Self-Paced Learning Based Graph Convolutional Neural Network for Mixed Integer Programming (Student Abstract)

Li Chen[1], Hua Xu[1]*, Ziteng Wang[1], Chengming Wang[2], Yu Jiang[2]

[1] State Key Laboratory of Intelligent Technology and Systems, Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China

[2] Meituan Inc., Block F&G, Wangjing International R&D Park, No.6 Wang Jing East Rd, Chaoyang District, Beijing, 100102, China

chenli19@mails.tsinghua.edu.cn, xuhua@tsinghua.edu.cn, wang-zt19@mails.tsinghua.edu.cn

wangchengming02@meituan.com, jiangyu31@meituan.com

Abstract

Graph convolutional neural network (GCN) based methods have achieved noticeable performance in solving mixed integer programming problems (MIPs). However, the generalization of existing work is limited due to the problem structure. This paper proposes a self-paced learning (SPL) based GCN network (SPGCN) with curriculum learning (CL) to make the utmost of samples. SPGCN employs a GCN model to imitate the branching variable selection during the branch and bound process, while the training process is conducted in a self-paced fashion. Specifically, SPGCN contains a loss-based automatic difficulty measurer, where the training loss of the sample represents the difficulty level. In each iteration, a dynamic training dataset is constructed according to the difficulty level for GCN model training. Experiments on four NP-hard datasets verify that CL can lead to generalization improvement and convergence speedup in solving MIPs, where SPL performs better than predefined CL methods.

Introduction

Mixed integer programming problems (MIPs) are significant parts of combinatorial optimization (CO) problems . Graph convolutional network (GCN) based methods have achieved remarkable enhancement on MIPs, such as Learn2branch (Gasse and et al. 2019). Although GCN networks produce strong baselines for CO problems, the generalization of the learned policy is limited due to the problem structure, and it is challenging to give any precise quantitative estimates a priori. Curriculum learning (CL) is a general training strategy (Wang, Chen, and Zhu 2020), which guides the model training process in a paradigm from simple to complex, making the utmost of samples with various difficulty levels. This paper presents a self-paced learning based graph convolutional network (SPGCN) for MIPs, in which self-paced learning is semi-automatic CL with a loss-based automatic difficulty measurer and a dynamic curriculum (Land and Doig 2010). SPGCN contains three parts, graph representation(Nair and et al. 2020), model prediction and training strategy. Experiments on four NP-hard benchmarks show good performance and high efficiency.

*Corresponding author

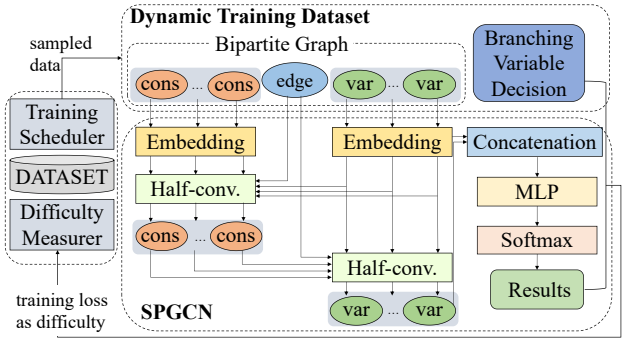

Figure 1: The overall structure of SPGCN.

Method

The overall structure of the proposed SPGCN is shown in Figure 1. Firstly, MIP instances are represented as bipartite graphs. Afterward, the bipartite graphs are used as inputs to the SPGCN model. SPGCN adopts two half-convolution operators to embed the bipartite graph, one from constraints to variables and one from variable to constraints. The learned variable node embeddings are employed to predict the branching decision variables based on the multilayer perceptron (MLP). Besides, the SPGCN is trained in a self-paced way, which contains a dynamic curriculum and a loss-based automatic difficulty measurer. The training scheduler constructs a dynamic training dataset in each iteration according to the sample difficulty measured by training loss.

Graph Representation. A bipartite graph (BG) was adopted to represent the MIP instance, as shown in Figure 2. Specifically, non-zero entries in objective function coefficients $\{k_1, \ldots, k_n\}$ are encoded as features of variable nodes $\{v_1, \ldots, v_n\}$, constraint bounds $\{b_1, \ldots, b_m\}$ as features of constraint nodes $\{c_1, \ldots, c_m\}$, and coefficient matrix A as features of $v - c$ edges in $\mathcal{E}$.

Dataset Generation. The dataset consists of state-action pairs $\{(s_i, a_i)_{i=1}^N\}$, where s_i is the BG state and a_i is the branching decision variable. When solving MIPs with SCIP, the decisions of strong branching and the new node states are recorded during the branch-and-bound, and normally several

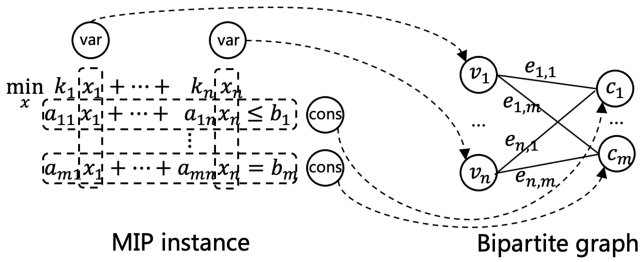

Figure 2: The BG representation for a MIP instance.

