One-shot learning for MIPs with SOS1 constraints

CORS-JOPT 2023

Charly Robinson La Rocca, Emma Frejinger, Jean-François Cordeau

Université de Montréal

Monday 29th May, 2023

Background

- Goal: quickly find good quality solutions for MIPs with SOS1 constraints.
- One-shot learning (OSL): approach exploits data driven tools with focus on sample efficiency.
- Key results:
  - Locomotive assignment problem (LAP):
    - 10x speed up on small instance
    - Up to 1% relative gap improvement on real large instances
  - MIPLIB: improvement on 58% of selected instances with SOS1 constraints.

Outline

What is SOS1?

Special ordered sets of type 1 (SOS1) constraints are useful when a choice involves multiple options or resources, and only one can be selected.

$$\sum_{k \in K_v} x_v^k = 1 \qquad \forall v \in V \tag{1}$$

$$x_v^k \in \{0, 1\} \qquad \forall v \in V, \forall k \in K_v \tag{2}$$

Why SOS1?

SOS1 can be modelled as a classification problem. The solution vector for the binary variables is analogous to the one-hot encoding of the optimal class.

$$x_v = [0, 0, 1, 0] \rightarrow k = 3 \tag{3}$$

Structuring effect: SOS1 constraints can model *important* decisions in the problem.

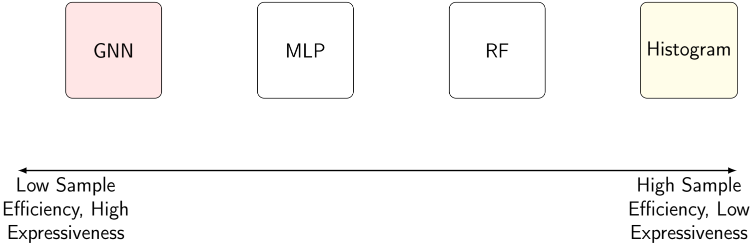
What is one-shot learning?

One-shot learning or few-shot learning aims to learn patterns where only one or a few training examples are available. Goal: Immitate the human ability to learn new concepts from one or a few examples. Examples: Matching Networks [Vinyals et al., 2016] and Prototypical Networks [Snell et al., 2017].

Why one-shot learning?

- Training data is expensive to obtain
- Easy to access and reproduce
- Act as a good baseline
- Can be used in combination with other methods: bagging or boosting
- Short improvement cycles

Efficiency vs expressiveness trade-off



Related works: Learning for MIPs

- MIP-GNN [Khalil et al., 2022]
- Entropic Branching (EB) [Gilpin and Sandholm, 2006]
- Rapid or Conflict Learning [Berthold et al., 2019]

Methodology: Probe and Freeze (PNF)

PNF is a one-shot learning heuristic that uses the probing data to build a model. It is composed of three routines:

- Probe
- Select
- Freeze

Probe

Hyperparameter: T_p is total probing time budget.
At every iteration, we compute the most likely class based on which variable has the highest value.

$$k_{vt} = \arg\max_{k \in K_v} \{x_{vt}^k\} \tag{4}$$

$$\mathrm{k}_v = [k_{v1}, k_{v2}, ..., k_{vn}] \tag{5}$$

Select

The selection strategy uses a scoring system to sort the SOS1 constraints based on the entropy H to infer uncertainty.

$$P(k|\mathrm{k}_v) = \frac{|\{z \in \mathrm{k}_v \mid z = k\}|}{|\mathrm{k}_v|} \tag{6}$$

$$H(\mathrm{k}_v) = -\sum_{k \in \mathrm{k}_v} P(k|\mathrm{k}_v) \log P(k|\mathrm{k}_v) \tag{7}$$

$$\mathrm{score}(v) = -H(\mathrm{k}_v) \tag{8}$$

Goal: Minimize potential assignment errors by selecting constraints with low entropy.

Predict

The predicted class k' is the most likely class in the probing vector k_v.

$$k' = \arg\max_k (P(k|\mathrm{k}_v)) \tag{9}$$

The underlying classifier uses the histogram method with discrete bins for each class.

Freeze

Hyperparameter: r is the ratio of constraints to freeze.
The freezing routine builds *freezing cuts* (FC) defined as follows:

$$FC(v, k') := \{x_v^{k'} = 1\}. \tag{10}$$

Goal: Create a reduced problem $\mathcal{P}'$ which is easier to solve.

Probe and Freeze (PNF)

1. **Probe**: Solve the problem with a probing time budget T_p.
2. **Select**: Sort the variables using the entropy.
3. **Freeze**: Freeze the variables based on the predicted class k'.
4. **Solve**: Solve the reduced problem $\mathcal{P}'$.

