RESEARCH STATEMENT

Runzhong Wang (runzhong.wang@sjtu.edu.cn)

My research interests span the areas of machine learning, especially graph learning, and related optimization problems. A common thread of my research is to exploit novel deep learning models to tackle optimization formulations with combinatorial natures, whereby such formulas commonly exist in graph learning. I have resorted to powerful machine learning and optimization approaches such as graph embedding, attention model, differentiable optimization, bi-level optimization, and optimal transport. Broadly speaking, my research work falls within the rising area of *Machine Learning for Combinatorial Optimization*, whereby its applications lie in various real-world decision-making scenarios. Some significant milestones have been achieved in this area, yet there still remain many open questions to answer.

Background and Current Work

Combinatorics is the inherent nature of many graph learning tasks, for example, node matching between two graphs requires solving the NP-hard quadratic assignment problem, and traversing a network is equivalent to the combinatorial traveling salesman problem. Tackling real-world tasks with combinatorial nature involves the following two steps: *problem modeling* and *problem solving*. To elaborate on these two steps, let's take graph matching problem as an example,

$$\max_{\mathbf{X}} \texttt{vec}(\mathbf{X})^\top \mathbf{K} \texttt{vec}(\mathbf{X}), \qquad s.t. \quad \mathbf{X} \in \{0,1\}^{n_1 \times n_2}, \mathbf{X}\mathbf{1} = 1, \mathbf{X}^\top \mathbf{1} \leq 1, \tag{1}$$

where $\mathbf{X}$ encodes the node-to-node correspondence between two graphs, $\mathbf{K}$ is the so-called affinity matrix containing node and edge affinity scores, and $\texttt{vec}(\cdot)$ means column-wise vectorization. How to build $\mathbf{K}$ from noisy real-world data is the *problem modeling* step, whereby novel neural networks may be a powerful clue if the network architecture and the learning process are designed properly. The formulation itself in Eq. (1) is an NP-hard problem, yielding challenges for the *problem solving* step, and a recent consensus is that machine learning will relieve the burden of human algorithm developers in a data-driven manner. However, achieving the expected efficacy via learning is not straightforward, and a poorly-developed machine learning method may be even inferior to a learning-free algorithm. My research works address deep learning models and combinatorial problems from two aspects: feature learning and prediction for the *problem modeling* step, and learning-fused optimization algorithms for the *problem solving* step. I describe them below.

Problem Modeling by Deep Neural Networks

When tackling real-world problems, the first step is to extract parameters to build the optimization form, known as the *problem modeling* step. A ubiquitous challenge is the noises in real-world data. If the *problem modeling* step contains too much error, the follow-up optimization step will be misled, and even the optimal solution may not equal the solution desired by the downstream task. For example, perfectly predicting the prices of financial assets is infeasible, and the portfolio optimizer will probably be misled by prediction error, leading to inferior decisions. Therefore, I tried to answer the following research question: *Can we mitigate the problem modeling error by tailoring the network architecture or the learning framework?* I explored end-to-end learning paradigms with input data from three orthogonal modalities (graphs, images, and time series), whereby neural networks for different modalities could be further coupled to fit the specific tasks.

1) Problem Modeling with Graphs. Deep learning with graph data has received increasing interest thanks to the simple yet effective structure of graph embedding models and graph convolutional networks. My research papers are usually based on basic graph neural networks (e.g. GCN, GraphSage), with additional technical improvements tailored for specific problems. For example, the cross-graph message passing module in [10, 8] effectively exchanges information in matching. A channel-independent embedding network and attention-based supervision are developed in [9] to enable higher model capacity and avoid overfitting. Besides, the graph structure could be further modified by the generative network proposed in [6]. The efficacy of the proposed graph modeling modules is validated on the corresponding tasks.

2) Problem Modeling with Images. For tasks involving images, the CNNs are not modified since their structures are well-developed and can be generally effective. Instead, I explore several improvements in training. In [7] I develop a label-free learning pipeline by maximizing the consensus between a CNN feature branch and a solver branch. In [11] and [3] I develop data augmentation tricks by simple copy-pasting and through adversarial attack, respectively.

3) Problem Modeling with Time Series. The time series of daily asset prices is studied in [12], which is modeled by an LSTM network. The LSTM network is followed by a differentiable portfolio optimization solver so that the prediction error is fully considered together with the optimization solver during training. An improvement of 20% in the annual return is achieved by jointly training LSTM and the optimization solver.

