DAG Convolutional Networks

Samuel Rey
Universidad Rey Juan Carlos
samuel.rey.escudero@urjc.es

Hamed Ajorlou
University of Rochester
hajorlou@ur.rochester.edu

Gonzalo Mateos
University of Rochester
gmateosb@ur.rochester.edu

Abstract

Directed acyclic graphs (DAGs) provide a natural framework for modeling directional and hierarchical relationships. We introduce the DAG Convolutional Network (DCN), a graph neural architecture specifically developed for convolutional learning on signals defined over DAGs. DCN employs causal graph filters that incorporate the inherent partial order of DAGs, introducing an inductive bias absent in conventional GNNs. Unlike existing approaches, DCN relies on formal convolutional operators that admit spectral interpretations, ensuring both theoretical grounding and efficient implementation. We further propose the Parallel DCN (PDCN), which processes shifted DAG signals through a shared multilayer perceptron, thereby decoupling parameter complexity from graph size while preserving predictive performance. Extensive evaluations confirm that (P)DCN achieves strong performance relative to state-of-the-art methods, combining accuracy, robustness, and computational efficiency within a principled signal processing framework.

1 Introduction

Graph signal processing (GSP) extends statistical learning to non-Euclidean domains by exploiting graph topology as prior knowledge. Recent progress in GSP has shaped Graph Neural Networks (GNNs), which learn node representations via graph convolutions, attention, and autoencoders, enabling applications across science and engineering [1, 2, 3, 4, 5, 6, 7]. While directed graphs naturally model flows, citations, and causal relations, incorporating directionality into GNNs remains challenging—especially for directed acyclic graphs (DAGs), whose nilpotent adjacency matrices collapse spectral information [8, 9]. Yet DAGs are central to causal inference, Bayesian networks, program analysis, and neural architecture optimization [10, 11, 12, 13, 14].

We introduce the *DAG Convolutional Network* (DCN), a convolutional GNN tailored for signals on DAGs. Building on recent extensions of GSP to partially ordered sets [15, 9], DCN layers use causal graph filters that encode the partial order of nodes, providing a strong inductive bias. This design connects naturally to linear structural equation models (SEMs), admits spectral interpretations. Unlike prior DAG-based GNNs [16, 17], DCN employs sparse matrix multiplications, enabling scalable learning of causal dependencies. To improve efficiency, we further propose the *Parallel DCN* (PDCN). Instead of recursive filtering, PDCN applies a shared-parameter MLP to shifted DAG signals in parallel and aggregates the outputs. This scheme decouples parameter count from graph size.

Related work. Recent work has increasingly focused on applying GNN-based machine learning to DAGs. D-VAE [16] introduces a variational autoencoder framework that embeds DAGs into a

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

continuous latent space. DAGNN [17] adopts a sequential aggregation strategy, where each node gathers information from its predecessors via an attention mechanism and updates its state through a gated recurrent unit (GRU), thereby leveraging the partial order of the DAG. However, both approaches rely on sequential operations, which impose high computational costs and limit their scalability to large graphs. DCN, by contrast, leverages weighted transitive closure filters [9] for efficient long-range aggregation, while PDCN further enhances scalability. Transformer-based DAG models [18] encode reachability with attention, but differ fundamentally from our convolutional operators.

Contributions. This work advances DAG representation learning by:

- Proposing DCN, a convolutional GNN for DAGs, and PDCN, a scalable parallel variant with parameter count independent of graph size.
- Demonstrating state-of-the-art performance on synthetic and real datasets, with robustness to additive noise.

Overall, we provide a unified framework for convolutional learning on DAGs, combining new architectures, and extensive experiments.

2 Preliminaries and Problem Statement

We consider signals supported on DAGs. A DAG is denoted $\mathcal{D} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of N nodes and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ the set of directed edges. An edge $(i, j) \in \mathcal{E}$ indicates a directed link $j \to i$. The absence of cycles induces a partial order over $\mathcal{V}$: node j precedes i whenever a directed path $j \to i$ exists [15]. Under a topological ordering, the adjacency matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$ is strictly lower triangular with zero diagonal.

A graph signal is a function $x : \mathcal{V} \to \mathbb{R}$, mapping each node to a real value. This function can be represented as a vector $\mathbf{x} \in \mathbb{R}^N$, where x_i denotes the signal value at node i. Examples include gene expression levels in regulatory networks or hydrological measurements along a river. We refer to these as DAG signals. The DAG signal processing framework of [9] extends classical GSP [2] by introducing graph-shift operators and transitive closures adapted to partially ordered structures.

