RL-PAO: PREDICTION AS ACTION IN DECISION MAKING UNDER UNCERTAINTY

Anonymous authors
Paper under double-blind review

ABSTRACT

Decision-making under uncertainty often relies on predicted parameters, yet accurate prediction does not necessarily lead to good operational decisions. Aligning prediction with downstream optimization requires learning from the consequences of the decisions those predictions induce. We introduce RL-PaO, a reinforcement learning framework that integrates system formulation, optimization, and decision execution into a single environment. This yields a Markov decision process in which prediction is regarded as action: it shifts the environment to produce subsequent context and reward that explicitly aligns prediction error with realized cost, and learning the optimal policy does not require differentiating through the black-box solver. We evaluate RL-PaO on day-ahead energy scheduling using real historical data. On the test year, RL-PaO achieves the lowest annual cost among the non-oracle baselines, achieving on average 10% cost reduction. Moreover, RL-PaO is capable of further analyses to provide strong interpretability both from the policy evolution perspective and the cost-accuracy trade-off.

1 INTRODUCTION

Decision-making under uncertainty is fundamental to energy systems, transportation, and supply chains (Mandi et al., 2024). Operational decisions must be made before parameters or constraints become known. Predictive models estimate these parameters, optimization solvers generate decisions based on the estimates and execution under realized conditions determines their value. It is vital to evaluate prediction quality since it changes the optimization objective and influences downstream execution. It remains a central challenge to align the predictor with the operational purpose.

Predict-then-optimize (PtO) trains the predictor independently of the optimizer, but its training objective such as MSE, MAE can be misaligned with downstream cost: lower estimation error does not necessarily translate to better decisions (Elmachtoub & Grigas, 2022). Prediction-and-optimization (PaO) addresses this mismatch by feeding back decision-loss surrogates to augment the prediction loss (Mandi et al., 2025; Silvestri et al., 2026), in the hope that by minimizing the augmented loss, a balance between prediction accuracy and decision quality could be reached so as to improve downstream performance (Mandi et al., 2024). However, training on the composite loss could lead to undesired solutions that are suboptimal for all of the member losses, see Table 1 for detailed comparison. Therefore, a paradigm that can effective align prediction and optimization is called for.

In this paper, we introduce **RL-PaO**. By treating system formulation, objective optimization and schedule execution as an integrated environment, we convert prediction into action that can shift the environment to produce next solution and a scalar reward that explicitly evaluates the cost-accuracy trade-off for the current solution. I.e., we build a Markov Decision Process (MDP) (Puterman, 1994) that closes the loop between the environment and an agent policy, see Figure 1. Though RL has been utilized in improving solver procedures such as MILP (Tang et al., 2020; Qi et al., 2021), RL-PaO differs from them in that RL-PaO concerns alignment between prediction and blackbox optimization, not accelerating a solver conditional on its internal states.

We evaluate RL-PaO on a day-ahead scheduling problem using real history data from the Osaka University. We train a Proximal Policy Optimization (PPO) agent (Schulman et al., 2017) on the train set and evaluate on a held-out test year. Results show that our method achieves the lowest annual cost among all baseline methods except the oracle. Moreover, unlike conventional PtO and

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

	Prior-based Optimization			Predict-Optimize Paradigm		
				PtO	PaO	
					Supervised PaO	RL-PaO **Ours**
	DO	RO	SO			
Uncertainty modeling	×	Robust uncertainty sets	Probability distributions	Open-loop prediction	Closed-loop prediction	Prediction-optimization alignment
Placement-free uncertainty	–	✓	✓	✓	×	✓
Decision-focused training	–	–	–	×	✓	✓
Model-agnostic optimization	–	–	–	✓	×	✓
Gradient-free	–	–	–	✓	×	✓

Table 1: Comparison of methods for optimization under uncertainty. Ours is the first RL-based PaO method that aims at improving the alignment between prediction and optimization. We emphasize the importance of *Decision-focused training* that optimizes by downstream decision cost instead of prediction error; and *Model-agnostic optimization* that operates without full knowledge of the underlying optimization formulation.

PaO, our formulation provides strong interpretability to characterize the cost–accuracy trade-off: prediction error is insufficient to assess decision quality In short, our contributions are:

- **(Novel paradigm)** We demonstrate that RL – conventionally leveraged for closed-loop system control – can also serve as a decision-oriented prediction module. On top of this, we formalize the two-stage prediction-optimization pipeline as a Markov Decision Process, based on which stable convergence and robustness towards hyperparameters is achieved.
- **(Parsimonious formulation)** We propose a counterintuitive state-action construction where the state is defined exclusively by the downstream system operational schedule. We prove that the schedule plan inherently encodes sufficient information to infer the upper bound of the constraints, removing the need for extra environmental input that supervised forecasting methods rely on.
- **(Real-world evaluation)** We evaluate RL-PaO on a real day-ahead scheduling problem at the Osaka University. By learning from offline data, the trained RL agent outperforms all non-oracle baselines in terms of annual cost, 9.42% better than supervised PaO and 12.07% than PtO. RL-PaO also provides strong interpretability from multiple perspectives.
- **(High generalizability)** Our framework treats the downstream task as a black box, and is inherently agnostic to the underlying system model, mathematical problem formulation, and prediction target. This yields strong generalizability and enables a truly model-free, problem-agnostic paradigm for decision-making problem.

2 Background and Problem Setting

In this paper we consider the following generic optimization problem parameterized by an uncertain quantity in vector $\boldsymbol{\xi} = \{\boldsymbol{\xi}_f, \boldsymbol{\xi}_g, \boldsymbol{\xi}_h\}$:

$$\begin{aligned} \boldsymbol{x}^*(\boldsymbol{\xi}) = \arg\min_{\boldsymbol{x}\in\mathcal{X}} & f(\boldsymbol{x}, \boldsymbol{\xi}_f), \\ \text{s.t.} \quad & \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{\xi}_g) \leq \mathbf{0}, \\ & \boldsymbol{h}(\boldsymbol{x}, \boldsymbol{\xi}_h) = \mathbf{0}, \end{aligned} \tag{1}$$

where $\boldsymbol{x}$ denotes the operational decision, f is the operational objective, and $\boldsymbol{g}$ and $\boldsymbol{h}$ describe inequality and equality constraints, respectively. In practice, the decision must be made before $\boldsymbol{\xi}$ becomes available. Therefore, from the perspective of how the uncertainty on $\boldsymbol{\xi}$ is incorporated, existing approaches can be broadly organized into two families. The first family leverages nominal values, uncertainty sets, or probability distributions. In the rest of the paper, we refer to this family collectively as Prior-based Optimization (PbO). The second family exploits observable contextual information to estimate the unknown quantities and uses them in the optimization process. We refer to this family as Prediction-Optimization (P-O). (Lahoud et al., 2025; Mandi et al., 2024)

Prior-based Optimization. Prior-based opimization specifies a representation of uncertainty before solving the optimization problem. Depending on how uncertainty is represented, three representative formulations are deterministic optimization (DO), robust optimization (RO), and stochastic optimization (SO). DO infers fixed estimate $\hat{\boldsymbol{\xi}}$ from historical data. SO assumes a predefined probability

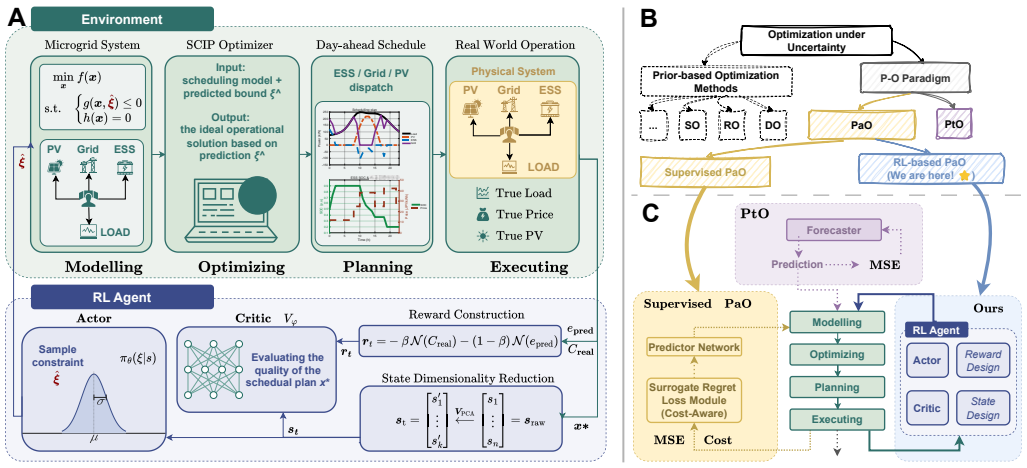

Figure 1: Overview of the proposed RL-PaO framework. **(A)** We explicitly include system modeling, optimization, scheduling and real-world operation into one environment. The RL agent observes current context to predict the unknown constraint in the optimization objective to be solved by an MILP solver. The solution influences execution process, yielding reward and next context. **(B)** our RL formulation is under the PaO category and different from the conventional supervised PaO methods. **(C)** Architectural comparison of different methods. PtO uses an open-loop predictor. Supervised PaO closes the training loop by incorporating downstream loss. By contrast, the proposed method aligns prediction and optimization by a novel MDP and RL training.

