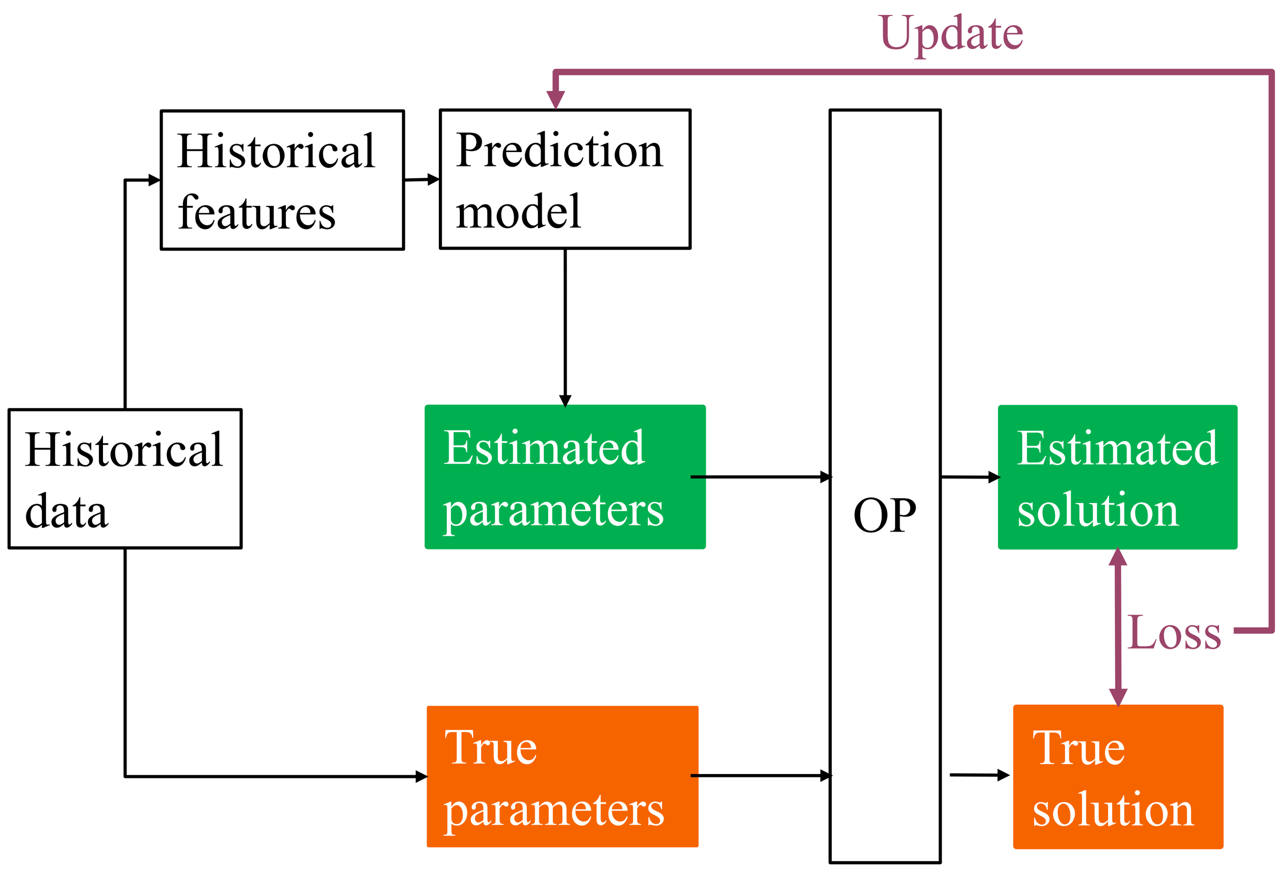
Two-Stage Predict+Optimize for Mixed Integer Linear Programs with Unknown Parameters in Constraints

Xinyi Hu [1], Jasper C.H. Lee [2], Jimmy H.M. Lee [1]
[1] The Chinese University of Hong Kong
[2] University of Wisconsin–Madison

Predict+Optimize (P+O)

