Difflog: Synthesizing Datalog Programs using Numerical Relaxation

Xujie Si*, Mukund Raghothaman*, Kihong Heo, Mayur Naik (*equal contribution)

Background and Motivation

Learning logical rules from relational input-output data

Traditionally learned using discrete combinatorial approaches

Gradient-based approaches are remarkably successful in machine learning.

Can gradient-based approaches greatly help to learn logical rules?

Importance of Learning Logic Programs

- Challenge problem in AI
- Good interpretability and compositionality
- An ideal model to represent the learned knowledge
- Logic programs widely used in many areas

Rule Selection Problem

Rule generation approaches
- Syntax-guided approach
- Inductive bias, e.g. meta-rules
- Meta-rule augmentation

Challenges and Our Approach

- How to relax logical rules?
- How to efficiently learn a relaxed program?
- How to recover classical program?

Key Ideas

- Associate each rule with a weight
- Each tuple will get a weight (depending on how it is derived)
- Turn combinatorial search into continuous optimization
- Recovery the desired logic program from the optimized weights

Performance Optimization

- Newton-Raphson Method: Focus on the exact program synthesis where zero loss is achievable
- Periodic MCMC Sampling: Help to escape local minimum
- Early Termination: Not ad hoc, a systematic approach based on a sound separation testing
- Parallelization: Leverage the fact of being a type of Las Vegas algorithm

Empirical Evaluation Results

- Significantly outperform the state-of-the-art synthesizer
- Scale to a large number of templates
- Hybrid optimization is essential to success

Numerical Relaxation of Logic Programs

- Easy to modify existing Datalog solvers with only $O(1)$ overhead.
- Connections to Classical Datalog

- The loss surface is still NOT monotonic.

- This enables us to recover classical program.

Contact: {xsi, rmukund, kheo, mhnai}@cis.upenn.edu