GNN-Parametrized Diffusion Policies for Wireless Resource Allocation

Yiğit Berkay Uslu **Samar Hadou** **Shirin Saeedi Bidokhti** **Alejandro Ribeiro**

Department of Electrical & Systems Engineering
University of Pennsylvania
Philadelphia, PA 19104
{ybuslu,selaraby,saeedi,aribeiro}@seas.upenn.edu

Abstract

This paper proposes a supervised training algorithm for learning stochastic resource allocation policies with generative diffusion models (GDMs). We formulate the allocation problem as the maximization of an ergodic utility function subject to ergodic Quality of Service (QoS) constraints. Given samples from a stochastic expert policy that yields a near-optimal solution to the constrained optimization problem, we train a GDM policy to imitate the expert and generate new samples from the optimal distribution. We achieve near-optimal performance through the sequential execution of the generated samples. To generalize across a family of network configurations, we parameterize the backward diffusion process with a graph neural network (GNN) architecture. We present numerical results in a case study of optimal power control.

1 Introduction

Most existing formulations and methods for optimal wireless resource allocation, whether classical or learning-based, seek deterministic solutions. In contrast, *optimal solutions* of many non-convex optimization problems (e.g., power control, scheduling) are *inherently probabilistic*, as the optimal solution may lie in the convex hull of multiple deterministic policies. By randomizing between multiple deterministic strategies, stochastic policies can achieve better performance by effectively convexifying the problem [1]. This phenomenon is also fundamental in multi-user information theory, where time sharing plays a critical role in achieving optimal performance across various communication channels [2–4]. In this work, we leverage diffusion models to learn generative solutions to stochastic network resource allocation problems.

Generative models (GMs) have shown significant success in generating samples from complex, multi-modal data distributions. Among the wide class of generative models, including variational autoencoders (VAEs) and generative adversarial networks (GANs), *generative diffusion models* (GDMs) stand out for their capability of generating high-quality and diverse samples with stable training [5, 6]. GDMs convert target data samples (e.g., images) to samples from an easy-to-sample prior (e.g., isotropic Gaussian noise) by a forward (noising) process, and then learn a backward (denoising) process to transform the prior distribution back to the target data distribution.

A substantial body of the existing literature utilizes GDMs, and GMs in general, for generating domain-specific synthetic data and for data augmentation to enhance machine-learning models in supervised and reinforcement learning tasks [7, 8]. Yet, research on the use of GMs for wireless network optimization, and GDMs in particular, is scant [9–13]. Concurrent works [14–19] propose generative model solvers for network optimization as a framework to learn solution distributions

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: New Perspectives in Advancing Graph Machine Learning.

that concentrate the probability mass around optimal deterministic solutions. The generative process then converts random noise into high-quality solutions by eliminating the noise introduced in the forward process. However, the problem formulation in the aforementioned studies is deterministic and ignores the probabilistic nature of the optimal solution.

Our work is one of the first to imitate *stochastic* expert policies using GDMs. We emphasize the stochastic nature of certain network optimization problems, where random solutions are not only essential for optimality but also are realized by leveraging iterative dual domain algorithms and time sharing. In our approach, the Quality of Service (QoS) near-optimality emerges through the sequential execution of solutions sampled from the optimal generated distribution. Moreover, we use a graph neural network (GNN) architecture as the backbone for the reverse diffusion process to enable learning families of solutions across network topologies. GNNs excel in learning policies from graph-structured data [20–23] and offer stability and scalability [24, 25].

This paper tackles imitation learning of stochastic wireless resource allocation policies. A GDM policy is trained to match an optimal solution distribution to a constrained optimization problem from which an expert policy can sample (see Section 2 and Section 3). We utilize a GNN-parametrization to condition the generative diffusion process directly on the network graphs (see Section 4). We evaluate the proposed GDM policy in a power control setup and demonstrate that the trained GDM policy closely matches the expert policy over a family of wireless networks (see Section 5).

2 Optimal Wireless Resource Allocation

We represent the channel state of a wireless network system with a matrix $\mathbf{H} \in \mathcal{H} \subseteq \mathbb{R}^{N \times N}$ and the allocation of corresponding resources with a vector $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^N$. Given $\mathbf{H}$, the choice of resource allocation $\mathbf{x}$ determines several QoS metrics that we represent with an objective utility $f_0 : \mathcal{X} \times \mathcal{H} \mapsto \mathbb{R}$ and a constraint utility $\mathbf{f} : \mathcal{X} \times \mathcal{H} \mapsto \mathbb{R}^c$. We define an optimal resource allocation $\mathbf{x}^*(\mathbf{H})$ as the argument that solves the constrained optimization problem,

$$\tilde{\mathrm{P}}(\mathbf{H}) = f_0\big(\mathbf{x}^*(\mathbf{H}), \mathbf{H}\big) = \underset{\mathbf{x} \in \mathcal{X}}{\text{maximum}} \;\; f_0\big(\mathbf{x}(\mathbf{H}), \mathbf{H}\big), \quad \text{subject to} \;\; \mathbf{f}\big(\mathbf{x}(\mathbf{H}), \mathbf{H}\big) \geq \mathbf{0}. \tag{1}$$