- Aim to incorporate OPs into the loss function
- Most focus on unknown parameters **only in objectives**

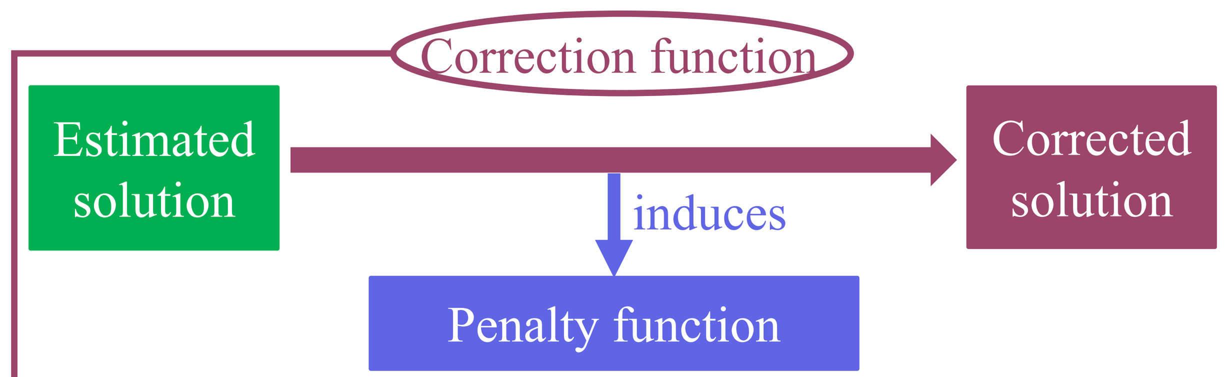


Loss function: Regret ([Demirović et al., 2019 & 2022]):

‖ True optimal value − Estimated optimal value ‖

Prior Work [Hu et al., AAAI 2023]

- The first & only approach for unknowns in constraints



Limitations: Correction function
1. Problem-specific and manually-designed
2. Difficult to give a general training method
3. Fails to handle the case when estimated solution is feasible but not good

Comparison

Two-Stage P+O VS Hu et al. framework

	Two-Stage P+O	Hu et al. framework
Correction function	No	Yes
Penalty function	Yes	Yes
Infeasible estimated solutions	Correct into feasible solutions	
Feasible estimated solutions	Modify into better solutions	No change

Proposition A.1.
- Given the same penalty function and prediction model
- Two-Stage P+O **always outputs as least as good** a corrected solution as the Hu et al. framework using any correction function

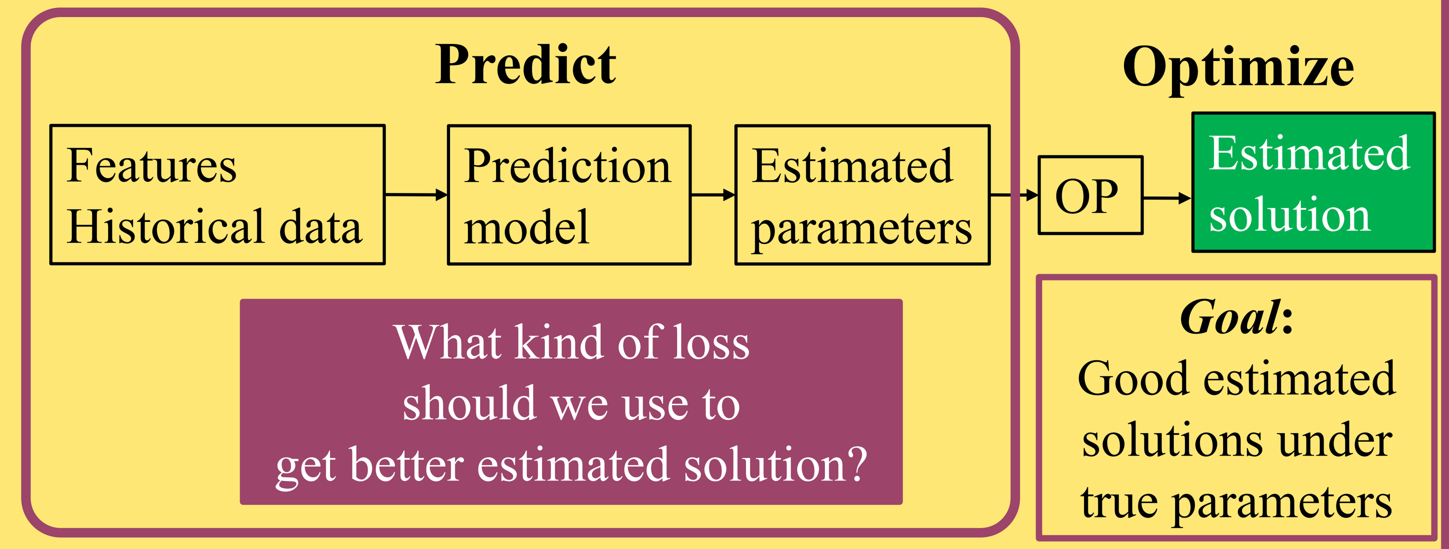
Problem: Optimization problems (OPs) with unknowns is challenging, especially when the unknowns are in the constraints

