Decomposing Temporal and Job-Induced Dynamics for Probabilistic Computing Workload Forecasting via Graph-Conditioned Dual-Branch Diffusion

Anonymous Author(s)
Affiliation
Address
email

Abstract

Electricity-computing synergy has emerged as a critical paradigm, which not only leverages green energy to satisfy the escalating energy demands of data centers but also exploits the operational flexibility of data centers to mitigate the uncertainty and volatility of renewable energy. Realizing this synergy requires efficient resource management strategies like job scheduling and auto-scaling, all of which depend on accurate probabilistic workload forecasting. However, workload forecasting confronts high-dimensional uncertainty driven by both temporal dynamics, arising from periodic user behavior, and job-induced dynamics, caused by distributed tasks running across multiple machines.To address this, we propose a Graph-Conditioned Dual-Branch Diffusion Model (GD-Diff). We identify implicit inter-machine correlations by analyzing job co-occurrence patterns and introduce a dual-branch structure embedded within the noise prediction network to decompose the modeling process, where one branch focuses on modeling temporal dynamics while the other captures sharp, job-induced dynamics. Experiments on public datasets demonstrate that our model achieves state-of-the-art performance in both deterministic and probabilistic forecasting, notably improving CRPS metrics by 5.43%.

1 Introduction

Workload forecasting is a fundamental task to realizing electricity-computing synergy, which is becoming increasingly important for modern data centers [6]. It supports proactive scheduling and resource allocation on the computing side [21, 28], while facilitating the alignment of energy demands with renewable generation and storage operations on the electricity side [7]. Through such coordination, data centers can align large-scale AI-driven electricity demand with the temporal variability and geographical distribution of wind and solar resources, realizing electricity-computing synergy [10].

However, workload forecasting remains challenging because workload traces in modern data centers exhibit substantial volatility [31, 36], as shown in Figure 1(a). Prior trace analyses show that production workloads are shaped by diverse operational factors, such as heterogeneous resource demands and dynamic scheduling decisions. These factors are entangled in observed traces, where irregular fluctuations from heterogeneous sources obscure predictable patterns and cannot be adequately captured by temporal correlations alone, ultimately degrading forecasting accuracy.

Existing methods typically formulate workload forecasting as a trace-based time-series prediction problem, extrapolating future resource utilization from historical observations using statistical models [3], recurrent neural networks [8, 2, 37, 35], and Transformer-based architectures [22]. These methods are deterministic and focus on point prediction, without explicitly characterizing the uncer-

Submitted to 40th Conference on Neural Information Processing Systems (NeurIPS 2026). Do not distribute.

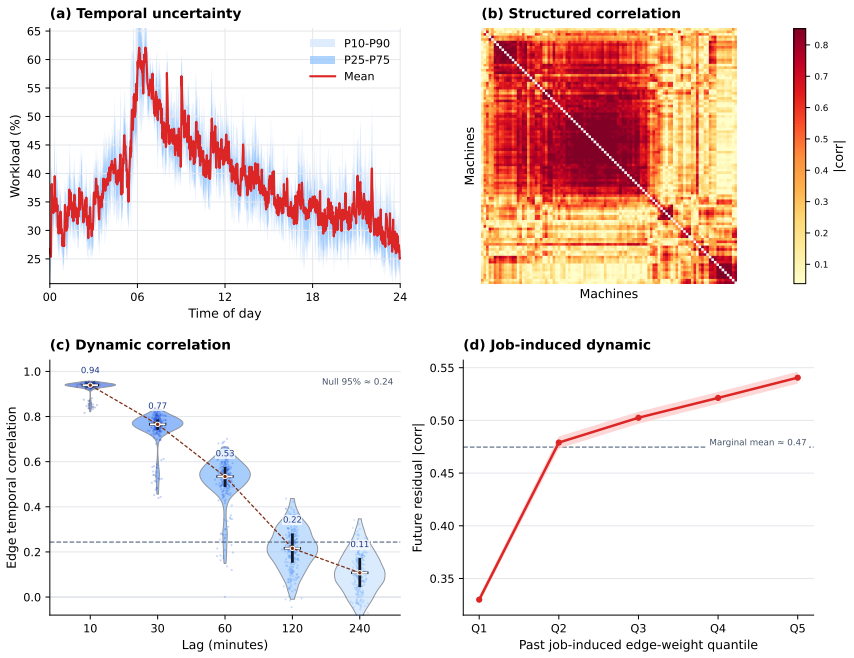


Figure 1: **Workload volatility is structured, dynamic, and aligned with job scheduling.** (a) Workload fluctuates substantially across days around the daily mean. (b) Cross-machine residuals form block-structured correlations rather than independent noise. (c) Inter-machine correlations decay over time yet stay above a random baseline, indicating dynamic yet edge-specific dependencies. (d) Stronger past job co-occurrence predicts stronger future cross-machine volatility correlation.

tainty caused by workload volatility. Point forecasts alone are insufficient for electricity-computing synergy, where scheduling and energy procurement decisions require quantified uncertainty to hedge against workload fluctuations. Recent studies have introduced probabilistic forecasting models to characterize predictive uncertainty [25, 12]. However existing approaches primarily rely on temporal correlations within individual sequences, lacking an explicit mechanism to model the sources of volatility. The structural origin of volatility remains unexploited.

To better understand this challenge, we analyze workload traces together with job scheduling logs, and observe that workload volatility exhibits structured and time-varying cross-machine patterns whose strength aligns with historical job co-occurrence (Figure 1). We refer to this structured component of volatility as *job-induced dynamics*: variations rooted in the scheduling layer that carry exploitable structure beyond regular temporal evolution. A detailed breakdown is deferred to Section 2.

Motivated by these observations, we propose **GD-Diff**, a job-aware graph-conditioned diffusion framework for probabilistic workload forecasting. (1) Inspired by the observed association between job co-occurrence and workload volatility, we first construct a job-induced weighted dynamic graph from scheduling logs. The graph transforms job co-occurrence into time-varying inter-machine dependencies, providing explicit structural signals of job-induced volatility. (2) Building on this graph, we design a dual-branch noise prediction network within a diffusion framework to decompose job-induced dynamics from regular temporal evolution. The temporal branch captures regular workload dynamics from historical traces, while the graph branch models job-induced dynamics. Their outputs are fused to guide diffusion denoising, enabling GD-Diff to derive reliable prediction intervals for risk-aware electricity-computing synergy.

Our main contributions are summarized as follows:

- We reveal that workload volatility exhibits exploitable cross-machine structure driven by job co-occurrence, and formalize this finding as a job-induced weighted dynamic graph that transforms scheduling logs into time-varying inter-machine dependencies.
- We propose GD-Diff, a graph-conditioned diffusion framework that decomposes job-induced dynamics from regular temporal evolution through a dual-branch noise prediction network.

