Neurosymbolic Learning

CIS 7000: Trustworthy Machine Learning, Apr 17
Ziyang Li, University of Pennsylvania
Joint Work With:
Jason Liu, Felix Zhu, Eric Zhao, William Dodds,
Neelay Velingker, Rajeev Alur
The need for Neurosymbolic AI
Two separate paradigms of programming

<table>
<thead>
<tr>
<th>Classical Algorithms</th>
<th>Deep Learning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Suited for exactly defined tasks on structured input domains</td>
<td>Suited for tasks which cannot be hand-programmed or have unstructured input</td>
</tr>
</tbody>
</table>
Two separate paradigms of programming

**Classical Algorithms**

Suited for exactly defined tasks on structured input domains

- Sort a list of numbers
- Find shortest path
- Solve boolean constraints

**Deep Learning**

Suited for tasks which cannot be hand-programmed or have unstructured input
Two separate paradigms of programming

Classical Algorithms

Suited for exactly defined tasks on structured input domains

- Sort a list of numbers
- Find shortest path
- Solve boolean constraints

Deep Learning

Suited for tasks which cannot be hand-programmed or have unstructured input

- Detecting objects in image
- Parse natural language text
- Control in physical environment
Neurosymbolic to combine both worlds...

Classical Algorithms

- Suited for exactly defined tasks on structured input domains

Deep Learning

- Suited for tasks which cannot be hand-programmed or have unstructured input

symbolic + neural = neurosymbolic
Neurosymbolic Learning...

= 

Machine learning with both neural and symbolic components
An Example of Neurosymbolic Learning
Neurosymbolic Learning of $\text{addition}(3, 5, 8)$
Neurosymbolic Learning of \(\text{addition}(3, 5, 8)\)

Logic Program

Neural Model

A Neurosymbolic Program
Neurosymbolic Learning of addition(3, 5, 8)

Logic Program

Neural Model

A Neurosymbolic Program
Neurosymbolic Learning of \( \text{addition}(3, 5, 8) \)

Logic Program

\[
\begin{align*}
& r & \rightarrow & p & \rightarrow & y \\
\end{align*}
\]

Neural Model

\[
\begin{align*}
& x & \rightarrow & M_\theta & \rightarrow & y \\
& \frac{\partial y}{\partial \theta} & \rightarrow & r \\
\end{align*}
\]

A Neurosymbolic Program

\[
\begin{align*}
& x & \rightarrow & M_\theta & \rightarrow & r & \rightarrow & p & \rightarrow & y \\
& \frac{\partial r}{\partial \theta} & \rightarrow & \frac{\partial y}{\partial r} \\
\end{align*}
\]
Neurosymbolic Learning of $\text{addition}(3, 5, 8)$

$3 + 5 \rightarrow 8$

$\text{rel addition}(a + b) = \text{digit}_1(a) \text{ and } \text{digit}_2(b)$

Logic Program

Neural Model

A Neurosymbolic Program
Neurosymbolic Learning of $\text{addition}(\overline{3}, \overline{5}, 8)$

**Relational Language**

$$\text{rel addition}(a + b) = \text{digit}_1(a) \text{ and } \text{digit}_2(b)$$

**Scallop (Li et. al., Datalog like syntax)**

```
addition(r) :- digit_1(a), digit_2(b), r is a + b
```

**DeepProblog (Manheave et. al., Prolog like syntax)**

```python
@ised.blackbox(a=IntRange(10), b=IntRange(10), output=IntRange(19))
def addition(a, b): return a + b
```

**ISED (Breslin et. al., Python like syntax)**
Neurosymbolic Learning of \( \text{addition}(3, 5, 8) \)

Neurosymbolic Frameworks aim to provide a programming interface for developers to write Neurosymbolic solutions.

A Neurosymbolic Program
A More Difficult Motivating Example: PacMan-Maze
A Motivating Example: PacMan-Maze

State: 200x200 colored image
Action: Up, Down, Left, Right
(Environments are 5x5 grids randomized for each session)
How to combine neural and symbolic components?

1. What is the **neural** component?
   - CNN that parses the image into entity positions

2. What does the **symbolic** program do?
   - Plan the optimal action to take at each state, given the possibly noisy entity positions

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

(Environments are 5x5 grids randomized for each session)
How to combine neural and symbolic components?