Given:
- Features
- Historical data (features, true parameters)

Predict unknown parameters and solve the OP

Challenge:
Solutions solved from *predicted* parameters may be **infeasible** under **true** setting

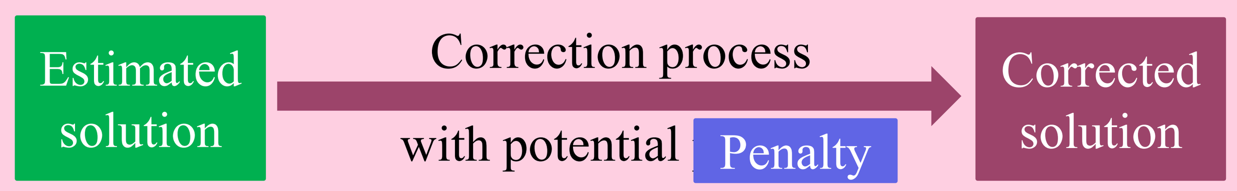
The Pipeline



Example: knapsack problem with unknown weights
If estimated weights are all 0
→ Estimated solution: select all items (may be infeasible)

Contribution 1: A conceptually simple and powerful framework

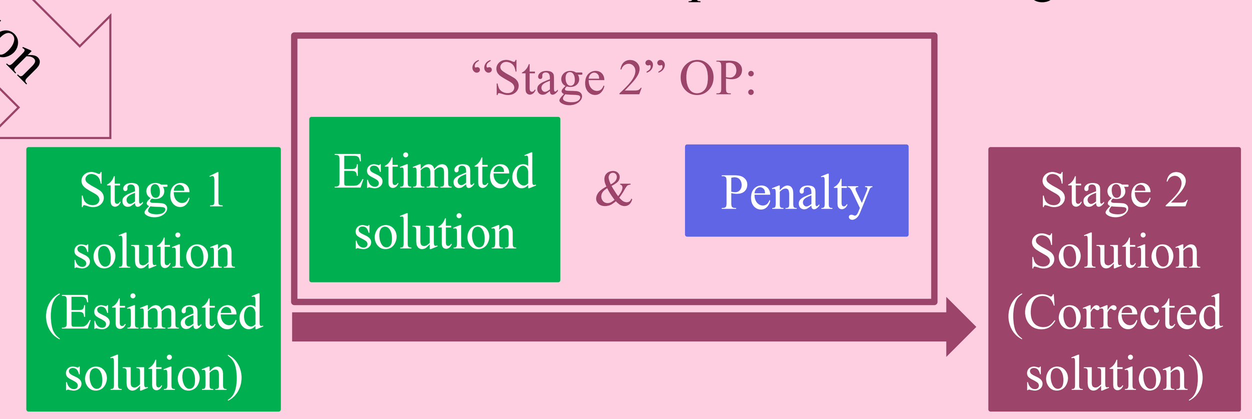
In certain applications, the estimated solution can be modified once the true parameters are revealed