distribution for $\boldsymbol{\xi}$, reformulating the problem using expectations; i.e., minimizing $\mathbb{E}_{\boldsymbol{\xi}_f}[f(\boldsymbol{x}, \boldsymbol{\xi}_f)]$ subject to $\mathbb{E}_{\boldsymbol{\xi}_g}[\boldsymbol{g}(\boldsymbol{x}, \boldsymbol{\xi}_g)] \leq \mathbf{0}$ and $\mathbb{E}_{\boldsymbol{\xi}_h}[\boldsymbol{h}(\boldsymbol{x}, \boldsymbol{\xi}_h)] = \mathbf{0}$. RO bounds $\boldsymbol{\xi}$ within a predefined uncertainty set. Optimizing the worst-case objective $\max_{\boldsymbol{\xi}_f} f(\boldsymbol{x}, \boldsymbol{\xi}_f)$ while ensuring constraints strictly hold for all realizations $\boldsymbol{\xi}_g$ and $\boldsymbol{\xi}_h$. These methods demand analytical priors for $\boldsymbol{\xi}$, which are often intractable in complex environments, especially for SO. Our approach directly learns optimal strategies, circumventing the requirements on mathematical priors and explicit uncertainty modeling.

PtO and PaO. Besides the prior-based methods above, addressing uncertainty through prediction in advance or jointly with optimization has been gaining popularity. Let $\boldsymbol{z}$ denote a latent variable available at the decision time, $\boldsymbol{\xi}$ an uncertain parameter required by the downstream optimization problem. A predictive model $q_{\boldsymbol{\theta}}$ parameterized by $\boldsymbol{\theta}$ maps the latent variable to an estimate, then fed into the objective function to construct a decision $\boldsymbol{x}^*$:

$$\hat{\boldsymbol{\xi}} = q_{\boldsymbol{\theta}}(\boldsymbol{z}), \qquad \boldsymbol{x}^*(\hat{\boldsymbol{\xi}}) = \arg\min_{\boldsymbol{x} \in \mathcal{X}} f(\boldsymbol{x}, \hat{\boldsymbol{\xi}}). \tag{2}$$

The decision $\boldsymbol{x}^*$ is then executed as a plan to obtain downstream cost. When the prediction model is trained independently of the optimizer, the method is called pedict-then-optimize (PtO). On the contrary, it is called prediction-and-optimization (PaO). PaO explicitly feeds back decision quality to the prediction model to help improve training.

We identify the following questions that cannot be readily answered by the existing methods:

1. Prior-based methods require knowing uncertain characterizations, which are typically intractable in complex environments.
2. standard PtO methods typically operate in an open-loop manner, relying on minimizing estimation errors without feedback from realized operational outcomes (Chen et al., 2022; Lahoud et al., 2025). However, lower estimation errors do not necessarily translate to better decisions or lower downstream costs (Mandi et al., 2024).
3. Existing PaO methods closes the loop by feeding back downstream costs to augment the estimation error (e.g. MSE), in a hope that minimizing the augmented loss would balance minimizing cost and MSE. However, performing gradient descent on the added loss could lead to solutions that are suboptimal for all member losses. (Shah et al., 2022)

Problem Formulation. We consider the following instance of Equation 1: day-ahead scheduling electricity use. The problem focuses on uncertainty in the inequality-constrained photovoltaic (PV) generation upper bound, a scenario that is particularly challenging for existing PaO methods. The system consists of four core components: (1) PV generation provides variable energy input; (2) Energy Storage Systems (ESS) shift power consumption and supply across time periods; (3) Load represents fixed electricity demand, and (4) power exchange with the main power grid. The problem can be cast as the Mixed Integer Linear Programming (MILP) form:

$$\begin{aligned} &\min_{\boldsymbol{P}^{\text{grid}}} \quad \boldsymbol{c}^\top \boldsymbol{P}^{\text{grid}} \\ &\text{s.t.} \quad \mathbf{1}_{\pm}^\top \boldsymbol{P}_t = \mathbf{0}, \quad \mathbf{0} \le \boldsymbol{P}_t \le \boldsymbol{U}_t, \quad \boldsymbol{E}_t = \boldsymbol{E}_{t-1} + \left(\eta_{\text{char}} \cdot \boldsymbol{p}_t^{\text{char}} - \eta_{\text{disc}}^{-1} \cdot \boldsymbol{p}_t^{\text{disc}}\right), \quad \forall t \\ &\text{where} \quad \boldsymbol{P}_t := \begin{bmatrix} P_t^{\text{pv}} \\ P_t^{\text{grid}} \\ P_t^{\text{load}} \\ \boldsymbol{p}_t^{\text{char}} \\ \boldsymbol{p}_t^{\text{disc}} \end{bmatrix}, \boldsymbol{U}_t := \begin{bmatrix} U_t^{\text{pv}} \\ U_t^{\text{grid}} \\ \infty \\ \boldsymbol{u}_t \cdot U_{\text{group}}^{\text{char}} \\ (1-\boldsymbol{u}_t) \cdot U_{\text{group}}^{\text{disc}} \end{bmatrix}, \mathbf{1}_{\pm} := \begin{bmatrix} +1 \\ +1 \\ -1 \\ -\mathbf{1} \\ +\mathbf{1} \end{bmatrix}. \end{aligned} \tag{3}$$

where $\boldsymbol{c}$ denotes the cost vector, $\boldsymbol{u}_t, \boldsymbol{p}_t^{\cdot}$ and $\mathbf{1}$ are $|\mathcal{M}|$-dimensional vectors, with $\mathcal{M}$ denoting the full set of ESS units. P_t^{pv} is actual PV generation, P_t^{load} is the fixed load demand. $\boldsymbol{p}_t^{\cdot}$ are the (dis)charging power. $\boldsymbol{E}$ the state of battery that should be kept in a fixed range. Intuitively, the objective minimizes total grid electricity purchase cost over the scheduling horizon, with $\boldsymbol{c}$ denoting the electricity price, P_t^{grid} the grid import power. The definitions of all variables and parameters are summarized in Table 2.

Following Equation 2, prediction enters the problem through U_t^{pv}, the per-step upper bound for the PV generation. A predictive model $q_{\boldsymbol{\theta}}$ is trained to output U_t^{pv}, then the objective is fed into an optimizer to obtain the solution $\boldsymbol{x}^*$. In this paper, we design a novel Markov Decision Process around $\boldsymbol{x}^*$ that closes the loop between prediction and optimization, allowing efficient reinforcement learning algorithms to tackle the problem.

Table 2: Notations for the problem.

Symbol	Description
$\boldsymbol{c}$	Electricity price
P_t^{pv}	Actual PV output
P_t^{grid}	Grid import power
P_t^{load}	Fixed load demand
$\boldsymbol{p}_t^{\cdot}$	ESS (dis)charging power
U_t^{pv}	PV upper bound†
U_t^{grid}	Grid exchange limit
$U_{\cdot}^{\text{group}}$	(Dis)charge power limit
$\boldsymbol{E}_t$	Battery state
$\eta_{\cdot}$	(Dis)charging efficiency
$\boldsymbol{u}_t$	Binary ESS mode indicator
$\mathcal{M}$	ESS unit set

† Uncertain parameter in the problem.

3 Solving the Scheduling Problem with RL-PaO

Predicting the uncertain constraint in Equation 3 shifts the objective problem and subsequently the solution schedule. Since the MILP optimizer internal states cannot be observed, conventional PtO/PaO methods assume that accurate prediction leads to optimized downstream costs. However, we show the assumption is false. Instead, we propose a novel MDP that permits an optimal policy to align prediction and optimization to improve downstream cost in a principled manner.

3.1 PtO and PaO

Figure 1C describes the standard training paradigm for PtO and PaO, respectively. PtO assumes that better estimation of the uncertain parameter can lead to lower downstream operational costs. However, this assumption rarely holds as the constraints can greatly influence the optimization objective and the resulting solution space. Recent studies show that there is no consistent correlation between prediction accuracy and downstream costs (Chen et al., 2022; Mandi et al., 2024). Motivated by this, supervised PaO methods augment the MSE prediction loss by some function of cost, in a hope that SGD training could also lead to a balanced prediction and cost-awareness:

$$\mathcal{L}_{\text{PaO}}(\boldsymbol{\theta}) = \alpha \cdot \underbrace{\widehat{\mathbb{E}}\left[(q_{\boldsymbol{\theta}}(\boldsymbol{z}) - \boldsymbol{\xi})^2\right]}_{\text{MSE}} + \beta \cdot \underbrace{f(\boldsymbol{x}, \hat{\boldsymbol{\xi}})}_{\text{Cost-aware}},$$

where $\alpha, \beta > 0$ are the weighting coefficients. In existing literature, α, β are typically computed through either heuristic search or a two-stage pre-training paradigm (Gabriele et al., 2026; Mandi et al., 2020; Shah et al., 2022).

