3D-Learning: Diffusion-Augmented Distributionally Robust Decision-Focused Learning

Jiaqi Wen
Department of Computer Science
University of Houston
Texas, USA
jwen5@cougarnet.uh.edu

Lei Fan
Department of Engineering Technology
University of Houston
Texas, USA
lfan8@central.uh.edu

Jianyi Yang*
Department of Computer Science
University of Houston
Texas, USA
jyang66@uh.edu

***Abstract*—Predict-then-Optimize (PTO) pipelines are widely employed in computing and networked systems, where Machine Learning (ML) models are used to predict critical contextual information for downstream decision-making tasks such as cloud LLM serving, data center demand response, and edge workload scheduling. However, these ML predictors are often vulnerable to out-of-distribution (OOD) samples at test time, leading to significant decision performance degradation due to large prediction errors. To address the generalization challenges under OOD conditions, we present the framework of Distributionally Robust Decision-Focused Learning (DR-DFL), which trains ML models to optimize decision performance under the worst-case distribution. Instead of relying on classical Distributionally Robust Optimization (DRO) techniques, we propose Diffusion-Augmented Distributionally Robust Decision-Focused Learning (`3D-Learning`), which searches for the worst-case distribution within the parameterized space of a diffusion model. By leveraging the powerful distribution modeling capabilities of diffusion models, `3D-Learning` identifies worst-case distributions that remain consistent with real data, achieving a favorable balance between average and worst-case scenarios. Empirical results on an LLM resource provisioning task demonstrate that `3D-Learning` outperforms existing DRO and Data Augmentation methods in OOD generalization performance.**

***Index Terms*—Diffusion Models, Distributionally Robust Learning, Decision-Focused Learning.**

I. Introduction

Many context-aware decision-making problems in computing and communication networks can be formulated within the Predict-then-Optimize (PTO) framework, where effective decision-making critically depends on the accurate prediction of system context [28]. One key example is resource provisioning for cloud (Large Language Model) LLM serving, where the accurate token-level workload prediction is essential for efficiently allocating computing resources (e.g., GPU cores or frequency) to mitigate over-provisioning or service degradation [4], [43], [46]. Another critical application is demand response in AI data centers, where accurate prediction of AI workloads of different types and renewable energy availability is essential for determining when how much to defer AI computation to reduce energy costs while maintaining service-level agreements [6], [21], [48].

In many PTO applications, decision performance is highly sensitive to specific types of prediction errors. For instance, in cloud resource provisioning, underestimating the workload can lead to sever service degradation, whereas overestimation may simply incur additional cost. As a result, the decision performance can be sub-optimal by training ML models solely to minimize prediction error without considering its impact on downstream decision objectives. Decision-Focused Learning (DFL) [9], [28] addresses the limitation of traditional prediction-focused learning by training ML models in an end-to-end manner to directly optimize the final decision objective. By aligning the learning process with decision performance, DFL provides more effective and robust decision-making strategies.

Despite its advantages over prediction-focused learning, DFL still struggles to generalize under Out-of-Distribution (OOD) testing scenarios—a common challenge in dynamic ML-based systems. In LLM serving, for instance, shifts in user demand, task types, or market dynamics can lead to fluctuating workload patterns that differ significantly from those seen during training. When faced with such distribution shifts, a DFL model trained solely on in-distribution data may make decisions with poor service quality or high operational costs. This vulnerability arises because standard DFL optimizes decision performance based only on the empirical training distribution, without accounting for potential distribution shifts at test time. To address this limitation, we introduce the Distributionally Robust Decision-Focused Learning (DR-DFL) framework, which seeks to optimize decision performance under the worst-case distribution within an ambiguity set around the training data. Theoretically, with a well-designed ambiguity set, DR-DFL enables more resilient decision-making under real-world OOD deployment scenarios [12], [18], [30].

A central challenge in DR-DFL lies in the modeling of the ambiguity set, which defines a distribution discrepancy measure to capture meaningful variations around the training distribution. However, Distributionally Robust Optimization (DRO) methods with traditional ambiguity modeling often lead to suboptimal DR-DFL performance. Ambiguity sets based on ϕ-divergences (e.g., KL divergence) restrict the

* Corresponding author: Jianyi Yang (jyang66@uh.edu). Jiaqi Wen and Jianyi Yang were supported in part by University of Houston Start-up Funds 74825 [R0512039] and 74833 [R0512042]. Lei Fan was supported by UH Energy Transition Institute.

distributions to be absolutely continuous with respect to the training distribution, thereby excluding test distributions with shifted support. As a result, DR-DFL with ϕ-divergence-based ambiguity sets may yield non-robust solutions under support shift, even when enhanced with data augmentation techniques. In contrast, Wasserstein distance-based ambiguity sets allow for support variation but often result in intractable optimization problems, particularly when the decision objective is non-convex. In such cases, solving the DRO problem typically requires relaxations, which may result in overly conservative training, ultimately degrading the average-case performance. These limitations highlight the need for a more expressive and computationally tractable ambiguity modeling in DR-DFL.

This paper focuses on addressing the challenges of ambiguity set design and proposes a novel DR-DFL framework based on diffusion models. The main contributions are summarized as follows:

- **Diffusion-based Ambiguity Modeling**. We introduce a new ambiguity modeling based on the score matching loss of diffusion models, offering several advantages. First, it allows the ambiguity set to include distributions with diverse and shifted support. Second, by constraining the score matching loss, we can ensure that candidate distributions remain consistent with the underlying data distribution. Finally, it enables the tractable search for the worst-case distribution within the parameterized space of the diffusion model.
- **Diffusion-Augmented Algorithm Design**. We propose the *<u>D</u>iffusion-Augmented <u>D</u>istributionally Robust <u>D</u>ecision-Focused Learning* (`3D-Learning`) algorithm, which integrates diffusion-based ambiguity modeling into the DR-DFL pipeline. It addresses the challenges of the constrained, non-convex inner maximization problem by combining the dual learning techniques with diffusion policy optimization tricks. Furthermore, given the inner maximization output, we design the min-max solver for DR-DFL based on the diffusion sampling. `3D-Learning` is the first training framework for DR-DFL in the parameterized diffusion space.
- **Performance Evaluation on LLM Resource Provisioning**. We formulate the resource provisioning problem for LLM inference as a PTO pipeline and evaluate `3D-Learning` against a range of DRO and data augmentation baselines. Simulation results demonstrate that `3D-Learning` significantly outperforms traditional DRO methods and data augmentation approaches in both average-case and worst-case performance across test datasets exhibiting various distribution shifts. Moreover, under various noisy perturbation scenarios, `3D-Learning` demonstrates exceptional noise robustness and stability.

