Deep Learning for Cost-Optimal Planning: Task-Dependent Planner Selection

Silvan Sievers\textsuperscript{1} Michael Katz\textsuperscript{2} Shirin Sohrabi\textsuperscript{2} Horst Samulowitz\textsuperscript{2} Patrick Ferber\textsuperscript{1}

\textsuperscript{1}University of Basel, Switzerland
\textsuperscript{2}IBM Research AI, Yorktown Heights, NY, USA

January 30, 2019
Setting

- (General) **Domain-independent** planning
- Problem: **no single best planner** for all domains
(General) **Domain-independent** planning

Problem: **no single best planner** for all domains

Combine planners in **portfolios**


Most prominent in satisficing planning/learning settings
Motivation

- Can we construct a good portfolio for optimal planning?
- **Online** portfolios: solve classification task for (single) planner selection
- Good technique for classification tasks: **deep learning**
**Problem**

**Motivation**
- Can we construct a good portfolio for *optimal* planning?
- **Online** portfolios: solve *classification task* for (single) planner selection
- Good technique for classification tasks: *deep learning*

**Contributions:**
- **Representation** of planning tasks consumable by deep learning
- Proper evaluation of techniques used in **Delfi1**, winner of last optimal IPC
- Discussion of encountered issues
Outline

1. Introduction
2. Planning Task Representation
3. Learning
4. Discussion
Planning Tasks

Given in a logic-based description (PDDL):

(:action pick-up
 :parameters (?x)
 :precondition
 (and (clear ?x) (ontable ?x) (handempty))
 :effect
 (and (not (ontable ?x))
 (not (clear ?x))
 (not (handempty))
 (holding ?x))
)
Representing Planning Tasks

Goal

Use image convolution for classification.
Representing Planning Tasks

Goal

Use **image convolution** for classification.

How to obtain representative images?
- SAT/CSP: convert textual problem description into images
- Here: focus on **structure** of planning tasks
Abstract structure graph: compact encoding

- Nodes for components of the PDDL description (predicates, objects, parameters, etc.)
- Edges to connect components if one is part of another
Representative Graphs

Abstract structure graph: compact encoding

- Nodes for components of the PDDL description (predicates, objects, parameters, etc.)
- Edges to connect components if one is part of another
Representative Images

Conversion of graphs into images:

- Encode **adjacency matrix** as black&white image
- Turn into grayscale by **clustering** pixels
- Resize to **fixed size**
Representative Images

Conversion of graphs into images:

- Encode **adjacency matrix** as black&white image
- Turn into grayscale by **clustering** pixels
- Resize to **fixed size**
Overview

- Goal: predict which planner(s) from the portfolio solve a given task
- Use simple **convolutional neural networks**
Goal: predict which planner(s) from the portfolio solve a given task

Use simple convolutional neural networks
Performance Representation

Multilabel classification:
- **Binary**: predict whether planners solve given task
- **Discretized** runtime (3 intervals): predict in which interval planners belong

Multilabel regression: predict . . .
- **Raw** runtime
- **Normalized** runtime
Performance Representation

Multilabel classification:
- **Binary**: predict whether planners solve given task
- **Discretized** runtime (3 intervals): predict in which interval planners belong

Multilabel regression: predict . . .
- **Raw** runtime
- **Normalized** runtime

Delfi1: binary
Planner Collections

- Fast Downward-based planners from Delfi1
- Those from Delfi1 + additional planners from IPC 2018
- **Minimal subset** of above to cover training data
Benchmarks

- Training set: domains from IPCs prior 2018
- Test set: domains from IPC 2018
Training Data Separation

- Two training data splits: random vs. domain-preserving
  random split
- Validation vs. no validation
Training Data Separation

- Two training data splits: **random** vs. **domain-preserving**
- Random split

**Validation** vs. **no** validation

**Choices of Delfi1:**

- **Hand-crafted** domain-preserving split
- No validation for final training (only for hyper parameter optimization)
48 settings, train 10 models for each
Results

48 settings, train 10 models for each

Comparison of Different Settings

- No domination of any setting over all others
- Delfi1 planner collection significantly better than other two
Results

48 settings, train 10 models for each

Comparison of Different Settings
- No domination of any setting over all others
- Delfi1 planner collection significantly better than other two

Further Observations:
- Mostly consistent planner selection within domains
- Not as strong as Delfi1 itself
**Issues**

- Somewhat **large variance** across different models
- Data is **not independently identically distributed** (i.i.d.)
Potential Future Work

- More sophisticated networks, graph conversion
- Use graphs convolution
- Automatically generate tasks with a certain structure:
  → i.i.d. distribution of tasks?
The End

Thank you for listening!
Poster tonight 7:00 – 8:30 pm: PRS 5097