Differentiable Learning of Logical Rules for Knowledge Base Reasoning
Fan Yang, Zhilin Yang, William W. Cohen
Carnegie Mellon University
{fanyang1,zhiliny,wcohen}@cs.cmu.edu

First-order logical rules are useful for knowledge base reasoning.
- **Interpretatable**
- **Transferrable** to unseen entities.

Learning probabilistic logical rules is difficult -- it requires learning
- the **discrete structure**, i.e. the particular set of rules to include, and
- the **continuous parameters**, i.e. confidence associated with each rule.

**Our Approach**
An end-to-end **differentiable** framework -- **Neural Logic Programming (Neural LP)**.

**TensorLog operators**
E = set of entities. R = set of binary relations.
- For each entity i, define v_i in \{0, 1\}|E|.
- For each relation R, define M_R in \{0, 1\}|E| x |E| where the (i, j) entry is 1 if and only if R(i, j).

**Key idea**
A neural controller that learns to compose TensorLog operators.

**Learning -- Objective function**
Learn weighted chain-like logical rules to reason over the knowledge base.

\[ \alpha \text{ query } (Y, X) \leftarrow R_n(Y, Z_n) \land \cdots \land R_1(Z_1, X) \]

Using TensorLog operators, the **objective** is:

\[ \max_{(\alpha, \beta)} \sum_{(y, x)} \text{score}(y | x) = \max_{(\alpha, \beta)} \sum_{(y, x)} v_y^T \left( \sum_{l} (\alpha_l \Pi_{R_l} \cdot M_{R_l} \cdot v_x) \right) \]

- l indexes over all possible rules
- \( \alpha_l \) is the confidence of the rule
- \( \beta_l \) is an ordered list of all relations in the rule

**Learning -- Recurrent formulation**
An equivalent but recurrent formulation to allow **end-to-end differentiable** optimization.

At each step,
- predict **attentions over** TensorLog operators,
- use hidden states to **read from memory**, 
- apply operators and write to memory.

**Experiments**
**Statistical relational learning**

<table>
<thead>
<tr>
<th></th>
<th>ISG</th>
<th>Neural LP</th>
</tr>
</thead>
<tbody>
<tr>
<td>T=2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>T=3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UMLS</td>
<td>43.5</td>
<td>93.2</td>
</tr>
<tr>
<td>Kinship</td>
<td>59.2</td>
<td>90.2</td>
</tr>
</tbody>
</table>

**WikiMovies with natural language queries.**

<table>
<thead>
<tr>
<th>Knowledge base</th>
<th>Model</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>directed_by</td>
<td>Key-Value Memory Network</td>
<td>93.9</td>
</tr>
<tr>
<td>written_by</td>
<td>Neural LP</td>
<td>94.6</td>
</tr>
</tbody>
</table>

**Knowledge base completion**

- **Example of learned rules**
  1.00 partially_contains(C, A) \land contains(B, A) \land contains(B, C)
  0.45 partially_contains(C, A) \land contains(B, A) \land contains(B, C)
  0.35 partiallyContains(C, A) \land contains(C, B) \land contains(B, A)
  1.00 marriage_location(C, A) \land nationality(C, B) \land contains(B, A)
  0.35 marriage_location(C, A) \land nationality(B, A)
  0.34 marriage_location(C, A) \land place_lived(C, B) \land contains(B, A)
  1.00 film_editor_by(B, A) \land nominated_for(A, B)
  0.30 film_editor_by(B, A) \land award_winner(B, A) \land nominated_for(B, C)

- **Inductive and transductive settings**

<table>
<thead>
<tr>
<th></th>
<th>WN18</th>
<th>FB15K</th>
<th>FB15KSelected</th>
</tr>
</thead>
<tbody>
<tr>
<td>TransE</td>
<td>0.01</td>
<td>0.48</td>
<td>0.53</td>
</tr>
<tr>
<td>Neural LP</td>
<td><strong>94.49</strong></td>
<td><strong>73.28</strong></td>
<td><strong>27.97</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Node+LinkFeat</th>
<th>DistMult</th>
<th>Neural LP</th>
</tr>
</thead>
<tbody>
<tr>
<td>WN18</td>
<td>94.3</td>
<td>94.2</td>
<td><strong>94.5</strong></td>
</tr>
<tr>
<td>FB15K</td>
<td>87.0</td>
<td>57.7</td>
<td>83.7</td>
</tr>
<tr>
<td>FB15KSelected</td>
<td>34.7</td>
<td>40.8</td>
<td>36.2</td>
</tr>
</tbody>
</table>