Neural Self-Improvement Learning for Domain-Independent Dynamic Programming

Minori Narita
Department of Mechanical and Industrial Engineering, University of Toronto, Canada

Eldan Cohen
Department of Mechanical and Industrial Engineering, University of Toronto, Canada

J. Christopher Beck
Department of Mechanical and Industrial Engineering, University of Toronto, Canada

Abstract

We propose a solver-in-the-loop self-improvement framework for training neural policies to guide domain-independent dynamic programming (DIDP) in solving combinatorial optimization problems. For each instance, DIDP-based beam search is run within a DIDP model, leveraging its pruning mechanisms such as dual bounds and state constraints, and the best solution found is extracted as an expert trajectory. The policy is then trained by minimizing a cross-entropy imitation loss. Experiments on the four combinatorial optimization problems show that the proposed method consistently reduces node expansions and achieves competitive solve time, while significantly outperforming the same DIDP models and solvers guided by Proximal Policy Optimization (PPO) and by the default dual bound guidance.

2012 ACM Subject Classification Computing methodologies → Discrete space search

Keywords and phrases Self-Improvement Learning, Dynamic Programming, Machine Learning

Digital Object Identifier 10.4230/LIPIcs.CVIT.2016.23

1 Introduction

Combinatorial optimization problems (COPs) arise in many domains, including routing, scheduling, and resource allocation, and are generally NP-hard. Neural combinatorial optimization (NCO) leverages deep neural networks to learn heuristics from data, either to directly construct solutions [2] or to guide search within solvers [1, 6]. Such neural guided search combines the representational power of deep learning with a systematic search mechanism to improve solution quality with additional computation. However, a common limitation of existing approaches is that policy training and deployment are largely decoupled. Policies are typically trained either via supervised learning from pre-computed near-optimal solutions [3] or via reinforcement learning (RL) in a standalone Markov Decision Process (MDP) [1, 6]. As a result, solver-side information such as pruning by constraints and dual bounds is not utilized during training nor is the learned policy trained for the context within which it will be deployed (i.e., as heuristic guidance for a solver).

We propose a self-improvement learning (SL) framework for training neural policies to guide domain-independent dynamic programming (DIDP) [4, 5]. Inspired by recent self-improvement learning approaches [7], our method collects trajectories by running beam search on a DIDP model. The best solution found is treated as a teacher trajectory, and the policy is trained by minimizing a cross-entropy imitation loss. This framework allows the policy to learn from solver-generated trajectories that respect problem constraints and objectives, without requiring a complicated reward function design. We evaluate our approach on the Traveling Salesperson Problem (TSP), TSP with Time Windows (TSPTW), Portfolio Optimization, and the Knapsack Problem and show that solver-in-the-loop SL is effective for training neural policies to guide combinatorial search.

2 Method

We consider combinatorial optimization problems modeled using DIDP, where problems are defined as DP models in the Dynamic Programming Description Language (DyPDL) [5]. In DyPDL, a state s represents a subproblem and a transition τ produces a successor state s'. Each transition τ from s incurs a cost $\mathsf{cost}_\tau(s)$ and is applicable only when its preconditions are satisfied. DyPDL models are solved by model-agnostic heuristic search algorithms [4], which *expand* states from the initial state s_0 by applying applicable transitions. At each successor state, the f-value is computed as $f(s') = g(s') + \eta(s')$, where $g(s')$ is the path cost from s_0 to s', and $\eta(s')$ is a dual-bound estimate of the remaining cost. For details, see Kuroiwa and Beck [5].

Our goal is to train a neural policy $\pi_\theta(\tau \mid s)$ that prioritizes promising transitions during search. Following previous work [6], f-values are weighted by accumulated policy values up to state s from the root node, i.e., $\pi^\dagger(\tau^- \mid s^-) = \pi(\tau_0 \mid s_0)\pi(\tau_1 \mid s_1)...\pi(\tau^- \mid s^-)$, to account for all decisions up to s. Therefore, the f-value is computed as $f(s) = (g(s) + \eta(s))/\pi^\dagger(\tau^- \mid s^-)$.

Rather than training the policy independently, we integrate the solver into the training loop to generate supervision signals. At each epoch, we sample instances from a fixed distribution and run beam search within the DIDP model with beam width b, guided by the current policy π^{best}. The search respects all pruning mechanisms of the solver (e.g., dual bounds, state constraints, and duplicate detection). Among all feasible solutions found, we select the best one and extract its trajectory $\{(s_0, \tau_0), (s_1, \tau_1), \dots\}$, adding the state-transition pairs to a replay buffer $\mathcal{D}$. The policy is updated by minimizing the cross-entropy loss: $\mathcal{L}(\theta) = -\frac{1}{B}\sum_{j=1}^{B} \log \pi_\theta(\tau_j^* \mid s_j)$, where B is the batch size. To stabilize training, we maintain a trajectory-collection policy π^{best} and update it only if the new policy improves performance on the validation set, in which case the buffer $\mathcal{D}$ is cleared. Our approach is inspired by self-improvement learning [7], but differs in that trajectory generation is performed within the DIDP solver, leveraging its pruning mechanisms to avoid unpromising states and to produce high-quality supervision signals. An overview is shown in Figure 1.