Loss function: Post-hoc regret [Hu et al., AAAI 2023]

‖ True optimal value − Corrected optimal value ‖ + Penalty

Core idea: formulate correction process as a "Stage 2" OP



Let: x^* (optimal solution) $\quad obj$ (objective function)
θ (true parameters) $\quad C$ (constraints)
$\hat{\theta}$ (estimated parameters)

Original OP:

$$x^*(\theta) = \underset{x}{\operatorname{argmin}}\, obj(x,\theta) \quad \text{s.t. } C(x,\theta)$$

Stage 1 OP: original OP using estimated parameters

$$x_1^* = \underset{x}{\operatorname{argmin}}\, obj(x,\hat{\theta}) \quad \text{s.t. } C(x,\hat{\theta})$$

Stage 2 OP: augment original OP by adding a penalty term

$$x_2^* = \underset{x}{\operatorname{argmin}}\, obj(x,\theta) + Pen(x_1^* \rightarrow x,\theta) \quad \text{s.t. } C(x,\theta)$$

Contribution 2: A general training method

More general:
- [Hu et al., AAAI 2023]: only for packing/covering LPs
- Proposed: for mixed integer linear programs (MILPs)

MILP in the standard form:

$$x^* = \underset{x}{\operatorname{argmin}}\, c^T x \quad s.t.\ Ax = b, Gx \geq h, x \geq 0, x_s \in \mathbb{Z}$$

Aim: train a neural network using Post-hoc regret

Main challenge: backward propagation

Let: w_e (edge e weight in the neural network)

$$\frac{dPReg(\hat{\theta},\theta)}{dw_e} = \left.\frac{\partial PReg(\hat{\theta},\theta)}{\partial x_2^*}\right|_{x_1^*} \frac{\partial x_2^*}{\partial x_1^*}\frac{\partial x_1^*}{\partial \hat{\theta}}\frac{\partial \hat{\theta}}{\partial w_e} + \left.\frac{\partial PReg(\hat{\theta},\theta)}{\partial x_1^*}\right|_{x_2^*} \frac{\partial x_1^*}{\partial \hat{\theta}}\frac{\partial \hat{\theta}}{\partial w_e}$$

In the case of MILP, easily calculable

Calculated by standard back propagation

Challenging:
MILP optima may not change under minor parameter perturbations
→ Uninformative gradients (0 or non-existent)

Our approach: define a surrogate loss, using x_1^* and x_2^* solved from a convex relaxation of the original MILP:

$$x^* = \underset{x,s}{\operatorname{argmin}}\, c^T x - \mu \sum_i \ln(x_i) - \mu \sum_i \ln(s_i)$$
$$s.t.\ Ax = b, Gx - s = h$$

Selected Experimental Results

Experimental comparison on two frameworks:
- Two-Stage P+O **always outperforms** Hu et al. framework