The standard PaO formulation therefore has two intrinsic drawbacks: (1) minimizing the augmented loss corresponds to learning multi-task behavior. However, the conventional SGD can lead to solutions that neither minimizes the MSE loss nor achieving cost-awareness. (2) Naively balancing MSE with cost-awareness the term by a weighted sum often yields models that underperform the extreme cases: i.e. either $\alpha = 0$ or $\beta = 0$.

3.2 RL-PaO: A New RL paradigm for Solving the PaO Problem

Our proposed method closes the loop by a novel Markov Decision Process in which the agent acts to predict $\boldsymbol{\xi}$ and receives a calibrated reward.

Environment. We deliberately include system modeling, optimizing, planning and execution into environment with which the agent interact. The RL agent observes current solution from the MILP optimizer as context to predict the unknown constraint for the next day-ahead schedule. The solution under goes execution and yields real cost as part of the reward. This way, predictions can indeed be regarded as actions as they shift the environment (internal states of the MILP optimizer).

State. The state includes sufficient information to predict the uncertain parameter $\boldsymbol{\xi}$ similar to other prediction methods. In our formulation, the continuous state space is $\mathbb{R}^{72}$, consisting of the optimal scheduling plan $\boldsymbol{x}^*$, which is a comprehensive operational profile (e.g., battery dispatch and grid purchasing) that implicitly reflects the latent meteorological patterns and temporal dynamics governing the PV upper bounds. To enable efficient learning, raw high-dimensional features are processed via dimensionality reduction before being fed into the agent.

Action. As indicated in the Problem Formulation, the predicted quantity is the per-step PV generation upper bound $\boldsymbol{\xi} := U_t^{\text{pv}}$ that resides in $\mathbb{R}^{22}$. The actions are generated conditional on the dimensionality-reduced states. Note that unlike the existing supervised PaO methods, our design naturally constructs a feedback loop since: (1) prediction as action can influence the optimization and plan execution (through the inequality constraints) due to the deliberate design, see Figure 1; (2) each action receives a reward designed so that maximizing cumulative reward improves the alignment between prediction and downstream optimization.

Transition. After producing an action U_t^{pv}, it is used in the inequality constraint in Equation 3. We then invoke the MILP optimizer to solve the day-ahead scheduling problem to output the optimal schedule $\boldsymbol{x}^* \in \mathbb{R}^{72}$. Real-world execution then reveals the true PV output and the realized operational cost. Note that revealing P_t^{true} is a long and highly complex process that does not readily permit online RL. Therefore, offline learning from the existing data logs is required.

Reward. Motivated by the observation that better prediction does not necessarily lead to better downstream cost performance (Gabriele et al., 2026), we design our reward to be an empirical mixture of prediction accuracy and decision quality measured by realized scheduling cost:

$$r_t = -\beta \cdot \mathtt{N}(C_{\text{real}}) - (1-\beta) \cdot \mathtt{N}(e_{\text{pred}}) \tag{4}$$

where C_{real} is the actual electricity cost under the schedule, e_{pred} is prediction error, and $\mathtt{N}$ denotes normalization that scales the two reward terms to the same range. β is a hyperparameter balancing decision quality and prediction stability. Note that unlike the existing supervised PaO methods that augments loss, the association of cost to reward allows dynamically adjusting the agent behavior.

Policy. It is well-known that an MDP permits an optimal policy if the states and actions are finite, rewards are bounded, discount factor is less than one (Puterman, 1994). RL-PaO satisfies these assumptions and therefore we are guaranteed that a well-calibrated reward function ideally can lead to optimal alignment between prediction and optimization to achieve lower downstream operational costs, provided the algorithm can attain global optimum. This optimality stands in sheer contrast to the empirically designed mixture supervised loss of the conventional PaO methods (Shah et al., 2022; Mandi et al., 2024) that may be stuck in local optima that minimize neither prediction error nor downstream cost surrogate.

Training Algorithm. Having described the MDP, we use Proximal Policy Optimization (PPO) (Schulman et al., 2017) to train the RL agent. It has been shown that PPO can efficiently learn from offline data (Zhuang et al., 2023) and can converge to the global optimum (Liu et al., 2019). PPO iteratively updates the policy π_θ by maximizing the clipped surrogate objective:

$$L(\theta) = \mathbb{E}_t \left[\min \left(r_t(\theta) A_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon) A_t \right) \right] \tag{5}$$

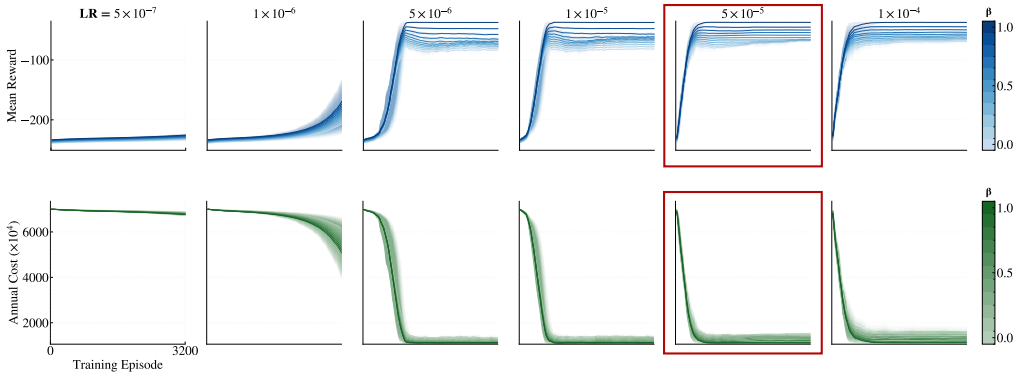

Figure 2: Cumulative rewards (upper row) and downstream operation costs (lower) across different learning rates. Transparency of lines set according to β value. RL-PaO is effective in (1) establishing correspondence between high rewards and lower costs; (2) the performance curves change smoothly along gradual increase of hyperparameters.

where $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}$ is the probability ratio, A_t is the advantage estimate, and ϵ controls the clipping range. The agent is trained in an offline fashion on historical data. In each training episode, the agent outputs an uncertain parameter estimate, then a black-box MILP optimizer is invoked to compute the corresponding optimal schedule. Based on the schedule and the revealed cost, reward is calculated against the realized true output. In the proposed framework, the MILP optimizer, its solution schedule plan and plan execution are all treated as part of the environment.

4 Experiments

We conduct experiments to answer the following research questions: **RQ1.** Is our proposed MDP well-posed in the sense that maximizing cumulative rewards equals minimizing downstream costs? **RQ2.** Can RL-PaO outperform the conventional PbO as well as PtO/PaO methods? **RQ3.** Can our method provide interpretability, including evidence that good prediction does not translate to better downstream operational costs? **RQ4.** Is our method robust towards hyperparameters?

4.1 Experimental Setup

Task. We use RL-PaO to tackle the day-ahead scheduling problem specified in Equation 3. We use a public dataset that records load, electricity price, temperature, etc. at the Osaka University in Japan[1]. The training set spans 2016–2018 and the test set covers 2019, all at hourly resolution. The MILP scheduling model is solved with the SCIP solver. The objective is to minimize the total annual operational cost of the microgrid subject to constraints. A training episode traverses the entire training set sequentially, with one training step being one day.

Baselines. We select all three types of methods as baselines. Oracle: The full-information benchmark that solves the scheduling problem using ground-truth PV generation, load and electricity price, serving as the theoretical lower bound of operation cost. PbO: DO uses point forecast; RO is with the worst-case uncertainty sets, and SO is based on scenario generation. PtO/PaO: We follow the standard PtO practice to train using MSE prediction loss. For PaO, we opt for the SPO+ surrogate regret loss (Dupont et al., 2024).