Problem Solving by Fusing Machine Learning and Optimization Solvers

Once the problem is modeled (either by a neural network or not), an optimization problem is formulated and it calls for the next *problem solving* step. A common challenge in *problem solving* is that finding the optimal solution is usually NP-hard. Algorithm developers have been working for decades to trade-off between efficiency and efficacy, whereby the

empirical best practices are coded into optimization solvers. A line of existing papers develops deep neural networks to fully replace learning-free solvers, abandoning the theoretical guarantees and empirical efficacy of the solvers. The drawback of purely relying on neural networks is already witnessed: on the traveling salesman problem, all pure neural network solvers are inferior to the learning-free LKH3 solver. My research was motivated to answer the following question: *Can we take the best of the two worlds of machine learning modules and optimization solvers?* I developed and explored three different strategies to fuse machine learning and optimization solvers, which are discussed as follows.

1) Optimization Solver Fused in Machine Learning Pipeline. If an optimization solver is differentiable or if we could estimate/approximate the gradient, then it could be fused as a layer in the neural network. In [1], the differentiable Sinkhorn solver is encoded as layers in our developed neural solver for Lawler's quadratic assignment problem. In [12], we exploit both Sinkhorn solver and perturbation-based gradient approximation tricks to differentiate through the cardinality constraint. In experiments, these learning solvers can either tackle pure optimization problems or be coupled with a *problem modeling* network to build an end-to-end joint learning pipeline of *problem modeling* and *problem solving*.

2) Machine Learning Module Fused in Optimization Solver. The inner mechanism of optimization solvers usually involves heuristic modules built with human experts, for example, deciding which node to explore next in a tree-search algorithm. Neural networks seem to be perfect replacements for such heuristic modules. In [5], we develop a neural network module based on graph neural networks and attention pooling to predict the heuristic score in the A* algorithm. The performance drop is nearly negligible w.r.t. learning-free A*, yet the speed-up is of magnitudes.

3) Machine Learning and Optimization Solver Fused Equally. A practical demand is that the fusing paradigm should be general and agnostic of the choices of optimization solvers. I propose one possible approach by treating the machine learning module and the optimization solver equally. A general paradigm is presented in [4] where the machine learning module and the optimization solver are designed as two optimization layers, formulating a bi-level optimization paradigm. The learning signal is passed by reinforcement learning, bypassing the non-differentiable issue of most solvers. Such a paradigm improves learning-free solvers on routing, matching, and scheduling problems on graphs.

Unifying Problem Modeling and Problem Solving — A Research Agenda

From my previous research, I notice the importance of machine learning in both *problem modeling* and *problem solving*. However, there seems missing a general machine learning approach being agnostic to problem types which can unify these two steps, worth being addressed in the future. Besides, I believe in the potential of applying state-of-the-art learning and optimization methods to tackle problems with broader social impact. I intend to answer the following questions: *1) Can we develop a general learning paradigm whereby both problem modeling and problem solving are unified? 2) Can we explore applications of machine learning with real-world social impacts?*

In the near future, I intend to extend existing machine learning and optimization solver fusion methods, making them differentiable and could handle more problem types. For example, a promising research direction is to explore 0/1 assignment problems with linear constraints. Simultaneously, based on my currently explored applications e.g. task scheduling [4] and finance [12], I intend to conduct research with broader social impacts, e.g. *bioinformatics*, *transportation*, *public policy*, or any other important scenarios that involve combinatorial and machine learning problems.

In the future, I will work towards the ultimate goal of a *unified problem modeling and problem solving framework*, whereby such a unified framework should further enable *large pretraining networks* covering a wide range of combinatorial problems. Besides, *causal inference* might be a promising tool for reliable *problem modeling*. *Out-of-distribution and adversarial robustness* is worth studying to further facilitate the *problem solving* step in application.

My research plan is a good mixture of futuristic and my current research. The unified *problem modeling* and *problem solving* could be based on extending the existing frameworks tailored for specific problems [1, 12]. I intend to work closely with my current collaborators who have expertise in *model pretraining*, *causal inference*, and *adversarial robustness*. Similarly, I intend to work closely with the industry in developing technologies targeting real-world problems. I believe my experience of collaborating with professors and students from the university and working with industrial experts will help me achieve these goals smoothly. I feel excited about learning, collaborating, and sharing with world-class researchers, and also contributing and shaping the future of machine learning , in terms of both technologies and applications.