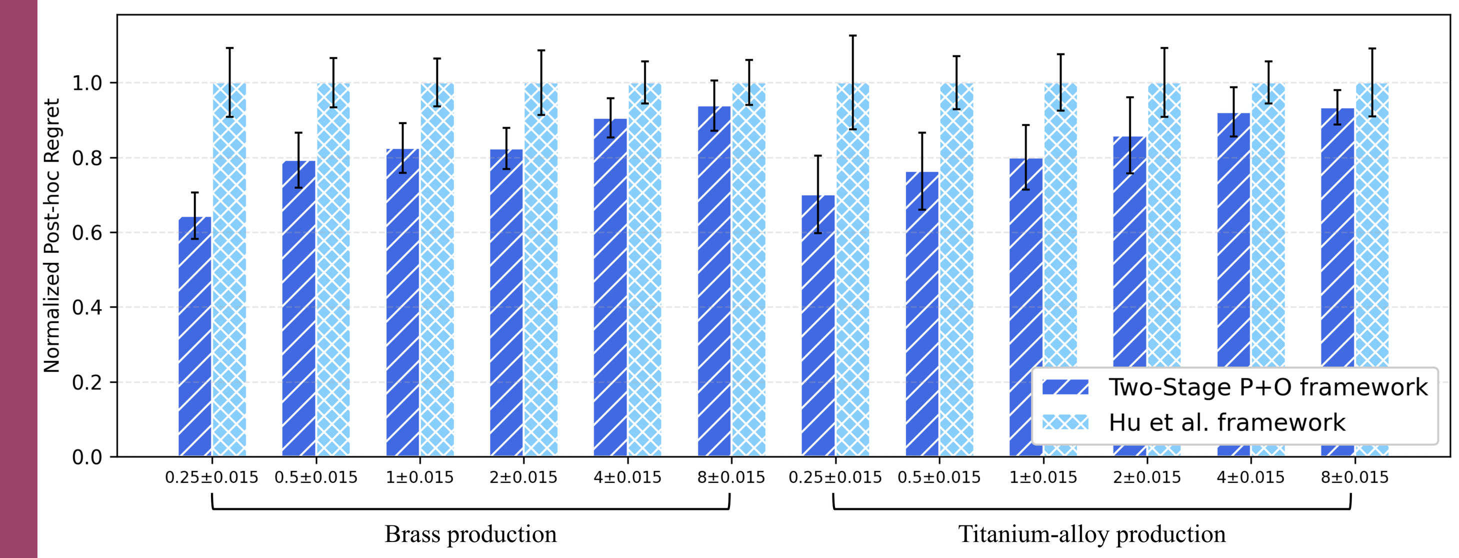


Figure 1: Framework comparison on the alloy production problem

Solution Quality:
- **Always outperform** state-of-the-art and classical methods

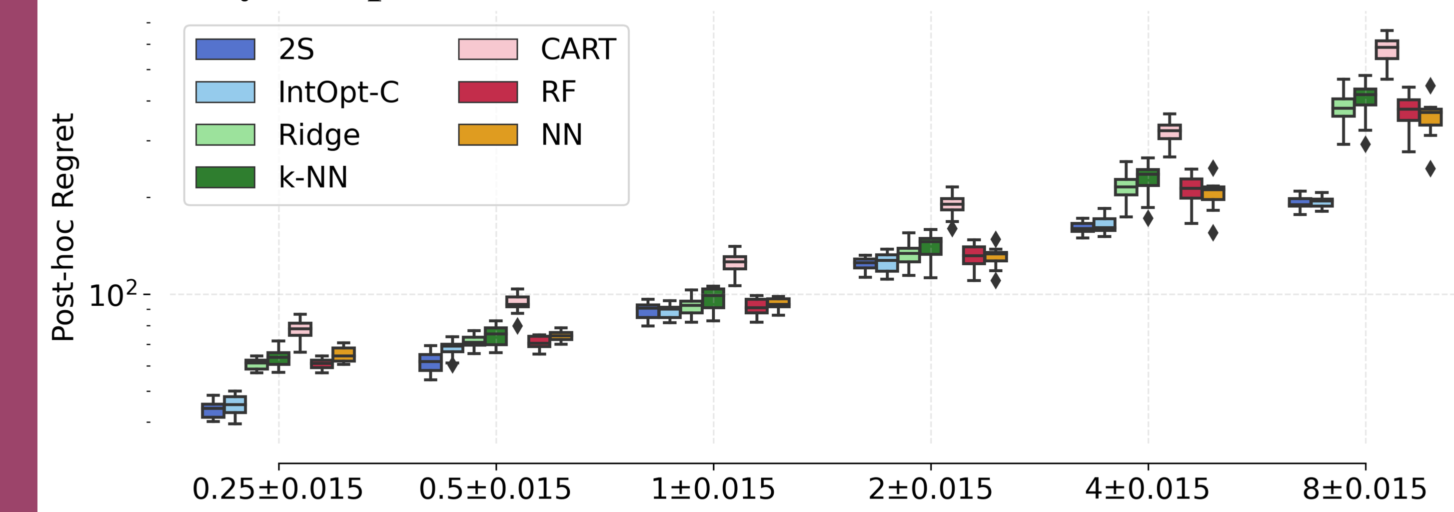


Figure 2: Selected experiment results on the brass production problem