- Extensive experiments on Alibaba Cluster and Google Cluster show that GD-Diff achieves state-of-the-art performance in both point and probabilistic forecasting. Ablations further validate the job-induced dynamic graph and dual-branch architecture.

2 Preliminary

2.1 Job–Workload Mechanism in High-Performance Clusters

In production clusters, machine-level workload is not an isolated time-series signal, but the continuous resource-usage outcome of a hierarchical job execution process. Users submit jobs to the cluster, and each job is decomposed into tasks or instances by the cluster management system. The scheduler then places these tasks or instances onto physical machines according to resource requirements, machine availability, and placement constraints. During execution, each task or instance consumes resources over time, and the observed workload of a machine is the aggregate resource usage of all running tasks or instances assigned to that machine. This mechanism establishes a natural connection between job scheduling and workload variation: when tasks or instances from the same distributed job are placed on multiple machines, their execution introduces temporally aligned workload variations across these machines, which we refer to as *job-induced dynamics*.

2.2 Empirical Analysis of Workload Volatility

We empirically validate the above mechanism on real cluster traces. For cross-machine analyses, the regular daily pattern is removed from each machine i via a *leave-one-day* template, where each day's residual is computed against the per-minute mean profile of the remaining days, yielding residual signals $r_i(t)$. Full preprocessing and analysis hyperparameters are provided in Appendix A.

Figure 1(a) shows that workload exhibits wide cross-day percentile bands around the daily mean profile, revealing substantial volatility on top of regular daily patterns. To examine whether such volatility is purely independent machine-level noise, the Pearson correlation matrix of $r_i(t)$ across machines is computed within sliding windows. As shown in Figure 1(b), a representative high-contrast window exhibits pronounced block structure, indicating that volatility carries structured cross-machine dependencies.

A natural follow-up question is whether these dependencies are static or evolve over time. Modeling each machine pair as an edge with time-varying weight $W_{ij}(t) = |\mathrm{corr}_{ij}(t)|$, the lagged correlation $\mathrm{corr}(W_{ij}(t), W_{ij}(t+\tau))$ decays with τ yet stays above a null baseline that randomizes edge identities across time while preserving each snapshot's weight distribution (Figure 1(c)), indicating that inter-machine dependencies are time-varying yet edge-specific rather than randomly reconfigured. To trace these dynamics back to job scheduling, we construct a job-induced edge weight from scheduling logs based on machine-pair co-occurrence in distributed jobs (formalized in Section 3), and group machine pairs into quintiles by this weight. The average future volatility correlation increases monotonically across quintiles (Figure 1(d)), showing that stronger historical job co-occurrence systematically leads to stronger future cross-machine synchronization.

Together, these observations show that workload volatility is substantial, structured across machines, dynamic, and aligned with job scheduling. Yet scheduling logs alone cannot fully explain workload variation, as unobserved runtime factors such as co-location interference and resource contention also shape resource usage. Workload variation is therefore decomposed into a *job-induced* component observable from scheduling logs and a *residual* component aggregating unobserved effects, motivating GD-Diff to represent job-induced dynamics through an explicit dynamic graph while accounting for volatility through a probabilistic formulation.

3 Methodology

Problem formulation. We address probabilistic multivariate time-series forecasting of machine-level workloads in production clusters. The data is denoted as $X = [X^{co}, X^{ta}] \in \mathbb{R}^{N\times(L+K)}$, where N is the number of machines, $X^{co} \in \mathbb{R}^{N\times L}$ the observed history over the past L steps, and $X^{ta} \in \mathbb{R}^{N\times K}$ the prediction target over the next K steps. The job scheduling records available at forecasting time t are denoted by $\mathcal{J}_{\leq t}$. They contain only the observed history of job executions up

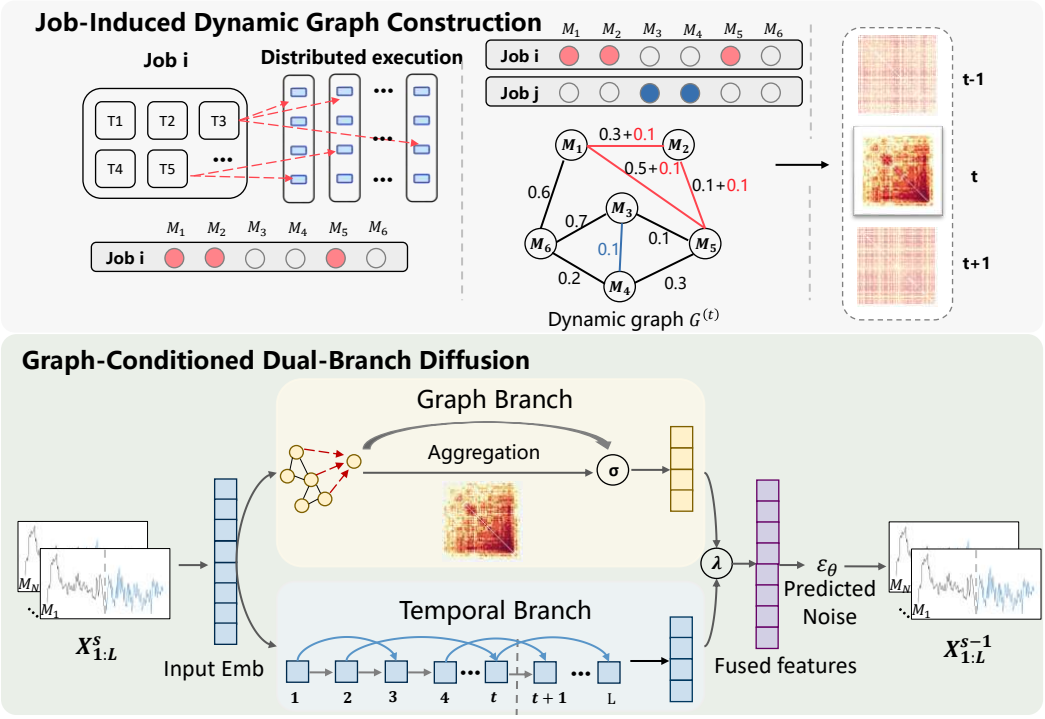


Figure 2: Overall architecture of the proposed Dual-Branch Diffusion Model. The framework decouples workload dynamics into temporal and job-induced components through a parallel noise prediction network.

to t, including job start records and the machines on which their tasks have been observed. Future job arrivals, future task placements, and completion information after t are not used. Our goal is to characterize the distribution of X^{ta} conditioned on the past workload and the job schedule

$$p_\theta\left(X^{ta} \,\middle|\, X^{co}, \mathcal{J}_{\le t}\right). \tag{1}$$

GD-Diff overview. GD-Diff models this distribution as a conditional denoising diffusion probabilistic model, in which a noise predictor ϵ_θ iteratively denoises Gaussian samples into target workloads while conditioning jointly on past observations X^{co} and a job-induced dynamic graph $\mathcal{G}^{co}$. As illustrated in Figure 2, GD-Diff comprises three components: (1) a job-induced dynamic graph $\mathcal{G}^{co}$ constructed from $\mathcal{J}_{\le t}$ that exposes inter-machine dependencies driven by distributed jobs (Section 3.1); (2) a dual-branch design of ϵ_θ that decouples regular temporal evolution from graph-driven structural fluctuations (Section 3.2); and (3) the conditional diffusion framework that turns ϵ_θ into a probabilistic forecaster (Section 3.3).