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

(Environments are 5x5 grids randomized for each session)
Results after combining neural and symbolic

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

<table>
<thead>
<tr>
<th></th>
<th><strong>Neurosymbolic</strong> (with Scallop)</th>
<th><strong>DQN</strong></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</td>
<td></td>
<td></td>
</tr>
<tr>
<td>within 50 steps)</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong># of Training</strong></td>
<td>50</td>
<td>50K</td>
</tr>
<tr>
<td><strong>episodes</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(to achieve the</td>
<td></td>
<td></td>
</tr>
<tr>
<td>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)
A Motivating Example: PacMan-Maze

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

(Environments are 5x5 grids randomized for each session)
A Motivating Example: PacMan-Maze

class EntityExtractor(nn.Module):
    def __init__(self):
        super(EntityExtractor, self).__init__()
        self.conv1 = nn.Conv2d(...)
        self.conv2 = nn.Conv2d(...)
        self.fc1 = nn.Linear(in_features=288, out_features=256)
        self.fc2 = nn.Linear(in_features=256, out_features=4)
        self.relu = nn.ReLU()

    def forward(self, x):
        batch_size, _, _, _ = x.shape
        x = self.relu(self.conv1(x))
        x = self.relu(self.conv2(x))
        x = x.view(batch_size, -1)
        x = self.fc2(self.relu(self.fc1(x)))
        return torch.softmax(x, dim=1)
A Motivating Example: PacMan-Maze

```
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)
```
Demo – Training the Agent!

Marker on the top-left indicating NN prediction on what the cell represents (Saturation indicates confidence)

- **Enemy (Red)**
- **PacMan (Blue)**
- **Goal (Green)**
Demo – Testing the Agent!

Marker on the top-left indicating NN prediction on what the cell represents (Saturation indicates confidence)

- **Enemy (Red)**
- **PacMan (Blue)**
- **Goal (Green)**
Key Take Aways

Decomposing end-to-end neural solutions into separated perception + reasoning solutions...

1. **Improves accuracy**
   Reasoning module is generalizable to combinatorically diverse situations

2. **Learns faster**
   Neural models are good at recognizing patterns; and they are used only for it

3. **Is explainable**
   Decomposition gives explicit meaning to intermediate information
Differentiating Symbolic Programs
Back to adding two MNIST digits...

A Neurosymbolic Program
Using Neural Network to Classify One MNIST Digit
Training Loop for MNIST Addition Task

Input \((x)\)

\[
\begin{align*}
2 & \\
5 & 
\end{align*}
\]

\(\Rightarrow\)

???

\(\Rightarrow\)

Ground Truth \((y)\)

\[
\begin{align*}
0 & \\
1 & \\
2 & \\
\ldots & \\
7 & \bullet \\
\ldots & \\
15 & \\
16 & \\
17 & \\
18 & 
\end{align*}
\]

Task Initially Proposed by DeepProbLog (R Manhaeve et. al, 2018)
Training Loop for MNIST Addition Task

Input \((x)\) --> Neural Network (???) --> Ground Truth \((y)\)

Input: 2, 5

Ground Truth: 7
Training Loop for MNIST Addition Task

Input \((x)\)  \hspace{1cm} \text{Parameters} \ (\theta) 

\[ \begin{array}{c}
2 \\
5 \\
\end{array} \hspace{1cm} \text{Neural Network} \]
Training Loop for MNIST Addition Task

Input \( (x) \) → Neural Network

Parameters \( (\theta) \)
Training Loop for MNIST Addition Task

Input ($x$)

Parameters ($\theta$)

Neural Network

<table>
<thead>
<tr>
<th>Input</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0.02 : digit_1(0)</td>
</tr>
<tr>
<td>5</td>
<td>0.02 : digit_1(1)</td>
</tr>
<tr>
<td></td>
<td>0.87 : digit_1(2)</td>
</tr>
<tr>
<td></td>
<td>0.01 : digit_1(3)</td>
</tr>
<tr>
<td></td>
<td>0.02 : digit_1(4)</td>
</tr>
<tr>
<td></td>
<td>0.01 : digit_1(5)</td>
</tr>
<tr>
<td></td>
<td>0.01 : digit_1(6)</td>
</tr>
<tr>
<td></td>
<td>0.02 : digit_1(7)</td>
</tr>
<tr>
<td></td>
<td>0.01 : digit_1(8)</td>
</tr>
<tr>
<td></td>
<td>0.01 : digit_1(9)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Input</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.02 : digit_2(0)</td>
</tr>
<tr>
<td></td>
<td>0.01 : digit_2(1)</td>
</tr>
<tr>
<td></td>
<td>0.01 : digit_2(2)</td>
</tr>
<tr>
<td></td>
<td>0.01 : digit_2(3)</td>
</tr>
<tr>
<td></td>
<td>0.02 : digit_2(4)</td>
</tr>
<tr>
<td></td>
<td>0.88 : digit_2(5)</td>
</tr>
<tr>
<td></td>
<td>0.01 : digit_2(6)</td>
</tr>
<tr>
<td></td>
<td>0.02 : digit_2(7)</td>
</tr>
<tr>
<td></td>
<td>0.01 : digit_2(8)</td>
</tr>
<tr>
<td></td>
<td>0.01 : digit_2(9)</td>
</tr>
</tbody>
</table>
Training Loop for MNIST Addition Task

\[ \text{rel addition}(x + y) = \text{digit}_1(x) \text{ and } \text{digit}_2(y) \]
Training Loop for MNIST Addition Task