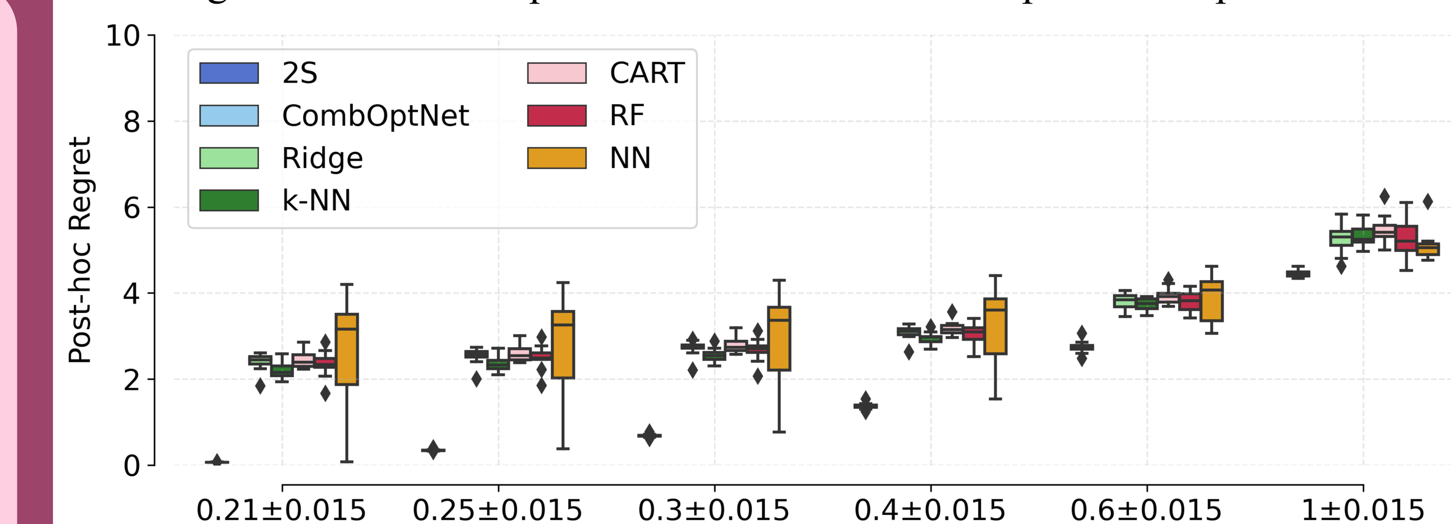


Figure 3: Selected experiment results on the 0-1 knapsack problem

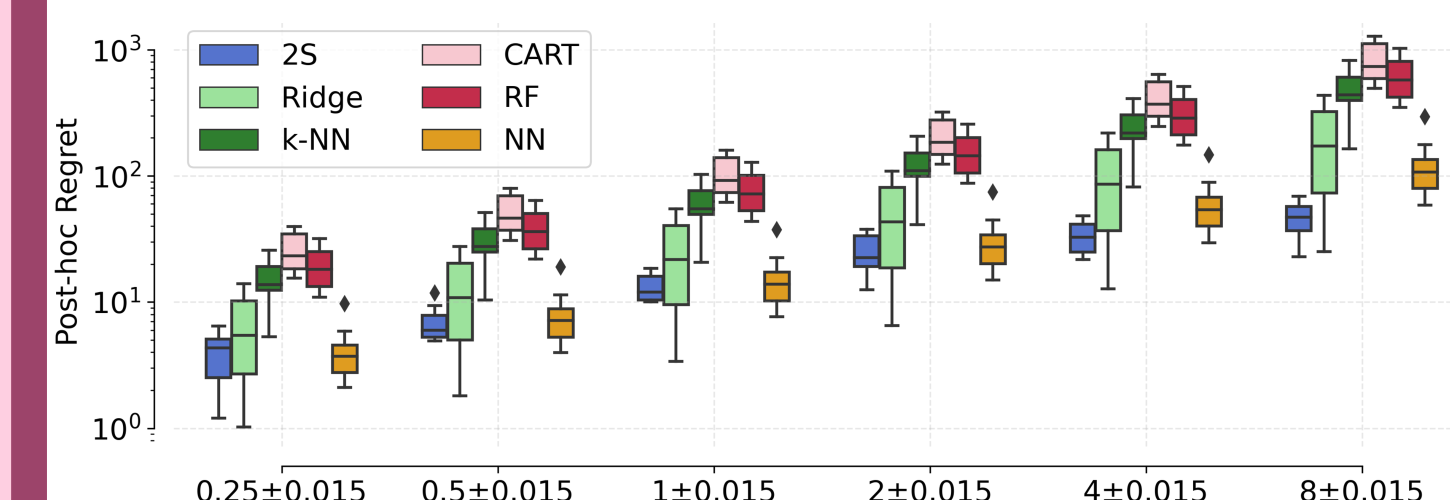


Figure 4: Selected experiment results on the nurse scheduling problem

Runtime:
- **On par with** IntOpt-C (state-of-the-art method)