In (1), we seek a resource allocation $\mathbf{x}^*(\mathbf{H})$ with the largest f_0 utility among those in which the components of the utility $\mathbf{f}$ are nonnegative. This abstract formulation encompasses channel and power allocation [26] in wireless networks (Section 5) as well as analogous problems, in, e.g., point-to-point [27], MIMO [28], broadcast [29], and interference channels [30]. In most cases of interest, the utilities f_0 and $\mathbf{f}$ in (1) are not convex. We introduce the convex relaxation in which optimization is over probability *distributions* of resource allocation variables, and QoS is measured in *expectation*,

$$\mathrm{P}(\mathbf{H}) = \underset{\mathcal{D}_{\mathbf{x}}}{\text{maximum}} \;\; \mathbb{E}_{\mathcal{D}_{\mathbf{x}}} \Big[f_0\big(\mathbf{x}(\mathbf{H}), \mathbf{H}\big) \Big], \quad \text{subject to} \;\; \mathbb{E}_{\mathcal{D}_{\mathbf{x}}} \Big[\mathbf{f}\big(\mathbf{x}(\mathbf{H}), \mathbf{H}\big) \Big] \geq \mathbf{0}. \tag{2}$$

In (2), we search over stochastic policies $\mathcal{D}_{\mathbf{x}}$ that maximize the *expected* utility $\mathbb{E}_{\mathcal{D}_{\mathbf{x}}}[\, f_0(\mathbf{x}(\mathbf{H}), \mathbf{H})]$ while satisfying the *expected* constraint $\mathbb{E}_{\mathcal{D}_{\mathbf{x}}}[\mathbf{f}\big(\mathbf{x}(\mathbf{H}), \mathbf{H}\big)] \geq \mathbf{0}$ when the resource allocation $\mathbf{x}(\mathbf{H})$ is drawn from the distribution $\mathcal{D}_{\mathbf{x}}$. For future reference, we introduce $\mathcal{D}^*_{\mathbf{x}}(\mathbf{H}) = \mathcal{D}^*_{\mathbf{x}}(\mathbf{x} \,\big|\, \mathbf{H})$ to denote a distribution that solves (2). In this distribution, the channel state $\mathbf{H}$ is given, and resource allocations $\mathbf{x}$ are sampled. The important point here is that the performance of stochastic policies is realizable through time sharing if we allocate resources at a faster time scale than QoS perception. Indeed, if we consider independent resource allocation policies $\mathbf{x}_\tau(\mathbf{H}) \sim \mathcal{D}_{\mathbf{x}}$, we have that for sufficiently large T,

$$\frac{1}{T} \sum_{\tau=1}^{T} f_0\big(\mathbf{x}_\tau(\mathbf{H}), \mathbf{H}\big) \approx \mathbb{E}_{\mathcal{D}_{\mathbf{x}}} \Big[f_0\big(\mathbf{x}(\mathbf{H}), \mathbf{H}\big) \Big], \tag{3}$$

with an analogous statement holding for the constraint utility $\mathbf{f}$. Since deterministic policies are particular cases of stochastic policies, we know that $\mathrm{P}(\mathbf{H}) \geq \tilde{\mathrm{P}}(\mathbf{H})$. In practice, it is often the case that $\mathrm{P}(\mathbf{H}) \gg \tilde{\mathrm{P}}(\mathbf{H})$, and for this reason, the stochastic formulation in (2) is most often preferred over the deterministic formulation in (1), [1–3].

Imitation Learning of Stochastic Policies. In this paper, we set out to learn to imitate the stochastic policies that solve (2). More to the point, consider a distribution $\mathcal{D}_{\mathbf{H}}$ of channel states $\mathbf{H}$. For each channel state realization $\mathbf{H}$, recall that the solution of (2) is the probability distribution $\mathcal{D}^*_{\mathbf{x}}(\mathbf{H}) = \mathcal{D}^*_{\mathbf{x}}(\mathbf{x} \,\big|\, \mathbf{H})$. Separate from these optimal distributions, we consider a parametric family of conditional

distributions $\mathcal{D}_{\mathbf{x}}(\mathbf{H};\boldsymbol{\theta}) = \mathcal{D}_{\mathbf{x}}(\mathbf{x} \,\big|\, \mathbf{H};\boldsymbol{\theta})$ in which the channel state $\mathbf{H}$ is given, and resource allocation variables are drawn. Our goal is to find the conditional distribution $\mathcal{D}^*_{\mathbf{x}}(\mathbf{H};\boldsymbol{\theta})$ that minimizes the expectation of the conditional KL-divergences $\mathrm{D}_{\mathrm{KL}}\big(\mathcal{D}^*_{\mathbf{x}}(\mathbf{H}) \,\big\|\, \mathcal{D}_{\mathbf{x}}(\mathbf{H};\boldsymbol{\theta})\big)$,

$$\mathcal{D}^*_{\mathbf{x}}(\mathbf{H};\boldsymbol{\theta}) \;=\; \underset{\mathcal{D}_{\mathbf{x}}(\mathbf{H};\boldsymbol{\theta})}{\operatorname{argmin}} \; \mathbb{E}_{\mathcal{D}_{\mathbf{H}}}\Big[\, \mathrm{D}_{\mathrm{KL}}\big(\mathcal{D}^*_{\mathbf{x}}(\mathbf{H}) \,\big\|\, \mathcal{D}_{\mathbf{x}}(\mathbf{H};\boldsymbol{\theta})\big)\,\Big], \tag{4}$$

In (4), the distributions $\mathcal{D}^*_{\mathbf{x}}(\mathbf{H})$ are given for all $\mathbf{H}$. The conditional distribution $\mathcal{D}_{\mathbf{x}}(\mathbf{H};\boldsymbol{\theta})$ is our optimization variable, which we compare with $\mathcal{D}^*_{\mathbf{x}}(\mathbf{H})$ through their KL divergence. KL divergences of different channel realizations are averaged over the channel state distribution $\mathcal{D}_{\mathbf{H}}$, which is also given. The optimal distribution $\mathcal{D}^*_{\mathbf{x}}(\mathbf{H};\boldsymbol{\theta}) = \mathcal{D}^*_{\mathbf{x}}(\mathbf{x} \,\big|\, \mathbf{H};\boldsymbol{\theta})$ minimizes the expected KL divergence among distributions that are representable by the parametric family $\mathcal{D}_{\mathbf{x}}(\mathbf{H};\boldsymbol{\theta})$.