Input \((x)\)

**Neural Network**

Parameters \((\theta)\)

Prediction \((y_{pred}/y^*)\)

Output

Scallop

<table>
<thead>
<tr>
<th>parameter</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\text{digit}_1(0))</td>
<td>0.02</td>
</tr>
<tr>
<td>(\text{digit}_1(1))</td>
<td>0.02</td>
</tr>
<tr>
<td>(\text{digit}_1(2))</td>
<td>0.87</td>
</tr>
<tr>
<td>(\text{digit}_1(3))</td>
<td>0.01</td>
</tr>
<tr>
<td>(\text{digit}_1(4))</td>
<td>0.02</td>
</tr>
<tr>
<td>(\text{digit}_1(5))</td>
<td>0.01</td>
</tr>
<tr>
<td>(\text{digit}_1(6))</td>
<td>0.01</td>
</tr>
<tr>
<td>(\text{digit}_1(7))</td>
<td>0.02</td>
</tr>
<tr>
<td>(\text{digit}_1(8))</td>
<td>0.01</td>
</tr>
<tr>
<td>(\text{digit}_1(9))</td>
<td>0.01</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>parameter</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\text{digit}_2(0))</td>
<td>0.02</td>
</tr>
<tr>
<td>(\text{digit}_2(1))</td>
<td>0.01</td>
</tr>
<tr>
<td>(\text{digit}_2(2))</td>
<td>0.01</td>
</tr>
<tr>
<td>(\text{digit}_2(3))</td>
<td>0.01</td>
</tr>
<tr>
<td>(\text{digit}_2(4))</td>
<td>0.02</td>
</tr>
<tr>
<td>(\text{digit}_2(5))</td>
<td>0.88</td>
</tr>
<tr>
<td>(\text{digit}_2(6))</td>
<td>0.01</td>
</tr>
<tr>
<td>(\text{digit}_2(7))</td>
<td>0.02</td>
</tr>
<tr>
<td>(\text{digit}_2(8))</td>
<td>0.01</td>
</tr>
<tr>
<td>(\text{digit}_2(9))</td>
<td>0.01</td>
</tr>
</tbody>
</table>

Page 37
Intermediate Representation

\( \mathbf{r} \)

Input \((x)\)

Parameters \((\theta)\)

Neural Network

Scallop

Prediction \((y_{\text{pred}}/ y^*)\)
Intermediate Representation ($ r $)

Prediction ($ y_{pred} / y^* $)

Neural Network

Parameters ($ \theta $)

Input ($ x $)

Scallop

\[
\frac{\partial y}{\partial r}
\]
Neural Network Parameters

Input \( (x) \)  

Parameters \( (\theta) \)  

Intermediate Representation \( (r) \)  

Prediction \( (y_{pred}/y^*) \)  

Neural Network

Scallop
Question 1 (Forward)
How do we get the result distribution?
Neural Network Parameters

Question 2 (Backward)
How do we get the gradient?

Input (
\( x \))

Parameters (
\( \theta \))

Intermediate Representation

Prediction
(\( y_{\text{pred}} / y^* \))

Scallop
Question 1: Forward probability estimation

A Neurosymbolic Program
Question 1: Forward probability estimation

\[
\begin{array}{ccc}
0 & \text{Pr}(\text{digit1} = 0) & 0.02 \\
1 & \text{Pr}(\text{digit1} = 1) & 0.02 \\
2 & \text{Pr}(\text{digit1} = 2) & 0.87 \\
3 & \text{Pr}(\text{digit1} = 3) & 0.01 \\
4 & \text{Pr}(\text{digit1} = 4) & 0.02 \\
5 & \text{Pr}(\text{digit1} = 5) & 0.01 \\
6 & \text{Pr}(\text{digit1} = 6) & 0.01 \\
7 & \text{Pr}(\text{digit1} = 7) & 0.02 \\
8 & \text{Pr}(\text{digit1} = 8) & 0.01 \\
9 & \text{Pr}(\text{digit1} = 9) & 0.01 \\
\end{array}
\]