3.1 Job-Induced Dynamic Graph Construction

To explicitly encode job-induced inter-machine dependencies, we construct a job-induced weighted dynamic graph from scheduling logs based on machine-level job co-occurrence.

From a job to machine-pair edges. At each historical time step $\tau \le t$, let $\mathcal{V}_m^{(\tau)} \subseteq \{1, \ldots, N\}$ denote the set of machines on which job J_m has tasks observed running at τ, as recorded in $\mathcal{J}_{\le t}$. An active job spanning multiple machines induces co-occurrence edges among those machines. For any pair (i, j) with $i, j \in \mathcal{V}_m^{(\tau)}$, machines i and j are considered to jointly execute job J_m at time τ. Thus, a job observed on q machines contributes to the $\binom{q}{2}$ pairwise co-occurrence edges among them.

Job co-occurrence intensity. Aggregating the edge contributions of all jobs at time τ, we define the *job co-occurrence intensity* between machines i and j as

$$C_{i,j}^{(\tau)} = \sum_{m} \mathbb{I}\Big(i \in \mathcal{V}_m^{(\tau)} \wedge j \in \mathcal{V}_m^{(\tau)}\Big), \quad i \neq j, \tag{2}$$

and set $C_{i,i}^{(\tau)} = 0$. Therefore, $C_{i,j}^{(\tau)}$ counts how many jobs are jointly observed on machines i and j at time τ. A larger $C_{i,j}^{(\tau)}$ indicates stronger job-induced inter-machine dependencies, and is empirically associated with stronger future volatility correlations, as shown in Figure 1(d). The matrix $C^{(\tau)} \in \mathbb{R}^{N\times N}$ therefore forms a weighted job co-occurrence graph at time τ.

Dynamic graph sequence. Repeating the above construction at each historical time step yields a sequence of raw job co-occurrence graphs, $\mathcal{C}_{t-L+1:t}^{co} = \{C^{(\tau)}\}_{\tau=t-L+1}^{t}$, which captures how job-induced inter-machine dependencies evolve with scheduling over time. Only job-machine assignments observed no later than the forecasting time t are used; future job arrivals, task-machine assignments, and completion events are excluded.

Graph normalization and sparsification. To stabilize the scale of edge weights, we normalize the co-occurrence intensities using min–max statistics computed only from the training set: $\widetilde{C}_{i,j}^{(\tau)} = (C_{i,j}^{(\tau)} - C_{\min}^{\text{train}})/(C_{\max}^{\text{train}} - C_{\min}^{\text{train}} + \delta)$, where δ is a small constant for numerical stability. We then retain, for each machine, only the k strongest normalized co-occurrence dependencies:

$$G_{i,j}^{(\tau)} = \widetilde{C}_{i,j}^{(\tau)} \cdot \mathbb{I}\Big(j \in \mathrm{TopK}\Big(\widetilde{C}_{i,:}^{(\tau)}, k\Big)\Big). \tag{3}$$

The resulting dynamic graph sequence $\mathcal{G}^{co} = \{G^{(\tau)}\}_{\tau=t-L+1}^{t}$ constructs time-varying inter-machine dependencies from job co-occurrence and is fed into the graph branch of the dual-branch noise predictor.

3.2 Dual-Branch Noise Prediction Network

As discussed in Section 2, workload volatility stems from two heterogeneous sources: regular temporal evolution driven by periodic user behavior, and job-induced dynamics driven by distributed scheduling. To disentangle them in the noise prediction process, we design ϵ_θ as a dual-branch network: a *temporal branch* captures regular temporal evolution along the time axis, and a *graph branch* captures job-induced dynamics along the machine axis through the dynamic graph $\mathcal{G}^{co}$.

Decomposed dual dynamics. Given a hidden representation H derived from X^{co} and X_k^{ta}, the two branches extract complementary representations in parallel with the diffusion-step embedding $e_k = \mathrm{Emb}(k)$ injected into both:

$$H_{\mathcal{T}} = \mathrm{Transformer}\big(H + e_k\big), \tag{4}$$

$$H_{\mathcal{G}} = \mathrm{GatedGCN}\big(H + e_k,\ \mathcal{G}^{co}\big). \tag{5}$$

The temporal branch operates along the time axis to capture regular temporal evolution. The graph branch propagates messages along the machine axis under the job co-occurrence edges in $\mathcal{G}^{co}$, with a learned gate that modulates how much neighbor information is injected. This replaces the inter-feature transformer used in standard CSDI [27] so that inter-machine dependencies are modeled via explicit job-induced dynamic structure rather than purely learned attention.

Adaptive gated fusion. The relative dominance of regular temporal evolution versus job-induced dynamics shifts over time. We fuse the two representations with a time-variant gate

$$\lambda = \sigma\big(\mathcal{W}_{gate}(H_{\mathcal{T}} + H_{\mathcal{G}})\big) \in (0, 1), \tag{6}$$

producing the fused representation

$$H_{fused} = \lambda \odot H_{\mathcal{T}} + (1 - \lambda) \odot H_{\mathcal{G}}, \tag{7}$$

from which the predicted noise $\hat{\epsilon}$ is obtained. A value of λ close to 1 indicates a state dominated by regular temporal evolution, while a value close to 0 signals strong job-induced dynamics.

Table 1: Deterministic forecasting results on Google Cluster Trace and Alibaba Cluster Trace. All results are reported as mean ± standard deviation over three random seeds. Lower values indicate better performance. The best results are highlighted in bold and the second-best results are underlined.