Notation: $PNF(r, T_p) == PNF\_r\_T_p$

Locomotive assignment problem (LAP)

Locomotive assignment problem (LAP) is a real-world problem that is used to assign locomotives to trains in a railway network.

$$Assignment\,(SOS1) : \sum_{c \in C_l} y_l^c = 1 \qquad \forall l \in L \qquad (11)$$

$$Flow : \sum_{l \in I[i]} x_l^q = \sum_{l \in O[i]} x_l^q \qquad \forall i \in N \qquad (12)$$

Metrics

Two metrics: runtime speed up (RS), and relative gap (RG).

$$\mathrm{RS} = \frac{\mathrm{runtime(CPLEX)}}{\mathrm{runtime(PNF)} + T_p} \tag{13}$$

$$\mathrm{RG} = \frac{c^T x' - c^T x_{\mathrm{CPLEX}}}{c^T x_{\mathrm{CPLEX}}} \times 100 \tag{14}$$

LAP CPLEX results

Table: Average summary metrics for LAP instances using CPLEX

Difficulty	Runtime (s)	Train count	Optimality gap (%)
E	206.42	141.77	0.01
M	2450.07	336.40	0.26
H	21600.20	5673.20	2.94

LAP speed up results

Table: Quantiles for runtime speed up in LAP instances

Scenario	Quantiles 0.25	Quantiles 0.75	Mean	Sample size
E+PNF_0.5_8	1.84	5.46	4.07	30.00
E+PNF_0.9_6	4.46	40.64	27.75	30.00
E+PNF_0.2_10	0.90	2.23	1.84	30.00
E+PNF_0.2_20	0.66	1.68	1.32	30.00
E+PNF_0.5_30	0.65	2.91	2.39	30.00
M+PNF_0.2_60	0.93	1.08	1.32	30.00
M+PNF_0.5_60	1.01	2.64	2.63	30.00
M+PNF_0.2_120	0.92	1.11	1.25	30.00
M+PNF_0.5_120	1.03	1.82	2.27	30.00
M+PNF_0.9_120	4.07	26.37	17.10	30.00
H+PNF_0.2_600	0.99	1.02	1.02	20.00
H+PNF_0.2_1800	0.99	1.02	1.00	20.00
H+PNF_0.5_1800	0.98	1.00	0.98	20.00

LAP gap results

Table: Quantiles for relative gap (%) in LAP instances

Scenario	Quantiles 0.25	Quantiles 0.75	Mean	Sample size
E+PNF_0.5_8	0.01	0.20	0.11	30.00
E+PNF_0.9_6	0.94	2.31	1.73	30.00
E+PNF_0.2_10	-0.00	0.02	0.02	30.00
E+PNF_0.2_20	-0.00	0.00	0.01	30.00
E+PNF_0.5_30	0.00	0.13	0.07	30.00
M+PNF_0.2_60	-0.12	0.00	-0.09	30.00
M+PNF_0.5_60	-0.11	0.04	-0.08	30.00
M+PNF_0.2_120	-0.05	0.00	-0.06	30.00
M+PNF_0.5_120	-0.03	0.03	-0.07	30.00
M+PNF_0.9_120	0.65	1.47	1.15	30.00
H+PNF_0.2_600	-1.09	-0.58	-0.10	20.00
H+PNF_0.2_1800	-0.99	-0.48	-0.52	20.00
H+PNF_0.5_1800	-1.16	-0.69	-1.03	20.00

Correlations

Table: Correlations with relative gap for LAP instances

Parameter	Spearman	Kendall	Average
Fixing ratio	0.61	0.51	0.56
Probing time budget (s)	-0.42	-0.33	-0.37

Gap over time

Table: Gap (%) for full LAP instances

Scenario	Time after probing (s)			
	1800	3600	5400	7200
cplex	nan	4.04	3.94	2.49
PNF_0.2_600	1.72	1.48	1.09	0.88
PNF_0.2_1800	2.01	1.67	1.18	1.17
PNF_0.5_1800	1.48	1.39	0.89	0.69

Table: Full LAP instances with feasible solution

Scenario	Time after probing (s)			
	1800	3600	5400	7200
cplex	0	6	7	7
PNF_0.2_600	7	9	9	9
PNF_0.2_1800	7	10	10	10
PNF_0.5_1800	8	9	9	9

MIPLIB 2017

The MIPLIB [Gleixner et al., 2021] dataset is an open source library of MIP instances. The instances in MIPLIB are generated independently and are not related to each other. This makes it a more challenging test bed for our approach.
Instance selection:

- Remove instance with high average entropy
- Remove instance with low sample size