To solve (4), we need access to the expert conditional distributions $\mathcal{D}^*_{\mathbf{x}}(\mathbf{H})$. This is impossible in general because algorithms that solve (2) do not solve for $\mathcal{D}^*_{\mathbf{x}}(\mathbf{H})$ directly, but rather generate samples $\mathbf{x}(\mathbf{H})$ drawn from the optimal distribution $\mathcal{D}^*_{\mathbf{x}}(\mathbf{H})$ [22]. Thus, we recast the goal of this paper as learning to generate samples $\mathbf{x} \,\big|\, \mathbf{H}$ from the distribution $\mathcal{D}^*_{\mathbf{x}}(\mathbf{H};\boldsymbol{\theta})$ when we are given samples $\mathbf{x}(\mathbf{H})$ of the expert conditional distributions $\mathcal{D}^*_{\mathbf{x}}(\mathbf{H})$ with channel states generated according to $\mathcal{D}_{\mathbf{H}}$:

> **Problem 1** *Given samples* $\mathbf{x}(\mathbf{H})$ *drawn from the expert distribution* $\mathcal{D}^*_{\mathbf{x}}(\mathbf{H})\mathcal{D}_{\mathbf{H}} = \mathcal{D}^*_{\mathbf{x}}(\mathbf{x} \,\big|\, \mathbf{H})\mathcal{D}_{\mathbf{H}}$ *[cf.* (2)*], we learn to generate samples* $\mathbf{x} \,\big|\, \mathbf{H}$ *drawn from the conditional distributions* $\mathcal{D}^*_{\mathbf{x}}(\mathbf{H};\boldsymbol{\theta}) = \mathcal{D}^*_{\mathbf{x}}(\mathbf{x} \,\big|\, \mathbf{H};\boldsymbol{\theta})$ *[cf.* (4)*].*

A solution to Problem 1 is illustrated in Fig. 2. For a given channel state realization $\mathbf{H}$, we show two-dimensional slices of *samples* of an optimal policy (in blue). As indicated by (1), these samples achieve optimal QoS metrics for (2) if executed sequentially (Fig. 1). We train a generative diffusion model (Section 3) that generates samples (in orange) that are distributed close to samples of an optimal distribution. When executed sequentially, the learned samples realize QoS metrics close to optimal values (Fig. 1). Neither the optimal distribution $\mathcal{D}^*_{\mathbf{x}}(\mathbf{H})\mathcal{D}_{\mathbf{H}} = \mathcal{D}^*_{\mathbf{x}}(\mathbf{x} \,\big|\, \mathbf{H})\mathcal{D}_{\mathbf{H}}$ nor the parametric distribution $\mathcal{D}^*_{\mathbf{x}}(\mathbf{H};\boldsymbol{\theta}) = \mathcal{D}^*_{\mathbf{x}}(\mathbf{x} \,\big|\, \mathbf{H};\boldsymbol{\theta})$ is computed.

Learning in the Dual Domain & Policy Randomization. Most learning approaches to allocating resources in wireless systems contend with the deterministic policy formulation in (1), e.g., [11, 14–17, 21, 26, 31]. This is due in part to the use of deterministic learning parameterizations [21, 26, 31], but even recent contributions that propose diffusion models, mostly do so for deterministic policies [11, 14–18]. This is a well-known limitation that has motivated, e.g., state-augmented algorithms that leverage dual gradient descent dynamics to randomize policy samples [22, 23]. These algorithms generate trajectories of primal and dual iterates by operating on a convex hull relaxation of the Lagrangian for the original problem and iteratively solving a sequence of Lagrangian maximization subproblems. Each subproblem is an unconstrained, deterministic problem to which regular learning methods apply, and near-optimality and feasibility guarantees are established neither for individual primal iterates nor their averages, but only for the sequential execution of the generated policy iterates.

A shortcoming of state-augmented algorithms—and iterative dual domain algorithms in general, regardless of whether parametrized or not—is that they incur a transient period during which suboptimal policies are executed. Reducing the length of this transient period typically requires larger step sizes, which in turn, introduces a trade-off with respect to solution optimality. Learning a generative model to sample from the stationary (optimal) policy distribution emerges as a promising approach for overcoming this trade-off. To the best of our knowledge, our paper is the first to demonstrate the imitation of stochastic policies that solve a constrained optimization problem with GDMs.

3 Policy Generative Models

GDMs involve a forward and a backward diffusion process. The *forward* process defines a Markov chain of diffusion steps to progressively add random noise to data. For a given $\mathbf{H}$ and a data sample $\mathbf{x}_0 = \mathbf{x}(\mathbf{H})$ drawn from the expert distribution $\mathcal{D}^*_{\mathbf{x}}(\mathbf{H})$, the forward chain follows

$$q(\mathbf{x}_k \,|\, \mathbf{x}_0; \mathbf{H}) = \mathcal{N}(\mathbf{x}_k; \sqrt{\bar{\alpha}_k}\mathbf{x}_0, (1-\bar{\alpha}_k)\mathbf{I}), \tag{5}$$

where $\bar{\alpha}_k := \prod_{i=1}^{k} \alpha_i$, $\alpha_k := 1 - \beta_k$, and β_k is a monotonically increasing noise schedule, e.g., a linear or cosine schedule. For a sufficiently large number of diffusion time steps K, (5) converts the data sample $\mathbf{x}_0$ into a sample that is approximately isotropic Gaussian distributed, i.e., $\mathbf{x}_K \approx \mathcal{N}(\mathbf{0}, \mathbf{I})$. The reverse process of (5) can be approximated as a chain of Gaussian transitions with some fixed variance $\sigma_k^2 \mathbf{I}$ and a parametrized mean $\boldsymbol{\mu}_{\boldsymbol{\theta}}$,

$$p_{\boldsymbol{\theta}}(\mathbf{x}_{k-1} \,|\, \mathbf{x}_k; \mathbf{H}) = \mathcal{N}\big(\mathbf{x}_{k-1}; \boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{x}_k, k; \mathbf{H}), \sigma_k^2 \mathbf{I}\big). \tag{6}$$