\[
\begin{array}{ccc}
0 & \text{Pr}(\text{digit2} = 0) & 0.02 \\
1 & \text{Pr}(\text{digit2} = 1) & 0.01 \\
2 & \text{Pr}(\text{digit2} = 2) & 0.01 \\
3 & \text{Pr}(\text{digit2} = 3) & 0.01 \\
4 & \text{Pr}(\text{digit2} = 4) & 0.02 \\
5 & \text{Pr}(\text{digit2} = 5) & 0.88 \\
6 & \text{Pr}(\text{digit2} = 6) & 0.01 \\
7 & \text{Pr}(\text{digit2} = 7) & 0.02 \\
8 & \text{Pr}(\text{digit2} = 8) & 0.01 \\
9 & \text{Pr}(\text{digit2} = 9) & 0.01 \\
\end{array}
\]
Question 1: Forward probability estimation

Pr(digit1 = 0) = 0.02
Pr(digit1 = 1) = 0.02
Pr(digit1 = 2) = 0.87
Pr(digit1 = 3) = 0.01
Pr(digit1 = 4) = 0.02
Pr(digit1 = 5) = 0.01
Pr(digit1 = 6) = 0.01
Pr(digit1 = 7) = 0.02
Pr(digit1 = 8) = 0.01
Pr(digit1 = 9) = 0.01

Pr(digit2 = 0) = 0.02
Pr(digit2 = 1) = 0.01
Pr(digit2 = 2) = 0.01
Pr(digit2 = 3) = 0.01
Pr(digit2 = 4) = 0.02
Pr(digit2 = 5) = 0.88
Pr(digit2 = 6) = 0.01
Pr(digit2 = 7) = 0.02
Pr(digit2 = 8) = 0.01
Pr(digit2 = 9) = 0.01

Pr(addition = 0) = ?
Question 1: Forward probability estimation

\[
\text{Pr}(\text{addition} = 0) = \text{Pr}(\text{digit1} = 0) \times \text{Pr}(\text{digit2} = 0)
\]
Question 1: Forward probability estimation

\[
\begin{align*}
\text{Pr}(\text{digit1} = 0) &= 0.02 \\
\text{Pr}(\text{digit1} = 1) &= 0.02 \\
\text{Pr}(\text{digit1} = 2) &= 0.87 \\
\text{Pr}(\text{digit1} = 3) &= 0.01 \\
\text{Pr}(\text{digit1} = 4) &= 0.02 \\
\text{Pr}(\text{digit1} = 5) &= 0.01 \\
\text{Pr}(\text{digit1} = 6) &= 0.01 \\
\text{Pr}(\text{digit1} = 7) &= 0.02 \\
\text{Pr}(\text{digit1} = 8) &= 0.01 \\
\text{Pr}(\text{digit1} = 9) &= 0.01 \\
\text{Pr}(\text{digit2} = 0) &= 0.02 \\
\text{Pr}(\text{digit2} = 1) &= 0.01 \\
\text{Pr}(\text{digit2} = 2) &= 0.01 \\
\text{Pr}(\text{digit2} = 3) &= 0.01 \\
\text{Pr}(\text{digit2} = 4) &= 0.02 \\
\text{Pr}(\text{digit2} = 5) &= 0.88 \\
\text{Pr}(\text{digit2} = 6) &= 0.01 \\
\text{Pr}(\text{digit2} = 7) &= 0.02 \\
\text{Pr}(\text{digit2} = 8) &= 0.01 \\
\text{Pr}(\text{digit2} = 9) &= 0.01
\end{align*}
\]

Pr(addition = 1) = ???
Question 1: Forward probability estimation

\[
\Pr(\text{addition} = 1) = \Pr(\text{digit1} = 0) \times \Pr(\text{digit2} = 1) + \Pr(\text{digit1} = 1) \times \Pr(\text{digit2} = 0)
\]
Question 1: Forward probability estimation

