Convolutional Learning on Directed Acyclic Graphs

Samuel Rey*, Hamed Ajorlou† and Gonzalo Mateos†
*Dept. of Signal Theory and Communications, Rey Juan Carlos University, Madrid, Spain
Email: samuel.rey.escudero@urjc.es.
†Dept. of Electrical and Computer Engineering, University of Rochester, NY, USA.
Emails: {gmateosb, hajorlou}@ur.rochester.edu

***Abstract*—We develop a novel convolutional architecture tailored for learning from data defined over directed acyclic graphs (DAGs). DAGs can be used to model causal relationships among variables, but their nilpotent adjacency matrices pose unique challenges towards developing DAG signal processing and machine learning tools. To address this limitation, we harness recent advances offering alternative definitions of causal shifts and convolutions for signals on DAGs. We develop a novel convolutional graph neural network that integrates learnable DAG filters to account for the partial ordering induced by the graph topology, thus providing valuable inductive bias to learn effective representations of DAG-supported data. We discuss the salient advantages and potential limitations of the proposed DAG convolutional network (DCN) and evaluate its performance on two learning tasks using synthetic data: network diffusion estimation and source identification. DCN compares favorably relative to several baselines, showcasing its promising potential.**

***Index Terms*—DAG, graph signal processing, graph neural networks, graph convolution.**

I. Introduction

The vast complexity and heterogeneity of modern networks are propelling the generation of data represented on non-Euclidean domains. Consequently, fields such as graph signal processing (GSP) have emerged, which model the underlying irregular signal structure as a graph and then exploit its topology to process said relational data [1], [2]. Graph neural networks (GNN) are non-linear, graph-aware machine learning (ML) models that serve as a prominent example of this new paradigm [3]–[6]. GNNs exploit prior information in the graph via convolutions [7]–[9], attention mechanisms [10], or graph autoencoders [11], [12], and have attained state-of-the-art performance in a gamut of ML applications.

Arguably most GNNs and graph-based methods focus on undirected graphs. However, accounting for directionality can play an important role in processing information, but this leap comes with well-documented challenges [13]–[15] that are exacerbated when dealing with a directed acyclic graph (DAG). For instance, DAGs have nilpotent adjacency matrices and hence a collapsed spectrum that prevents us from directly applying spectral-based tools [16]. These difficulties notwithstanding, DAGs are prevalent across domains including causal inference [17], learning Bayesian networks and structural causal models [18]–[21], code parsing [22], or performance prediction for neural architectures [23], just to name a few.

This work was supported in part by the Spanish AEI under Grants PID2019-105032GBI00, TED2021-130347B-I00 and PID2022-136887NBI00 funded by MCIN/AEI/10.13039/501100011033.

This work develops a novel *convolutional* GNN tailored for learning from signals defined over DAGs. As DAGs impose a partial ordering on the set of nodes, a key challenge is to incorporate this stronger inductive bias into our architecture in a principled manner. To that end, we build on a recent framework that extends GSP tools to partially ordered sets (posets) [16], [24], to arrive at a DAG convolutional network (DCN). These foundational design principles can be traced to linear structural equation models (SEMs), making DCN well-suited to capture causal relations in the data. Moreover, the resulting architecture admits a spectral representation and is permutation equivariant. Unlike related alternatives [25], [26], the convolutional layers primarily involve sparse matrix multiplications, resulting in manageable computational complexity.

Related work and contributions in context. Developing GNNs to process data defined over DAGs is an important problem that is starting to draw attention. Recently, [25] advocated embedding DAGs into a smooth latent space and introduced a variational autoencoder for DAGs, dubbed D-VAE. The DAG neural network (DAGNN) from [26] sequentially aggregates the node representations from predecessors via an attention mechanism, and then uses a gated recurrent unit (GRU) to encode the causal structure of the DAG during a combination step. However, the sequential scheme followed by both D-VAE and DAGNN results in a large computational burden, limiting their applicability to moderately-sized graphs. Furthermore, DAGNN needs to combine the true and reverse directions of the DAG in a non-intuitive way to process the data. Later, [27] proposed encoding the DAG ordering via a reachability-based attention mechanism, extending graph transformers to DAGs.

All in all, the main contributions of this work are as follows:

- We propose DCN, the first convolutional GNN designed to learn from data defined over DAGs. We elaborate on the key benefits of the model and discuss its limitations.
- Different from existing approaches [25], [26], DCN layers rely on formal convolutions derived in [16] for a signal model over posets, that can be linked to linear SEMs.
- We evaluate the performance of DCN and show it compares favorably to several baselines in different graph learning tasks. We share the code to reproduce our results.

II. Preliminaries and problem statement

We first define basic concepts and notation about DAGs, as well as the required background on GSP and GNNs. We then formally state the problem of learning from DAG signals.

A. Preliminaries: DAGs, graph signals and GNNs

DAGs and graph signals. Let $\mathcal{D} = (\mathcal{V}, \mathcal{E})$ be a directed acyclic graph (DAG), where $\mathcal{V}$ denotes the set of N nodes and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ denotes the set of edges. An edge $(i, j) \in \mathcal{E}$ exists if and only if there is a link from node j to node i. Since $\mathcal{D}$ contains no cycles, we can sort $\mathcal{V}$ topologically, ensuring that node j precedes node i whenever $(i, j) \in \mathcal{E}$. Consequently, every DAG induces a unique partial order on $\mathcal{V}$, where $j < i$ if j is a *predecessor* of i, i.e., if there exists a path from j to i [24]. Note that in posets not all the elements are comparable. In fact, for every pair $i, j \in \mathcal{V}$ without a path from i to j or vice-versa, we have that $i \not\leq j$ and $j \not\leq i$.