A *backward* diffusion process samples $\mathbf{x}_K \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and iteratively runs the backward chain in (6) for $k = K, \ldots, 1$. With reparametrizing (5) as $\mathbf{x}_k(\mathbf{x}_0, \boldsymbol{\epsilon}) = \sqrt{\bar{\alpha}_k}\mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_k}\boldsymbol{\epsilon}$ [5], sampling $\mathbf{x}_{k-1} \sim p_{\boldsymbol{\theta}}(. \,|\, \mathbf{x}_k; \mathbf{H})$ amounts to updating

$$\mathbf{x}_{k-1} = \frac{1}{\sqrt{\alpha_k}} \left(\mathbf{x}_k - \frac{\beta_k}{\sqrt{1 - \bar{\alpha}_k}} \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{x}_k, k; \mathbf{H}) \right) + \sigma_k \mathbf{w}, \tag{7}$$

where $\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, and $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{x}_k, k; \mathbf{H})$ predicts the noise $\boldsymbol{\epsilon}$ added to $\mathbf{x}_0 \sim \mathcal{D}_{\mathbf{x}}^*(\mathbf{H})$ from the noisy sample $\mathbf{x}_k$ at time step k. An *optimal GDM-policy parametrization* $\boldsymbol{\theta}^*$ minimizes the $\mathbf{H}$-expectation of the DDPM loss function [5] given by

$$\mathcal{L}_{\text{GDM}}(\boldsymbol{\theta}) = \mathbb{E}_{\mathbf{x}_0, k, \mathbf{H}, \boldsymbol{\epsilon}} \; \omega(k) \left\| \boldsymbol{\epsilon}_{\boldsymbol{\theta}}\big(\mathbf{x}_k(\mathbf{x}_0, \boldsymbol{\epsilon}), k; \mathbf{H}\big) - \boldsymbol{\epsilon} \right\|^2. \tag{8}$$

In (8), $\omega(k)$ is a time-dependent weighting function that is usually omitted for simplicity, and the expectation is over random time steps $k \sim \text{Uniform}([1, K])$, Gaussian noise $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, expert (data) samples $\mathbf{x}_0 \sim \mathcal{D}_{\mathbf{x}}^*(\mathbf{H})$, and conditioning networks $\mathbf{H} \sim \mathcal{D}_{\mathbf{H}}$. The DDPM loss in (8) is a variational upper bound on the expected KL divergence loss in (4) and becomes tight for $\boldsymbol{\theta} = \boldsymbol{\theta}^*$. Thus, running (7) with optimal parametrization $\boldsymbol{\epsilon}_{\boldsymbol{\theta}^*}$ for a given $\mathbf{H}$ generates samples from the expert conditional distribution, i.e., $\mathbf{x}_0 \sim \mathcal{D}_{\mathbf{x}}(\mathbf{H}; \boldsymbol{\theta}^*) = \mathcal{D}_{\mathbf{x}}^*(\mathbf{H}; \boldsymbol{\theta}) \approx \mathcal{D}_{\mathbf{x}}^*(\mathbf{H})$.

4 GNN-Parametrizations for GDM Policies

We employ GNNs for GDM parameterization, as they are well-suited for processing network data, such as resource allocations. Moreover, GNNs inherently take graphs as input, making them a natural fit for GDMs conditioned on $\mathbf{H}$. GNNs process graph data through a cascade of L graph convolutional layers [32]. Inputs are node signals (features) and graph shift operators (GSO) while outputs are node embeddings. Each layer $\boldsymbol{\Psi}^{(\ell)}$ is a nonlinear aggregation function obtained by the composition of a graph convolutional filter and a pointwise nonlinearity φ (e.g., relu),

$$\mathbf{Z}^{(\ell)} = \boldsymbol{\Psi}^{(\ell)} \left(\mathbf{Z}^{(\ell-1)}; \mathbf{H}, \Theta^{(\ell)} \right) = \varphi \left[\sum_{m=0}^{M} \mathbf{H}^m \mathbf{Z}^{(\ell-1)} \boldsymbol{\Theta}_m^{(\ell)} \right]. \tag{9}$$

In (9), $\Theta^{(\ell)} = \{\boldsymbol{\Theta}_m^{(\ell)} \in \mathbb{R}^{F_{\ell-1} \times F_\ell}\}_{m=0}^{M}$ is a set of learnable weights, M denotes the number of hops and $\mathbf{Z}^{(\ell-1)} \in \mathbb{R}^{N \times F_{\ell-1}}$ is the input node signal to layer ℓ. The GSO, $\mathbf{H}$, encodes the underlying connectivity of the network, which is the network state in our case. For improved and more stable training, we take advantage of normalization layers and residual connections. To this end, we redefine φ in (9) as the composition of a normalization layer followed by a pointwise nonlinearity while the first term in the sum, $\mathbf{Z}\boldsymbol{\Theta}_0$, inherently represents a learnable residual connection. We view $\mathbf{x}_k$ and $\mathbf{k} = k\mathbf{1}_N$ as node signals and introduce a read-in layer $\boldsymbol{\Phi}^{(0)} = (\boldsymbol{\Phi}_{\mathbf{x}}, \boldsymbol{\Phi}_{\mathbf{k}})$ that adds sinusoidal-time embeddings to the input node features. That is, we have