Dataset	Model	60→6		60→30	
		MAE	RMSE	MAE	RMSE
Google	GRU	0.0650 ± 0.0022	0.0890 ± 0.0028	0.0727 ± 0.0054	0.0973 ± 0.0046
	TimeMixer	0.0543 ± 0.0003	0.0849 ± 0.0045	0.0643 ± 0.0004	0.0942 ± 0.0005
	iTransformer	0.0544 ± 0.0001	0.0818 ± 0.0000	0.0642 ± 0.0000	0.0949 ± 0.0000
	AGCRN	0.0540 ± 0.0000	$\underline{0.0803 \pm 0.0005}$	$\underline{0.0633 \pm 0.0005}$	$\underline{0.0923 \pm 0.0005}$
	PatchTST	0.0541 ± 0.0000	0.0815 ± 0.0000	0.0637 ± 0.0000	0.0945 ± 0.0000
	TimeFilter	0.0539 ± 0.0001	0.0813 ± 0.0002	0.0643 ± 0.0001	0.0949 ± 0.0002
	TMDM+NSTrans	0.0665 ± 0.0008	0.0941 ± 0.0009	0.0740 ± 0.0020	0.1050 ± 0.0028
	DiffSTG	–	–	–	–
	D3U	0.0585 ± 0.0039	$0.0596 \pm 0.0372^{\dagger}$	0.0672 ± 0.0028	0.0973 ± 0.0022
	CSDI	$\underline{0.0533 \pm 0.0005}$	0.0807 ± 0.0006	0.0634 ± 0.0010	0.0940 ± 0.0015
	GD-Diff	$\mathbf{0.0524 \pm 0.0001}$	$\mathbf{0.0795 \pm 0.0001}$	$\mathbf{0.0615 \pm 0.0007}$	$\mathbf{0.0905 \pm 0.0009}$
Alibaba	GRU	6.4890 ± 0.3387	8.4730 ± 0.2897	8.5113 ± 0.3771	10.8090 ± 0.5466
	TimeMixer	4.5947 ± 0.0349	6.4605 ± 0.0464	5.7344 ± 0.0792	8.2124 ± 0.0939
	iTransformer	4.5817 ± 0.0048	6.4548 ± 0.0080	5.7588 ± 0.0137	8.1880 ± 0.0143
	AGCRN	4.8393 ± 0.0314	6.7420 ± 0.0419	6.0683 ± 0.0165	8.6990 ± 0.0955
	PatchTST	4.5402 ± 0.0045	$\underline{6.3860 \pm 0.0050}$	$\underline{5.6522 \pm 0.0197}$	$\underline{8.0808 \pm 0.0109}$
	TimeFilter	$\underline{4.5156 \pm 0.0053}$	$\mathbf{6.3829 \pm 0.0110}$	5.7179 ± 0.0382	8.2162 ± 0.0964
	TMDM+NSTrans	5.3833 ± 0.1039	7.4365 ± 0.1260	6.6318 ± 0.4736	9.2438 ± 0.5939
	DiffSTG	8.2000 ± 0.3197	12.7300 ± 1.1630	8.4633 ± 1.3056	11.8733 ± 1.4346
	D^3U	4.9502 ± 0.2565	6.6798 ± 0.1718	5.9452 ± 0.1019	8.3266 ± 0.0356
	CSDI	4.7331 ± 0.0604	6.6396 ± 0.0859	6.1252 ± 0.2708	9.2129 ± 0.7587
	GD-Diff	$\mathbf{4.4281 \pm 0.0603}$	6.4637 ± 0.1068	$\mathbf{5.4736 \pm 0.0265}$	$\mathbf{8.0390 \pm 0.0685}$

Note. †The RMSE result of D3U on Google 60→6 contains an abnormal seed value and is not considered when marking the best and second-best results.

3.3 Graph-Conditioned Diffusion Framework

With the noise predictor ϵ_θ defined above and the dynamic graph $\mathcal{G}^{co}$ defined in Section 3.1, we now situate them within the conditional DDPM framework [13, 27] that completes GD-Diff. The forward process gradually corrupts the clean target X_0^{ta} into Gaussian noise over T steps under a fixed variance schedule $\{\beta_k\}_{k=1}^{T}$. The reverse process learns to invert this corruption by repeatedly applying ϵ_θ, jointly conditioning on the observed history X^{co} and the dynamic graph $\mathcal{G}^{co}$. The model is trained with the standard noise-matching objective:

$$\mathcal{L}(\theta) = \mathbb{E}_{X_0^{ta},\, X^{co},\, \mathcal{G}^{co},\, \epsilon,\, k}\left[\left\| \epsilon - \epsilon_\theta\big(X_k^{ta},\, k \,\big|\, X^{co},\, \mathcal{G}^{co}\big) \right\|_2^2 \right], \tag{8}$$

where $k \sim \mathrm{Uniform}\{1, \ldots, T\}$ and $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. At inference, the trained ϵ_θ is applied iteratively to denoise Gaussian samples into plausible workload trajectories, yielding both point estimates and prediction intervals for risk-aware downstream decision-making.

4 Experiments

4.1 Experimental Setup

Experimental Setup. We evaluate GD-Diff on two public production cluster traces: Alibaba Cluster Trace 2018 [11], which records resource usage and job scheduling logs of about 4,000 machines over 8 days, and Google Cluster Trace 2019 [34], which records resource usage and scheduling logs of a Borg cell over one month. For both datasets, we construct a 100-machine subset, with 1-minute resolution for Alibaba and 5-minute resolution for Google according to their resolution. The look-back window is fixed to $L = 60$, and we evaluate two forecasting horizons, $K \in \{6, 30\}$. We compare GD-Diff with deterministic forecasting models, including GRU [5], AGCRN [17], TimeMixer [29], iTransformer [20], PatchTST [23], and TimeFilter [14], as well as generative forecasting models, including CSDI [27], DiffSTG [32], TMDM [19], and D^3U [16]. All models are trained with three random seeds $\{2024, 2025, 2026\}$ on NVIDIA RTX 4090 GPUs, and we report the mean and standard deviation across runs. At inference, GD-Diff draws 100 samples from the diffusion process. We use MAE and RMSE to evaluate point forecasting accuracy and CRPS

Table 2: Probabilistic forecasting results on Google Cluster Trace and Alibaba Cluster Trace. All results are reported as mean ± standard deviation over three random seeds. Lower CRPS indicates better probabilistic forecasting quality. The best results are highlighted in bold and the second-best results are underlined.

Model	Google		Alibaba	
	60→6	60→30	60→6	60→30
TMDM+NSTrans	0.0517 ± 0.0007	0.0556 ± 0.0005	4.2222 ± 0.1048	4.8792 ± 0.3192
DiffSTG	–	–	–	–
D^3U	0.0431 ± 0.0024	0.0495 ± 0.0010	3.5133 ± 0.1503	$\underline{4.2715 \pm 0.0180}$
CSDI	$\underline{0.0393 \pm 0.0003}$	$\underline{0.0468 \pm 0.0008}$	$\underline{3.4727 \pm 0.0775}$	4.4099 ± 0.2149
GD-Diff	$\mathbf{0.0381 \pm 0.0000}$	$\mathbf{0.0452 \pm 0.0005}^{\dagger}$	$\mathbf{3.2244 \pm 0.0252}$	$\mathbf{4.0320 \pm 0.0303}$

Note. †The CRPS result of GD-Diff on Google 60→30 is computed from the available seeds and should be updated after the missing seed is completed.