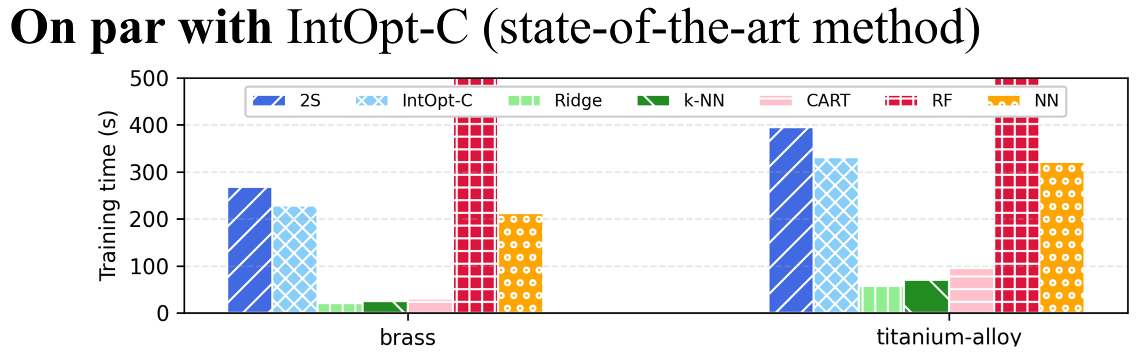


Figure 5: Average runtime (in seconds) for the alloy production

- **Much faster** than CombOptNet (state-of-the-art method) & RF

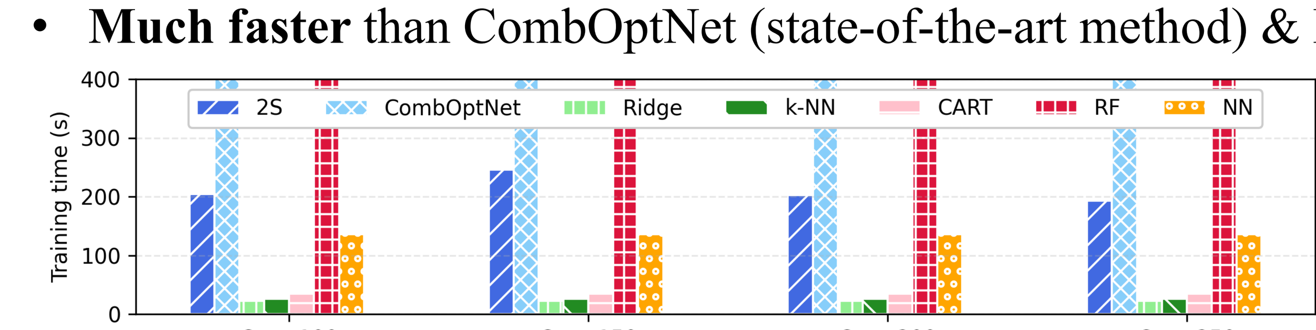


Figure 6: Average runtime (in seconds) for the 0-1 knapsack problem