$$\mathbf{Z}^{(0)} = \boldsymbol{\Phi}^{(0)}(\mathbf{x}_k, \mathbf{k}) = \boldsymbol{\Phi}_{\mathbf{x}}(\mathbf{x}_k) + \boldsymbol{\Phi}_{\mathbf{k}}(\mathbf{k}), \tag{10}$$

where $\boldsymbol{\Phi}_{\mathbf{x}} : \mathbb{R}^N \mapsto \mathbb{R}^{N \times F_0}$ is a multi-layer-perceptron (MLP) layer, and $\boldsymbol{\Phi}_{\mathbf{k}} : \mathbb{R}^N \mapsto \mathbb{R}^{N \times F_0}$ is a cascade of a sinusoidal time embedding and MLP layers. Finally, we add a readout MLP layer $\boldsymbol{\Phi}^{(L)} : \mathbb{R}^{N \times F_L} \mapsto \mathbb{R}^N$ that learns to predict the noise $\boldsymbol{\epsilon}$ from the output node embeddings.

5 Case Study: Power Control in Multi-User Interference Networks

Wireless Network & Power Control Setup. We consider the problem of power control in N-user interference channels. To summarize the setup briefly, all network realizations are sampled from a

family of network configurations with $N = 100$ transmitter-receiver (tx-rx) pairs (also the nodes in our graphs) and an average density of 12 tx-rx pairs/km^2. For each network, we first drop the transmitters randomly in a square grid world, and each transmitter is paired with a neighboring receiver. Receivers treat signals coming from all but their respective transmitters as interference. We note that similar setups have been investigated in [22, 23] and can be referred to for more details.

We optimize the transmit power levels $\mathbf{x} \in [0, P_{\max}]^N$ where $P_{\max} = 10$ mW is the maximum transmit power budget. The channel bandwidth and noise power spectral density (PSD) are set to $W = 20$ MHz and $N_0 = -174$ dBm/Hz, respectively. The network state $\mathbf{H}$ is the matrix of long-term channel gains which follow a log-normal shadowing with a standard deviation of 7 plus the standard dual-slope path-loss model. For a given $\mathbf{H}$, the short-term (instantaneous) channel gains $\widetilde{\mathbf{H}}$ vary following Rayleigh fading, and we define the instantaneous rate of receiver i given an allocation $\mathbf{x}$ as

$$\widetilde{r}_i(\mathbf{x}, \widetilde{\mathbf{H}}) = \log_2 \left(1 + \frac{x_i \cdot |\widetilde{h}_{ii}|^2}{W N_0 + \sum_{j \neq i} x_j \cdot |\widetilde{h}_{ji}|^2} \right), \tag{11}$$

where x_i is the ith entry of $\mathbf{x}$ and $\widetilde{h}_{ji}$ is the (j, i)th element in matrix $\widetilde{\mathbf{H}}$. Note that a policy $\mathcal{D}_{\mathbf{x}}(\mathbf{H})$ is determined only by the long-term gains, whereas the instantaneous rate depends on the short-term channel gains. Ergodic rates are given by the joint expectation over the policy and the fading,

$$\mathbf{r}(\mathcal{D}_{\mathbf{x}}(\mathbf{H}), \mathbf{H}) := \mathbb{E}_{\mathcal{D}_{\mathbf{x}}(\mathbf{H}),\, \widetilde{\mathbf{H}}|\mathbf{H}} \left[\widetilde{\mathbf{r}}\big(\mathbf{x}, \widetilde{\mathbf{H}}\big) \right]. \tag{12}$$

In our experiments, we draw 200 samples from the trained GDM policy for each network and evaluate the joint expectation over 200 time steps, with each time step spanning 10 ms. We impose a minimum ergodic rate requirement of $f_{\min} = 0.6$ bps/Hz for all receivers by setting the utility constraints $\mathbf{f}(\mathcal{D}_{\mathbf{x}}(\mathbf{H}), \mathbf{H}) := \mathbf{r}(\mathcal{D}_{\mathbf{x}}(\mathbf{H}), \mathbf{H}) - \mathbf{1}_N f_{\min}$, whereas the utility objective is the network-wide average of the ergodic rates given by $f_0(\mathcal{D}_{\mathbf{x}}(\mathbf{H}), \mathbf{H}) := \mathbf{1}_N^\top \mathbf{r}(\mathcal{D}_{\mathbf{x}}(\mathbf{H}), \mathbf{H})/N$.

Expert Policy & Baselines. To generate samples from an optimal solution distribution, we first train a GNN-parametrized model via a state-augmented primal-dual (SA) learning algorithm as in [23]. The trained model is executed online for each given network $\mathbf{H}$ over a sufficiently long time window to generate trajectories of primal and dual iterates with near-optimality and feasibility guarantees. We collect the resulting primal iterates $\{\mathbf{x}_b^\dagger(\mathbf{H})\}_{b=0}^{B-1}$, i.e., resource allocation vectors, in a buffer with capacity $B = 500$. During GDM policy training, expert policy samples are uniformly drawn from the buffer, i.e., we have $\mathcal{D}_{\mathbf{x}}^*(\mathbf{H}) = \mathrm{Uniform}\big[\big\{\mathbf{x}_0^\dagger(\mathbf{H}), \ldots, \mathbf{x}_{B-1}^\dagger(\mathbf{H})\big\}\big]$ for all $\mathbf{H} \sim \mathcal{D}_{\mathbf{H}}$. While we parametrize the expert policy and run a state-augmented training algorithm over a training dataset of networks $\mathcal{D}_{\mathbf{H}}$, one can alternatively run the dual descent without the parametrization and store the resource allocation iterates for each instance $\mathbf{H} \sim \mathcal{D}_{\mathbf{H}}$.