Softwares. I believed in the value of open-source, which motivates me to develop the following repositories:
1) An open-source deep graph matching repository (670 stars): `https://github.com/Thinklab-SJTU/ThinkMatch`;
2) A user-friendly graph matching toolbox (155 stars): `https://github.com/Thinklab-SJTU/pygmtools`;
3) General bi-level framework for combinatorial optimization (66 stars): `https://github.com/Thinklab-SJTU/PPO-BiHyb`;
4) Awesome learning for optimization papers (803 stars): `https://github.com/Thinklab-SJTU/awesome-ml4co`;
5) A data-augmentation protocol for vision tasks (392 stars): `https://github.com/GothicAi/Instaboost`.

References

[1] **Runzhong Wang**, Junchi Yan and Xiaokang Yang. "Neural Graph Matching Network: Learning Lawler's Quadratic Assignment Problem with Extension to Hypergraph and Multiple-graph Matching." *IEEE Transactions on Pattern Analysis and Machine Intelligence* (**PAMI**), 2022.

[2] Chang Liu, Chenfei Lou, **Runzhong Wang**, Alan Yuhan Xi, Li Shen, Junchi Yan. "Deep Neural Network Fusion via Graph Matching with Applications to Model Ensemble and Federated Learning." *International Conference on Machine Learning* (**ICML**), 2022.

[3] Qibing Ren, Qingquan Bao, **Runzhong Wang**, Junchi Yan. "Appearance and structure aware robust deep visual graph matching: Attack, defense and beyond." *IEEE Conference on Computer Vision and Pattern Recognition* (**CVPR**), 2022.

[4] **Runzhong Wang**, Zhigang Hua, Gan Liu, Jiayi Zhang, Junchi Yan, Feng Qi, Shuang Yang, Jun Zhou, Xiaokang Yang. "A Bi-Level Framework for Learning to Solve Combinatorial Optimization on Graphs." *Neural Information Processing Systems* (**NeurIPS**), 2021.

[5] **Runzhong Wang**, Tianqi Zhang, Tianshu Yu, Junchi Yan and Xiaokang Yang. "Combinatorial Learning of Graph Edit Distance via Dynamic Embedding." *IEEE Conference on Computer Vision and Pattern Recognition* (**CVPR**), 2021.

[6] Tianshu Yu, **Runzhong Wang**, Junchi Yan and Baoxin Li. "Deep Latent Graph Matching." *International Conference on Machine Learning* (**ICML**), 2021.

[7] **Runzhong Wang**, Junchi Yan and Xiaokang Yang. "Graduated Assignment for Joint Multi-Graph Matching and Clustering with Application to Unsupervised Graph Matching Network Learning." *Neural Information Processing Systems* (**NeurIPS**), 2020.

[8] **Runzhong Wang**, Junchi Yan and Xiaokang Yang. "Combinatorial Learning of Robust Deep Graph Matching: an Embedding based Approach." *IEEE Transactions on Pattern Analysis and Machine Intelligence* (**PAMI**), 2020.

[9] Tianshu Yu, **Runzhong Wang**, Junchi Yan and Baoxin Li. "Learning Deep Graph Matching with Channel-Independent Embedding and Hungarian Attention." *International Conference on Learning and Representations* (**ICLR**), 2020.

[10] **Runzhong Wang**, Junchi Yan and Xiaokang Yang. "Learning Combinatorial Embedding Networks for Deep Graph Matching." *International Conference on Computer Vision* (**ICCV Oral**), 2019.

[11] **Runzhong Wang***, Hao-shu Fang*, Jianhua Sun*, Minghao Gou, Yong-Lu Li, Cewu Lu. "InstaBoost: Boosting Instance Segmentation via Probability Map Guided Copy-Pasting." *International Conference on Computer Vision* (**ICCV**), 2019.

[12] "Relaxed Combinatorial Optimization Networks with Self-Supervision: Theoretical and Empirical Notes on the Cardinality-Constrained Case." with Li Shen, Yiting Chen, Xiaokang Yang, Dacheng Tao and Junchi Yan. *In submission to International Conference on Learning and Representations* (**ICLR**), 2023.