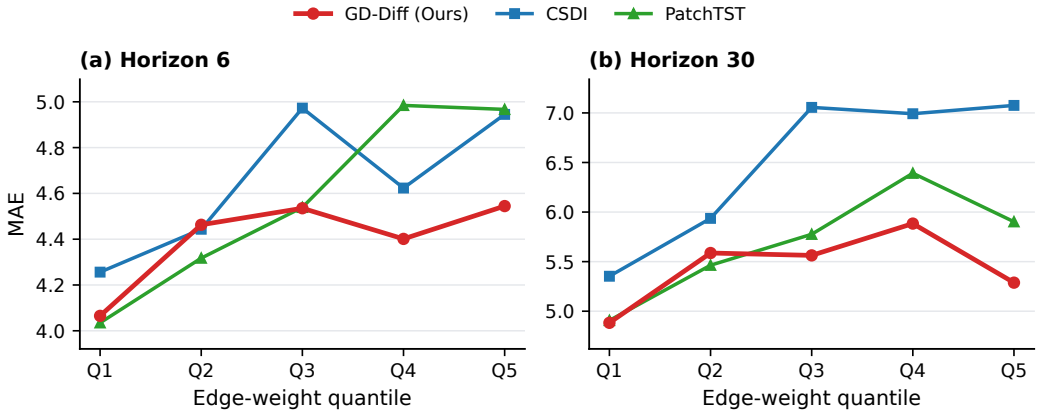


Figure 3: Ablation study results comparing GD-Diff against three controlled variants. Reported values represent the average performance across both 6-step and 30-step forecasting horizons.

to evaluate probabilistic forecasting quality, where lower values indicate better performance for all metrics. Full hyperparameter settings are provided in Appendix B.

4.2 Main Results

Table 1 reports deterministic forecasting results on Google and Alibaba datasets, averaged over three random seeds. GD-Diff achieves the best MAE in all four (dataset, horizon) settings and the best RMSE in three of four, demonstrating that job-induced structural conditioning consistently improves the central forecast while remaining competitive under squared-error evaluation. Among the baselines, recent Transformer-based methods (TimeFilter, PatchTST, iTransformer) form the strongest pool but treat each machine independently and miss the scheduling-induced cross-machine structure. AGCRN, equipped with a self-adaptive AVWGCN graph, reaches the level of the strongest temporal baselines, indicating that data-driven adaptive graphs recover part of the inter-machine dependency, whereas DiffSTG, built on a predefined adjacency, cannot adapt to the dynamic dependencies driven by job scheduling.

Table 2 reports probabilistic metric CRPS. GD-Diff attains the lowest CRPS across all settings, outperforming all probabilistic diffusion baselines including CSDI, TMDM, and the recent D^3U. This indicates that architectural refinements to the diffusion framework alone do not translate into gains on cluster workload forecasting; calibrated uncertainty additionally requires modeling the structural source of volatility, which GD-Diff captures through its job-induced graph and preserves through the dual-branch noise predictor, driving consistent state-of-the-art performance.

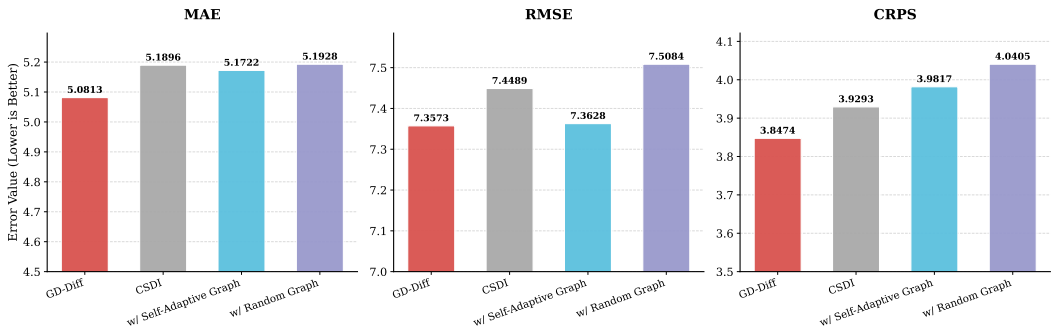


Figure 4: Ablation study results comparing GD-Diff against three controlled variants. Reported values represent the average performance across both 6-step and 30-step forecasting horizons.

4.3 Ablation Study

To examine the contribution of the two core designs in GD-Diff (the job-induced dynamic graph and the parallel dual-branch noise predictor), we compare the full model against two controlled variants:

- **w/ Self-Adaptive Graph**: replaces $\mathcal{G}^{co}$ with a learnable latent graph inferred from workload data, removing the explicit scheduling-derived structure.
- **w/ Cascaded T→G**: replaces the parallel dual-branch design with a serial T→G structure (the temporal module followed by the graph module), keeping $\mathcal{G}^{co}$ unchanged.

Effect of the job-induced graph. The w/ Self-Adaptive Graph variant recovers part of the loss but still trails GD-Diff, indicating that purely data-driven graph learning struggles to infer the dynamic inter-machine dependencies. Together, these results show that GD-Diff's gains come from the inter-machine structure induced by job co-occurrence

Effect of the parallel dual-branch design. The w/ Cascaded T→G variant also underperforms GD-Diff, showing that simply stacking the temporal and graph modules is less effective than modeling them in parallel. In the serial T→G structure, temporal modeling is applied before graph branch, which weakens the independent extraction of job-induced dynamics. This supports our parallel dual-branch design, where regular temporal evolution and job-induced dynamics are modeled separately and then fused adaptively.

5 Related Work

5.1 Deterministic Workload Forecasting

Early statistical approaches were limited to linear patterns and sensitive to volatility [3, 1, 18]. With the rapid development of deep learning, Recurrent Neural Networks particularly LSTM and GRU, have been employed to handle non-linear sequential dependencies in workload forecasting [8, 2, 37]. Researchers also proposed hybrid models to better capture features: Chen et al. [4] fused SG filter, CNN, BiLSTM with attention to extract local features; Zhao et al. [38] combine Time-Frequency Enhanced GRU with Attention extract features from both the frequency and temporal domains; and esDNN [35] modeled multivariate correlations to capture cross-series dependencies. Most recently, CATScaler [22] have advanced long-sequence forecasting by developing a convolution-augmented transformer to accurately predict workload changes at both local and global levels.