II. Formulation and Appications

A. Decision-Focused Learning

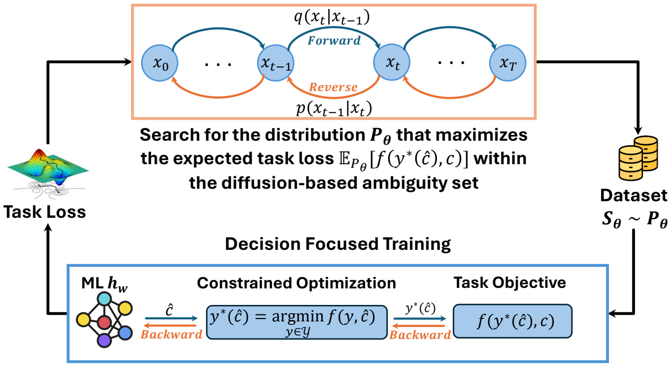
$q(x_t|x_{t-1})$
Forward
x_0
x_{t-1}
x_t
x_T
Reverse
$p(x_{t-1}|x_t)$
Search for the distribution P_θ that maximizes the expected task loss $\mathbb{E}_{P_\theta}[f(y^*(\hat{c}),c)]$ within the diffusion-based ambiguity set
Task Loss
Dataset
$S_\theta \sim P_\theta$
Decision Focused Training
ML h_w
Constrained Optimization
Task Objective
$\hat{c}$
Backward
$y^*(\hat{c}) = \operatorname{argmin}_{y \in \mathcal{Y}} f(y,\hat{c})$
$y^*(\hat{c})$
Backward
$f(y^*(\hat{c}),c)$

Fig. 1. Framework of Decision Focused Learning

PTO problems have wide applications in computing and communication networks [42], [44], [47]. In a PTO pipeline as shown in 1, a ML predictor $h_w \in \mathcal{H}$ with weight w maps an input v (e.g. historical context or side information) into a context prediction $\hat{c}$. Then a decision-making step is taken to optimize the decision objective based on the predicted context $\hat{c}$. We consider a general decision-making objective as

$$y^*(c) = \arg\min_{y \in \mathcal{Y}} f(y, c), \tag{1}$$

where f is the objective functions and $\mathcal{Y}$ incorporates the constraints on the decision variable y.

In a standard ML training loss, we usually define the training loss by some discrepancy measure $l(\hat{c}, c)$ such as Mean Squared Error (MSE) or Cross-Entropy (CE) between the predicted label $\hat{c}$ and the ground-truth label c. Denote $x = (v, c)$ as a labeled sample. We minimize the empirical loss $\mathbb{E}_{S_0}[l(h_w(v), c)]$ based on a training dataset $S_0 = \{x_1, \cdots, x_{|S_0|}\}$ with $|S_0|$ samples drawn from an underlining distribution P_0. Existing works [28] have demonstrated that such standard ML training may not achieve satisfactory decision performance because the information of decision objective (1) is not incorporated in the ML training. Therefore, DFL is proposed to train the ML model in an end-to-end style by directly minimizing the decision objective $f(y^*(\hat{c}), c)$. The training objective of DFL is expressed as

$$h_w = \arg\min_{h_w \in \mathcal{H}} \mathbb{E}_{S_0}[f(y^*(\hat{c}), c)], \tag{2}$$

where $y^*(\hat{c})$ is the solution of (1) given a predicted context $\hat{c} = h_w(v)$. DFL is more general than standard ML training because it reduces to a standard ML training by choosing the decision objective as the label discrepancy measures.

B. Distributionally Robust Decision-Focused Learning

Although DFL directly optimizes the decision objective, it suffers from significant performance drop in the OOD testing environment in the same way as standard ML. In real applications with examples in Section II-C, the distribution of testing context can shift a lot over time, resulting in unreliable decision performance.

DRO is a widely-adopted framework to improve OOD generalization performance. In the PTO pipeline, we are concerned about the decision objective in (1), so we introduce

DR-DFL which trains predictor to minimize the worst-case decision objective by solving a min-max optimization:

$$h_w = \arg \min_{h_w \in \mathcal{H}} \max_{P \in \mathcal{B}(P_0, \epsilon)} \mathbb{E}_P[f(y^*(\hat{c}), c)], \tag{3}$$

where P_0 is the underlining distribution of the training dataset S_0, and $\mathcal{B}(P_0, \epsilon)$ is the ambiguity set which contains possible testing distributions and is typically modeled as a distribution ball $\mathcal{B}(P_0, \epsilon) = \{P \mid D(P, P_0) \leq \epsilon\}$ given a distribution discrepancy measure D and a budget ϵ. Based on the discrepancy measure D in the ambiguity set, we can get different DRO methods. The widely-adopted discrepancy measures include Wasserstein distance and the family of $\phi-$divergence which lead to Wasserstein DRO [5], [10], [30], [49] and $\phi-$divergence DRO [12]–[14], [16] respectively.

The choice of the ambiguity set has a large effect on the performance of DRO. As the decision objective in the PTO pipeline can be very sensitive to the distribution shifts, it becomes critical to choose a proper ambiguity set in DR-DFL. As the focus of this paper, we will discuss the challenges of ambiguity modeling and present our diffusion-based ambiguity modeling in Section III.

C. Applications in Computing and Communication Networks

The considered PTO problem has wide applications for context-aware decision-making problems in computing and communication networks.

1) Workload-Aware Resource Provisioning for LLM Inference: With the rapid deployment of AI, particularly large language models (LLMs), the substantial energy costs of AI workloads have become a critical concern. In AI data centers, inference workloads often exceed training workloads in volume [4], [43], [46]. However, reducing the energy consumption of inference is challenging due to its latency sensitivity and limited temporal flexibility. Hardware-based techniques such as Dynamic Voltage and Frequency Scaling (DVFS) and power capping have been explored to improve energy efficiency [27], but they face new challenges in the context of LLM serving.

As LLMs are increasingly adopted, serving systems must process a large volume of inference requests. To ensure stable performance, computational capacity is typically fixed over short time slots, requiring data center operators to provision resources in advance based on predicted workloads. However, inference demand can fluctuate significantly, making it difficult to forecast workloads accurately and to strike an effective balance between energy efficiency and performance guarantees based on the prediction. We develop a distributionally robust LLM workload predictor targeting on both serving performance and energy costs across diverse workload patterns with a detailed case study in Section V.

2) Demand Response in AI Data Centers: The rapid advancement of AI technologies has driven an exponential increase in the demand for high-performance AI data centers, which places a substantial burden on power grids and contributes to high energy costs [7]. Despite their high power consumption, the flexibility in AI training workloads opens up opportunities for demand response (DR), where data centers adapt their energy usage in response to grid signals such as real-time electricity prices and carbon intensity [6], [21], [48]. The power usage can be controlled by workload shifting (e.g., deferring non-urgent jobs to off-peak hours) or power capping (e.g., reducing server utilization or GPU frequency). In this way, AI data centers can significantly lower energy costs while satisfying Service Level Agreements (SLAs) for AI workloads.

However, the performance of DR strategies relies on the accurate prediction of the workloads and the energy price. This task is complicated by the high variability of AI workloads and the integration of renewable energy sources, which introduce substantial uncertainty into workload demand and energy supply, respectively. To address these challenges, ML models can be employed to forecast future workload patterns and power price. However, ML models are known to suffer from severe performance degradation due to distribution shifts on workload or energy price in real-world environments. Therefore, distributionally robust and decision-focused ML models are essential to provide accurate predictions and directly support downstream objectives under uncertainty.

3) Channel-Aware Edge Data Center Selection: Edge Data Centers (EDCs) offer heterogeneous hardware and software resources—including CPUs, GPUs, memory, and pre-deployed AI models—to support Mobile Edge Computing (MEC) services. These EDCs enable low-latency computation for latency-sensitive applications. In a typical MEC system, edge users generate diverse computational tasks that must be assigned to suitable EDCs in order to meet service quality objectives, such as minimizing end-to-end latency and maximizing inference accuracy [25], [29], [40].

A key factor that significantly impacts the latency of MEC services is the quality of the wireless channel between the user and the selected EDCs. Unlike traditional cloud computing environments, where network latency is relatively stable, edge computing environments are highly dynamic due to user mobility, wireless conditions, and network congestion. As such, channel-aware EDC selection becomes essential to ensure adequate utilization of communication resources and efficient task offloading. However, accurately predicting channel conditions is challenging since mobile channel quality can be influenced by numerous factors. To address this challenge, ML models can be leveraged for their ability to integrate various information and predict short-term channel conditions. More importantly, to fight against uncertainty in real-world edge environments, a robust and decision-focused channel quality predictor is critically needed for optimizing the user-to-EDC assignments with their service objectives.