4.2 Results

RQ1 & RQ2: Performance. Figure 2 answers **RQ1** with an affirmative. It is visible that both cumulative rewards (upper) and downstream operational costs (lower) are improved along with training after learning rates are sufficient large, under all possible β values. This verifies that (1) our novel

[1] https://www.data.jma.go.jp/stats/etrn/

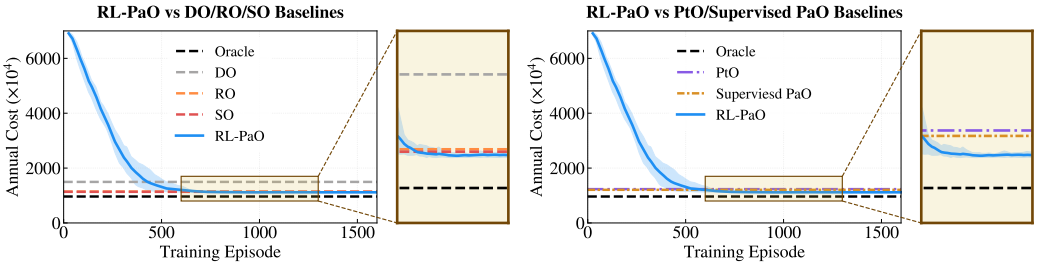

Figure 3: Comparison on annual cost. Blue solid line and shade are mean and standard deviation over 5 independent runs. **Left**: RL-PaO compared against the conventional PbO baselines that require model knowledge in uncertainty (DO, RO, SO). **Right**: comparison against PtO, PaO methods. In both cases the model-free RL-PaO achieves the best cost.

design that exploits prediction as action to change downstream execution is well-posed, since converged policies can indeed achieve lower downstream costs; (2) the RL-PaO formulation is smooth in hyperparameters in the sense that the performance curves change smoothly along with the increase of learning rates. There is no abrupt change after $\texttt{lr} \geq 5 \times 10^{-6}$, though an excessively large value (e.g. 10^{-3}) induces noticeable training oscillation. We consider $\texttt{lr} = 5 \times 10^{-5}, \beta = 0.9$ to be the optimal configuration for subsequent analysis.

Regarding **RQ2**, Figure 3 shows that RL-PaO also outperforms the baselines in terms of annual cost. The left hand side compares against the PtO baselines that require model knowledge on uncertainty. Yet, it is visible in the zoom-in plot that model-free RL attains the best final annual cost, underperforming only the oracle.

In the RHS figure we see a similar trend. RL-PaO is also the best performer among PtO and supervised PaO, leading to around 9×10^5 cost reduction than the second best method. This is because supervised PaO methods such as SPO+ require uncertain parameters to appear in the objective function, an assumption that is violated for the constraint-side PV uncertainty in our scheduling problem. RL-PaO naturally bypasses this limitation by directly optimizing end-to-end decision quality in a black-box manner. Table 3 further consolidates the observation: RL-PaO is the top player among all baseline methods. Comparing to Oracle, RL-PaO achieves the lowest cost gap 15.91% by model-free learning from data.

Table 3: Overall performance comparison on the 2019 test set. Annual cost is measured in the unit of $\times 10^4$. $\downarrow$ indicates lower is better.

Category	Method	Annual Cost ↓	Cost Gap (%) ↓
PbO	Oracle	963.76	0.00
	DO	1496.20	55.25
	RO	1145.20	18.83
	SO	1133.90	17.65
PtO/PaO	PtO	1233.40	27.98
	Supervised PaO	1207.90	25.33
	RL-PaO	**1117.10**	**15.91**

RQ3: Interpretability. Different from PtO/PaO baselines that do not provide interpretability for their predictions, RL-PaO offers insights from its policy space. Figure 4 shows policy evolution (Zhu et al., 2025) of the learned Gaussian distribution across 12 action dimensions at training day 0, 100, and 200. Prediction of the uncertain parameter is sampled from the policy. Starting from a broad and nearly uniform distribution at day 0, the policy becomes progressively narrower and moves toward the optimal decision region that aligns prediction with downstream cost optimization. The shift of the Gaussians at each dimension identifies different important regions that together characterize the optimal solution space for the studied problem.

Fig. 5 visualizes the joint distribution of daily prediction MAE and daily operation cost for four methods on the test dataset. It is visible that PtO clusters in the region with low prediction error but relatively high operation cost, as its training objective purely minimizes fitting error without considering downstream scheduling impact. This observation also supports the finding that lower estimation errors does do not translate to lower costs. It should also be noted that PtO has outliers shaded in purple, indicating extremely high costs are incurred – a phenomenon unique to PtO. We attribute this to its open-loop nature. Supervised PaO shifts slightly toward lower cost but remains confined to a similar error-cost regime. In contrast, RL-PaO disperses its error estimations but achieve significantly lower cost. This is even more clear from the last joint distribution plot that

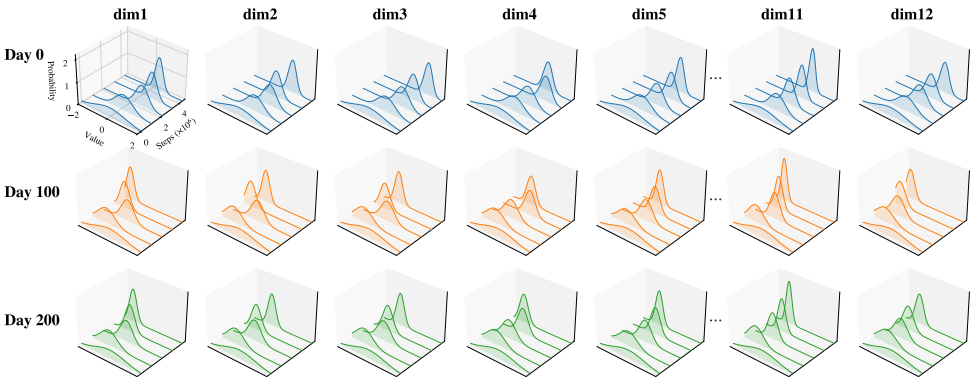

Figure 4: Policy evolution plot of the learned Gaussian distribution across 12 action dimensions at training day 0, 100, and 200. Recall that prediction of the uncertain parameter is sampled from the policy. As learning proceeds, the policies gradually narrow and shift toward the optimal decision region that aligns prediction with downstream cost optimization.

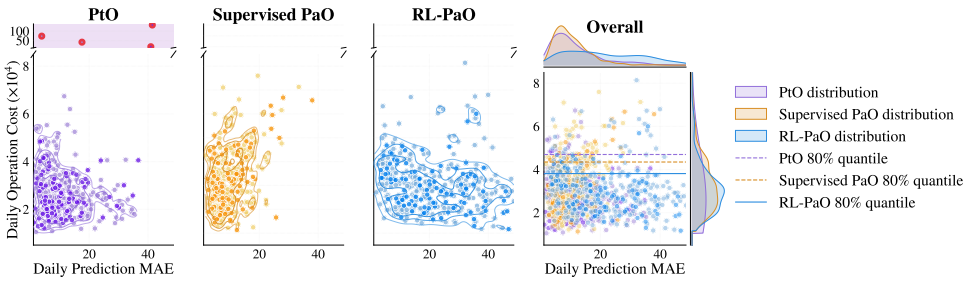

Figure 5: Prediction error versus operation cost, daily level. The first three windows show scatter plots on the test set. The last window shows a joint-distribution plot. Transparency distinguishes weekdays and weekends. PtO and PaO points are concentrated in a low estimation error region but the average cost is high. Outliers with extremely high costs are identified by red dots with purple background. By contrast, RL-PaO error distribution is more dispersed, but the average cost is lower, as can be seen from the 80% quantile line from the last window.

estimates the densities of MAE and costs for all compared methods. It can be seen that the RL-PaO's MAE distribution is more uniform, but its cost distribution places a majority of mass in lower-cost region, as indicated by the 80% quantile line. This observation further supports that high prediction accuracy does not equal better decision quality.

Figure 6 further quantifies this relationship at the annual level. The LHS shows the Pareto front of prediction error versus annual total cost. PtO achieves the lowest prediction error among the baselines but suffers from higher operation cost, lying in a Pareto-dominated region. By adjusting the reward weight β, RL-PaO spans a continuous trade-off curve and sits on the Pareto frontier, indicating that no competing methods can simultaneously achieve lower prediction error and lower operation cost. The RHS shows the overall correlation across all RL-PaO configurations with different β and learning rates. The Pearson correlation coefficient is 0.973, indicating a strong overall positive relationship between prediction error and operation cost. It is intuitive that for some of the configurations that lower MAE indeed indicates lower annual cost. But this relationship no longer holds for the majority of points scattered in the lower left corner.

RQ4: Hyperparameter Sensitivity. Figure 7 shows a Mahanttan plot of the proposed method across learning rates and reward weights β. Transparent dots are final operational costs of independent runs, with deep blue dots indicating the mean of the corresponding combinations. The upper row shows results of fixing β and varying `lr`, while the lower row fixing `lr` and varying β. It is visible that the superior performance of the proposed method is not a result of specific hyperparameter setting but can be attained from various combinations.