Despite these advancements, most existing approaches provide only deterministic point estimates, thereby failing to quantify the uncertainty risks essential for reliable resource management. Moreover, these methods fail to leverage job information, which fundamentally drives workload dynamics, thereby ignoring the implicit structural connections between machines established by distributed scheduling.

5.2 Probabilistic Forecasting

Probabilistic forecasting quantifies uncertainty by modeling the conditional distribution of future values [9]. Representative general-purpose approaches include MQ-CNN [33], which adopts a Sequence-to-Sequence framework for direct quantile prediction, and DeepAR [26], which utilizes autoregressive RNNs to learn distribution parameters. Extending these concepts to cloud computing, Rossi et al. [25] developed uncertainty-aware prediction frameworks specifically for cloud workloads, emphasizing their critical role in improving the reliability of resource management strategies.

However, these approaches often struggle to capture the complex, multi-modal, and non-stationary nature of cloud workloads. For instance, parametric methods rely on fixed distribution assumptions, while quantile-based approaches may lack the generative expressiveness to model full data densities. Denoising Diffusion Probabilistic Models offer a more powerful paradigm, providing superior capability in modeling complex and arbitrary distributions [13]. Representative works include TimeGrad [24], which combines diffusion models with RNNs to learn the data gradient in an autoregressive manner. In contrast, CSDI [27] proposes a conditional score-based framework that treats forecasting as a specialized imputation task, generating future values non-autoregressively based on observed history. Despite their effectiveness, these general-purpose time-series models primarily focus on temporal dynamics, failing to capture the latent spatial correlations between machines.

5.3 Spatio-temporal Graph Forecasting

To capture correlations in multivariate systems, Spatio-Temporal Graph Neural Networks have become an important paradigm, particularly in domains with explicit topologies like traffic forecasting [**?** 30]. Recently, researchers have integrated diffusion models with ST-GNNs to enhance generative capabilities. For example, DiffSTG [32] combines denoising diffusion models with graph neural networks to capture spatio-temporal connection and enable probabilistic forecasting. To address scenarios with unknown topologies, approaches like DiffGSL [15] incorporate self-adaptive graph structure into the diffusion process to infer possible latent dependencies.

Despite these advancements, directly applying these methods to data center workloads remains non-trivial. Unlike traffic networks with fixed physical connections, the structural dependencies in data centers are implicit and highly dynamic, driven by job co-occurrence. Existing generic spatio-temporal diffusion models fail to effectively mine these implicit correlations and lack a mechanism to explicitly decouple the smooth temporal trends from the sharp, job-induced dynamics. Moreover, these models typically rely on vision-based U-Net architectures, which treat time-series as images and often struggle with long-term dependencies.

6 Conclusion

We presented GD-Diff, a graph-conditioned dual-branch diffusion model for probabilistic workload forecasting in production clusters. By constructing a job-induced dynamic graph from scheduling logs and decomposing the noise predictor into a temporal branch and a graph branch, GD-Diff explicitly disentangles regular temporal evolution from job-induced dynamics. Experiments on the Alibaba and Google cluster traces show that GD-Diff consistently improves both point accuracy and probabilistic calibration over state-of-the-art deterministic and generative baselines, providing a reliable forecasting foundation for risk-aware resource management in electricity-computing synergy.

Limitations. GD-Diff inherits the inference cost of diffusion sampling (100 samples $\times$ 50 denoising steps), which is slower than point-prediction baselines and may exceed the latency budget of sub-second control loops. The operational time scales most relevant to our target setting, however, are on the order of minutes to hours: day-ahead and intra-day energy market participation, renewable generation forecasting, energy storage dispatch, and renewable-aware workload scheduling all operate at this granularity, leaving ample headroom for GD-Diff's inference cost.

References

References

[1] Danilo Ardagna, Sara Casolari, Michele Colajanni, and Barbara Panicucci. Dual time-scale distributed capacity allocation and load redirect algorithms for cloud systems. *Journal of Parallel and Distributed Computing*, 72(6):796–808, 2012.

[2] Deep Bodra and Sushil Khairnar. Machine learning-based cloud resource allocation algorithms: a comprehensive comparative review. *Frontiers in Computer Science*, 7:1678976, 2025.

[3] Rodrigo N Calheiros, Enayat Masoumi, Rajiv Ranjan, and Rajkumar Buyya. Workload prediction using arima model and its impact on cloud applications' qos. *IEEE transactions on cloud computing*, 3(4):449–458, 2014.

[4] Lei Chen, Weiwen Zhang, and Haiming Ye. Accurate workload prediction for edge data centers: Savitzky-golay filter, cnn and bilstm with attention mechanism. *Applied Intelligence*, 52(11): 13027–13042, 2022.

[5] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734. Association for Computational Linguistics, 2014. doi: 10.3115/v1/D14-1179.

[6] Philip Colangelo, Ayse K Coskun, Jack Megrue, Ciaran Roberts, Shayan Sengupta, Varun Sivaram, Ethan Tiao, Aroon Vijaykar, Chris Williams, Daniel C Wilson, et al. Ai data centres as grid-interactive assets. *Nature Energy*, 11(2):254–261, 2026.

[7] Mehiar Dabbagh, Bechir Hamdaoui, Ammar Rayes, and Mohsen Guizani. Shaving data center power demand peaks through energy storage and workload shifting control. *IEEE Transactions on Cloud Computing*, 7(4):1095–1108, 2017.

[8] Martin Duggan, Karl Mason, Jim Duggan, Enda Howley, and Enda Barrett. Predicting host cpu utilization in cloud computing using recurrent neural networks. In *2017 12th international conference for internet technology and secured transactions (ICITST)*, pages 67–72. IEEE, 2017.

[9] Tilmann Gneiting and Adrian E Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American statistical Association*, 102(477):359–378, 2007.

[10] Jiaju Guo, Wenjuan Hou, Xiaoyue Wang, Xueliang Zhang, Shaohong Wu, Linsheng Yang, Wenhui Jiang, and Lunwei Zhang. Toward a new spatial paradigm of electricity–computing synergy under renewable energy endowments and geographical environmental constraints. *Applied Energy*, 409:127495, 2026.

[11] Jing Guo, Zihao Chang, Sa Wang, Haiyang Ding, Yihui Feng, Li Mao, and Yungang Bao. Who limits the resource efficiency of my datacenter: An analysis of alibaba datacenter traces. In *2019 IEEE/ACM 27th International Symposium on Quality of Service (IWQoS)*, pages 1–10. IEEE, 2019. doi: 10.1109/IWQoS.2019.8765655.