III. Diffusion Ambiguity Modeling

A. Challenges in Ambiguity Modeling

The choice of ambiguity set in DR-DFL (3) has a large effect on the robustness performance and the solution tractability. The advantages and limitations of the commonly-used

Wasserstein and $\phi-$divergence based ambiguity sets are introduced as below.

If we choose the discrepancy measure as the Wasserstein distance $D_{\mathrm{W}}(P, P_0)$, we get the Wasserstein DRO (`W-DRO`). Wasserstein measure has no restrictions on the support of the distribution P, so the obtained ambiguity set is broad enough to include diverse testing distributions. However, it is difficult to find a tractable solution for `W-DRO`. Some methods [5], [10], [30] reformulate Wasserstein-constrained DRO into a finite optimization based on the assumption of convex objectives which typically do not hold in deep learning. Other methods convert `W-DRO` into an adversarial training with norm constraints on samples [10], [37], but these solutions can be overly conservative and cannot fully exploit the benefits of probabilistic ambiguity modeling to improve generalization.

Alternatively, the distribution discrepancy can be measured by the family of ϕ-divergence $D_\phi(P\|P_0) = \mathbb{E}_{P_0}[\phi(\frac{\mathrm{d}P}{\mathrm{d}P_0})]$ where ϕ is a convex function with $\phi(1) = 0$ [12], [13], [16], [17], [22]. If we choose $\phi(\tau) = \tau \ln(\tau)$, we get `KL-DRO` with a KL-divergence-based ambiguity set. A closed-form solution to the inner maximization (3) of `KL-DRO` is provided in [12]. Nevertheless, the definition of ϕ-divergence requires any distribution P in the ambiguity set to be absolutely continuous with respect to the training distribution P_0 (denoted as $P << P_0$), which means for any measurable set $\mathcal{A}$, $P_0(\mathcal{A}) = 0$ implies $P(\mathcal{A}) = 0$. This implicit restriction narrows the ambiguity set and consequently limits the robustness of DRO when facing test scenarios with support shift. While we can exploit data augmentation methods to improve the generalization of `KL-DRO`, the worst-case distribution searched by `KL-DRO` is still restricted to share the same support with the finite training samples, potentially resulting in a DRO solution that is not robust enough.

The intrinsic difficulty of modeling ambiguity sets in DR-DFL stems from the infinite dimension of the probability space. This motivates us to model the ambiguity set in a parameterized space. Therefore, unlike these traditional ambiguity modeling, we leverage diffusion models to directly learn the worst-case distribution in the context of DR-DFL exploiting their strong distribution modeling capability [11], [34]–[36] introduced as follows.

B. Distribution Modeling via Diffusion Models

The diffusion models learn the underlining distribution from a finite dataset and can generate more samples from the underlining distribution. They rely on forward and backward stochastic processes introduced as follows.

Forward Process. The forward process incrementally injects noise into the data, generating a sequence of perturbed samples. It begins with an initial sample $x_0 \in \mathcal{R}^d$ drawn from the underlining distribution P_0, and evolves according to a stochastic process as:

$$\mathrm{d}x = k(x, t)\mathrm{d}t + g(t)\mathrm{d}w, \tag{4}$$

where $k(\cdot, t) : \mathcal{R}^d \to \mathcal{R}^d$ is a vector-valued function, $g(t) \in \mathcal{R}$, w is a standard Wiener process and $\mathrm{d}w$ is white Gaussian noise. By the forward process, we get a collection of random variables $\{x_t\}_{t\in[0,T]}$. We use P_t to represent the distribution of x_t and $P_{t|0}$ to denote the conditional distribution of x_t given $x_0 \sim P_0$. With a sufficiently long time T, the marginal distribution $P_T(x_T)$ approximates a tractable prior distribution $\pi(x)$ which is typically chosen as a standard Gaussian distribution.

Reverse Process. A reverse diffusion process is associated with the forward equation in (4) and is expressed as

$$\mathrm{d}x = \left(k(x, t) - g(t)^2 \nabla_x \log P_t(x)\right) \mathrm{d}t + g(t)\mathrm{d}\bar{w}, \tag{5}$$

where $\bar{w}$ is a standard Wiener process in the reverse-time direction, $\nabla_x \log P_t(x)$ is the time-dependent score function.

Score Matching. In the reverse process, the score function $\nabla_x \log P_t(x)$ plays a critical role in directing the dynamics. To estimate the score function $\nabla_x \log P_t(x)$, we train a score-based model $s_\theta(x, t)$ based on samples generated from the forward diffusion process. The score-based model should minimize the following score-matching loss:

$$J_{\mathrm{SM}}(\theta) = \int_0^T \mathbb{E}_{P_t(x)} \left[\lambda(t) \left\|\nabla_x \log P_t(x) - s_\theta(x, t)\right\|^2\right] \mathrm{d}t,$$

where $\lambda(t) > 0$ is a positive weighting function. We usually approximate the score-matching loss by a tractable denoising score-matching loss up to a constant that does not rely on θ:

$$J(\theta)=\int_0^T \mathbb{E}_{P_0(x)P_{t|0}(x'|x)}\Big[\lambda(t)\Big\|\nabla_{x'} \log P_{t|0}(x'|x) - s_\theta(x, t)\Big\|^2\Big]\mathrm{d}t, \tag{6}$$

Sampling. If we discretize the reverse process, initialize $x_T \sim \pi$ and replace $\nabla_x \log P_t(x)$ with the score-based model $s_\theta(x, t)$, we can generate samples with a Markov chain with T steps:

$$x_{t-1} = x_t + [k(x_t, t) - g^2(t)s_\theta(x_t, t)]\Delta t + g(t)\sqrt{|\Delta t|}z_t, \tag{7}$$

where Δ_t is a small enough constant and $z_t \sim \mathcal{N}(0, \mathbf{I})$. Most existing diffusion models generate samples following the Markov chain [8], [11], [34], [36] and a common expression for the joint distribution of the reverse outputs is

$$P_\theta(x_{0:T}) = \pi(x_T) \prod_{t=1}^{T} P_\theta(x_{t-1} \mid x_t), \tag{8}$$

where $P_\theta(x_{t-1} \mid x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t))$.

The following lemma shows that constraint on the denoising score-matching loss (6) implies that the constraint on the KL-divergence between P_0 and P_θ is satisfied.

Lemma 1 (Theorem 1 and Corollary 1 in [35])**.** *Given the assumptions listed in Appendix A of [35]* [1]*, if the denoising*

[1] The assumptions require that the expected squared norm over P_0 and π are bounded by any finite value, the functions $k(\cdot, t)$, $\nabla_x \log P_t(x)$, and $s_\theta(\cdot, t)$ are Lipschitz continuous and upper bounded by a value related to $\|x\|$, $g(t)$ is a non-zero function, and $\int_{t=0}^T \int_{\mathcal{O}} \|P_t(x)\|_2^2 + dg(t)^2 \|\nabla_x P_t(x)\|_2^2 \mathrm{d}x\mathrm{d}t$ for any open bounded set $\mathcal{O}$ and $\mathbb{E}\left[\exp(\frac{1}{2}\int_{t=0}^T \|\nabla_x \log P_t(x) - s_\theta(x, t)\|_2^2 \mathrm{d}t)\right]$ are bounded by any finite value.

score-matching loss (6) *satisfies* $J(\theta) \leq \epsilon$, *the output distribution of the diffusion model* $P_\theta(x_0)$ *satisfies*

$$D_{\mathrm{KL}}(P_0||P_\theta) \leq \epsilon + D_{\mathrm{KL}}(P_T||\pi) + C_1,$$

where P_T *is the final-step output distribution of the forward process and* $P_T \approx \pi$ *by the design of diffusion models, and* C_1 *is a constant that does not rely on* θ.