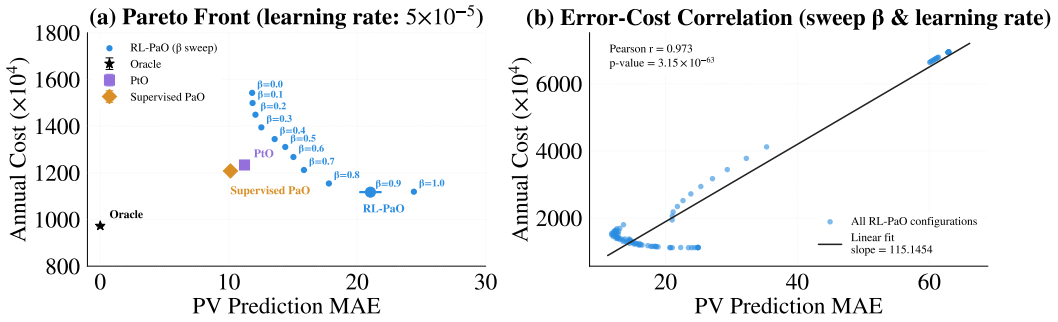

Figure 6: Prediction error versus operation cost, annual level. (a) Pareto front of annual cost against prediction MAE, recorded for $\texttt{lr} = 5 \times 10^{-5}$. The reward weight β spans a continuous trade-off curve and sits on the Pareto frontier, indicating that no competing methods can simultaneously achieve lower prediction error and lower operation cost. (b) Overall correlation across all β and learning rate configurations. The blue points scattered in the lower left corner supports the finding that lower prediction error does not equal lower cost.

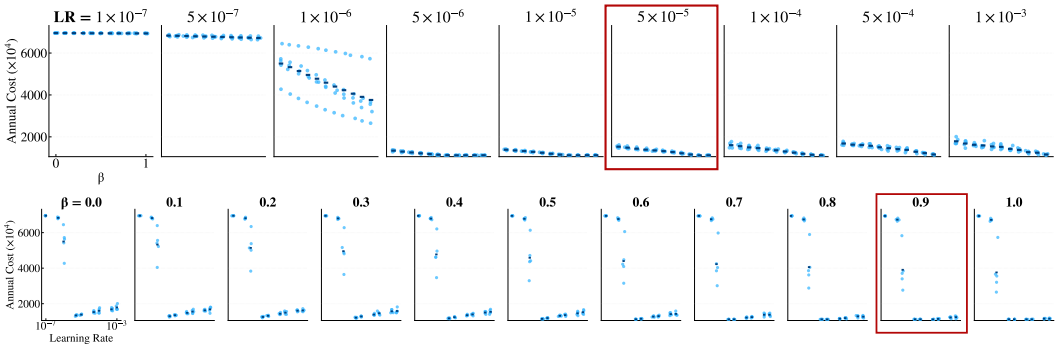

Figure 7: Mahanttan plot of the proposed method across learning rates and reward weights β. Transparent dots show final operational costs, with deep blue dots indicating the mean. (Upper) Results of fixing β and varying $\texttt{lr}$. (Lower) Fixing $\texttt{lr}$ and varying β. The superior performance of RL can be attained from various combinations.

5 Conclusion

Predicting the parameters is crucial for decision making under uncertainty. PtO and PaO predict the parameter through independent prediction loss and decision-augmented surrogate, respectively. This paper presented RL-PaO, an RL framework that treats prediction as action and integrates system formulation, optimization, and decision execution into a single environment, yielding an MDP that could be solved effectively by existing RL methods. Experiments on the day-ahead scheduling problem using real data at the Osaka University verified that RL-PaO achieved lowest annual cost than all non-oracle baselines. Extensive analysis showed that our RL-PaO design can also provide strong interpretability and is robust towards hyperparameters.

We identify a potential limitation for this work: while the black-box treatment of the downstream problem delivers strong generalizability, it also introduces computational overhead during training. Every training step requires solving the full downstream optimization problem and runtime can become prohibitive for large-scale complex systems with a vast set of decision variables.

Future work can be pursued in two directions. First, it would be helpful to extend to joint prediction of multiple uncertain parameters of different physical natures, including load, electricity prices and distributed renewable output, and examine the performance of a single RL agent in such multi-variable settings. Second, generating longer-horizon schedules such as one-week dispatch plans could also help alleviate computational burden.

AI USE STATEMENT

In this work, we used AI for polishing the manuscript. We have not used generative AI tools for research problem identification, core idea construction, algorithm design, and the key design of reward and state in our RL agent, and the rest of the required disclosure tasks are not applicable to this work. Specifically, all core research ideas, algorithm designs are fully developed by the authors. The training and test datasets are built on real-world data from Osaka University, and none of these data are AI-generated. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We have taken measures to ensure the reproducibility of all results in our work. The implementation code of RL-PaO and PPO is uploaded as supplementary material. The use of the MILP model and the SCIP solver, experimental settings such as hyperparameter sweeps, random seeds are detailed in Appendix A.2.

REFERENCES

Brandon Amos and J. Zico Kolter. Optnet: Differentiable optimization as a layer in neural networks. In Proceedings of the 34th International Conference on Machine Learning (ICML), volume 70 of Proceedings of Machine Learning Research, pp. 136–145. PMLR, 2017.

Gah-Yi Ban and Cynthia Rudin. The big data newsvendor: Practical insights from machine learning. Operations Research, 67(1):90–108, 2019.

A. Ben-Tal and A. Nemirovski. Robust optimization – methodology and applications. Mathematical Programming, 92(3):453–480, 2002.

Q. Berthet, M. Blondel, O. Teboul, M. Cuturi, J.-P. Vert, and F. Bach. Learning with differentiable perturbed optimizers. In Advances in Neural Information Processing Systems, volume 33, pp. 9508–9519, 2020.

D.P. Bertsekas. Nonlinear Programming. Athena Scientific, 2nd edition, 1999.

Dimitris Bertsimas and Melvyn Sim. The price of robustness. Operations Research, 52(1):35–53, 2004.

J.R. Birge and F. Louveaux. Introduction to Stochastic Programming. Springer, 2nd edition, 2011.

Stephen Boyd and Lieven Vandenberghe. Convex Optimization. Cambridge University Press, Cambridge, UK, 2004.

Xianbang Chen, Yafei Yang, Yikui Liu, and Lei Wu. Feature-Driven Economic Improvement for Network-Constrained Unit Commitment: A Closed-Loop Predict-and-Optimize Framework. IEEE Transactions on Power Systems, 37(4):3104–3118, 2022.

Priya L. Donti, Brandon Amos, and J. Zico Kolter. Task-based end-to-end model learning in stochastic optimization. In Advances in Neural Information Processing Systems, volume 30, 2017.

Chloé Dupont, Pietro Favaro, Fran¸cois Vallée, Bruno Francois, and Jean-Fran¸cois Toubeau. Decision-focused learning for optimized participation of an industrial consumer in energy-only and reserve markets. In 2024 IEEE PES Innovative Smart Grid Technologies Europe (ISGT EUROPE), pp. 1–5, 2024.

Adam N. Elmachtoub and Paul Grigas. Smart "predict, then optimize". Management Science, 68 (1):9–26, 2022.

A.N. Elmachtoub, J.C.N. Liang, and R. McNellis. Decision trees for decision-making under the predict-then-optimize framework. In International Conference on Machine Learning, pp. 2858–2867. PMLR, 2020.

A. Ferber, B. Wilder, B. Dilkina, and M. Tambe. Mipaal: Mixed integer program as a layer. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 34, pp. 1504–1511, 2020.

Kris Johnson Ferreira, Bin Hong Alex Lee, and David Simchi-Levi. Analytics for an online retailer: Demand forecasting and price optimization. Manufacturing & Service Operations Management, 18(1):69–88, 2016.

Giuseppe Gabriele, Fabio Pavirani, Seyed Soroush Karimi Madahi, and Chris Develder. Forecasting what matters: Decision-focused rl for controlled ev charging with unknown departure times. In Proceedings of the 17th ACM International Conference on Future and Sustainable Energy Systems (E-Energy '26), 2026.

Jérémie Gallien, Adam J. Mersereau, Andres Garro, Alberte Dapena Mora, and Martín Nóvoa Vidal. Initial shipment decisions for new products at Zara. Operations Research, 63(2):269–286, 2015.

L. Kong, J. Cui, Y. Zhuang, R. Feng, B.A. Prakash, and C. Zhang. End-to-end stochastic optimization with energy-based model. In Advances in Neural Information Processing Systems, volume 35, pp. 11341–11354, 2022.

J. Kotary, F. Fioretto, P. Van Hentenryck, and B. Wilder. End-to-end constrained optimization learning: A survey. In Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, pp. 4475–4482, 2021.

Alan Anis Lahoud, Ahmad Saeed Khan, Erik Schaffernicht, Marco Trincavelli, and Johannes Andreas Stork. Predict-and-optimize techniques for data-driven optimization problems: A review. Neural Processing Letters, 57:40, 2025.