3 Experimental Results

We evaluate the proposed approach on TSP, TSPTW, Portfolio Optimization, and Knapsack. For all domains, the BQ network is used as the policy model [2] and is trained for up to 72 hours with early stopping. Training instances follow the same distributions as in Narita et al. [6]. We compare our approach against dual-bound guidance (the default setting in DIDP), greedy heuristic guidance based on manually designed domain-specific heuristics, and PPO-based neural guidance. All baselines are implemented following Narita et al. [6],

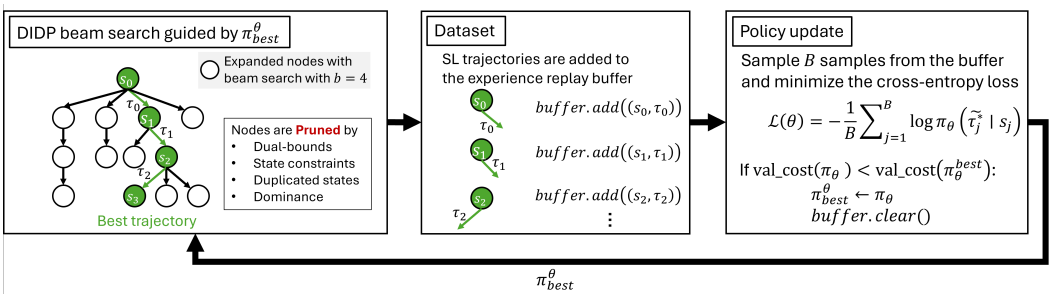

Figure 1 One epoch of self-improvement learning for DIDP. Here $b = 4$; training uses $b = 256$.

and we refer the reader to that work for details. Solutions are generated using Complete Anytime Beam Search (CABS) [4].

Figure 2 shows the mean relative error (MRE) versus expanded nodes (top row) and runtime (bottom row). The proposed method (π=SL-BQ) consistently reduces the number of node expansions required to reach high-quality solutions. Notably, it significantly outperforms PPO guidance, especially on TSPTW, where the problem is highly constrained by time-windows and PPO struggles with handling dead-ends. The runtime of SL-BQ is also competitive, achieving the best performance on TSP and Portfolio, and second-best performance on TSPTW and Knapsack, beaten by greedy and dual bound guidance, respectively. These results demonstrate that leveraging solver-generated trajectories can lead to more efficient and stable learning compared to RL-based approaches.

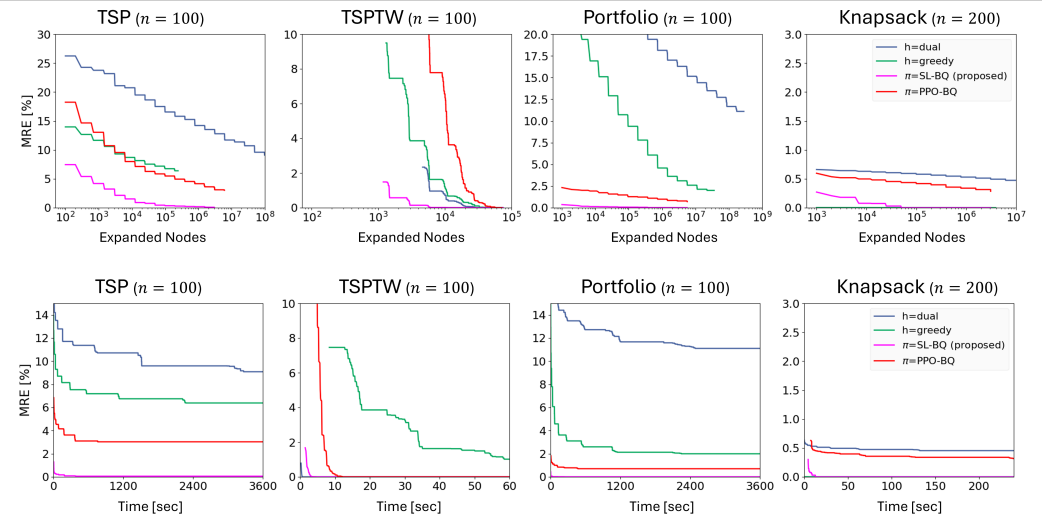

Figure 2 Node expansion (top row) and solve-time (bottom row) performance of different heuristics to guide CABS, averaged over 20 instances. Note the differing scales for both axes.

References

1 Quentin Cappart, Thierry Moisan, Louis-Martin Rousseau, Isabeau Prémont-Schwarz, and Andre A Cire. Combining reinforcement learning and constraint programming for combinatorial optimization. In *AAAI*, volume 35, pages 3677–3687, 2021. doi:10.1609/aaai.v35i5.16484.

2 Darko Drakulic, Sofia Michel, Florian Mai, Arnaud Sors, and Jean-Marc Andreoli. BQ-NCO: Bisimulation quotienting for efficient neural combinatorial optimization. In *NeurIPS*, 2023.

3 Chaitanya K Joshi, Thomas Laurent, and Xavier Bresson. An efficient graph convolutional network technique for the travelling salesman problem. *arXiv preprint arXiv:1906.01227*, 2019.

4 Ryo Kuroiwa and J. Christopher Beck. Solving domain-independent dynamic programming problems with anytime heuristic search. In *ICAPS*, 2023. doi:10.1609/icaps.v33i1.27201.

5 Ryo Kuroiwa and J. Christopher Beck. Domain-independent dynamic programming. *Artificial Intelligence*, 354:104506, 2026. doi:10.1016/j.artint.2026.104506.

6 Minori Narita, Ryo Kuroiwa, and J. Christopher Beck. Reinforcement learning-based heuristics to guide domain-independent dynamic programming. In *CPAIOR*, page 137–154, 2025. doi:10.1007/978-3-031-95976-9_9.

7 Jonathan Pirnay and Dominik G. Grimm. Self-improvement for neural combinatorial optimization: Sample without replacement, but improvement. *Transactions on Machine Learning Research*, 2024.