Note that the KL-divergence $D_{\mathrm{KL}}(P_0||P_\theta)$ in Lemma 1 is not the KL-divergence $D_{\mathrm{KL}}(P_\theta||P_0)$ commonly used in `KL-DRO`. The former KL-divergence allows P_θ to have broader support space than P_0 ($P_0 << P$).

C. Diffusion-Based Ambiguity Set

Lemma 1 implies that if we find a diffusion modeled distribution P_θ whose parameters satisfy $J(\theta) \leq \epsilon$, then P_θ stays close to the training distribution P_0 through a KL-divergence $D_{\mathrm{KL}}(P_0||P_\theta)$ up to a budget related to ϵ. This KL-divergence is the reversed KL-divergence in `KL-DRO` and allows P_θ to have broader support space than P_0. Therefore, we can define a parameterized ambiguity set based on the diffusion models without the support shift issue. The DR-DFL with diffusion-based ambiguity set is expressed as below.

$$\min_{w \in \mathcal{W}} \max_{\theta \in \Theta} \mathbb{E}_{P_\theta(x)}[f(y^*(h_w(v)), c)], \quad \text{s.t. } J(\theta, S_0) \leq \epsilon, \quad (9)$$

where $P_\theta(x), x = (v, c)$ is the output distribution of the diffusion model, $J(\theta, S_0)$ is the denoising score-matching loss of a diffusion model based on a training dataset S_0.

The diffusion-based ambiguity set leverages the powerful distribution modeling capabilities of diffusion models to enhance the generalization performance of DR-DFL. Specifically, diffusion models can generate diverse samples beyond the support of the training distribution, enabling the discovery of distributions with the worst decision-making performance, thereby yielding robust solutions. By constraining on the score-matching loss, the distributions in the ambiguity set remains consistent with the training data, striking a balance between average-case and worst-case performance. Moreover, the inner maximization in (9) is conducted over a finite parameterized space rather than an infinite probability space, making the inner maximization tractable.

IV. 3D-Learning Algorithm

Despite the advantages of diffusion-based ambiguity set, it is challenging to solve DR-DFL in (9) due to the complexity of diffusion models. We propose `3D-Learning` algorithm in this section to solve this challenge.

A. Inner Maximization of `3D-Learning`

Solving the inner maximization of (9) presents two key challenges. First, the problem is a constrained non-convex optimization, making it difficult to maximize the objective and minimize the constraint violation simultaneously. Second, the objective function depends on the diffusion parameter θ through the probability function inside the expectation, which is computationally expensive to evaluate and differentiate.

Algorithm 1 Inner Maximization of `3D-Learning` (`IMAX`)

```
Input: Training dataset $S_0$; ML model $h_w$, Adversary budget $\epsilon > 0$; Step size $\eta > 0$.
Initialization. Initialize the the diffusion parameter $\theta$ and Lagrangian weight $\alpha > 0$.
for $k = 1, 2, \cdots, K$ do
    Update diffusion model $\theta_k$ by solving (10) given $\alpha$.
    Update the Lagrangian parameter $\alpha$: $\alpha \leftarrow \max\{\alpha + \eta(J(\theta_k, S_0) - \epsilon), 0\}$.
end for
return Adversarial diffusion model $\theta_K$.
```

We propose Algorithm 1 to tackle the challenges of inner maximization. Observing that the constraint in (9) is a budget constraint ($J(\theta) \geq 0$ and $\epsilon > 0$), we can solve the constrained optimization by a dual learning method [1], [23]. Algorithm 1 adaptively learns a Lagrangian dual $\alpha > 0$ to convert the inner constrained maximization into multi-step unconstrained optimizations below

$$\max_{\theta} \mathbb{E}_{P_\theta}[f(y^*(h_w(v)), c)] - \alpha J(\theta, S_0). \quad (10)$$

We update α by dual gradient descent based on the denoising score-matching loss J: Increase α to emphasize more on the constraint satisfaction if J violates the budget and decrease α otherwise. After enough iterations, the dual variable converges to a near-optimal one that balances the objective maximization and constraint violation.

Next, we transform the objective in (10) into a differentiable term. By the Markov chain in (8), we can express the joint probability of denoising outputs as

$$P_\theta(x_{0:T}) = C \cdot e^{-\frac{\|x_T\|^2}{2}} \cdot e^{-\sum_{t=1}^{T} \frac{\|x_{t-1} - \mu_\theta(x_t, t)\|^2}{2\sigma_t^2}}, \quad (11)$$

where C is a nomalizing constant, and σ_t^2 is the variance of the reserve noise at step t. Then, we can exploit tricks in policy optimization algorithms to transform the expected objective.

By the trick in vanilla policy gradient [39], we can derive the gradient of the expected objective in (10) as

$$\begin{aligned} &\nabla_\theta \mathbb{E}_{P_\theta(x)}[f(y^*(h_w(v)), c)] = \\ &\quad \mathbb{E}_{P_\theta(x_{0:T})}[\nabla_\theta \ln P_\theta(x_{0:T}) \cdot f(y^*(h_w(v)), c)], \end{aligned} \quad (12)$$

where $\ln P_\theta(x_{0:T}) = \sum_{t=1}^{T} [x_{t-1} - \mu_\theta(x_t, t)]^2 + C_2$ where C_2 is a constant. We can empirically calculate the expected gradient in (12) based on a dataset sampled from the reverse process of the diffusion model P_θ.

Proximal Policy Optimization (PPO) [33] is believed to have more stable performance than vanilla policy gradient. By PPO, we can transform the expected objective in (10) into a differentiable form as

$$\begin{aligned} \mathbb{E}_{P_\theta}[f(y^*(h_w(v)), c)] = \mathbb{E}_{P_{\theta_0}}[&\min(r_\theta f(y^*(h_w(v)), c)), \\ &\mathrm{clip}(r_\theta(x_{0:T}), 1 - \kappa, 1 + \kappa) \cdot f(y^*(h_w(v)), c))], \end{aligned} \quad (13)$$

where clip is the clipping function in PPO [33] with the clipping parameter $\kappa \in (0, 1)$, the probability ratio is

Algorithm 2 `3D-Learning` Algorithm

```
Input: Training dataset $S_0$; Adversary budget $\epsilon > 0$.
Initialization. Initialize the ML model parameter $w$.
for epoch $= 1, 2, \cdots, E$ do
    Run IMAX ($h_w$,$\epsilon$) in Algorithm 1 to update the diffusion model parameter $\theta$.
    Generate adversarial dataset $S_\theta$ based on the diffusion model $P_\theta$.
    Update the ML model parameter $w$ based on $S_\theta$.
end for
return ML model $h_w$.
```

$r_\theta(x_{0:T}) = \frac{P_\theta(x_{0:T})}{P_{\theta_0}(x_{0:T})} = \exp\{-\sum_{t=1}^{T}(\frac{\|x_{t-1}-\mu_\theta(x_t,t)\|^2}{2\sigma_t^2} - \frac{\|x_{t-1}-\mu_{\theta_0}(x_t,t)\|^2}{2\sigma_t^2})\}$, and the reference model P_{θ_0} can be a pre-trained diffusion model on the training dataset S_0. Similar as (12), we can empirically calculate expected objective based on a the dataset sampled from the reverse process of the diffusion model P_{θ_0}. To reduce the training complexity, we can fix the parameters of the first $T - T'$ steps and only optimize the last T' steps of the reverse process by choosing $r_\theta(x_{0:T'}) = \exp\{-\sum_{t=1}^{T'}(\frac{\|x_{t-1}-\mu_\theta(x_t,t)\|^2}{2\sigma_t^2} - \frac{\|x_{t-1}-\mu_{\theta_0}(x_t,t)\|^2}{2\sigma_t^2})\}$.