<table>
<thead>
<tr>
<th>Value</th>
<th>Pr(digit1 = 0)</th>
<th>Pr(digit1 = 1)</th>
<th>Pr(digit1 = 2)</th>
<th>Pr(digit1 = 3)</th>
<th>Pr(digit1 = 4)</th>
<th>Pr(digit1 = 5)</th>
<th>Pr(digit1 = 6)</th>
<th>Pr(digit1 = 7)</th>
<th>Pr(digit1 = 8)</th>
<th>Pr(digit1 = 9)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.02 : \text{digit}_1(0)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0.02 : \text{digit}_1(1)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0.87 : \text{digit}_1(2)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0.01 : \text{digit}_1(3)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0.02 : \text{digit}_1(4)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>0.01 : \text{digit}_1(5)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>0.01 : \text{digit}_1(6)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>0.02 : \text{digit}_1(7)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>0.01 : \text{digit}_1(8)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>0.01 : \text{digit}_1(9)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Value</th>
<th>Pr(digit2 = 0)</th>
<th>Pr(digit2 = 1)</th>
<th>Pr(digit2 = 2)</th>
<th>Pr(digit2 = 3)</th>
<th>Pr(digit2 = 4)</th>
<th>Pr(digit2 = 5)</th>
<th>Pr(digit2 = 6)</th>
<th>Pr(digit2 = 7)</th>
<th>Pr(digit2 = 8)</th>
<th>Pr(digit2 = 9)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.02 : \text{digit}_2(0)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0.01 : \text{digit}_2(1)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0.91 : \text{digit}_2(2)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0.01 : \text{digit}_2(3)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0.02 : \text{digit}_2(4)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>0.88 : \text{digit}_2(5)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>0.01 : \text{digit}_2(6)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>0.02 : \text{digit}_2(7)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>0.01 : \text{digit}_2(8)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>0.01 : \text{digit}_2(9)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\[
\text{Pr(}\text{addition} = 2\text{)} = \text{???}\]
Question 1: Forward probability estimation

\[
\text{Pr}(\text{addition} = 2) = \text{Pr}(\text{digit1} = 0) \times \text{Pr}(\text{digit2} = 2) + \text{Pr}(\text{digit1} = 1) \times \text{Pr}(\text{digit2} = 1) + \text{Pr}(\text{digit1} = 2) \times \text{Pr}(\text{digit2} = 0) + 
\]
Question 1: Forward probability estimation

```
0  Pr(digit1 = 0)  0.02
1  Pr(digit1 = 1)  0.02
2  Pr(digit1 = 2)  0.02
3  Pr(digit1 = 3)  0.01
4  Pr(digit1 = 4)  0.02
5  Pr(digit1 = 5)  0.01
6  Pr(digit1 = 6)  0.01
7  Pr(digit1 = 7)  0.02
8  Pr(digit1 = 8)  0.01
9  Pr(digit1 = 9)  0.01
```

```
0  Pr(addition = 0)  0.02
1  Pr(addition = 1)  0.02
2  Pr(addition = 2)  0.87
3  Pr(addition = 3)  0.01
4  Pr(addition = 4)  0.01
5  Pr(addition = 5)  0.01
6  Pr(addition = 6)  0.02
7  Pr(addition = 7)  0.01
8  Pr(addition = 8)  0.01
9  Pr(addition = 9)  0.01
```
Question 2: Backward gradient estimation

A Neurosymbolic Program
\[ \frac{\partial y}{\partial r}^T \]

\[ \begin{align*}
\partial Pr(\text{addition} = 1) / \partial Pr(\text{digit1} = 0) \\
\partial Pr(\text{addition} = 1) / \partial Pr(\text{digit1} = 1) \\
\ldots \\
\partial Pr(\text{addition} = 1) / \partial Pr(\text{digit2} = 0) \\
\partial Pr(\text{addition} = 1) / \partial Pr(\text{digit2} = 1) \\
\ldots 
\end{align*} \]
\[ \frac{\partial y}{\partial r} \]

\[ r 
\]

\[ y^* \]

\[ \begin{align*} 
\frac{\partial \Pr(\text{addition} = 1)}{\partial \Pr(\text{digit1} = 0)} & = ??? \\
\Pr(\text{addition} = 1) & = \Pr(\text{digit1} = 0) \times \Pr(\text{digit2} = 1) + \\
& \quad \Pr(\text{digit1} = 1) \times \Pr(\text{digit2} = 0) 
\end{align*} \]

Note:
\[
\frac{\partial y}{\partial r} = \begin{pmatrix}
0.02 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot
\end{pmatrix}
\]

\[
y^* = \begin{pmatrix}
0 & 1 & 2 & \ldots & 7 & \ldots & 15 & 16 & 17 & 18
\end{pmatrix}
\]

\[
\partial \Pr(\text{addition} = 1) / \partial \Pr(\text{digit1} = 0) = \Pr(\text{digit2} = 1)
\]

**Note:**

\[
\Pr(\text{addition} = 1) = \Pr(\text{digit1} = 0) \times \Pr(\text{digit2} = 1) + \Pr(\text{digit1} = 1) \times \Pr(\text{digit2} = 0)
\]
The Questions Are Solved!

A Neurosymbolic Program
The Questions Are Solved! Or are we...?

A Neurosymbolic Program
The Questions Are Solved! Or are we...?