Problem statement. We are given a dataset $\mathcal{T} = \{(\mathbf{X}_m, \mathbf{y}_m)\}_{m=1}^M$, where each input $\mathbf{X}_m \in \mathbb{R}^{N \times F}$ has F features per node, and $\mathbf{y}_m \in \mathbb{R}^N$ is a single-feature output signal. The goal is to learn a nonlinear map $f_{\mathbf{\Theta}} : \mathbb{R}^{N \times F} \mapsto \mathbb{R}^N$ that depends on $\mathcal{D}$ and leverages its structure. Parameters $\mathbf{\Theta}$ are trained by minimizing the empirical risk

$$\min_{\mathbf{\Theta}} \frac{1}{M} \sum_{m=1}^{M} \mathcal{L}(\mathbf{y}_m, f_{\mathbf{\Theta}}(\mathbf{X}_m; \mathcal{D})), \tag{1}$$

with $\mathcal{L}$ a task-specific loss (e.g., mean squared error for regression or cross-entropy for classification). A typical convolutional GNN layer is $\mathbf{X}^{(\ell)} = \sigma\Big(\sum_{r=0}^{R-1} \mathbf{A}^r \mathbf{X}^{(\ell-1)} \mathbf{\Theta}_r^{(\ell)}\Big)$, where $\mathbf{X}^{(\ell)}$ are node features, $\mathbf{\Theta}_r^{(\ell)}$ learnable filter parameters, and σ a nonlinearity [3, 19, 20]. While effective on general graphs, these filters are not convolutional for DAG signals: $\mathbf{A}$ is nilpotent with all eigenvalues zero, eliminating a spectral domain and voiding the Fourier interpretation. Moreover, DAGs encode a partial order that such models do not exploit. Building on the DAG signal processing framework [9], we develop convolutional architectures that explicitly encode the partial order and causal structure inherent in DAGs. As introduced in Section 4, these architectures help address the limitations of existing approaches.

3 DAG Signal Processing

DAG signal processing [9] provides a Fourier analysis and convolution framework for signals on DAGs. Consider a DAG $\mathcal{D} = (\mathcal{V}, \mathcal{E})$ with adjacency matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$ and a signal $\mathbf{x} \in \mathbb{R}^N$. Following [9], $\mathbf{x}$ is modeled as $\mathbf{x} = \mathbf{W}\mathbf{c}$ with exogenous inputs $\mathbf{c} \in \mathbb{R}^N$, where $\mathbf{W}$ is the weighted transitive closure: $W_{ij} \neq 0$ if j is a predecessor of i, $W_{ii} = 1$, and $\mathbf{W}$ is lower triangular and invertible. A canonical choice is $\mathbf{W} = (\mathbf{I} - \mathbf{A})^{-1} = \mathbf{I} + \mathbf{A} + \cdots + \mathbf{A}^{N-1}$, which implies

$\mathbf{x} = \mathbf{A}\mathbf{x} + \mathbf{c}$, i.e., a linear SEM [10]. For each node $k \in \mathcal{V}$, the causal graph-shift operator is $\mathbf{S}_k = \mathbf{W}\mathbf{D}_k\mathbf{W}^{-1}$, where $\mathbf{D}_k$ is diagonal with $[\mathbf{D}_k]_{ii} = 1$ if $i \leq k$ and 0 otherwise; hence all $\mathbf{S}_k$ are diagonalized by $\mathbf{W}$, whose columns form a Fourier basis, and $\mathbf{c} = \mathbf{W}^{-1}\mathbf{x}$ are the Fourier coefficients. The most general shift-invariant filter, called a causal graph filter, is

$$\mathbf{H} = \sum_{k\in\mathcal{V}} \theta_k \mathbf{S}_k = \mathbf{W}\Big(\sum_{k\in\mathcal{V}} \theta_k \mathbf{D}_k\Big)\mathbf{W}^{-1}, \tag{2}$$

with coefficients $\boldsymbol{\theta} \in \mathbb{R}^N$. The convolution of $\mathbf{x}$ with filter $\mathbf{h}$ is $\mathbf{y} = \mathbf{H}\mathbf{x} = \mathbf{h} *_{\mathcal{D}} \mathbf{x}$, whose frequency response is given by the diagonal of $\sum_k \theta_k \mathbf{D}_k$, i.e., a pointwise scaling of Fourier coefficients $\mathbf{c}$. This provides a principled definition of Fourier analysis and convolution on DAGs that respects both reachability and partial order, forming the foundation of the architectures in Section 4.

4 DAG Convolutional Network

We now introduce two architectures for learning from DAG signals: the DAG Convolutional Network (DCN) and its parallel counterpart (PDCN). The DCN builds directly on causal graph filters from (2), defining each layer as

$$\mathbf{x}^{(\ell)} = \sigma\left(\sum_{k\in\mathcal{V}} \theta_k^{(\ell)} \mathbf{S}_k \mathbf{x}^{(\ell-1)}\right), \quad \ell = 1, \ldots, L, \tag{3}$$

where $\theta_k^{(\ell)}$ are learnable filter coefficients, $\sigma(\cdot)$ is a nonlinearity such as ReLU, and the input is $\mathbf{x}^{(0)} = \mathbf{x}$. Spectrally, the operation $\mathbf{S}_k\mathbf{x}^{(\ell-1)} = \mathbf{W}\mathbf{D}_k\mathbf{W}^{-1}\mathbf{x}^{(\ell-1)}$ selects the exogenous inputs (or "causes") associated with the predecessors of k, which are then diffused through the reachability DAG using $\mathbf{W}$, thereby injecting the inductive bias from the partial order. From a message-passing perspective, each $\mathbf{S}_k$ aggregates features from common predecessors of nodes i and k, producing multiple messages that are combined through the weighted sum in (3) and updated by σ. To increase expressiveness and accommodate multi-feature signals, the single-filter recursion generalizes to a filter bank:

$$\mathbf{X}^{(\ell)} = \sigma\left(\sum_{k\in\mathcal{V}} \mathbf{S}_k \mathbf{X}^{(\ell-1)} \boldsymbol{\Theta}_k^{(\ell)}\right), \tag{4}$$

where $\mathbf{X}^{(\ell)} \in \mathbb{R}^{N\times F_o^{(\ell)}}$ is the feature matrix at layer ℓ, and $\boldsymbol{\Theta}_k^{(\ell)} \in \mathbb{R}^{F_i^{(\ell)}\times F_o^{(\ell)}}$ are learnable filter parameters, with $F_i^{(\ell)} = F_o^{(\ell-1)}$. The DCN thus resembles a GNN but with aggregation restricted to causal GSOs, which ensures that convolution respects the DAG structure. Its advantages include: (i) a spectral interpretation since $\mathbf{S}_k$ has well-defined eigenvalues (binary diagonals of $\mathbf{D}_k$), enabling analyses of stability and transferability; (ii) numerical stability under depth, as $\mathbf{S}_k$ are idempotent and well-conditioned, avoiding the exploding/vanishing spectrum issues common in GNNs; and (iii) efficient computation, since each $\mathbf{S}_k$ is typically quite sparse, so layer complexity is $\mathcal{O}(NF(S+NF))$ with $S = \max_k \|\mathbf{S}_k\|_0$. However, DCN requires one $\boldsymbol{\Theta}_k^{(\ell)}$ per node k, and hence the number of parameters grows linearly with N, which may become a limitation for large graphs. To address scalability, we introduce the Parallel DCN (PDCN), which replaces depth with width. Instead of sequentially stacking layers as in (4), PDCN defines N parallel branches. Each branch processes a shifted input $\mathbf{S}_k\mathbf{X}$ through a shared multilayer perceptron (MLP), i.e.,

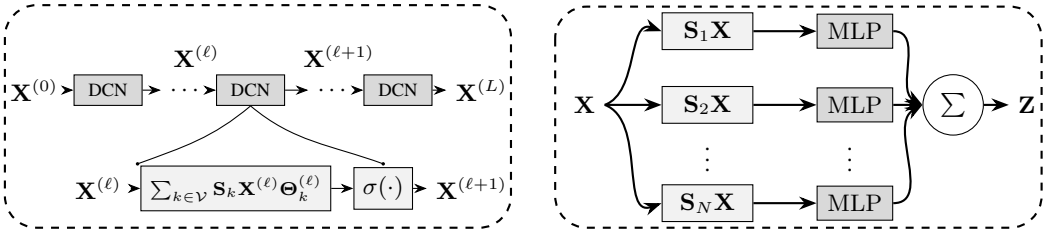
$$\bar{\mathbf{Z}}_k = \mathrm{MLP}(\mathbf{S}_k\mathbf{X}) = \sigma\Big(\ldots\sigma\big(\mathbf{S}_k\mathbf{X}\,\boldsymbol{\Theta}^{(1)}\big)\ldots\boldsymbol{\Theta}^{(L)}\Big), \tag{5}$$

where the parameters $\boldsymbol{\Theta}^{(\ell)}$ are identical across all branches, following a siamese design. The branch outputs are then aggregated as

$$\mathbf{Z} = \sum_{k\in\mathcal{V}} \bar{\mathbf{Z}}_k, \tag{6}$$

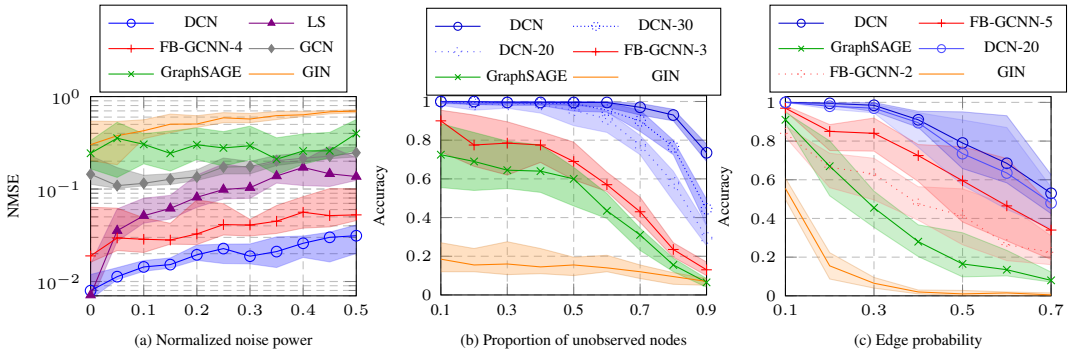
which defines the final PDCN representation.

