4, 7, 5) \rightarrow 17 \)
- \( \text{less-than}(3, 4) \rightarrow \text{false} \)
- \( \text{not-3-or-4}(5) \rightarrow \text{true} \)
- \( \text{count-3}(3, 5, \ldots, 7) \rightarrow 1 \)
- \( \text{count-3-or-4}(4, 3, \ldots, 5) \rightarrow 2 \)

8 images

**CLUTRR** 10K
- Output: Kinship Relation

**Passage:** Rich's daughter Christine made dinner for her sister Kim. Beth went to her brother Rich's birthday party. Anne went shopping with her sister Kim.

**Query:** Rich is Anne's ...?

**Answer:** Father

**Structured Kinship Graph** (CLUTRR-G only)

**Mugen** 1K
- Output: Aligned?

**Video:**
- Video clips

**Text:** Mugen climbs up on a ladder, and walks to the right and collects a few coins

**Aligned?: true**

**HWF** 10K
- Output: Answer

\[ 1 + 3 - 5 \rightarrow 1.6 \]

**CLEVR** 50K
- Output: Answer

**Image:** (on the right)

**Question:** How many objects are there behind the purple cube?

**Answer:** 3

**VQAR** 10K
- Output: Object ID

**Image:** (on the right)

**KB:** is_a(giraffe, mammal)
- is_a(mammal, animal)

... (3,390 axioms)

**Programmatic Query:**
- target(o) = name(o, "animal"), left(o, op), attr(o, "tall")

**Answer:** 012
Benchmark Suite

Involves Computer Vision
(Images & Videos)

<table>
<thead>
<tr>
<th>Benchmark Suite</th>
<th>Data Size</th>
<th>Output Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>MNIST-R</strong></td>
<td>60K</td>
<td>Summation</td>
</tr>
<tr>
<td><strong>CLUTRR</strong></td>
<td>10K</td>
<td>Kinship Relation</td>
</tr>
<tr>
<td><strong>Mugen</strong></td>
<td>1K</td>
<td>Video Alignment</td>
</tr>
<tr>
<td><strong>HWF</strong></td>
<td>10K</td>
<td>Answer</td>
</tr>
<tr>
<td><strong>CLEVR</strong></td>
<td>50K</td>
<td>Answer</td>
</tr>
<tr>
<td><strong>VQAR</strong></td>
<td>10K</td>
<td>Object ID</td>
</tr>
<tr>
<td><strong>Pathfinder</strong></td>
<td>600K</td>
<td>Path?</td>
</tr>
</tbody>
</table>

**MNIST-R** examples:
- \( \text{sum2}(3, 2) \rightarrow 5 \)
- \( \text{sum3}(3, 2, 7) \rightarrow 12 \)
- \( \text{sum4}(3, 2, 7, 9) \rightarrow 17 \)
- \( \text{less-than}(3, 3) \rightarrow \text{false} \)
- \( \text{not-3-or-4}(5) \rightarrow \text{true} \)
- \( \text{count-3}(2, 5, \ldots, 7) \rightarrow 1 \)
- \( \text{count-3-or-4}(4, 3, \ldots, 3) \rightarrow 2 \)

**CLUTRR** passage:

**Query:** Rich is Anne’s ...?
**Answer:** Father

**Mugen** video:
- Video: Mugen climbs up on a ladder, and walks to the right and collects a few coins.
- Aligned?: true

**HWF** example:
- \( 1 + 3 - 5 \rightarrow 1.6 \)

**CLEVR** question:
- **Question:** How many objects are there behind the purple cube?
  - **Answer:** 3

**VQAR** image:
- **Image:** (on the right)
- **KB:** is_a(giraffe, mammal)
- **Programmatic Query:** target(o) = name(o, "animal"), left(o, op), attr(o, "tall")
  - **Answer:** o12
Benchmark Suite

Involves Natural Language Processing
(Natural Text)
Benchmark Suite

Requires Multi-Modal Capability
(Combination of CV & NLP)
Performance: Scallop vs. Baselines

Testing Accuracy (%) on Selected Benchmark Tasks
Scallop 👫 Foundation Models
Foundation Models

- Claude 2
- ANTHROP\C
- OWL-ViT
- T5
- OpenAI CLIP
- Stable Diffusion
- ViLT
- LLaMA-2
- ResNet
- DSFD
- Midjourney
- GitHub Copilot
- Segment Anything
- Research by Meta AI
- S3D
What is the right abstraction layer to program the foundation models?
Context:

[Cristina] was afraid of heights just like her daughters, [Sheila] and [Diana]. However, [Diana]'s father, [Jonathan], loved heights and even went skydiving a few times. [Ruth] and her son, [Jeremy], went to the park, and had a wonderful time. [Jeremy] went to the bakery with his uncle [Jonathan] to pick up some bread for lunch.