Algorithm 1: The Self-Paced GCN Model

Input: Training Dataset $\{(s_i, a_i)_{i=1}^N\}$; GCN model f_w; The maximum iteration number T;
Output: The parameters w of f_w.

Initializing w, v, $\lambda = \lambda_0$.
for t = 1 : T **do**
  for i = 1 : n **do**
    $v_i^* = \arg\min_{v_i \in [0,1]} v_i \cdot L(f_w(s_i), a_i)$.
  end for
  $w^* = \arg\min_w \sum_{i=1}^N v_i^* \cdot L(f_w(s_i), a_i)$.
  Update λ_t according to the predefined sequence N.
end for

pairs are recorded to the dataset. However, these pairs may be of uneven quality and heterogeneous, as they come from different instances and different branching stages.

Self-Paced GCN Model. To handle the issues above, a self-paced learning based GCN model described in Algorithm 1 is proposed to make the training faster, more generalized, and more robust. The goal of the model is to minimize the empirical cross-entropy loss on the whole training set,

$$\min_{w,x} \mathbb{E}(w, v, \lambda) \sum_{i=1}^{N} v_i \cdot L_i(f_w(s_i), a_i) \tag{1}$$

where f_w is the GCN model with parameter w which maps the bipartite state s to branch variable action a. $v \in [0,1]^N$ is the mask vector where $v_i = 1$ indicates that the ith sample is chosen for training. λ represents the age parameter which is dynamically adjusted to make sure exactly N_t examples be assigned with non-zero masks v_i in the tth epoch to control the learning pace. w and v are optimized iteratively. Based on SPGCN, CLGCN is the simplified version of SPGCN, where the curriculum is constructed from easy to hard based on the predefined difficulty levels.

Experiments and Analysis

Experimental Settings. SPGCN is compared with CLGCN and Learn2brach on Set Covering (SC), Combinatorial Auction (CA), Capacitated Facility Location (CFL) and Maximum Independent Set (MIS). Each benchmark consists of easy, medium and hard instances. All experiments are repeated five times. The details of benchmarks and other additional experimental details are given in the appendix.

Problem		Learn2branch	CLGCN	SPGCN
SC	acc@1	63.2	63.7	**65.9**
	acc@5	91.2	92.3	**93.5**
	acc@10	96.3	97.8	**98.9**
CA	acc@1	59.7	60.3	**63.1**
	acc@5	90.1	91.9	**92.8**
	acc@10	95.8	96.4	**97.2**
CFL	acc@1	71.2	71.5	**72.8**
	acc@5	97.9	98.1	**98.5**
	acc@10	**99.9**	**99.9**	**99.9**
MIS	acc@1	56.7	58.2	**60.1**
	acc@5	81.2	83.8	**86.3**
	acc@10	89.7	91.2	**93.3**

Table 1: Comparison of prediction accuracy on the test sets.

Problem		Learn2branch	CLGCN	SPGCN
SC	Medium	1582	1449	**1329**
	Hard	30347	29849	**28479**
CA	Medium	692	653	**642**
	Hard	5342	5144	**5023**
CFL	Medium	338	348	**323**
	Hard	339	351	**326**
MIS	Medium	2034	1837	**1749**
	Hard	3013	2934	**2649**

Table 2: Comparison of resulting node counts for medium and hard difficulty datasets.

Results and Analysis. Table 1 shows the prediction accuracy of branching variables on the test sets representing the models' imitation ability. Table 2 shows the final node counts required to solve the instances by algorithms. In general, SPGCN significantly outperforms learn2branch and predefined CL methods on all benchmarks, demonstrating the effectiveness of self-paced learning.

Conclusion

In this paper, a data-driven approach SPGCN is proposed that integrates self-paced learning into the GCN model. Experiments on four NP-hard datasets verify the efficiency of SPGCN. Since SPL learning is a general training strategy, it has the potential to be effective on various MIP tasks, which provides a new idea for the MIP community.

Acknowledgements

This research was supported by Meituan.

References

Gasse, M.; and et al., C. 2019. Exact combinatorial optimization with graph convolutional neural networks. *Advances in Neural Information Processing Systems*, 32.

Land, A. H.; and Doig, A. G. 2010. An automatic method for solving discrete programming problems. In *50 Years of Integer Programming 1958-2008*, 105–132. Springer.

Nair, V.; and et al., B. 2020. Solving mixed integer programs using neural networks. *arXiv preprint arXiv:2012.13349*.

Wang, X.; Chen, Y.; and Zhu, W. 2020. A Comprehensive Survey on Curriculum Learning. *arXiv preprint arXiv:2010.13166*.