This parallel design yields several advantages. (i) The number of trainable parameters becomes independent of N, scaling as $\mathcal{O}(F^2L)$ under weight sharing. (ii) Computations can be parallelized across branches, and in practice the shifted inputs $\mathbf{S}_k\mathbf{X}$ may be precomputed to reduce runtime. (iii) The architecture belongs to the family of parallel GNNs with injective aggregation and readout [21], thereby satisfying the Weisfeiler–Lehman (WL) test for distinguishing non-isomorphic DAGs [22].



(a) DCN architecture (b) PDCN architecture

Figure 1: Comparison of the DCN (a) and PDCN (b) architectures. DCN is sequential, applying causal convolutions layer by layer with a representative computation $\sum_{k\in\mathcal{V}} \mathbf{S}_k \, \mathbf{X}^{(\ell)} \, \mathbf{\Theta}_k^{(\ell)}$ followed by a nonlinearity. PDCN processes multiple shifted signals $\mathbf{S}_k\mathbf{X}$ in parallel through weight-shared MLP branches and aggregates $\mathbf{Z} = \sum_k \overline{\mathbf{Z}}_k$, keeping the parameter count independent of N.



(a) Normalized noise power (b) Proportion of unobserved nodes (c) Edge probability

Figure 2: (a) NMSE in the network diffusion task as the noise in the observations increases. For the source identification task, we plot accuracy as a function of (b) the proportion of unobserved nodes and (c) the edge probability. We report the median performance across 25 realizations and values between the first and third quartile in the shaded area.

5 Numerical Evaluation

In this section, we evaluate the proposed DCN across diverse tasks, datasets, and conditions. We aim to demonstrate its ability to learn from DAG-supported signals, and to benchmark its performance against state-of-the-art baselines, highlighting accuracy as well as robustness to noise.

Baselines. We compare (P)DCN with both DAG-agnostic and DAG-aware models. The former include GCN [3], FB-GCNN [19], GraphSAGE [23], GIN [4], GAT [5], and an MLP [24] that ignores graph structure. The latter comprise DAGNN [17] and D-VAE [16], from which we adopt only the encoder modules, since our task focuses on direct learning from DAG signals. Full implementation and training details are given in Appendix A.1.

Tasks and metrics. We assess (P)DCN on both synthetic and real data. In the synthetic setting, we study diffusion learning, where the objective is to predict the output of a causal graph filter applied to input signals on a DAG, with performance measured by normalized mean squared error (NMSE). We also consider source identification, where the model must recover the hidden origin of a propagated signal; this is treated as a node-classification problem, with accuracy defined as the fraction of correctly predicted source nodes. To demonstrate practical relevance, we further evaluate (P)DCN on hydrological data from the River Thames, forecasting pollutant concentrations using a DAG constructed from hydrological dependencies, thereby illustrating its applicability to real-world scientific domains. In all cases, the data is split into 70% for training, 20% for validation, and 10% for testing.

Table 1: Performance on diffusion learning (NMSE) and source identification (Accuracy). Results show mean $\pm$ std and average runtime across 25 runs.

	Diffusion learning		Source identification	
Method	NMSE	Time (s)	Accuracy	Time (s)
DCN	$\mathbf{0.014 \pm 0.010}$	7.9	0.057 ± 0.008	22.2
DCN-30	0.029 ± 0.017	6.0	0.053 ± 0.012	20.1
DCN-10	0.049 ± 0.021	5.9	0.054 ± 0.005	17.0
DCN-T	0.073 ± 0.024	8.1	$\mathbf{0.997 \pm 0.018}$	22.5
DCN-30-T	0.153 ± 0.030	6.0	0.996 ± 0.032	17.8
DCN-10-T	0.181 ± 0.030	5.9	0.933 ± 0.141	19.5
DAGNN	0.173 ± 0.040	5111.8	0.957 ± 0.031	9344.5
D-VAE	0.163 ± 0.041	14024.1	0.987 ± 0.007	12841.9
LS	0.048 ± 0.022	0.4	0.060 ± 0.016	0.45
PDCN	0.098 ± 0.015	22.6	0.288 ± 0.114	79.5
FB-GCNN	0.138 ± 0.028	5.8	0.676 ± 0.172	18.6
GraphSAGE	0.341 ± 0.039	9.9	0.620 ± 0.163	40.0
GIN	0.404 ± 0.079	9.7	0.176 ± 0.098	39.8
GCN	0.167 ± 0.037	5.2	0.052 ± 0.018	7.6
GAT	0.653 ± 0.089	19.9	0.071 ± 0.018	38.5
MLP	0.392 ± 0.012	4.1	0.050 ± 0.016	16.9