Question:

What is the relationship between Ruth and Sheila?
Relational Knowledge Extraction with GPT

Context:
[Cristina] was afraid of heights just like her daughters, [Sheila] and [Diana]. However, [Diana]'s father, [Jonathan], loved heights and even went skydiving a few times. [Ruth] and her son, [Jeremy], went to the park, and had a wonderful time. [Jeremy] went to the bakery with his uncle [Jonathan] to pick up some bread for lunch.
Relational Knowledge Extraction with GPT

Context:

[Cristina] was afraid of heights just like her daughters, [Sheila] and [Diana]. However, [Diana]'s father, [Jonathan], loved heights and even went skydiving a few times. [Ruth] and her son, [Jeremy], went to the park, and had a wonderful time. [Jeremy] went to the bakery with his uncle [Jonathan] to pick up some bread for lunch.
Relational Knowledge Extraction with GPT

```python
@gpt_extract_relation(
    prompt="Please extract the kinship relationships from the context: ",
    examples=[["Alice is Bob’s mother", ["alice", "bob", "son"], ...]], ...
)
type parse_relations(bound context: String, sub: String, obj: String, rela: String), ...
```
Relational Knowledge Extraction with GPT

Context: [Cristina] was afraid of heights just like her daughters, [Sheila] and [Diana]. However, [Diana]'s father, [Jonathan], loved heights and even went skydiving a few times. [Ruth] and her son, [Jeremy], went to the park, and had a wonderful time. [Jeremy] went to the bakery with his uncle [Jonathan] to pick up some bread for lunch. What is the relationship between Sheila and Ruth?

@gpt_extract_relation(  
prompt="Please extract the kinship relationships from the context:",  
examples=[("Alice is Bob’s mother", ["alice", "bob", "son"], ...), ...])  
type parse_relations(bound context: String, sub: String, obj: String, rela: String), ...
Relational Knowledge Extraction with GPT

@gpt_extract_relation(
    prompt="Please extract the kinship relationships from the context:",
    examples=[("Alice is Bob’s mother", ["alice", "bob", "son"], ...), ...])
type parse_relations(bound context: String, sub: String, obj: String, rela: String), ...

<table>
<thead>
<tr>
<th>sub</th>
<th>obj</th>
<th>rela</th>
</tr>
</thead>
<tbody>
<tr>
<td>cristina</td>
<td>diana</td>
<td>daughter</td>
</tr>
<tr>
<td>jeremy</td>
<td>jonathan</td>
<td>uncle</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
@gpt_extract_relation(
    prompt="Please extract the kinship relationships from the context:",
    examples=[("Alice is Bob’s mother", [[“alice”, “bob”, “son”], ...]), ...])

rel kinship(p1,p2,rela) = context(ctx) and parse_relations(ctx,p1,p2,rela)
rel kinship(p1,p3,r3) = kinship(p1,p2,r1) and kinship(p2,p3,r2) and composition(r1,r2,r3)
rel answer(r) = question(p1,p2) and kinship(p1,p2,r)
Image Classification as Probabilistic Relation

@clip_classifier(['cat', 'dog'])

type cat_or_dog(
    bound img: Tensor,
    free label: String,
)

OpenAI CLIP
Image Classification as Probabilistic Relation

<table>
<thead>
<tr>
<th>id</th>
<th>image</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>![Dog Image]</td>
</tr>
<tr>
<td>1</td>
<td>![Cat Image]</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

@clip_classifier(["cat","dog"]) type cat_or_dog(
  bound img: Tensor,
  free label: String,
)
Image Classification as Probabilistic Relation

```python
@clip_classifier(['cat', 'dog'])
type cat_or_dog(
    bound img: Tensor,
    free label: String,
)
```

<table>
<thead>
<tr>
<th>id</th>
<th>image</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>![dog image]</td>
</tr>
<tr>
<td>1</td>
<td>![cat image]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>prob</th>
<th>id</th>
<th>label</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.00</td>
<td>0</td>
<td>cat</td>
</tr>
<tr>
<td>0.99</td>
<td>0</td>
<td>dog</td>
</tr>
<tr>
<td>0.98</td>
<td>1</td>
<td>cat</td>
</tr>
<tr>
<td>0.02</td>
<td>1</td>
<td>dog</td>
</tr>
</tbody>
</table>

... ... ...
Image Segmentation as Probabilistic Relation

@segmentanything
type image_segment(
    bound img: Tensor,
    free id: u32,
    free segment: Tensor,
)

Segment Anything
Research by Meta AI
Image Segmentation as Probabilistic Relation

```rust
@segmentAnything

type imageSegment(
    bound img: Tensor,
    free id: u32,
    free segment: Tensor,
)
```
Image Segmentation as Probabilistic Relation

@segment Anything

type image_segment(
    bound img: Tensor,
    free id: u32,
    free segment: Tensor,
)

<table>
<thead>
<tr>
<th>prob</th>
<th>id</th>
<th>segment</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.99</td>
<td>0</td>
<td><img src="image" alt="Green Cube" /></td>
</tr>
<tr>
<td>0.98</td>
<td>1</td>
<td><img src="image" alt="Purple Cube" /></td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Combining Foundation Models

@segment_anything
type image_segment(
    bound img: Tensor,
    free id: u32,
    free segment: Tensor)

@clip_classifier(["green","red",...])
type obj_color(
    bound object_segment: Tensor,
    free label: String)

@gpt_complete(prompt=
    "Please semantically parse the
    following question...")
type semantic_parse(
    bound question: String,
    free answer: Expr)

Question: How many green objects are there in the image?

<table>
<thead>
<tr>
<th>prob</th>
<th>count</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.00</td>
<td>0</td>
</tr>
<tr>
<td>0.03</td>
<td>1</td>
</tr>
<tr>
<td>0.02</td>
<td>2</td>
</tr>
<tr>
<td>0.91</td>
<td>3</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Evaluation