[12] Haitian Hang, Xiu Tang, Jianling Sun, Lingfeng Bao, David Lo, and Haoye Wang. Robust auto-scaling with probabilistic workload forecasting for cloud databases. In *2024 IEEE 40th International Conference on Data Engineering (ICDE)*, pages 4016–4029. IEEE, 2024.

[13] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[14] Yifan Hu, Guibin Zhang, Peiyuan Liu, Disen Lan, Naiqi Li, Dawei Cheng, Tao Dai, Shu-Tao Xia, and Shirui Pan. Timefilter: Patch-specific spatial-temporal graph filtration for time series forecasting. *arXiv preprint arXiv:2501.13041*, 2025.

[15] Chanyoung Jung and Yun Jang. Diffgsl: A graph structure learning diffusion model for dynamic spatio-temporal forecasting. In *2024 IEEE International Conference on Big Data (BigData)*, pages 3124–3133. IEEE, 2024.

[16] Qi Li, Zhenyu Zhang, Lei Yao, Zhaoxia Li, Tianyi Zhong, and Yong Zhang. Diffusion-based decoupled deterministic and uncertain framework for probabilistic multivariate time series forecasting. In *The Thirteenth International Conference on Learning Representations*, 2025.

[17] Ruoyu Li, Sheng Wang, Feiyun Zhu, and Junzhou Huang. Adaptive graph convolutional neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[18] Ta-Hsin Li. A hierarchical framework for modeling and forecasting web server workload. *Journal of the American Statistical Association*, 100(471):748–763, 2005.

[19] Yuxin Li, Wenchao Chen, Xinyue Hu, Bo Chen, Baolin Sun, and Mingyuan Zhou. Transformer-modulated diffusion models for probabilistic multivariate time series forecasting. In *The Twelfth International Conference on Learning Representations*, 2024.

[20] Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. itransformer: Inverted transformers are effective for time series forecasting. *arXiv preprint arXiv:2310.06625*, 2023.

[21] Tania Lorido-Botran, Jose Miguel-Alonso, and Jose A Lozano. A review of auto-scaling techniques for elastic applications in cloud environments. *Journal of grid computing*, 12(4): 559–592, 2014.

[22] Fan'an Meng, Hongjun Dai, Guoqing Cong, Bo Zhu, and Hailiang Zhao. Catscaler: A convolution-augmented transformer scaling framework for cloud-native applications. *IEEE Transactions on Services Computing*, 2025.

[23] Yuqi Nie, Nam H Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. *arXiv preprint arXiv:2211.14730*, 2022.

[24] Kashif Rasul, Calvin Seward, Ingmar Schuster, and Roland Vollgraf. Autoregressive denoising diffusion models for multivariate probabilistic time series forecasting. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 8857–8868. PMLR, 2021.

[25] Andrea Rossi, Andrea Visentin, Diego Carraro, Steven Prestwich, and Kenneth N Brown. Forecasting workload in cloud computing: towards uncertainty-aware predictions and transfer learning. *Cluster Computing*, 28(4):258, 2025.

[26] David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. Deepar: Probabilistic forecasting with autoregressive recurrent networks. *International journal of forecasting*, 36(3): 1181–1191, 2020.

[27] Yusuke Tashiro, Jiaming Song, Yang Song, and Stefano Ermon. Csdi: Conditional score-based diffusion models for probabilistic time series imputation. *Advances in neural information processing systems*, 34:24804–24816, 2021.

[28] Abhishek Verma, Luis Pedrosa, Madhukar Korupolu, David Oppenheimer, Eric Tune, and John Wilkes. Large-scale cluster management at google with borg. In *Proceedings of the tenth european conference on computer systems*, pages 1–17, 2015.

[29] Shiyu Wang, Haixu Wu, Xiaoming Shi, Tengge Hu, Huakun Luo, Lintao Ma, James Y Zhang, and Jun Zhou. Timemixer: Decomposable multiscale mixing for time series forecasting. *arXiv preprint arXiv:2405.14616*, 2024.

[30] Xiaoyang Wang, Yao Ma, Yiqi Wang, Wei Jin, Xin Wang, Jiliang Tang, Caiyan Jia, and Jian Yu. Traffic flow prediction via spatial-temporal graph neural network. In *Proceedings of the Web Conference 2020*, pages 1082–1092, Taipei, Taiwan, 2020. ACM.

[31] Yuxin Wang, Yuhan Chen, Zeyu Li, Xueze Kang, Yuchu Fang, Yeju Zhou, Yang Zheng, Zhenheng Tang, Xin He, Rui Guo, et al. Burstgpt: A real-world workload dataset to optimize llm serving systems. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pages 5831–5841, 2025.

[32] Haomin Wen, Youfang Lin, Yutong Xia, Huaiyu Wan, Qingsong Wen, Roger Zimmermann, and Yuxuan Liang. Diffstg: Probabilistic spatio-temporal graph forecasting with denoising diffusion models. In *Proceedings of the 31st ACM International Conference on Advances in Geographic Information Systems*, pages 1–12, Hamburg, Germany, 2023. ACM.

[33] Ruofeng Wen, Kari Torkkola, Balakrishnan Narayanaswamy, and Dhruv Madeka. A multi-horizon quantile recurrent forecaster. *arXiv preprint arXiv:1711.11053*, 2017.

[34] John Wilkes. Yet more Google compute cluster trace data. Google research blog, April 2020. Posted at `https://ai.googleblog.com/2020/04/yet-more-google-compute-cluster-trace.html`.

[35] Minxian Xu, Chenghao Song, Huaming Wu, Sukhpal Singh Gill, Kejiang Ye, and Chengzhong Xu. esdnn: deep neural network based multivariate workload prediction in cloud computing environments. *ACM Transactions on Internet Technology (TOIT)*, 22(3):1–24, 2022.

[36] Bin Yang, Wei Xue, Tianyu Zhang, Shichao Liu, Xiaosong Ma, Xiyang Wang, and Weiguo Liu. End-to-end i/o monitoring on leading supercomputers. *ACM Transactions on Storage*, 19(1): 1–35, 2023.

[37] Haitao Yuan, Jing Bi, Shuang Li, Jia Zhang, and MengChu Zhou. An improved lstm-based prediction approach for resources and workload in large-scale data centers. *IEEE Internet of Things Journal*, 11(12):22816–22829, 2024.

[38] Feiyu Zhao, Weiwei Lin, Shengsheng Lin, Haocheng Zhong, and Keqin Li. Tfegru: time-frequency enhanced gated recurrent unit with attention for cloud workload prediction. *IEEE Transactions on Services Computing*, 2024.

A Details of the Empirical Analysis

This appendix provides the dataset preprocessing and analysis hyperparameters used to produce Figure 1 in Section 2.