We compare the GDM and expert policies with two baseline deterministic policies: *(i) Average-power transmission policy*: For each given $\mathbf{H}$, we compute the time average of the expert policy samples – similar to primal averaging in the optimization literature – and fix it, i.e., we set $\mathbf{x}(\mathbf{H}) \approx \mathbb{E}_{\mathcal{D}_{\mathbf{x}}^*(\mathbf{H})}[\mathbf{x}^\dagger(\mathbf{H})]$ at all time steps. *(ii) Full-power transmission policy (FP)*: All transmitters use all the transmission power available at all time steps, i.e., $\mathbf{x}(\mathbf{H}) = P_{\max}\mathbf{1}_N$.

Performance of GNN-Parametrized GDM Policies. To evaluate our method, we draw a total of 128 network realizations. Following a $5:1:2$ split, we obtain training, validation, and test datasets of size $|\mathcal{D}_{\mathbf{H}}| = 80$, $|\mathcal{V}_{\mathbf{H}}| = 16$, and $|\mathcal{T}_{\mathbf{H}}| = 32$ networks, respectively. We train the GDM policy over the training dataset $\mathcal{D}_{\mathbf{H}}$ and test it on $\mathcal{T}_{\mathbf{H}}$. The expert policy solutions are obtained across all datasets for evaluation and benchmarking purposes. Due to limited space, we defer the implementation and training details of the GDM policy to Appendix A.

Fig. 1 showcases the test performance of the GDM policy, expert policy, and the baselines over a time horizon of 200 time steps. We estimate the ergodic rate vector for a given network $\mathbf{H}$ and time step τ as $(1/\tau)\sum_{s=0}^{\tau-1} \widetilde{\mathbf{r}}\big(\mathbf{x}_s, \widetilde{\mathbf{H}}\big)$ where $\mathbf{x}_s$ denotes the power allocation decision at an earlier time step $s < \tau$ [c.f. (12)]. In the first two plots, we report the time evolution of the average ergodic rate, i.e., the mean-rate objective utility, and the 5th-percentile of ergodic rates, both of which are computed across all $|\mathcal{T}_h| \times N = 32 \times 100 = 3200$ tx-rx pairs in the test dataset. The rightmost violin plot shows the histogram of ergodic rates of all receivers evaluated at two different time steps. We verify that the expert policy eventually satisfies almost all the constraints unlike the baselines. Although

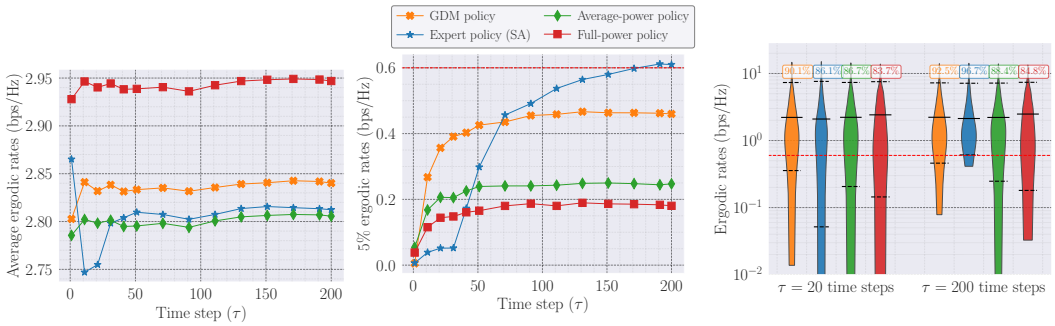

Figure 1: Comparison of the test performance of GDM policy with the expert policy (SA) and other baselines. Leftmost and middle plots show the time evolution of the average and 5th percentile of ergodic rates, respectively. The minimum rate requirement $f_{\min}$ is shown with a dashed, red line. Rightmost plot shows the distribution of ergodic rates and constraint satisfaction percentages, evaluated up to $\tau = 20$ and $\tau = 200$ time steps. We drew a solid black line at the median values while the dashed black lines indicate the 5th and 95th percentile values.

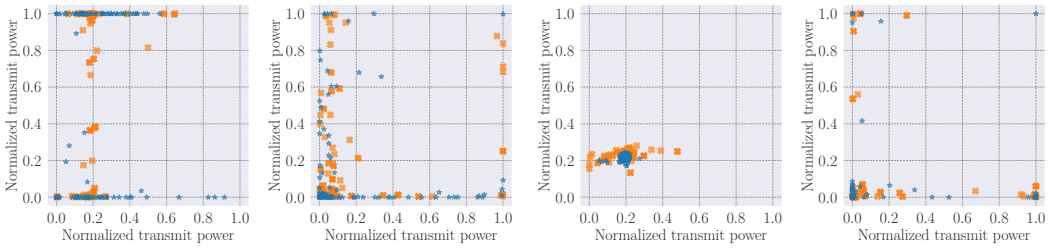

Figure 2: Example 2D slices of expert policy (blue) and GDM policy (orange) samples are shown for two pairs of neighboring nodes from the test dataset. Transmit powers are normalized by $P_{\max}$.

the GDM policy does not exhibit strict feasibility, it converges to a near-feasible and near-optimal policy in very few time steps compared to the expert policy and does not incur a long transient period. Moreover, both the full-power and average-power baselines are outperformed by the GDM policy in terms of percentile rates and constraint violations.