1. **Scalability Issue**
   
   Doing exact probabilistic estimation is too time consuming! We need to speed it up!
   
   **Solution:** approximation, heuristics, etc.

2. **Expressiveness Issue**
   
   This framework is too primitive, need to support more operations than simply summations.
   
   **Solution:** a programming language that supports negation, recursion, aggregations, etc.

A Neurosymbolic Program
Scallop, A Neurosymbolic Programming Language
Relational Programming and Scallop

Relational Programming
Datalog, Prolog, Scallop

rel grandmother(a, b) =
father(a, x) and mother(x, b)

Scallop, a Neurosymbolic Programming Language
Scallop: A Neurosymbolic Programming Language

Neurosymbolic Programming Language based on Datalog
Scallop: A Neurosymbolic Programming Language

Neurosymbolic Programming Language based on Datalog

Expressive Logic:
- Recursion
- Negation
- Aggregation
Scallop: A Neurosymbolic Programming Language

Neurosymbolic Programming Language based on Datalog

Expressive Logic:
- Recursion
- Negation
- Aggregation

Rich Reasoning Framework:
- Discrete
- Probabilistic
- Differentiable
Scallop: A Neurosymbolic Programming Language

- Neurosymbolic Programming Language based on Datalog
- Expressive Logic:
  - Recursion
  - Negation
  - Aggregation
- Foreign Interface:
  - Functions
  - Predicates
  - Aggregators
  - Attributes
- Rich Reasoning Framework:
  - Discrete
  - Probabilistic
  - Differentiable
Scallop: A Neurosymbolic Programming Language

Neurosymbolic Programming Language based on Datalog

Expressive Logic:
- Recursion
- Negation
- Aggregation

Foreign Interface:
- Predicates
- Attributes
- Aggregators

Rich Reasoning Framework:
- Discrete
- Probabilistic
- Differentiable

Goal: Provide a unified language for AI developers
Scallop has been applied to many ML tasks...
CLUTRR: Kinship Reasoning
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?
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?
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?

Answer:
Sheila is Ruth’s niece.
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.

For each pair of names, classify them into 21 types of kinship

Symbolic Context

0.951::context(DAUGHTER, “Cristina”, “Sheila”)
0.002::context(MOTHER, “Cristina”, “Sheila”)
0.004::context(FATHER, “Cristina”, “Sheila”)
...
0.001::context(NA, “Cristina”, “Sheila”)
0.942::context(DAUGHTER, “Christina”, “Diana”)
0.015::context(MOTHER, “Christina”, “Diana”)
...
0.002::context(NA, “Sheila”, “Diana”)
Mugen: Text/Video Retrieval
Mugen is a dataset containing multi-modality data (video, text, audio, etc)

In this task, we consider the video-text-alignment. The model takes in video and text, and returns whether they are aligned or not.

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

Aligned? Yes
Mugen using Scallop...

Video timeline

Text description

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

Scallop Program

0.99::action(FRAME0, “climb”); ...
0.85::action(FRAME5, “walk”); ...
0.85::mod(FRAME5, “collect-coin”); ...
0.01::action(FRAME10, “jump”); ...

expr(0, “climb”)
expr(1, “walk”)
expr(2, “collect-coin”)

0.99::match()
Mugen using Scallop...

Video timeline

rel match_single(tid, vid, vid + 1) = expr(tid, a), action(vid, a)
rel match_sub(tid, tid, vid_start, vid_end) =
    match_single(tid, vid_start, vid_end)
rel match_sub(tid, tid, vid_start, vid_end) =
    match_sub(tid, tid, vid_start, vid_mid), match_single(tid, vid_mid, vid_end)
rel match_sub(tid_start, tid_end, vid_start, vid_end) =
    match_sub(tid_start, tid_end - 1, vid_start, vid_mid),
    match_single(tid_end, vid_mid, vid_end)
rel match() = expr_start(tid_start), expr_end(tid_end), action_start(vid_start),
    action_end(vid_end), match_sub(tid_start, tid_end, vid_start, vid_end)

0.99::action(FRAME0, "climb"); ...
0.85::action(FRAME5, "walk"); ...
0.85::mod(FRAME5, "collect-coin"); ...
0.01::action(FRAME10, "jump"); ...
expr(0, "climb")
expr(1, "walk")
expr(2, "collect-coin")
0.99::match()
Mugen using Scallop...

In this process, Scallop helps to extract detailed actions from the video, providing better interpretability and explainability.
Mugen using Scallop...
Training video- and text-retrieval models under constrastive learning

Potential Text Descriptions...