5.1 Synthetic Setup

Unless specified otherwise, we generate synthetic graphs from an Erdős–Renyi random DAG model with $N = 100$ nodes and edge probability $p = 0.2$. A total of 2000 input–output signal pairs are simulated according to $\mathbf{y} = \mathbf{H}\mathbf{x}$. The input $\mathbf{x}$ is sparse, with nonzero entries restricted to the first 25 nodes, while $\mathbf{H}$ is a causal graph filter (Eq. 2) constructed from 25 causal GSOs selected uniformly at random, with filter taps sampled uniformly from $[-1, 1]$. $\mathbf{y}$ represents the diffused output signal. Both input and output signals are perturbed with zero-mean additive white Gaussian noise, normalized relative to the signal power to achieve 5% noise power. For the source identification task, the input $\mathbf{x}$ is one-hot with the nonzero entry restricted to the first 25 topologically ordered nodes of $\mathcal{V}$. Since the diffusion $\mathbf{y} = \mathbf{H}\mathbf{x}$ preserves the source value, the output $\mathbf{y}$ is masked at the source node to prevent trivial identification. All reported results are averaged over 25 trials

Findings and discussion. Table 1 shows that the proposed architectures—particularly DCN—consistently outperform baselines. In the network diffusion task, using subsets of 30 or 10 GSOs (DCN-30 and DCN-10) decouples parameters from graph size while retaining strong accuracy, with DCN-30 second only to the full DCN. Compared to Least Squares (LS), DCN is more robust to noisy inputs (see Test Case 1). Conversely, reversing edge directions with transposed GSOs (DCN-T) sharply reduces performance, especially with fewer GSOs. In source identification, the trend reverses: DCN variants struggle, while DCN-T variants reach near-perfect accuracy. This matches intuition, as source detection requires traversing DAGs in reverse. These results highlight the importance of DAG directionality. In terms of cost, PDCN—implemented with a 128-unit MLP—trains and infers slower than DCN, which uses only a 32-dimensional vector per layer ($L = 2$). With comparable neurons, runtimes converge (results not shown). Overall, DAG-agnostic models underperform DAG-specific ones such as PDCN, DAGNN, and D-VAE. Notably, DCN surpasses DAGNN and D-VAE in accuracy while requiring much less computation.

Test Case 1. For the diffusion task, Fig. 2(a) explores model performance under varying noise levels. In the absence of noise, LS achieves the lowest NMSE overall. However, its accuracy deteriorates sharply as noise power increases. In contrast, DCN achieves NMSE comparable to LS under noiseless conditions, while maintaining significantly greater robustness to noise. This demonstrates that despite the underlying generative process being linear, the nonlinear DCN is advantageous in noisy regimes, particularly when sample sizes are limited.

Test Case 2. For the source identification task, Fig. 2(b) examines classification accuracy as the proportion of unobserved nodes increases. Larger values on the x-axis represent more difficult scenarios with fewer observed nodes and more possible sources. Following the results of Fig. 2, all models in this test use transposed GSOs. The plots show that DCN maintains high accuracy even when a substantial fraction of nodes is unobserved. Among DCN variants, DCN-30 and DCN-20 illustrate the trade-off between reducing the number of GSOs and preserving accuracy. With 50% or

Table 2: Performance comparison of models on the River Thames network. Results for four chemical indicators are reported as mean NMSE $\pm$ std and mean training + testing time (s) over 25 trials. Rightmost column shows trainable parameter counts. Best NMSE values are bolded.

	Silicon		Sulphate		Nitrate		Chloride		Params
Method	NMSE	Time	NMSE	Time	NMSE	Time	NMSE	Time	Count
DCN	$\mathbf{0.006 \pm 0.010}$	2.26	$\mathbf{0.003 \pm 0.003}$	2.20	$\mathbf{0.004 \pm 0.003}$	2.27	$\mathbf{0.020 \pm 0.100}$	2.26	1313
DCN-15	0.049 ± 0.020	2.28	0.031 ± 0.007	2.19	0.073 ± 0.020	2.27	0.076 ± 0.096	2.26	993
PDCN	0.037 ± 0.071	1.77	0.109 ± 0.015	1.74	0.120 ± 0.052	1.77	0.129 ± 0.095	1.76	385
DCN-T	0.054 ± 0.064	2.10	0.116 ± 0.028	2.23	0.069 ± 0.022	2.29	0.117 ± 0.097	2.29	1313
Linear	0.009 ± 0.015	0.01	0.004 ± 0.003	0.01	0.006 ± 0.005	0.01	0.022 ± 0.099	0.01	20
DAGNN	0.025 ± 0.028	144.9	0.044 ± 0.010	112.8	0.086 ± 0.021	84.4	0.049 ± 0.098	144.9	888454
D-VAE	0.012 ± 0.010	183.4	0.004 ± 0.003	177.6	0.086 ± 0.043	182.8	0.040 ± 0.100	183.4	460694
GraphSAGE-A	0.029 ± 0.036	3.90	0.022 ± 0.012	3.81	0.022 ± 0.018	3.89	0.040 ± 0.101	3.89	161
GIN-A	0.057 ± 0.056	3.87	0.174 ± 0.028	3.73	0.204 ± 0.031	3.83	0.192 ± 0.074	3.83	33409
MLP	0.073 ± 0.103	1.63	0.171 ± 0.028	1.58	0.126 ± 0.033	1.62	0.159 ± 0.099	1.62	97
GCN	0.537 ± 0.077	2.01	0.628 ± 0.040	1.96	0.670 ± 0.059	2.00	0.657 ± 0.058	2.02	97
FB-GCNN-2	0.046 ± 0.027	2.27	0.010 ± 0.003	2.21	0.021 ± 0.007	2.24	0.032 ± 0.101	2.25	161
GAT	0.042 ± 0.034	7.88	0.084 ± 0.018	7.65	0.059 ± 0.023	7.82	0.069 ± 0.092	7.85	163