Boyi Liu, Qi Cai, Zhuoran Yang, and Zhaoran Wang. Neural trust region/proximal policy optimization attains globally optimal policy. In Advances in Neural Information Processing Systems, volume 32, pp. 1–12, 2019.

J. Mandi and T. Guns. Interior point solving for lp-based prediction + optimisation. Advances in Neural Information Processing Systems, 33:7272–7282, 2020.

J. Mandi, P.J. Stuckey, and T. Guns. Smart predict-and-optimize for hard combinatorial optimization problems. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 34, pp. 1603–1610, 2020.

Jayanta Mandi, James Kotary, Senne Berden, Maxime Mulamba, Victor Bucarey, Tias Guns, and Ferdinando Fioretto. Decision-focused learning: Foundations, state of the art, benchmark and future opportunities. Journal of Artificial Intelligence Research, 81:1623–1701, 2024.

Jayanta Mandi, Marianne Defresne, Senne Berden, and Tias Guns. Feasibility-aware decision-focused learning for predicting parameters in the constraints. In Advances in Neural Information Processing Systems, volume 38, 2025.

M.V. Pogančić, A. Paulus, V. Musil, G. Martius, and M. Rolinek. Differentiation of blackbox combinatorial solvers. In International Conference on Learning Representations, 2020.

Martin L. Puterman. Markov Decision Processes: Discrete Stochastic Dynamic Programming. John Wiley & Sons, Inc., New York, NY, USA, 1st edition, 1994.

Meng Qi, Mengxin Wang, and Zuo-Jun Shen. Smart feasibility pump: Reinforcement learning for (mixed) integer programming. In ICML Workshop: Reinforcement Learning for Real Life, 2021.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. arXiv:1707.06347, 2017.

Sanket Shah, Kai Wang, Bryan Wilder, Andrew Perrault, and Milind Tambe. Decision-focused learning without decision-making: Learning locally optimized decision losses. In Advances in Neural Information Processing Systems, volume 35, 2022.

Alexander Shapiro, Darinka Dentcheva, and Andrzej Ruszczyński. Lectures on Stochastic Programming: Modeling and Theory. Number 9 in MOS-SIAM Series on Optimization. SIAM, Philadelphia, PA, 2009.

Mattia Silvestri, Senne Berden, Gaetano Signorelli, Ali İrfan Mahmutoğulları, Jayanta Mandi, Brandon Amos, Tias Guns, and Michele Lombardi. Score function gradient estimation to widen the applicability of decision-focused learning. Journal of Artificial Intelligence Research, 85, 2026. doi: 10.1613/jair.1.19498.

Yunhao Tang, Shipra Agrawal, and Yuri Faenza. Reinforcement learning for integer programming: Learning to cut. In Proceedings of the 37th International Conference on Machine Learning, Proceedings of Machine Learning Research, pp. 9367–9376, 2020.

K. Wang, B. Wilder, A. Perrault, and M. Tambe. Automatically learning compact quality-aware surrogates for optimization problems. Advances in Neural Information Processing Systems, 33: 9586–9596, 2020.

Lulu Wen, Kaile Zhou, Shanlin Yang, and Xinhui Lu. Optimal load dispatch of community microgrid with deep learning based solar power and load forecasting. Energy, 171:1053–1065, 2019.

Dafang Zhao, Daichi Watari, Yuki Ozawa, Ittetsu Taniguchi, Toshihiro Suzuki, Yoshiyuki Shimoda, and Takao Onoye. Data-driven online energy management framework for hvac systems: An experimental study. Applied Energy, 352:121921, 2023.

Lingwei Zhu, Haseeb Shah, Han Wang, Yukie Nagai, and Martha White. q-exponential policy optimization. In International Conference on Learning Representations (ICLR), 2025.

Zifeng Zhuang, Kun LEI, Jinxin Liu, Donglin Wang, and Yilang Guo. Behavior proximal policy optimization. In The Eleventh International Conference on Learning Representations, 2023.

A Appendix

A.1 Related Work

PbO. DO replaces uncertainties parameters with deterministic estimates (Boyd & Vandenberghe, 2004; Bertsekas, 1999) but loses feasibility and optimality under time-varying uncertainty. RO guarantees worst-case feasibility over a predefined uncertainty set (Ben-Tal & Nemirovski, 2002; Bertsimas & Sim, 2004) but yields over-conservative solutions with excessive costs. SO optimizes expected cost via probability distributions (Shapiro et al., 2009; Birge & Louveaux, 2011), yet relies on valid distributions, neglects action risks, and is computationally expensive at scale.

PtO. PtO decouples prediction and optimization: a predictor minimizes forecast error, and a solver generates decisions from predicted parameters (Elmachtoub & Grigas, 2022). This modular framework and allows independent module tuning, with applications spanning microgrid dispatch (Wen et al., 2019), inventory management (Gallien et al., 2015), and retail pricing. (Ferreira et al., 2016) combined random forest forecasts with integer programming for pricing; (Gallien et al., 2015) paired linear regression with MILP for inventory optimization. In energy systems (Zhao et al., 2023; Chen et al., 2022), LSTMs predict renewable generation for downstream solvers. However, PtO suffers from prediction-decision mismatch, and lower forecast error does not equal better decision.

PaO. To address prediction-decision mismatch, PaO optimizes directly for downstream cost rather than pure prediction error (Kotary et al., 2021; Lahoud et al., 2025; Mandi et al., 2024). Gradient-based differentiable PaO embeds solvers as differentiable layers for end-to-end training, including KKT-based OptNet (Amos & Kolter, 2017; Donti et al., 2017; Ferber et al., 2020) and interior-point regularized methods (Mandi & Guns, 2020); both require problem-specific derivation and are sensitive to downstream model complexity. Surrogate loss and perturbation approaches avoid explicit differentiation: SPO constructs surrogate losses via duality theory (Elmachtoub & Grigas, 2022), and perturbation-based schemes estimate gradients from solver outputs for black-box compatibility (Berthet et al., 2020; Pogančić et al., 2020; Shah et al., 2022), trading generality for reduced gradient fidelity. Non-gradient discrete PaO bypasses gradient computation and optimizes decision loss directly (Elmachtoub et al., 2020; Ban & Rudin, 2019), but is limited to simple architectures and lacks high-dimensional scalability. Engineering techniques including warm-starting, approximate solvers, and pre-computed surrogates cut PaO training cost (Mandi et al., 2020; Wang et al., 2020; Kong et al., 2022). More recently, (Mandi et al., 2025) propose losses that balance infeasibility and suboptimality without restricting the downstream problem to LP/MIP.

A.2 Implementation Details

Table 4: Hyperparameters and configurations.

Hyperparameter	Value
Reward weight β	Swept in $\{0.0, 0.1, 0.2, \ldots, 1.0\}$
Learning rate	Swept in $\{10^{-7}, 5{\times}10^{-7}, 10^{-6}, 5{\times}10^{-6}, 10^{-5}, 5{\times}10^{-5}, 10^{-4}, 5{\times}10^{-4}, 10^{-3}\}$
Discount factor γ	0.99
PPO clip range	0.2
GAE parameter λ	0.95
Value function coefficient v_f	0.5
Max gradient norm	0.5
Epochs per policy update	10
Rollout steps per update	2048
Mini-batch size	64
Hidden layer size (actor / critic)	64
Number of hidden layers (actor / critic)	2
Base random seed	42
Number of independent random seeds	5
Steps per training episode	1096
MILP relative optimality gap	0.01
MILP time limit per solve	0.3 s

Dataset and Environment. The electricity price, solar irradiance, ambient temperature data etc. are sourced from public datasets.[2] The training set spans years 2016–2018, and the held-out test set covers year 2019. All data are at 2-hour resolution, yielding 12 time steps per day for the day-ahead scheduling horizon.

Optimization Model. The day-ahead energy scheduling problem is formulated as a mixed-integer linear program (MILP). The objective minimizes total operational cost. The model is solved via the SCIP solver, with a 1% relative optimality gap and a 0.3 s time limit per solve to ensure computational efficiency during reinforcement learning training.

RL-PaO Agent. The RL agent is built on the Proximal Policy Optimization (PPO) algorithm with a standard actor-critic MLP architecture. Raw high-dimensional scheduling solution vectors from the MILP solver are compressed into low-dimensional observations via principal component analysis (PCA). The composite reward function is defined as $r = -\beta \cdot J_{\text{cost}} - (1-\beta) \cdot J_{\text{MSE}}$, where $\beta \in [0, 1]$ balances the operational cost term and the prediction error term.