In Fig. 2, example two-dimensional (2D) slices of 100-dimensional learned GDM policies are overlaid with expert policy samples. The GDM policy generalizes to unseen test networks drawn from $\mathcal{T}_{\mathbf{H}}$, and the conditional distribution of GDM policy samples significantly resembles that of the expert policy. A peculiarity of optimal power control policies is that they tend to be probabilistic and involve multiple transmission modes. That is especially true for tx-rx pairs with less favorable channel conditions for which the policy randomization becomes more nuanced. In such cases, similar to time-sharing strategies, pairs that would otherwise generate considerable mutual interference adopt a policy-switching mechanism where they take turns to transmit at high power during periods of minimal interference (e.g., the top-left and bottom-right corners in the rightmost plot of Fig. 2). By alternating their transmissions, all tx-rx pairs satisfy their minimum ergodic rate requirements.

Strong test generalization evidenced in both figures notwithstanding, the GDM policy exhibits a small feasibility gap compared to the expert policy in Fig. 1. We attribute this gap primarily to the supervised training algorithm not accounting directly for the sensitivity of the constraints and the aforementioned policy-switching phenomenon. The feasibility gap and overall performance of the GDM policies can be further improved by incorporating the QoS requirements and additional variance constraints directly into the training loss and/or generative process.

6 Conclusion

This work demonstrated that generative diffusion processes can imitate expert policies that sample from optimal solution distributions of stochastic network optimization problems. We employed a GNN to condition the generative process on a family of wireless network graphs. More broadly, we anticipate that our attempt at generative diffusion-based sampling of random graph signals will be of

interest beyond resource optimization in wireless networks. We leave constraint-aware, unsupervised training of GDM policies across a wider range of network topologies as future research directions.

References

[1] M. Neely, *Stochastic network optimization with application to communication and queueing systems*. Morgan & Claypool, 2010.

[2] R. Gallager, "A perspective on multiaccess channels," *IEEE Transactions on Information Theory*, vol. 31, no. 2, pp. 124–142, 1985.

[3] I. Sason, "On achievable rate regions for the Gaussian interference channel," *IEEE Transactions on Information Theory*, vol. 50, no. 6, pp. 1345–1356, 2004.

[4] Q. He, D. Yuan, and A. Ephremides, "Optimal scheduling for emptying a wireless network: Solution characterization, applications, including deadline constraints," *IEEE Transactions on Information Theory*, vol. 66, no. 3, pp. 1882–1892, 2020.

[5] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," *Advances in Neural Information Processing Systems*, vol. 33, pp. 6840–6851, 2020.

[6] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *Proceedings of CVPR*, pp. 10684–10695, June 2022.

[7] A. Kasgari, W. Saad, M. Mozaffari, and H. V. Poor, "Experienced deep reinforcement learning with generative adversarial networks (GANs) for model-free ultra reliable low latency communication," *IEEE Transactions on Communications*, vol. 69, no. 2, pp. 884–899, 2020.

[8] W. Njima, A. Bazzi, and M. Chafii, "DNN-based indoor localization under limited dataset using GANs and semi-supervised learning," *IEEE Access*, vol. 10, pp. 69896–69909, 2022.

[9] E. M. Diallo, "Generative model for joint resource management in multi-cell multi-carrier NOMA networks," 5 2024.

[10] Y. Hua, R. Li, Z. Zhao, X. Chen, and H. Zhang, "GAN-powered deep distributional reinforcement learning for resource management in network slicing," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 2, pp. 334–349, 2019.

[11] H. Du, R. Zhang, Y. Liu, J. Wang, Y. Lin, Z. Li, D. Niyato, J. Kang, Z. Xiong, S. Cui, B. Ai, H. Zhou, and D. I. Kim, "Enhancing Deep Reinforcement Learning: A Tutorial on Generative Diffusion Models in Network Optimization," *IEEE Communications Surveys and Tutorials*, 2024.

[12] S. Nouri, M. K. Motalleb, and V. Shah-Mansouri, "Diffusion-RL for scalable resource allocation for 6G networks," 2025.

[13] F. You, H. Du, X. Hou, Y. Ren, and K. Huang, "DRESS: Diffusion reasoning-based reward shaping scheme for intelligent networks," 2025.

[14] R. Liang, B. Yang, Z. Yu, B. Guo, X. Cao, M. Debbah, H. V. Poor, and C. Yuen, "DiffSG: A Generative Solver for Network Optimization with Diffusion Model," 8 2024.

[15] R. Liang, B. Yang, P. Chen, X. Li, Y. Xue, Z. Yu, X. Cao, Y. Zhang, M. Debbah, H. V. Poor, and C. Yuen, "Diffusion models as network optimizers: Explorations and analysis," *IEEE Internet of Things Journal*, pp. 1–1, 2025.

[16] A. B. Darabi and S. Coleri, "Diffusion model based resource allocation strategy in ultra-reliable wireless networked control systems," *IEEE Communications Letters*, 2024.

[17] X. Lu, Z. Feng, J. Sun, J. Lou, C. Wu, W. Bao, and J. Li, "Generative diffusion model-based energy management in networked energy systems," in *ICASSP*, pp. 1–5, 2025.

[18] Y. Xue, R. Liang, B. Yang, X. Cao, Z. Yu, M. Debbah, and C. Yuen, "Joint task offloading and resource allocation in low-altitude MEC via graph attention diffusion," 2025.

[19] X. Wang, L. Feng, J. Wang, H. Du, C. Zhao, W. Li, Z. Xiong, D. Niyato, and P. Zhang, "Graph diffusion-based AeBS deployment and resource allocation for RSMA-enabled URLLC low-altitude economy networks," 2025.

[20] Y. Shen, Y. Shi, J. Zhang, and K. B. Letaief, "Graph neural networks for scalable radio resource management: Architecture design and theoretical analysis," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 1, pp. 101–115, 2020.

[21] Z. Wang, M. Eisen, and A. Ribeiro, "Learning decentralized wireless resource allocations with graph neural networks," *IEEE Transactions on Signal Processing*, vol. 70, pp. 1850–1863, 2022.