more observed nodes, DCN-30 performs comparably to the full DCN, while DCN-20 continues to outperform non-DAG-based baselines.

Test Case 3. We further study the effect of graph edge density on the source identification task. For Erdős–Rényi graphs with increasing edge probability p, Fig. 2(c) shows that denser graphs make source identification more difficult, since diffusion tends to produce homogeneous signals that differ only by source distance. Importantly, DCN and its variant DCN-20 consistently outperform FB-GCNN-R (with $R = 2$ and 5), validating the advantages of convolutional filters tailored to DAGs over conventional GSP-based graph filters. Even as density increases, DCN accuracy decreases only gradually, underscoring its robustness.

5.2 Real World Setup

Hydrological Data Forecasting. We evaluate the proposed (P)DCN on the River Thames dataset [25], which contains weekly hydrological and chemical measurements from 20 monitoring sites between 2010 and 2016 (excluding years with excessive missing data). Variables include pH, alkalinity, suspended solids, and concentrations of chloride, nitrate, sulphate, silicon, and metals, along with flow rates. Graph signals studied consist of 343 realizations and each graph signal $\mathbf{x} \in \mathbb{R}^{20}$ records simultaneous chemical concentrations at all sites, with diffusion modeled along the river's upstream-to-downstream DAG structure (Fig. 3). We focus on four representative chemical markers: dissolved silicon, sulphate, nitrate, and chloride. The task is cast as graph signal imputation: a subset of intermediary or sink nodes (cyan in Fig. 3) is masked, while source nodes (purple) remain observed. The goal is to interpolate missing downstream values from observed upstream signals, a setting that reflects practical monitoring challenges such as sensor failures. Results in Table 2 show that DCN achieves the lowest NMSE across all indicators. The DCN-15 variant offers a trade-off between accuracy and parameter efficiency, incurring an error increase while reducing model size. In contrast, DAGNN [17] and D-VAE [16] require orders of magnitude more parameters and runtime without outperforming DCN. The parallelized PDCN also performs competitively, surpassing several DAG-agnostic GNNs.

6 Conclusions, Limitations, and Future Work

We introduced DCN, a graph neural architecture tailored for convolutional learning on signals over DAGs. Unlike conventional GNNs, DCN employs causal graph filters that encode node partial orders, yielding sparse operators with clear spectral interpretations. We also proposed PDCN, a parallel and parameter-efficient variant that retains strong predictive performance. Empirically, experiments on synthetic and real datasets, showed consistent gains over state-of-the-art baselines in accuracy, noise robustness, and efficiency. DCN's main limitation lies in its reliance on multiple causal GSOs, with parameter counts growing with graph size. PDCN mitigates this, but future work will pursue adaptive GSO selection, extensions to dynamic or probabilistic DAGs, and Bayesian formulations for uncertainty quantification. Studying stability under graph perturbations also remains an open challenge. Overall, this work offers a principled and efficient framework for deep learning on DAGs, with wide applicability in domains governed by directional or causal structure.

References

[1] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst, "Geometric deep learning: Going beyond Euclidean data," vol. 34, no. 4, pp. 18–42, July 2017.

[2] A. Ortega, P. Frossard, J. Kovačević, J. M. F. Moura, and P. Vandergheynst, "Graph signal processing: Overview, challenges, and applications," *Proc. IEEE*, vol. 106, no. 5, pp. 808–828, 2018.

[3] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. Int. Conf. Learn. Representations*, 2017, pp. 1–14.

[4] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?" in *Proc. Int. Conf. Learn. Representations*, 2019, pp. 1–17.

[5] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *Proc. Int. Conf. Learn. Representations*, 2018, pp. 1–12.

[6] C. Wang, S. Pan, G. Long, X. Zhu, and J. Jiang, "MGAE: Marginalized graph autoencoder for graph clustering," 2017, pp. 889–898.

[7] S. Rey, V. M. Tenorio, S. Rozada, L. Martino, and A. G. Marques, "Overparametrized deep encoder-decoder schemes for inputs and outputs defined over graphs," in *Proc. European Signal Process. Conf. (EUSIPCO)*. IEEE, 2021, pp. 855–859.

[8] A. G. Marques, S. Segarra, and G. Mateos, "Signal processing on directed graphs: The role of edge directionality when processing and learning from network data," vol. 37, no. 6, pp. 99–116, 2020.