Training and Hyperparameter Sweep. We sweep the reward weight $\beta \in [0, 1]$ and the policy learning rate $\text{lr} \in [10^{-7}, 10^{-3}]$. All results are averaged over 5 independent random-seed runs, with min–max ranges shown as shaded areas in the figures. The full framework is implemented in Python. Each training step corresponds to one day-ahead scheduling instance. One full training episode comprises 1096 steps, forming a complete sequential pass over the entire 2016–2018 training set.

A.3 Additional Results

This section provides supplementary daily cost breakdowns and detailed ablation analyses to support the findings presented in the main paper.

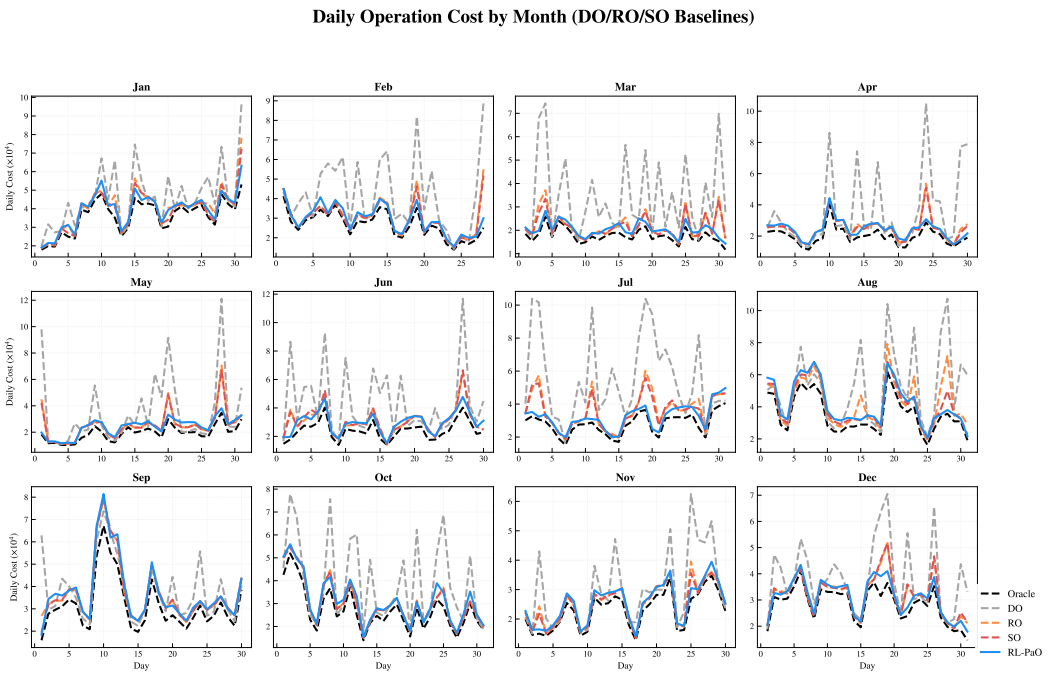

Figure C1: Daily operation cost by month compared with conventional optimization baselines on the 2019 test set. Each subplot corresponds to one calendar month, comparing Oracle, DO, RO, SO and the proposed RL-PaO.

Figure C1 decomposes the daily operational cost on a per-month basis over the 2019 test set, comparing the proposed RL-PaO with three conventional DO, RO, and SO, as well as the full-information Oracle lower bound. The DO baseline shows the highest cost volatility and overall cost level, with

[2] https://www.data.jma.go.jp/stats/etrn/

particularly large deviations. RO and SO deliver more stable performance but still remain significantly above the Oracle benchmark. By contrast, RL-PaO closely follows the Oracle costs across all twelve months, maintaining near-ideal performance under diverse seasonal PV generation conditions.

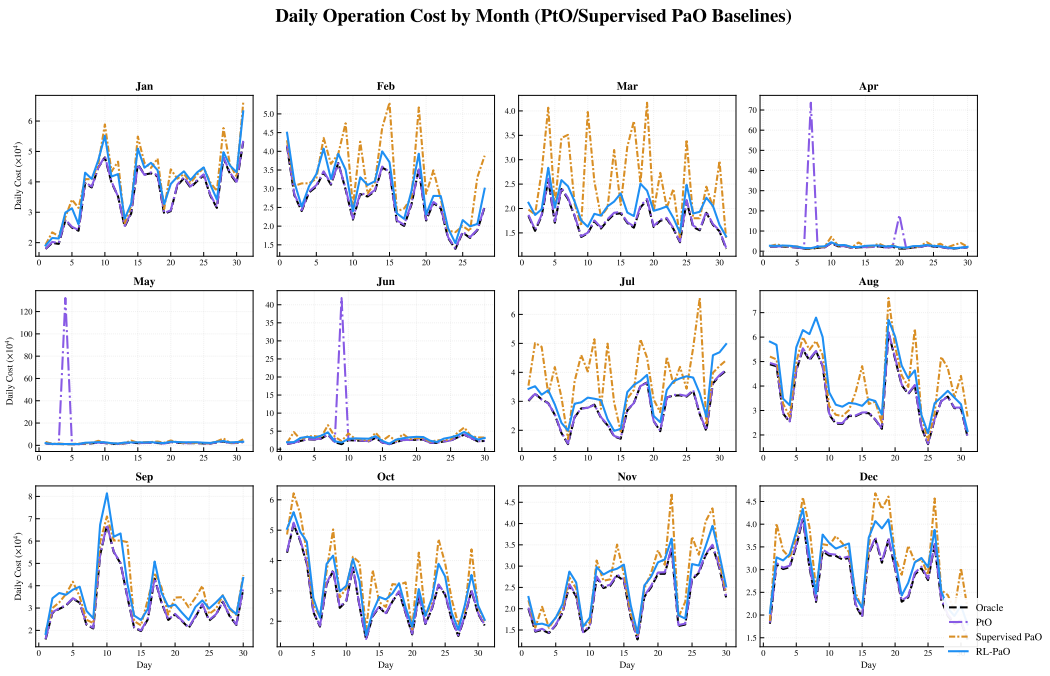

Figure C2: Daily operation cost by month compared with predict-and-optimize baselines on the 2019 test set. Each subplot corresponds to one calendar month, comparing Oracle, predict-then-optimize (PtO), supervised PaO, improved PaO and the proposed RL-PaO.

Figure C2 presents the monthly daily cost comparison against P-O baselines, including the standard two-stage PtO framework and supervised PaO. The PtO approach exhibits extreme cost spikes in spring and summer months (April–June), reflecting the unstableness of pure prediction-error-driven training under constraint uncertainty. Supervised PaO improves upon PtO but still suffers from notable performance fluctuations. The proposed RL-PaO achieves cost levels closest to the Oracle across all months without anomalous peaks, demonstrating substantially stronger robustness than both supervised prediction-optimization baselines.

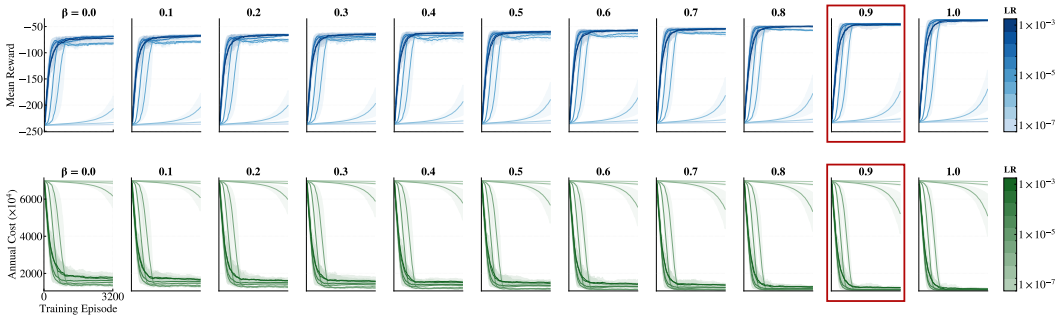

Figure C3: Convergence behavior of RL-PaO across different reward weights β and learning rates. Top: mean reward convergence across training episodes. Bottom: annual operation cost convergence across training episodes. Each column corresponds to a fixed β value, and each colored curve corresponds to a different learning rate.

Figure C3 presents the full convergence behavior of RL-PaO across a grid of reward weights β and learning rates, with mean reward shown in the top row and annual operation cost in the bottom row. Learning rate dominates convergence speed: configurations with very low learning rates fail to converge within the training horizon, while moderate to high rates converge within 500 episodes. Reward weighting β determines the final performance level: higher β consistently yields better final reward and lower operational cost across all learning rates. This figure is also presented and discussed in the main experimental section.