B. Min-Max Solution of `3D-Learning`

Now we are ready to solve the min-max problem in (9). The min-max solution is extended from the algorithm of Gradient Descent with Max-Oracle (GDMO) in [15]. For nonconvex-nonconcave min-max optimization problems, GDMO is proved to guarantee an approximate stationary solution with the approximation error depending on the error of inner-maximization.

The algorithm flow of `3D-Learning` is provided in Algorithm 2. Following the GDMO framework, `3D-Learning` first runs `IMAX` in Algorithm 1 to search for the adversarial diffusion model P_θ that maximizes the expected loss of the current ML model h_w within the diffusion ambiguity set. Next, given the updated diffusion model P_θ, we need to update the ML parameter w to minimize the expected decision objective $\mathbb{E}_{P_\theta}[f(y^*(h_w(v)), c)]$. One choice is to perform the gradient descent on the PPO transformation in (13). However, in order to provide the ML model with more diverse samples, we generate an adversarial dataset S_θ by the diffusion model P_θ and directly approximate the expected decision objective $\mathbb{E}_{P_\theta}[f(y^*(h_w(v)), c)]$ by S_θ. Next, we can perform a gradient descent on the decision-focused objective based on the adversarial dataset S_θ. The objective can be differentiated by existing differentiable optimization layers [28].

V. Case Study

In this section, we evaluate the performance of `3D-Learning` based on a simulation study on resource management for LLM inference serving. We present the problem statement and setups, followed by the empirical comparison of `3D-Learning` and baselines.

A. System Model

We give the system model for the application of Cloud Resource Provisioning for LLM Inference in Section II-C1. Our objective is to develop a robust LLM workload predictor that achieves a good trade-off between utility performance and energy costs across diverse workload patterns. At each time step $i \in [N]$, the LLM inference workload is c_i, measured as the total number of input and output tokens assuming the best achievable LLM performance. Since the exact workload c_i is unknown at the beginning of step i, the operator assigns an LLM instance with capacity a_i based on the predicted workload $\hat{c}_i$. While real-world resource provisioning involves multiple dimensions—including CPU, GPU cores, and memory—we abstract these complexities by defining a_i as the token-handling capacity of the allocated LLM instance at time i. In other words, a_i represents the number of tokens the instance can process in the slot i while maintaining optimal LLM performance (no output length limit).

To jointly capture the inference performance and the corresponding energy costs, we define an objective function that depends on the workload c_i and the allocated capacity a_i. Specifically, the utility of assigning an LLM instance with capacity a_i to process a workload of size c_i is quantified by *Utility*(a_i, c_i), which reflects the service quality achieved under the resource allocation a_i. When the allocated capacity fully satisfies the incoming workload ($a_i \geq c_i$), all requests can be processed with the highest performance, resulting in the maximum average utility $s(1)$. In contrast, when the allocated capacity is insufficient ($a_i < c_i$), the platform may either reject non-critical requests or restrict the output lengths of LLMs, leading to degraded service quality [20], [45]. In such cases, the average utility is modeled as $s\left(\frac{a_i}{c_i}\right)$, where $s(\cdot)$ is an increasing function of the ratio of the allocated capacity to the demanded capacity. The overall *Utility* model is defined as:

$$\textit{Utility}(a_i, c_i) = \begin{cases} s(1) \cdot c_i, & \text{if } a_i \geq c_i, \\ s\left(\frac{a_i}{c_i}\right) \cdot c_i, & \text{if } a_i < c_i. \end{cases} \tag{14}$$

The function *Cost*(a_i, c_i) captures the total energy cost associated with assigning an LLM instance with capacity a_i for a time slot i with an inference workload c_i. The instance processes the workload of $\min(a_i, c_i)$ using activated computing resources, incurring energy consumption of $P_{\text{act}} \cdot \min(a_i, c_i)$. If the allocated capacity exceeds the actually demanded workload ($a_i > c_i$), the overly allocated capacity $(a_i - c_i)^+$ incurs additional energy consumption $P_{\text{idle}} \cdot (a_i - c_i)^+$ at a lower power rate , due to power-saving techniques such as GPU frequency scaling. The total energy cost is scaled by Power Usage Effectiveness (PUE) ω, and is modeled as:

$$\textit{Cost}(a_i, c_i) = \omega \cdot \left(P_{\text{act}} \cdot \min(a_i, c_i) + P_{\text{idle}} \cdot (a_i - c_i)^+\right) \tag{15}$$

The overall objective of capacity provisioning is to maximize the net utility of LLM inference serving, accounting for both service performance and energy cost. The optimization problem is formulated as:

$$\max_{\forall i, a_i \in [a_{\min}, a_{\max}]} \mathcal{R}(a_{1:N}, c_{1:N}) := \sum_{i=1}^{N} \Big[\mathit{Utility}(a_i, c_i) - \gamma \cdot \mathit{Cost}(a_i, c_i) \Big], \tag{16}$$

where $[a_{\min}, a_{\max}]$ denote the range of allowable LLM processing capacities, and $\gamma > 0$ is a scaling coefficient that unifies the units of *Utility* and *Cost*.

B. Experiment Setups

1) System Setups: In the LLM inference serving system, the capacity for an LLM instance is decided at the beginning of each time slot and remains constant within the time slot to avoid unstable service quality. A sequence example includes consecutive $N = 28$ time slots. In the simulation, we set the maximum processing speed of an LLM instance with multiple GPUs as 4×10^5 tokens per time slot. In addition, we set the power consumption per token as $P_{\text{act}} = 4 \times 10^{-6}$kWh based on the estimation of GPT-3 [3] that GPT-3 consumes an order of 0.4 kWh of electricity to generate 100 pages of content. The idle power consumption is set as $P_{\text{idle}} = 1.4 \times 10^{-6}$kWh which is about one third of the activated power consumption. The PUE is set as $\omega = 1.1$.

For the utility model in (14), we adopt a logarithmic function $s(X) = B \log(AX + 1)$, which captures the diminishing returns of resource allocation. Logarithmic utility functions are widely used in network economics and resource allocation. Similar logarithmic utility functions have also been employed by Low *et al.* [24] to model the utility of network flows in TCP congestion control. Moreover, Stephen *et al.* [2] highlight that such utility functions possess desirable properties—monotonicity and strict concavity—which make them well-suited for modeling fair resource allocation problems. This form can be readily substituted with alternative utility functions that reflect revenue or service quality in practical LLM serving scenarios. We choose the parameters in the utility function as, $A = 20$, and $B = 0.2$. The cost coefficient μ is set as 0.34

2) Baselines: The baselines which are compared with our algorithms in our experiments are introduced as below.

Decision-Focused Learning ML (DFL): This method [28] trains the ML to optimize the decision objective without considering distributionally robustness.

Wasserstein-based DRO (W-DRO): This is a DRO algorithm where the ambiguity set is defined by the Wasserstein measure. In the experiments, we choose the FWDRO algorithm in [37] which applies to general objectives, and replace its loss function with our decision-focused objective.

KL-divergence-based DRO (KL-DRO): This is a DRO algorithm where the ambiguity set is defined by the KL divergence. We choose the commonly-used KL-DRO solution derived in [12] and replace its loss function with our decision-focused objective.

Data Augmentation (DA): Data augmentation techniques are commonly used to improve the generalization performance of ML by incorporating more diverse training samples [19]. In the experiments, we inject new samples to the training datasets by adding Gaussian, Perlin or Cutout noise.

3) Datasets: The experiments are conduct based on the dataset of Azure LLM Inference Traces [31], [38]. The dataset captures time series of input and output token counts for each service request in the years 2023 and 2024 from two production-grade LLM inference services deployed within Azure, targeting code-related and conversational tasks, respectively. To assess the generalization performance in different testing distributions, we split the datasets into a training dataset and several testing datasets with different distribution shifts.