MIPLIB summary results

Table: Number of solved MIPLIB instances

Scenario	Number of instances	Solved	Unsolved
E+PNF_0.2_1	68	67	1
E+PNF_0.2_2	68	57	11
M+PNF_0.2_60	44	39	5
M+PNF_0.2_120	44	37	7
H+PNF_0.2_300	76	61	15
H+PNF_0.2_600	76	58	18
Total	376	319	57

MIPLIB results

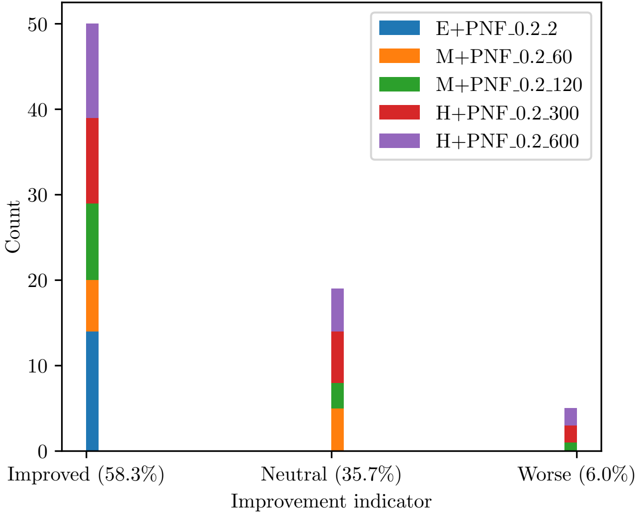
Table: Quantiles for runtime speed up in selected MIPLIB instances

Scenario	Quantiles 0.25	0.75	Mean	Sample size
E+PNF_0.2_2	1.07	1.24	1.19	14.00
M+PNF_0.2_60	0.38	3.62	2.80	11.00
M+PNF_0.2_120	0.30	1.34	1.23	13.00
H+PNF_0.2_300	0.29	1.01	1.32	18.00
H+PNF_0.2_600	0.38	1.31	1.22	18.00

Table: Quantiles for relative gap (%) in selected MIPLIB instances

Scenario	Quantiles 0.25	0.75	Mean	Sample size
E+PNF_0.2_2	-0.00	0.00	0.06	14.00
M+PNF_0.2_60	0.00	2.32	4.80	11.00
M+PNF_0.2_120	0.00	0.00	1.57	13.00
H+PNF_0.2_300	-0.06	0.00	-1.16	18.00
H+PNF_0.2_600	-0.00	0.06	-4.87	18.00

MIPLIB summary outcome results



Conclusion

Our results are consistant with Occam's razor principle which states that the simplest explanation is usually the best.

- Implication: Machine learning methods with *low expressiveness* can be used to accelerate the discovery of solutions for MIPs with SOS1 constraints.
- Limitation: PNF heuristic is unable to find a feasible solution for 11% of selected MIPLIB instances (21/188).
- Future work: risk management strategy to mitigate the risk of infeasibility.

Acknowledgments: We acknowledge the support of the Institute for Data Valorization (IVADO), the Canada First Research Excellence Fund (Apogée/CFREF), and the Canadian National Railway Company for their financial backing and provision of valuable data which was instrumental in advancing this research.

Berthold, T., Stuckey, P. J., and Witzig, J. (2019).
Local rapid learning for integer programs.
In *Integration of Constraint Programming, Artificial Intelligence, and Operations Research: 16th International Conference, CPAIOR 2019, Thessaloniki, Greece, June 4–7, 2019, Proceedings 16*, pages 67–83. Springer.

Gilpin, A. and Sandholm, T. (2006).
Information-theoretic approaches to branching in search.
In *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*, pages 545–547.

Gleixner, A., Hendel, G., Gamrath, G., Achterberg, T., Bastubbe, M., Berthold, T., Christophel, P. M., Jarck, K., Koch, T., Linderoth, J., Lübbecke, M., Mittelmann, H. D., Ozyurt, D., Ralphs, T. K., Salvagnin, D., and Shinano, Y. (2021).
MIPLIB 2017: Data-Driven Compilation of the 6th Mixed-Integer Programming Library.

Mathematical Programming Computation.

Khalil, E. B., Morris, C., and Lodi, A. (2022).
MIP-GNN: A data-driven framework for guiding combinatorial solvers.
In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 10219–10227.

Snell, J., Swersky, K., and Zemel, R. (2017).
Prototypical networks for few-shot learning.
Advances in neural information processing systems, 30.

Vinyals, O., Blundell, C., Lillicrap, T., Wierstra, D., et al. (2016).
Matching networks for one shot learning.
Advances in neural information processing systems, 29.