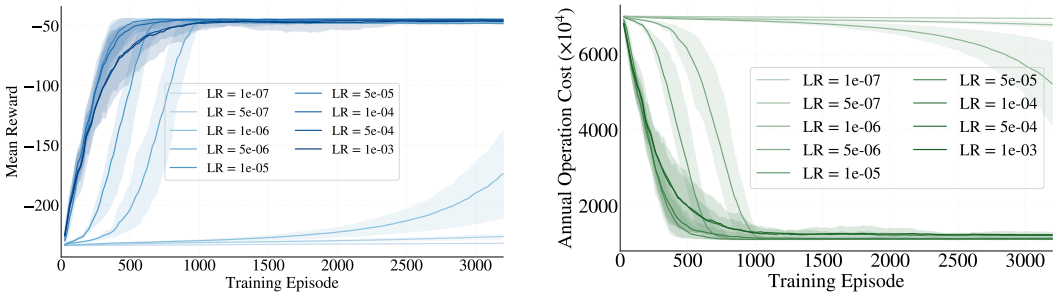

Figure C4: **Left:** Reward and operation cost convergence under different learning rates with β fixed at 0.90. Each colored band corresponds to one learning rate configuration. **Right:** Reward convergence under different reward weights β with learning rate fixed at 5×10^{-5}. Each colored band corresponds to one β configuration.

Figures C4 present mean reward convergence and annual operation cost convergence, respectively, across different learning rates with β fixed at 0.90. Learning rate is the dominant factor governing convergence speed: very low learning rates (e.g., 10^{-7}) converge extremely slowly and fail to approach the optimal performance level even after 3000 training episodes. Convergence accelerates markedly as the learning rate increases, with moderate rates in the 5×10^{-5} to 10^{-4} range reaching a stable performance plateau within approximately 500 episodes. In terms of final solution quality, moderate learning rates yield the highest mean reward (Figure C4) and the lowest operational cost (Figure C5) with the tightest variance across random seeds. Excessively high learning rates (e.g., 10^{-3}) do not further speed up convergence, but instead cause marginal performance degradation and larger fluctuation, as overly large parameter update steps tend to overshoot the optimal policy. This jointly confirms that the learning rate imposes a trade-off between convergence speed and final scheduling performance, and the 5×10^{-5} setting used in the main experiment strikes the optimal balance between fast convergence, near-optimal cost, and training stability.

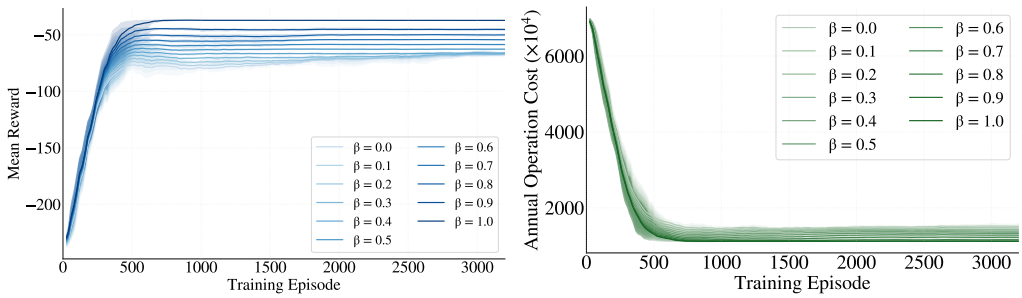

Figure C5: **Left:** Annual operation cost convergence under different reward weights β with learning rate fixed at 5×10^{-5}. Each colored band corresponds to one β configuration. **Right:** Annual operation cost convergence under different learning rates with β fixed at 0.90. Each colored band corresponds to one learning rate configuration.

Figures C5 present reward convergence and annual operation cost convergence, respectively, across different reward weights β with the learning rate fixed at 5×10^{-5}. All β configurations exhibit similar convergence speeds, reaching a stable performance plateau around 500 training episodes, indicating that the cost weighting coefficient barely affects the convergence rate of the RL training pipeline. In terms of final performance, there is a consistent monotonic trend: higher β values yield

higher mean reward (Figure C6) and lower operational cost (Figure C7) as the model places greater emphasis on minimizing downstream scheduling cost. Notably, the performance gain diminishes as β approaches 1.0: $\beta = 0.9$ already achieves near-optimal cost levels comparable to $\beta = 1.0$, while maintaining tighter variance across random seeds. This jointly confirms that cost-oriented reward weighting effectively improves decision quality, and $\beta = 0.9$ strikes the optimal balance between final scheduling performance and training stability.

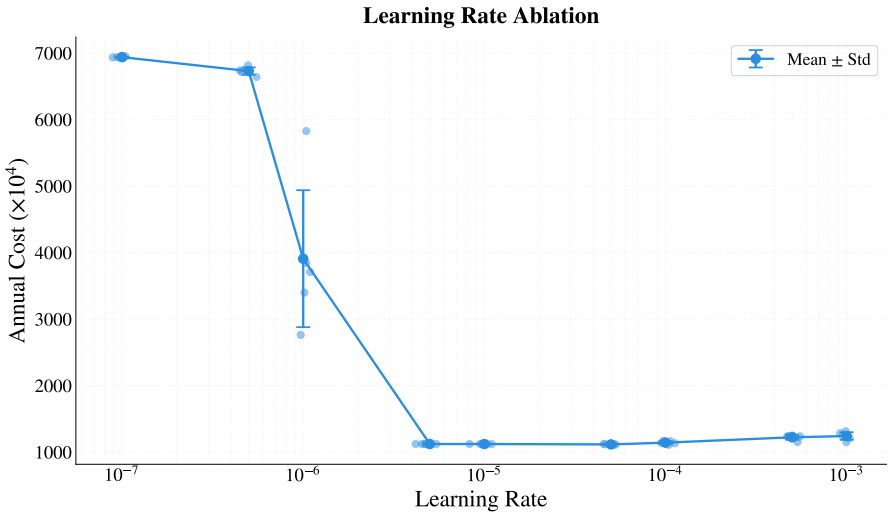

Figure C6: Final annual operation cost with mean and standard deviation across different learning rates, with β fixed at 0.90.

Figure C6 quantifies the final annual operational cost (mean and standard deviation) across different learning rates with β fixed at 0.90. The cost stays at an extremely high level when the learning rate falls below 10^{-6}, as the algorithm fails to converge sufficiently within the training horizon. It drops to the lowest plateau across the 10^{-5} to 10^{-4} range, and climbs slowly as the learning rate increases further past 5×10^{-5}. This pattern indicates that the learning rate is not the higher the better: excessively high learning rates produce overly large parameter update steps that tend to overshoot the optimal solution and induce training oscillation, which leads to marginally suboptimal final scheduling performance and degraded stability. The 5×10^{-5} setting adopted in the main experiment achieves the balance between convergence speed and solution quality.

Figure C7 quantifies the final annual operational cost (mean and standard deviation) across different reward weights β with the learning rate fixed at 5×10^{-5}. Overall, the cost decreases monotonically as β increases from 0.0 to 0.9, reaching the minimum (optimal) level at $\beta = 0.9$. However, the cost rebounds slightly when β further increases to 1.0. This pattern indicates that excessively weighting the operational cost term while compromising the reward regularization component degrades training stability and leads to marginally suboptimal final performance. The dashed line marks the $\beta = 0.9$ value adopted in the main experiment, which achieves the balance between cost-oriented decision optimization and training stability.

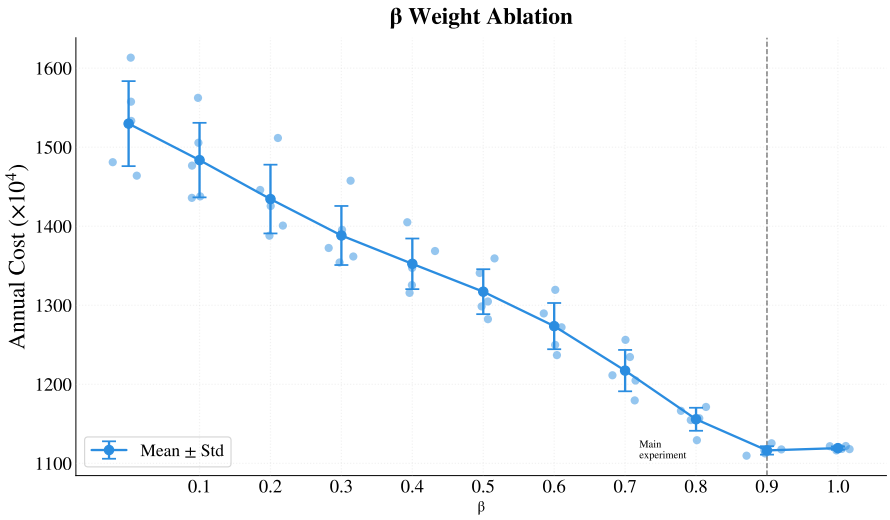

Figure C7: Final annual operation cost with mean and standard deviation across different reward weights β, with learning rate fixed at 5×10^{-5}. The dashed line marks the β value adopted in the main experiment.