All ML models are trained on the 2023-Conversations (23V (Train)) dataset with 751 sequence samples and evaluated on different testing datasets with dataset sizes ranging from 798 to 4320. The distributional discrepancy between each testing set and the training set is quantified by Wasserstein distance shown under the dataset names in Table I. The testing sets are listed as below with their time, LLM task and acronym: 2023-Conversations (23V (Test)), 2023-Code (23D), 2024-Code (24D), and 2024-Conversations (24V). To increase the diversity of the testing sets, we merge instances from two original datasets in an half-to-half way and get three additional testing sets: 2023-Code&2024-Code (23D24D), 2024-Code&2024-Conversations (24D24V) and 2023-Code&2024-Conversations (23D24V).

4) Training Setups: The experimental setup is divided into the following parts:

Predictor: The workload predictors in 3D-Learning and all the baselines share the same two-layer stacked LSTM architecture with 128 and 64 hidden neurons.

Diffusion Model: The diffusion model in 3D-Learning is DDPM [11] which has $T = 500$ steps in a forward or a backward process.

Training: For 3D-Learning, we adopt the PPO-based reformulation in (13) for inner maximization. We train the reference DDPM θ_0 in (13) based on the original training dataset 23V (Train) and use it to generate an initial dataset Z_0 to calculate r_θ in (13). The sampling variance of DDPM is chosen from a range $[0.05, 0.1]$. To improve training efficiency, only the last $T' = 10$ backward steps of the DDPM model are fine-tuned by (13). We choose a slightly higher clipping parameter $\kappa = 0.4$ in (13) to encourage the maximization while maintaining stability. We choose $\epsilon = 0.03$ as IMAX's adversarial budget which gives the best average performance over all validation datasets. We choose $\eta = 0.01$ as the rate to update the Lagrangian parameter α in Algorithm 1. We use the Adam optimizer with a learning rate of 10^{-6} for both the diffusion training in the maximization and the predictor update in minimization. The diffusion model is trained for 10 inner epochs with a batch size of 64. The predictor is trained for 15 epochs with a batch size of 64.

For the baseline methods, we choose the same neural network architecture as 3D-Learning. We carefully tuned the hyperparameters of the baseline algorithms to achieve optimal average performance over all validation datasets. For W−DRO, we consider the Wasserstein distance with respect to l_2−norm and set the adversarial budget as $\epsilon = 2$. For KL−DRO, we choose the adversarial budget $\epsilon = 2$. The predictors in both baseline DRO methods are trained by Adam optimizer with a learning rate of 2×10^{-5}. Both baselines are trained for 100 epochs with a batch size of 64.

C. *Experiment Results*

TABLE I
TEST REGRET ON DIFFERENT DATASETS.

Dataset	**Algorithms**					
	3D−Learning	**KL−DRO**	**W−DRO**	**Cutout**	**Gaussian**	**DFL**
23V(Test) (0.0001)	**0.0518**	0.0797	0.0682	0.0967	0.0949	0.1298
24V (0.0961)	**0.0283**	0.0604	0.0828	0.0707	0.0716	0.1003
23D (0.1459)	**0.2213**	0.2967	0.3087	0.3241	0.3304	0.3993
24D (0.1011)	**0.2775**	0.4558	0.6266	0.4648	0.5004	0.6103
23D24D (0.1565)	**0.3140**	0.5071	0.6894	0.5215	0.5643	0.6828
23D24V (0.1357)	**0.0703**	0.1214	0.1680	0.1317	0.1376	0.1791
24D24V (0.0687)	**0.1814**	0.3072	0.3976	0.3259	0.3471	0.4360
Average	**0.1635**	0.2612	0.3345	0.2765	0.2923	0.3625
Maximum	**0.3140**	0.5071	0.6894	0.5215	0.5643	0.6828

1) Default Setting: We give a comprehensive comparison between 3D−Learning and different decision-focused baseline algorithms on various testing datasets with different distribution shifts. We evaluate the performance by the normalized *Regret*(alg) $= (\bar{\mathcal{R}}_{\mathrm{opt}} - \bar{\mathcal{R}}_{\mathrm{alg}})/\bar{\mathcal{R}}_{\mathrm{opt}}$, where $\bar{\mathcal{R}}_{\mathrm{alg}}$ denotes the mean performance of the algorithm on a testing dataset and $\bar{\mathcal{R}}_{\mathrm{opt}}$ represents the optimal mean performance on the same testing dataset. The regrets and their average and maximum values over all testing datasets are shown in Table I, We can find that 3D−Learning outperforms all DRO algorithms across all datasets. Specifically, while KL−DRO and W−DRO also achieve notable improvements over the standard DFL method, 3D−Learning attains an average regret of 0.1635 over all datasets, exceeding the average performance of KL−DRO and W−DRO by 37.4% and 51.1%, respectively. Furthermore, on the **23D24D** dataset with the largest distribution shift, all baselines reach the maximum regret, but 3D−Learning achieves a maximum regret of 0.3140, surpassing KL−DRO and W−DRO by 38.0% and 54.4%, respectively. The advantages of 3D−Learning come from the use of diffusion model to construct the ambiguity set. Compared to the ambiguity sets with KL divergence, diffusion-based ambiguity set allows 3D−Learning to generate diverse samples beyond the support of the training distribution, discover distributions that lead to the worst-case decision-making performance. Meanwhile, by restricting the denoising score matching loss by a budget ϵ, 3D−Learning ensures that the distributions in the ambiguity set are consistent with the underlining data distribution. This avoids the overly-broad relaxation of ambiguity set in the W−DRO solution and achieve performance improvements far exceeding W−DRO. The comparison with DRO baselines demonstrate that 3D−Learning with the diffusion-based ambiguity set is superior in achieving a favorable performance balance between average and worst-case testing environments.

In addition, in Table I, we compare 3D−Learning with two common data augmentation methods which inject new samples by adding *Cutout* or *Gaussian* noise. For DA with *Cutout* noise, each training data point at each time step is masked to zero with a probability of 5%, and the cutout dataset is added to original training dataset. For DA with *Gaussian* noise, a standard Gaussian noise scaled by 5% of the maximum value of the original data point is added to each data point. at each time step. Then, the decision-focused training is applied on the augmented training datasets. As shown in Table I, both data augmentation methods provide obvious performance improvements for both average and maximum performance compared to DFL since DA makes the training data more diverse and so enhances the generalization performance. *Cutout* augmentation performs better than Gaussian augmentation, achieving an average Regret of 0.2765, which represents a 23.7% improvement over DFL. However, DA methods have a limited performance improvement especially for the maximum regret. This is because DA methods do not optimize for the worst-case performance and they cannot effectively inject samples that are important for the decision objective. By contrast, 3D−Learning effectively optimizes for the worst-case and decision aware objectives, thus significantly outperforming DA methods on both average and worst-case performance.

2) Evaluation on Corrupted Datasets: In Fig. 3, we access the performance of distributionally robust algorithms on corrupted testing environments. We create corrupted testing datasets by injecting Cutout, Perlin, or Gaussian noise to the original testing dataset (23D24D) which has the largest Wasserstein distribution discrepancy compared to the training dataset. For Cutout noise, each per-round data point is masked to zero with a probability of 0.5%. The added Perlin noise has a magnitude of 5% of the maximum value of the original data point. Standard Gaussian noise scaled by 10% of the original maximum value is added to the original dataset. The regret ratios (*Regret*(alg)/*Regret*(DFL) × 100 %) with the regret of DFL as the baseline are illustrated in Fig 2. Under various noisy perturbations, 3D−Learning achieves an average Regret Ratio of only 46.8% of DFL, while significantly outperforming KL−DRO and W−DRO by 27.7% and 53.0%, respectively. These results demonstrate that by diffusion augmented DRO, 3D−Learning exhibits outstanding robustness against noisy corruptions compared to existing methods.