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

**Choice B:** Mugen collects 5 coins before jumping to kill an enemy

**Choice C:** Mugen jumps twice to the right and uses the key to open the door
20bn: Video Reasoning via Linear-Temporal Specifications
Grounding predicates through actions with Scallop...

Given a real life video of people doing a pre-specified task, ground the details of the shown objects indicated in the video.

Example:

- Task: attach(“sticky note”, “cabinet”)
- Predicates:
  - far(“sticky note”)
  - touching(“sticky note”, “hand”)
  - visible(“sticky note”)
  - ...

```
rel precond("attach", a, b) =
touching(a, "hand") and visible("hand") and visible(a)
rel postcond("attach", a, b) =
not touching(a, "hand") and touching(a, b) and far(a)
```
Grounding predicates through actions with Scallop...

rel precond("approach", x) =
  far(x) and visible(x)
rel postcond("approach", x) =
  close(x) and not far(x) and visible(x)

rel precond("attach", a, b) =
  touching(a, "hand") and visible("hand") and visible(a)
rel postcond("attach", a, b) =
  not touching(a, "hand") and touching(a, b) and far(a)
Grounding predicates through actions with Scallop...

**Video timeline**

Event

- approach("doorknob")

Scallop Program

\[
\text{rel precond}(\text{"approach"}, x) = \text{far}(x) \text{ and visible}(x)
\]

\[
\text{rel postcond}(\text{"approach"}, x) = \text{close}(x) \text{ and not far}(x) \text{ and visible}(x)
\]

**Neural Model**

- 0.85::far("doorknob")
- 0.99::visible("doorknob")
- 0.01::far("doorknob")
- 0.95::visible("doorknob")

Grounding predicates through actions (Migimatsu et al. 2022)
Foundation Models

Claude 2
ANTHROP\C

DSFD

OWL-ViT

T5

OpenAI CLIP

Stable Diffusion

ViLT

LLaMA-2

ResNet

Midjourney

GitHub Copilot

Segment Anything
Research by Meta AI

S3D
Foundation Models: Recap

What is the right **abstraction layer** to **program** with **foundation models**?
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.

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.
[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.
@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"], ...), ...])

```python
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), ...

<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>
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?

```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), ...

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)
```

answer
niece
Domain Specific Language (DSL) in Scallop

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

```
Count(FilterColor(Scene(), "green"))
```
Domain Specific Language (DSL) in Scallop

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

```
Count(FilterColor(Scene(), "green"))
```

```
type Expr = Scene() | FilterColor(Expr, String) | Count(Expr) | Exists(Expr) | ...
```
Domain Specific Language (DSL) in Scallop

Syntax of a Query DSL:

type Expr = Scene() | FilterColor(Expr, String) | Count(Expr) | Exists(Expr) | ...

Semantics:

type eval<T>(bound expr: Expr, free output: T)
rel eval<Object>(e, o) = case e is Scene() and object(o)
rel eval<Object>(e, o) = case e is FilterShape(e1, s) and eval<Object>(e1, o) and shape(o, s)
rel eval<Object>(e, o) = case e is FilterColor(e1, c) and eval<Object>(e1, o) and color(o, c)
rel eval<usize>(e, n) = n := count(o: eval<Object>(e1, o) where e: case e is Count(e1))
rel eval<bool>(e, b) = b := exists(o: eval<Object>(e1, o) where e: case e is Exists(e1))
Semantic Parsing with GPT

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

**Syntax of a query DSL:**

```plaintext
type Expr = Scene() | FilterColor(Expr, String) | Count(Expr) | Exists(Expr) | ...
```
Semantic Parsing with GPT

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

**Syntax of a query DSL:**

```plaintext
type Expr = Scene() | FilterColor(Expr, String) | Count(Expr) | Exists(Expr) | ...
```

```bash
@gpt_complete(prompt="The programmatic representation of "{{question}}" is {{answer}}", examples=[("Is there a sphere?", "Exists(FilterShape(Scene(), "sphere\"))")])
type semantic_parse(bound question: String, answer: Expr)
```
Semantic Parsing with GPT

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

**Syntax of a query DSL:**

```plaintext
type Expr = Scene() | FilterColor(Expr, String) 
| Count(Expr) | Exists(Expr) | ...
```