[9] B. Seifert, C. Wendler, and M. Püschel, "Causal Fourier analysis on directed acyclic graphs and posets," *IEEE Trans. Signal Process.*, vol. 71, pp. 3805–3820, 2023.

[10] J. Peters, D. Janzing, and B. Schölkopf, *Elements of Causal Inference: Foundations and Learning Algorithms*. The MIT Press, 2017.

[11] X. Zheng, B. Aragam, P. K. Ravikumar, and E. P. Xing, "DAGs with no tears: Continuous optimization for structure learning," *Proc. Adv. Neural. Inf. Process. Syst.*, vol. 31, 2018.

[12] S. S. Saboksayr, G. Mateos, and M. Tepper, "CoLiDE: Concomitant linear DAG estimation," in *Proc. Int. Conf. Learn. Representations*, 2024.

[13] M. Allamanis, E. T. Barr, P. Devanbu, and C. Sutton, "A survey of machine learning for big code and naturalness," *ACM Computing Surveys (CSUR)*, vol. 51, no. 4, pp. 1–37, 2018.

[14] C. Zhang, M. Ren, and R. Urtasun, "Graph hypernetworks for neural architecture search," in *Proc. Int. Conf. Learn. Representations*, 2019.

[15] M. Püschel, B. Seifert, and C. Wendler, "Discrete signal processing on meet/join lattices," vol. 69, pp. 3571–3584, 2021.

[16] M. Zhang, S. Jiang, Z. Cui, R. Garnett, and Y. Chen, "D-VAE: A variational autoencoder for directed acyclic graphs," in *Proc. Adv. Neural. Inf. Process. Syst.*, 2019.

[17] V. Thost and J. Chen, "Directed acyclic graph neural networks," in *Int. Conf. Learn. Representations*, 2021.

[18] Y. Luo, V. Thost, and L. Shi, "Transformers over directed acyclic graphs," in *Proc. Adv. Neural. Inf. Process. Syst.*, vol. 36, 2023, pp. 47 764–47 782.

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

[20] E. Isufi, F. Gama, D. I. Shuman, and S. Segarra, "Graph filters for signal processing and machine learning on graphs," *IEEE Trans. Signal Process.*, vol. 72, pp. 4745–4781, 2024.

[21] S. Doshi and S. P. Chepuri, "Graph neural networks with parallel neighborhood aggregations for graph classification," *IEEE Trans. Signal Process.*, vol. 70, pp. 4883–4896, 2022.

[22] B. Y. Weisfeiler and A. A. Lehman, "A reduction of a graph to a canonical form and an algebra arising during this reduction," *Nauchno-Technicheskaya Informatsia*, vol. 2, no. 9, pp. 12–16, 1968.

[23] W. L. Hamilton, R. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. Adv. Neural. Inf. Process. Syst.*, 2017, pp. 1025–1035.

[24] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[25] M. J. Bowes, L. K. Armstrong, S. A. Harman, D. J. E. Nicholls, H. D. Wickham, P. M. Scarlett, and M. D. Juergens, "Weekly water quality data from the River Thames and its major tributaries (2009–2017)," 2020.

[26] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Representations*, 2015.

A Technical Appendices and Supplementary Material

A.1 Implementation Details

All experiments were run on Google Colab GPUs, with the more computationally demanding cases in Table 1 executed on Tesla T4 GPUs. We first describe the configurations of the proposed (P)DCN models, followed by the baseline architectures and training settings. The code to reproduce the results is available on GitHub.[1] .

(P)DCN Details

For all experiments we adopted a $L = 2$-layer DCN with 32 hidden units, selected through a grid search over the hyperparameter space. The DCN-T variant corresponds to the same architecture but with reversed edge directions, i.e., using transposed causal GSOs. Additionally, we evaluated reduced variants of DCN in which only a randomly chosen subset of the causal GSOs is used to build the convolutional filters. These models are denoted as DCN-$|\mathcal{U}|$, where $|\mathcal{U}|$ indicates the number of GSOs included.

For PDCN, the shared MLP is composed of a single hidden layer with 128 units. In this parallel scheme, each shifted version of the input signal is passed through the shared MLP, and the outputs are aggregated, ensuring that the parameter count remains independent of the graph size while preserving expressive capacity.

Baseline Methods

We next summarize the baseline models used in the experimental comparisons presented in Section 5.

GCN [3]. The Graph Convolutional Network applies first-order graph convolutions based on a degree-normalized adjacency matrix augmented with self-loops. Each layer aggregates features from neighbors, followed by a linear transformation and nonlinearity. While effective, GCNs are prone to oversmoothing. In our setup, the model has $L = 2$ layers with hidden dimension 32.

FB-GCNN [19]. The Filter Bank GCN is a spatial convolutional model that applies a filter bank of parameters to different powers of the graph shift operator $\mathbf{A}$, thereby combining multi-scale neighborhood information. Unless otherwise noted, we use a filter order $R = 2$. The parameter vector dimension is set to 32, with a depth of $L = 2$ layers. Variants with different R are denoted as FB-GCNN-R.