A.1 Dataset and Residual Computation

We use the Alibaba Cluster Trace 2018 [11]. From the trace we subsample $N{=}100$ machines and aggregate their CPU utilization at 1-minute resolution over the 8-day trace, yielding $T{=}11{,}520$ time steps per machine.

The regular daily pattern is removed via a **leave-one-day daily template**: for each day d and each minute-of-day m, the template $T_{i,d,m}$ is the mean of machine i's CPU utilization at minute m across all days other than d. The residual is

$$r_i(t) = x_i(t) - T_{i,\,d(t),\,m(t)}, \tag{9}$$

where $d(t)$ and $m(t)$ are the day and minute-of-day indices of timestamp t. This leave-one-out construction prevents the day under analysis from contaminating its own template.

A.2 Figure 1(a) — Temporal Uncertainty

Daily CPU profiles are obtained by reshaping each machine's trace into $(\text{days}) \times 1440$ and averaging across the 100 machines, producing one daily profile per day. We visualize the across-day P10–P90 and P25–P75 percentile bands together with the cross-day mean.

A.3 Figure 1(b) — Structured Residual Pattern

We slide a $W{=}120$-minute window with step 1 minute across the residual signal. For each window we compute the cross-machine $|\mathrm{corr}|$ matrix and select the window maximizing the contrast score $(\mathrm{P}_{90} - \mathrm{P}_{50}) \times \mathrm{median_std}$, ensuring the chosen window has both pronounced inter-machine dependencies and meaningful residual energy. Within this window, we keep the top-100 machines ranked by residual standard deviation, reorder them via average-linkage hierarchical clustering on $1 - |\mathrm{corr}|$, and visualize the resulting submatrix with the diagonal masked.

A.4 Figure 1(c) — Dynamic Correlation

We construct an edge-weight time series $W_{ij}(t) = |\mathrm{corr}_{ij}(t)|$ by sliding a 120-minute window with step 1 minute over the residuals. For each lag $\tau \in \{10, 30, 60, 120, 240\}$ minutes, we compute the per-edge Pearson autocorrelation $\mathrm{corr}(W_{ij}(t), W_{ij}(t+\tau))$ over all snapshots.

The null baseline preserves each snapshot's edge-weight distribution but randomizes edge identities across time: for each time point we independently sample edge indices uniformly at random, so the marginal weight distribution at each t is preserved while the temporal binding between an edge and its identity is broken. We sample $20{,}000$ such null edge series and report the 95th percentile of the null lagged-correlation distribution as the dashed reference line.

A.5 Figure 1(d) — Job-Induced Volatility Coupling

For each anchor time t, the past job-induced edge weight is the per-minute mean of the job-induced adjacency matrix over the past $L{=}60$ minutes; the future response is the residual $|\mathrm{corr}|$ computed over the future $H{=}30$ minutes. Anchors are sampled every 5 minutes. Machine pairs are grouped into 5 quantile bins by their past edge weight; for each bin we report the mean future $|\mathrm{corr}|$ together with a 95% confidence interval over all sampled anchors.

B Implementation Details

This appendix lists the hyperparameters of GD-Diff used in our experiments. When the two datasets differ, we report the Alibaba configuration as the canonical reference.

B.1 Data

For both datasets we use $N{=}100$ machines. Alibaba Cluster Trace 2018 is sampled at 1-minute resolution over 8 days (2018-01-01 to 2018-01-08); Google Cluster Trace 2019 is sampled at 5-minute resolution over one month (2019-05-01 to 2019-06-01). Each trace is split chronologically into train/validation/test with ratio $0.7/0.1/0.2$. The look-back window is $L{=}60$ steps and we evaluate on horizons $K \in \{6, 30\}$.

B.2 Backbone and Diffusion

The conditional noise predictor ϵ_θ consists of 4 residual blocks with $C{=}64$ hidden channels, 8 attention heads, and a diffusion-step embedding of dimension 128. The DDPM uses $T{=}50$ diffusion steps with a quadratic β-schedule from $\beta_1{=}10^{-4}$ to $\beta_T{=}0.5$.

B.3 Job-Induced Graph Branch

The graph branch is implemented as a GatedGCN with Chebyshev order $K_{\mathrm{cheb}}{=}2$, node-embedding dimension 16, and top-k sparsification with $k{=}10$. Edge weights are normalized via min–max statistics computed on the training set.

B.4 Optimization

We train with Adam at initial learning rate 10^{-3} and batch size 4, for at most 100 epochs with early stopping (patience 20). The learning rate is decayed by $\gamma{=}0.1$ at epochs $\{15, 20\}$. Each model is trained with three random seeds $\{2024, 2025, 2026\}$ and we report mean$\pm$std.

B.5 Inference

GD-Diff draws 100 samples from the diffusion process for the main results and 10 samples for ablations and architectural variants.

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 [N/A].
- [N/A] 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 [N/A]).

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 also be asked to 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 [N/A] 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: **[TODO]**

   Justification: **[TODO]**

   Guidelines:

   - The answer [N/A] 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 [N/A] 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: **[TODO]**

   Justification: **[TODO]**

Guidelines:

- The answer [N/A] 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: **[TODO]**

   Justification: **[TODO]**

   Guidelines:

   - The answer [N/A] 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: **[TODO]**

   Justification: **[TODO]**

   Guidelines:

   - The answer [N/A] 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: **[TODO]**

   Justification: **[TODO]**

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.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://neurips.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) necessary to understand the results?

   Answer: **[TODO]**

   Justification: **[TODO]**

   Guidelines:

   - The answer [N/A] 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: **[TODO]**

   Justification: **[TODO]**

   Guidelines:

   - The answer [N/A] 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: **[TODO]**

   Justification: **[TODO]**

   Guidelines:

   - The answer [N/A] 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: **[TODO]**

   Justification: **[TODO]**

   Guidelines:

   - The answer [N/A] 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: **[TODO]**

    Justification: **[TODO]**

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] 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., pre-trained language models, image generators, or scraped datasets)?

    Answer: **[TODO]**

    Justification: **[TODO]**

    Guidelines:

    - The answer [N/A] 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: **[TODO]**

    Justification: **[TODO]**

    Guidelines:

    - The answer [N/A] 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: **[TODO]**

    Justification: **[TODO]**

    Guidelines:

    - The answer [N/A] 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: **[TODO]**

    Justification: **[TODO]**

Guidelines:

- The answer [N/A] 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: **[TODO]**

    Justification: **[TODO]**

    Guidelines:

    - The answer [N/A] 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 rigor, or originality of the research, declaration is not required.

    Answer: **[TODO]**

    Justification: **[TODO]**

    Guidelines:

    - The answer [N/A] 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 in the NeurIPS handbook for what should or should not be described.