```plaintext
@gpt_complete(prompt="The programmatic representation of "{{question}}" is {{answer}}", examples=[("Is there a sphere?", "Exists(FilterShape(Scene(), "sphere\"))")])
type semantic_parse(bound question: String, answer: Expr)
```

<table>
<thead>
<tr>
<th>question</th>
<th>answer</th>
</tr>
</thead>
<tbody>
<tr>
<td>How many green objects are there in the image?</td>
<td>Count(FilterColor(Scene(), &quot;green&quot;))</td>
</tr>
</tbody>
</table>
Image Classification as Probabilistic Relation

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

@type cat_or_dog(
    bound img: Tensor,
    free label: String,
)
Image Classification as Probabilistic Relation

<table>
<thead>
<tr>
<th>id</th>
<th>image</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td><img src="image1.png" alt="Dog" /></td>
</tr>
<tr>
<td>1</td>
<td><img src="image2.png" alt="Cat" /></td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

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

```haskell
type cat_or_dog(
  bound img: Tensor,
  free label: String,
)
```
Image Classification as Probabilistic Relation

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

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

<table>
<thead>
<tr>
<th>id</th>
<th>image</th>
<th>prob</th>
<th>id</th>
<th>label</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td><img src="image1.png" alt="Dog" /></td>
<td>0.00</td>
<td>0</td>
<td>cat</td>
</tr>
<tr>
<td>1</td>
<td><img src="image2.png" alt="Cat" /></td>
<td>0.99</td>
<td>0</td>
<td>dog</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td>0.98</td>
<td>1</td>
<td>cat</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td>0.02</td>
<td>1</td>
<td>dog</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Image Segmentation as Probabilistic Relation

Segment Anything
Research by Meta AI

```rust
@segment Anything
type image_segment(
    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,
)
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>![Green object]</td>
</tr>
<tr>
<td>0.98</td>
<td>1</td>
<td>![Purple object]</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)
Domain Specific Language (DSL) in Scallop

Syntax of a Query DSL:

\[
\text{type Expr } = \text{Scene()} \mid \text{FilterColor(Expr, String)} \\
\mid \text{Count(Expr)} \mid \text{Exists(Expr)} \mid \ldots
\]

Semantics:

\[
\begin{align*}
\text{type eval<T>}(\text{bound expr: Expr, free output: T}) \\
\text{rel eval<Object>}(e, o) &= \text{case } e \text{ is Scene()} \text{ and object(o)} \\
\text{rel eval<Object>}(e, o) &= \text{case } e \text{ is FilterShape(e1, s) and eval<Object>}(e1, o) \text{ and shape(o, s)} \\
\text{rel eval<Object>}(e, o) &= \text{case } e \text{ is FilterColor(e1, c) and eval<Object>}(e1, o) \text{ and color(o, c)} \\
\text{rel eval<usize>}(e, n) &= n := \text{count(o: eval<Object>}(e1, o) \text{ where e: case e is Count(e1))} \\
\text{rel eval<bool>}(e, b) &= b := \text{exists(o: eval<Object>}(e1, o) \text{ where e: case e is Exists(e1))}
\end{align*}
\]
Scallop + LLM for Program Analysis

type input_program(program: String)

@gpt_extract_info(
    header="""Please point out the dataflow graph in the given Java program""",
    prompts=["What are the dataflow edges?",
             "What are the sources of user inputs?",
             "What are the sinks that may result in vulnerabilities?"],
    examples=[
        ["""""""]]
    )

type gen_dataflow_edge(bound program: String, from: String, to: String),
    gen_source(bound program: String, source: String),
    gen_sink(bound program: String, sink: String, loc: String)

rel source(s) = input_program(pgm) and gen_source(pgm, s)
rel sink(s, l) = input_program(pgm) and gen_sink(pgm, s, l)
rel edge(a, b) = input_program(pgm) and gen_dataflow_edge(pgm, a, b)
rel path(a, b) = edge(a, b) or (path(a, c) and edge(c, b))

rel vul(loc) = source(src) and sink(snk, loc) and path(src, snk)
Scallop + LLM for Program Analysis

Sample Java File:

```java
public void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
    // some code
    response.setContentType("text/html;charset=UTF-8");
    javax.servlet.http.Cookie[] theCookies = request.getCookies();
    String param = "noCookieValueSupplied";
    if (theCookies != null) {
        for (javax.servlet.http.Cookie theCookie : theCookies) {
            if (theCookie.getName().equals("Cdsr92")) {
                param = java.net.URLDecoder.decode(theCookie.getValue(), "UTF-8");
                break;
            }
        }
    }
    String fileName = null;
    java.io.FileOutputStream fos = null;
    try {
        fileName = org.pck.bcks.helpers.Utils.TESTFILES_DIR + param;
        fos = new java.io.FileOutputStream(fileName, false);
        response.getWriter().println(ESAPI.encoder().encodeForHTML(fileName));
    } catch (Exception e) {
        // System.out.println("File exception caught and swallowed");
    } finally {
        // we tried...
    }
}
```

Extracted dataflow and source/sink information
Questions