GraphSAGE [23]. GraphSAGE learns inductive embeddings by sampling and aggregating features from fixed-size neighborhoods. Different aggregation functions can be used, including mean, LSTM, or pooling; here we employ mean aggregation. Our implementation uses $L = 2$ layers, each with hidden dimension 32.

GIN [4]. The Graph Isomorphism Network was designed to match the discriminative power of the Weisfeiler–Lehman test by using injective sum aggregation. Each GIN layer aggregates neighbor features via summation and processes them with a learnable MLP. We implement the standard GIN with $L = 2$ layers, each with 32 hidden units.

[1] `https://github.com/reysam93/dag_conv_nn/`

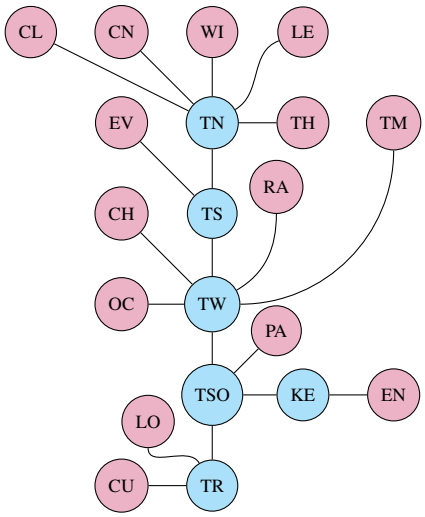


Figure 3: A DAG illustrating the flow of the River Thames and the positions of monitoring sites along its course. Purple nodes mark source locations, while cyan nodes denote intermediate or sink sites. Acronyms for the sites are defined in [25].

GAT [5]. The Graph Attention Network employs self-attention to adaptively weigh neighbor contributions during feature aggregation. Each layer applies multiple attention heads to capture diverse relational patterns. In our experiments, GAT uses $L = 2$ layers, each with 16 hidden units and 2 attention heads.

DAGNN [17]. The Directed Acyclic Graph Neural Network modifies message passing to respect the topological order of a DAG. It uses an attention mechanism to weight inputs from parent nodes, and a gated recurrent unit (GRU) to sequentially integrate this information. Our implementation employs $L = 2$ layers with hidden dimension 128 and dropout rate 0.2.

D-VAE [16]. The DAG Variational Autoencoder encodes DAGs into continuous latent spaces by recursively aggregating node and predecessor information via GRUs, and subsequently reconstructing the DAG from the latent code. For our comparisons, we only use the encoder component of D-VAE. The hidden layer size is set to 128.

MLP [24]. As a graph-agnostic baseline, the Multilayer Perceptron treats each node independently without considering graph connectivity. It consists of two fully connected layers with 32 units each. This baseline allows us to isolate the benefits of incorporating graph structure into the models.

Training Hyperparameters

We detail here the hyperparameter choices used for training the models reported in Section 5.

Learning rate. Candidate learning rates were explored in the range 5×10^{-4} to 5×10^{-3}. A rate of 5×10^{-4} was selected for the River Thames forecasting and diffusion learning tasks, whereas source identification was trained with 5×10^{-3}.

Batch size. All experiments employed a batch size of 25.

Number of epochs. In all cases, training was run for 100 epochs.

Weight decay. A weight decay coefficient of 10^{-4} was used uniformly across all tasks.

Optimizer. All models were optimized using Adam [26].

B River Thames structure

The Fig. 3 demonstrates the structure of River Thames as we used in experiments.

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: The first contribution is fully explained in Section 4, and the experimental results provided in Section 5 demonstrate the second contribution.

   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: Limitations are discussed in Section 4 and 6.

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: No theoretical results are provided or proved in this paper.

   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: The contribution is a new architecture which is fully described, supported by experiments and reproducible. All the experimental setup is explained in 5 and A.1. The code is available on GitHub.[2]

[2] `https://github.com/reysam93/dag_conv_nn/`

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: [Yes]

   Justification: The code is available on GitHub.[3]

   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.

[3] `https://github.com/reysam93/dag_conv_nn/`.

- 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: Please see section 5 and section A.1.

   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: Please see section 5.

   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: Please see section 5 and section A.1.

   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: To the best of our knowledge, our research conforms to the NeurIPS Code of Ethics in every aspect.

   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: [Yes]

    Justification: This paper introduces a new neural architecture that can be used for hydrological data forecasting as showed in section 5, so the positive societal impacts are mentioned. We don't see any negative societal impact of the work.

    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: This paper proposes a new architecture but does not release pretrained models, large-scale generative systems, or scraped datasets that could pose risks of misuse. To the best of our knowledge, the proposed architecture does not carry foreseeable risks of misuse.

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: [Yes]

Justification: The experiments are conducted on a publicly available dataset [25], which has been properly cited. No licensed data have been used.

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: [Yes]

Justification: The proposed architectures are thoroughly explained and documented; however, the code is not included with this submission in order to comply with the double-blind review process. The code will be released publicly upon acceptance.

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:

    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:

    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:

    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.