3) Effects of Training Objectives: Next, we investigate the effects of training objectives (decision-focused or MSE training) on 3D−Learning. The MSE training replaces the decision objective in 3D−Learning with the mean squared prediction error. The regret ratios of both training strategies are

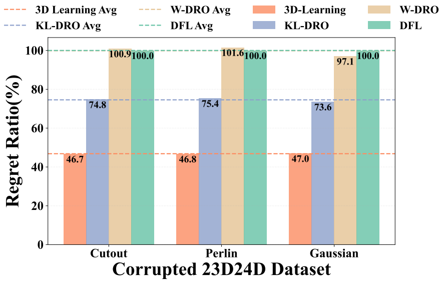

Fig. 2. Robustness evaluation under diverse noisy corruptions on **23D24D** dataset.

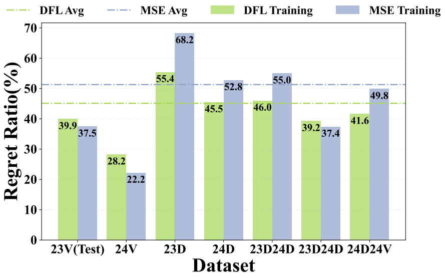

Fig. 3. `3D-Learning` with decision-focused training and MSE training.

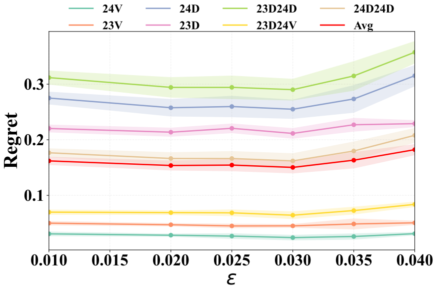

Fig. 4. Effect of budget ϵ on `3D-Learning` performance.

illustrated in Fig 3. Decision-focused training achieves, on average, a 6.1% improvement over MSE training in decision performance across various datasets. However, decision-focused training does not win for all datasets: it performs slightly worse on the datasets `23V` (Test) `24V` and `23D24D`. This is because distributionally robust learning optimizes for the worst-case distribution but not for every distinct distribution. Nevertheless, by the end-to-end training strategy, decision-focused training demonstrates superior performance in most cases compared to prediction-focused training.

4) Effects of DRO Budget: Finally, we investigate the effects of the critical budget parameter ϵ in (9) on the performance of `3D-Learning`. As shown in Fig 4, the regret-ϵ curves across all datasets exhibit a concave shape, achieving the optimal average performance at around $\epsilon = 0.03$. When ϵ falls below this threshold and continues to decrease, the diffusion-modeled distribution becomes overly constrained to the training data, limiting the ability of `3D-Learning` to generalize to OOD datasets. Conversely, when ϵ exceeds this threshold and continues to increase, `3D-Learning` with an overly large ambiguity set can conservatively over-optimize the decision objective on irrelevant distributions, which can result in degraded performance on real OOD datasets. Therefore, it is critical to select an appropriate budget ϵ to generate effective adversarial distributions, striking a desirable trade-off between average and worst-case performance.

VI. Related Works

Our work is closely related to the literature on DRO. Most existing DRO algorithms construct ambiguity sets using either Wasserstein distances [5], [10], [30], [49] or ϕ-divergences [12]–[14], [16]. However, these approaches often involve optimization over an infinite-dimensional probability space, which poses significant computational challenges and can result in sub-optimal solutions. Recently, Ren *et al.* [32] proposed DRAGEN, which performs Wasserstein-based DRO on the latent space of a generative model like Variational Auto-Encoder (VAE) so that the generated environments are more consistent with the realistic world. However, DRAGEN still relies on solving the Wasserstein-based DRO in an infinite probability space. Thus, the intractability and over-relaxation issues of Wasserstein-based DRO still exist. Moreover, applying the decoding model to latent variables may constrain the expressiveness of the ambiguity set. In contrast, `3D-Learning` constructs a novel ambiguity set based on the score-matching loss of the diffusion model and directly searches for adversarial distributions within its parameterized space, which offers greater flexibility in exploring adversarial distributions.

Our work is naturally related to DFL which mainly study end-to-end training strategies for decision task objectives [9], [28]. Robust DFL is also considered in recent literature. Ma *et al.* [26] introduce a differentiable parameterized Second-Order Cone (SOC) to define the ambiguity set and propose an end-to-end framework that trains ML to predict the ambiguity set for the downstream DRO task. In comparison, our DR-DFL framework is fundamentally different, as it addresses the distributionally robustness problem during the training of the context predictor, rather than focusing on inference-time DRO tasks as in [26]. Moreover, `3D-Learning` constructs ambiguity sets using diffusion models, which capture distributions with diverse and shifted support, thereby offering greater expressiveness. Wang *et al.* [41] proposed a Generate-then-Optimize framework that trains a diffusion model to generate data for downstream statistical optimization, targeting the conditional value-at-risk (CVaR) objective. While this approach is related to our work, it primarily addresses risk mitigation under in-distribution scenarios, whereas `3D-Learning` is designed to enhance robustness in out-of-distribution (OOD) environments.

VII. Conclusion

Our work focuses on DR-DFL which models many critical PTO applications in networking. We propose a diffusion augmented algorithm `3D-Learning` to improve the OOD generalization of DR-DFL. Specifically, by leveraging diffusion-based ambiguity modeling, `3D-Learning` enables the search of worst-case distributions in the parameterized space of diffusion models, which achieves a good balance between average and worst-case performances. Extensive simulation results on resource provisioning for LLM inference confirm the effectiveness of `3D-Learning`, demonstrating substantial performance gains and enhanced robustness under perturbations. These findings highlight the potential of diffusion models as a powerful tool for distributionally robust, decision-driven learning in dynamic and uncertain environments.

References

[1] Santiago Balseiro, Haihao Lu, and Vahab Mirrokni. Dual mirror descent for online allocation problems. In Hal Daumé III and Aarti Singh, editors, *ICML*, volume 119 of *Proceedings of Machine Learning Research*, pages 613–628. PMLR, 13–18 Jul 2020.

[2] Stephen P Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[3] Tom Brown, Benjamin Mann, et al. Language models are few-shot learners. 33:1877–1901, 2020.

[4] René Caspart, Sebastian Ziegler, Arvid Weyrauch, Holger Obermaier, Simon Raffeiner, Leon Pascal Schuhmacher, Jan Scholtyssek, Darya Trofimova, Marco Nolden, Ines Reinartz, Fabian Isensee, Götz, and Charlotte Debus. Precise energy consumption measurements of heterogeneous artificial intelligence workloads. In *International Conference on High Performance Computing*, pages 108–121. Springer, 2022.

[5] Ruidi Chen and Ioannis Ch. Paschalidis. Distributionally robust learning. *Foundations and Trends® in Optimization*, 4(1-2):1–243, 2020.

[6] Philip Colangelo, Ayse K Coskun, Jack Megrue, et al. Turning ai data centers into grid-interactive assets: Results from a field demonstration in phoenix, arizona. *arXiv preprint arXiv:2507.00909*, 2025.

[7] Carly Davenport, CFA Singer, N Mehta, B Lee, and J Mackay. Ai data centers and the coming us power demand surge. *PDF). Goldman Sachs. Archived from the original (PDF) on*, 26, 2024.

[8] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. 34:8780–8794, 2021.

[9] Priya Donti, Brandon Amos, and J. Zico Kolter. Task-based end-to-end model learning in stochastic optimization. 30, 2017.