[22] N. NaderiAlizadeh, M. Eisen, and A. Ribeiro, "State-augmented learnable algorithms for resource management in wireless networks," *IEEE Transactions on Signal Processing*, 2022.

[23] Y. B. Uslu, N. NaderiAlizadeh, M. Eisen, and A. Ribeiro, "Fast state-augmented learning for wireless resource allocation with dual variable regression," 2025.

[24] L. Ruiz, F. Gama, and A. Ribeiro, "Graph neural networks: Architectures, stability, and transferability," *Proceedings of the IEEE*, vol. 109, no. 5, pp. 660–682, 2021.

[25] L. Testa, C. Battiloro, S. Sardellitti, and S. Barbarossa, "Stability of graph convolutional neural networks through the lens of small perturbation analysis," in *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6865–6869, IEEE, 2024.

[26] F. Liang, C. Shen, W. Yu, and F. Wu, "Towards optimal power control via ensembling deep neural networks," *IEEE Transactions on Communications*, vol. 68, no. 3, pp. 1760–1776, 2019.

[27] W. Yu, "Sum-capacity computation for the Gaussian vector broadcast channel via dual decomposition," *IEEE Transactions on Information Theory*, vol. 52, no. 2, pp. 754–759, 2006.

[28] X. Lin, R. W. Heath, and J. G. Andrews, "The interplay between massive MIMO and underlaid D2D networking," *IEEE Transactions on Wireless Communications*, vol. 14, no. 6, pp. 3337–3351, 2015.

[29] D. Tse, "Optimal power allocation over parallel Gaussian broadcast channels," in *Proc. of IEEE International Symposium on Information Theory*, p. 27, 1997.

[30] K. Chaitanya A, U. Mukherji, and V. Sharma, "Power allocation for interference channels," in *2013 National Conference on Communications (NCC)*, pp. 1–5, 2013.

[31] H. Sun, X. Chen, Q. Shi, M. Hong, X. Fu, and N. D. Sidiropoulos, "Learning to optimize: Training deep neural networks for wireless resource management," in *IEEE 18th SPAWC*, pp. 1–6, 2017.

[32] F. Gama, A. G. Marques, G. Leus, and A. Ribeiro, "Convolutional neural network architectures for signals supported on graphs," *IEEE Transactions on Signal Processing*, vol. 67, no. 4, pp. 1034–1049, 2018.

[33] A. Q. Nichol and P. Dhariwal, "Improved denoising diffusion probabilistic models," in *Proceedings of the 38th International Conference on Machine Learning* (M. Meila and T. Zhang, eds.), vol. 139 of *Proceedings of Machine Learning Research*, pp. 8162–8171, PMLR, 18–24 Jul 2021.

[34] D. P. Kingma and J. Ba, "ADAM: A method for stochastic optimization," 2014.

A Implementation & Training Details for GDM Policy

For the GNN-parametrizations, the GSO representation of the network state $\mathbf{H}$ is a fully connected graph where nodes correspond to transmitter-receiver pairs, and the edge weights between nodes i and j, e_{ij}, are set to log-normalized long-term channel gains given by $e_{ij} \propto \log_2\left(1 + \frac{P_{\max}|h_{ij}|^2}{WN_0}\right)$. The GNN has $L = 6$ layers, each with $F_\ell = 128$ hidden features and filter size of $M = 2$. We set the number of diffusion time steps to $K = 500$, use a cosine noise schedule β_k [33] and train the GDM policy to minimize the DDPM objective in (8) with a log-SNR weighting function $\omega(k) = \log\big(\mathrm{SNR}(k)\big)$, where $\mathrm{SNR}(k) := \alpha_k^2/\sigma_k^2$.

We train the GDM policy over the training dataset $\mathcal{D}_{\mathbf{H}}$ for a maximum of 10^4 epochs with an ADAM optimizer [34], an initial learning rate of 10^{-2}, and a learning rate schedule that follows a cosine decay with warm restarts. In each epoch, we iterate over the whole dataset with mini-batches of 16 graphs and sample 250 graph signals, i.e., expert policy data $\mathbf{x}(\mathbf{H}) \sim \mathcal{D}^*_{\mathbf{x}}(\mathbf{H})$, for each graph in the batch. Every 200 epochs, we evaluate the checkpointed GDM model on the validation dataset $\mathcal{V}_{\mathbf{H}}$ and save the best model in terms of 5th percentile ergodic rates. We ran our experiments on a single NVIDIA RTX 3090 GPU with 24 GB of memory.

We apply an affine transform $[0, P_{\max}] \mapsto [-1/2, 1/2]$ to map the policy space to a centered diffusion space. To sample from the diffusion model, we run the DDPM sampling equation (7) with standardized variables, invert the affine transform, and project the generated policy samples to the support $[0, P_{\max}]^N$. We observed negligible difference in the quality of generations when we swapped the DDPM sampler with other samplers, e.g., a DDIM sampler or its accelerated counterpart with fewer denoising time steps.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes] , [No] , or [NA] .
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for NA).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will be asked to also include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While "[Yes] " is generally preferable to "[No] ", it is perfectly acceptable to answer "[No] " provided a proper justification is given (e.g., "error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering "[No] " or "[NA] " is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: [NA]

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: [NA]

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

   Justification: [NA]

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: [NA]

   Guidelines:

   - The answer NA means that the paper does not include experiments.

- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: [NA]

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).

- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: [NA]

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: [NA]

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: [NA]

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.

- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: [NA]

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: [NA]

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: [NA]

    Guidelines:

    - The answer NA means that the paper poses no such risks.

- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [NA]

    Justification: [NA]

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: [NA]

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: [NA]

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: [NA]

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: [NA]

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.