[10] Rui Gao and Anton Kleywegt. Distributionally robust stochastic optimization with wasserstein distance. *Mathematics of Operations Research*, 48(2):603–655, 2023.

[11] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *NIPS*, volume 33, pages 6840–6851. Curran Associates, Inc., 2020.

[12] Zhaolin Hu and L Jeff Hong. Kullback-leibler divergence constrained distributionally robust optimization. *Available at Optimization Online*, 1(2):9, 2013.

[13] Hisham Husain, Vu Nguyen, and Anton van den Hengel. Distributionally robust bayesian optimization with ϕ-divergences. 2023.

[14] Ruiwei Jiang and Yongpei Guan. Data-driven chance constrained stochastic program. *Mathematical Programming*, 158(1):291–327, 2016.

[15] Chi Jin, Praneeth Netrapalli, and Michael Jordan. What is local optimality in nonconvex-nonconcave minimax optimization? In Hal Daumé III and Aarti Singh, editors, *ICML*, volume 119 of *Proceedings of Machine Learning Research*, pages 4880–4889. PMLR, 13–18 Jul 2020.

[16] Johannes Kirschner, Ilija Bogunovic, Stefanie Jegelka, and Andreas Krause. Distributionally robust bayesian optimization. In Silvia Chiappa and Roberto Calandra, editors, *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 2174–2184. PMLR, 26–28 Aug 2020.

[17] Burak Kocuk. Conic reformulations for kullback-leibler divergence constrained distributionally robust optimization and applications. *IJOCTA*, 11(2):139–151, April 2021.

[18] Daniel Kuhn, Peyman Mohajerin Esfahani, Shafieezadeh Nguyen, et al. Wasserstein distributionally robust optimization: Theory and applications in machine learning. pages 130–166. 2019.

[19] Misha Laskin, Kimin Lee, Adam Stooke, Lerrel Pinto, Pieter Abbeel, and Aravind Srinivas. Reinforcement learning with augmented data. 33:19884–19895, 2020.

[20] Baolin Li, Yankai Jiang, Vijay Gadepally, and Devesh Tiwari. Llm inference serving: Survey of recent advances and opportunities. In *2024 IEEE HPEC*, pages 1–8, 2024.

[21] Zhenhua Liu, Iris Liu, Steven Low, and Adam Wierman. Pricing data center demand response. *ACM SIGMETRICS Performance Evaluation Review*, 42(1):111–123, 2014.

[22] Zijian Liu, Qinxun Bai, Jose Blanchet, Perry Dong, Wei Xu, Zhengqing Zhou, and Zhengyuan Zhou. Distributionally robust q-learning. In *ICML*, pages 13623–13643. PMLR, 2022.

[23] Alfonso Lobos, Paul Grigas, and Zheng Wen. Joint online learning and decision-making via dual mirror descent. In Marina Meila and Tong Zhang, editors, *ICML*, volume 139 of *Proceedings of Machine Learning Research*, pages 7080–7089. PMLR, 18–24 Jul 2021.

[24] S.H. Low. A duality model of tcp and queue management algorithms, 2003.

[25] Quyuan Luo, Shihong Hu, Changle Li, Guanghui Li, and Weisong Shi. Resource scheduling in edge computing: A survey. *IEEE Communications Surveys and Tutorials*, 23(4):2131–2165, 2021.

[26] Xutao Ma, Chao Ning, and Wenli Du. Differentiable distributionally robust optimization layers. 2024.

[27] Paul Joe Maliakel, Shashikant Ilager, and Ivona Brandic. Investigating energy efficiency and performance trade-offs in llm inference across tasks and dvfs settings. *arXiv preprint arXiv:2501.08219*, 2025.

[28] 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*, 80:1623–1701, August 2024.

[29] Erfan Meskar and Ben Liang. Fair multi-resource allocation with external resource for mobile edge computing. In *INFOCOM WKSHPS*, pages 184–189, 2018.

[30] Peyman Mohajerin Esfahani and Daniel Kuhn. Data-driven distributionally robust optimization using the wasserstein metric: Performance guarantees and tractable reformulations. *Mathematical Programming*, 171(1):115–166, 2018.

[31] Pratyush Patel, Esha Choukse, Chaojie Zhang, Aashaka Shah, Íñigo Goiri, Saeed Maleki, and Ricardo Bianchini. Splitwise: Efficient generative llm inference using phase splitting. In *2024 ACM/IEEE 51st Annual ISCA*, pages 118–132, 2024.

[32] Allen Z. Ren and Anirudha Majumdar. Distributionally robust policy learning via adversarial environment generation. *IEEE Robotics and Automation Letters*, 7(2):1379–1386, 2022.

[33] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. 2017.

[34] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. 2022.

[35] Yang Song, Conor Durkan, Iain Murray, and Stefano Ermon. Maximum likelihood training of score-based diffusion models. 34:1415–1428, 2021.

[36] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. 32, 2019.

[37] Matthew Staib and Stefanie Jegelka. Distributionally robust deep learning as a generalization of adversarial training. In *NIPS workshop on Machine Learning and Computer Security*, volume 3, page 4, 2017.

[38] Jovan Stojkovic, Chaojie Zhang, Íñigo Goiri, Josep Torrellas, and Esha Choukse. Dynamollm: Designing llm inference clusters for performance and energy efficiency, 2024.

[39] Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. 12, 1999.

[40] Shreshth Tuli, Fatemeh Mirhakimi, Samodha Pallewatta, Giuliano Zawad, et al. Ai augmented edge and fog computing: Trends and challenges. *Journal of Network and Computer Applications*, 216:103648, 2023.

[41] Prince Zizhuang Wang, Jinhao Liang, Shuyi Chen, Ferdinando Fioretto, and Shixiang Zhu. Gen-dfl: Decision-focused generative learning for robust decision making. 2025.

[42] Xinyu Wang, Yiyang Peng, and Wei Ma. An end-to-end smart predict-then-optimize framework for vehicle relocation problems in large-scale vehicle crowd sensing. *arXiv preprint arXiv:2411.18432*, 2024.

[43] Grant Wilkins, Srinivasan Keshav, and Richard Mortier. Hybrid heterogeneous clusters can lower the energy consumption of llm inference workloads. In *Proceedings of the 15th ACM International Conference on Future and Sustainable Energy Systems*, pages 506–513, 2024.

[44] Jiajun Wu, Chengjian Sun, and Chenyang Yang. Proactive optimization with machine learning: Femto-caching with future content popularity, 2020.

[45] Yuqing Yang, Lei Jiao, and Yuedong Xu. A queueing theoretic perspective on low-latency llm inference with variable token length. In *2024 22nd International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*, pages 273–280, 2024.

[46] Zhisheng Ye, Wei Gao, Qinghao Hu, Peng Sun, Xiaolin Wang, Yingwei Luo, Tianwei Zhang, and Yonggang Wen. Deep learning workload

scheduling in gpu datacenters: A survey. *ACM Comput. Surv.*, 56(6), January 2024.

[47] Jinlei Zhang, Ergang Shan, Lixia Wu, Jiateng Yin, Lixing Yang, and Ziyou Gao. An end-to-end predict-then-optimize clustering method for stochastic assignment problems. *IEEE Transactions on Intelligent Transportation Systems*, 25(9):12605–12620, 2024.

[48] Yijia Zhang, Daniel Curtis Wilson, Ioannis Ch. Paschalidis, and Ayse K. Coskun. Hpc data center participation in demand response: An adaptive policy with qos assurance. *IEEE Transactions on Sustainable Computing*, 7(1):157–171, 2022.

[49] Chaoyue Zhao and Yongpei Guan. Data-driven risk-averse stochastic optimization with wasserstein metric. *Operations Research Letters*, 46(2):262–267, 2018.