Let $\mathbf{A} \in \mathbb{R}^{N \times N}$ denote the (possibly weighted) adjacency matrix of $\mathcal{D}$, where $A_{ij} \neq 0$ if and only if $(i, j) \in \mathcal{E}$. Due to the aforementioned partial ordering of $\mathcal{V}$, we can sort the entries of $\mathbf{A}$ to obtain a strictly lower-triangular matrix. The diagonal is null because DAGs cannot have self-loops.

In addition to the graph $\mathcal{D}$, we consider signals defined on the set of nodes $\mathcal{V}$. Formally, graph signals are functions from the vertex set to the real field $x : \mathcal{V} \mapsto \mathbb{R}$, which can be represented as a vector $\mathbf{x} \in \mathbb{R}^N$, where x_i is the signal value at node i. A signal processing theory can be constructed for graph signals supported on DAGs (a.k.a. DAG signals), which was put forth in [16] and we review in Section III-A.

Convolutional GNNs. A GNN can be represented as a parametric non-linear function that depends on the underlying graph structure. While several GNN models have been proposed, most of them build upon an *aggregation function* driven by the graph topology. In this work we consider a class of GNNs where the aggregation is accomplished through graph convolutions [7], [9]. More precisely, we are interested in GNNs employing a bank of learnable convolutional graph filters, where the output of layer $\ell = 1, \ldots, L$ is given by

$$\mathbf{X}^{(\ell+1)} = \sigma\left(\sum_{r=0}^{R-1} \mathbf{A}^r \mathbf{X}^{(\ell)} \mathbf{\Theta}_r^{(\ell)}\right). \tag{1}$$

Here, $\mathbf{\Theta}_r^{(\ell)} \in \mathbb{R}^{F_i^{(\ell)} \times F_o^{(\ell)}}$ collects the learnable filter coefficients, $F_i^{(\ell)}$ and $F_o^{(\ell)}$ are the number of input and output features of the ℓ-th layer, and the powers $\mathbf{A}^r$ determine the radius of the aggregation neighborhood. If necessary, the adjacency matrix could be substituted with any desired *graph-shift operator* (GSO) that encodes the graph structure; see e.g., [1], [9]. The point-wise nonlinear activation function is often chosen to be a ReLU, i.e., $\sigma(x) = \max(0, x)$.

B. Problem statement

This work addresses the problem of learning from signals defined over a DAG $\mathcal{D}$. We are given $\mathcal{T} = \{\mathbf{X}_m, \mathbf{y}_m\}_{m=1}^M$, the training set containing M input-output observations from a network process on $\mathcal{D}$. The input graph signals are denoted as $\mathbf{X}_m \in \mathbb{R}^{N \times F}$ and the corresponding outputs as $\mathbf{y}_m \in \mathbb{R}^N$. Here outputs are single-feature graph signals $\mathbf{y}_m$ for simplicity, and generalizations to multi-feature signals or nodal/graph labels are straightforward. We use $\mathcal{T}$ to learn the map relating $\mathbf{X}_m$ and $\mathbf{y}_m$, which is assumed to be accurately represented by a non-linear parametric function $f_{\mathbf{\Theta}}(\cdot|\mathcal{D}) : \mathbb{R}^{N \times F} \mapsto \mathbb{R}^N$. To that end, we estimate the weights $\mathbf{\Theta}$ of $f_{\mathbf{\Theta}}(\cdot|\mathcal{D})$ by solving

$$\min_{\mathbf{\Theta}} \frac{1}{M} \sum_{m=1}^{M} \mathcal{L}(\mathbf{y}_m, f_{\mathbf{\Theta}}(\mathbf{X}_m|\mathcal{D})), \tag{2}$$

where $\mathcal{L}$ is a loss function, e.g., mean squared error (MSE) for regression or the cross entropy loss for classification.

III. Convolutional learning on DAGs

Among the many graph-aware functions available, we confine our selection of $f_{\mathbf{\Theta}}(\cdot|\mathcal{D})$ to the class of graph convolutional neural networks (GCNN); see e.g., [7], [9] for extensive discussion and justification on their merits. The first step towards designing a GCNN tailored to DAG signals is to define the convolution operation. Naively, one could use the graph filters in (1), as it is customary when graphs have cycles [9], [28]. However, for a DAG all the eigenvalues of $\mathbf{A}$ are 0, due to its strict lower triangular structure. This collapsed spectrum deprives us of a spectral interpretation for such filters. Moreover, recall that DAGs impose a partial ordering and can potentially capture causal relations, two properties we wish to encode in our GCNN. To overcome these challenges and design $f_{\mathbf{\Theta}}(\cdot|\mathcal{D})$ in a principled manner (Section III-B), we bring to bear the novel DAG signal processing framework introduced in [16], which we briefly outline next.

A. Graph-shift operators and convolution for DAGs

Let $\mathbf{c} \in \mathbb{R}^N$ be a vector collecting the causes or nodal contributions of a DAG signal $\mathbf{x}$. The signal model introduced in [16] postulates that $\mathbf{x}$ can be described by the causes at predecessor nodes. Specifically, we write $\mathbf{x} = \mathbf{W}\mathbf{c}$, where $\mathbf{W} \in \mathbb{R}^{N \times N}$ is the *weighted transitive closure* of $\mathcal{D}$ with $W_{ij} \neq 0$ if there is a path from j to i (i.e., if $j < i$) or $i = j$. Note that $\mathbf{W}$ can be interpreted as the adjacency matrix of the reachability graph of $\mathcal{D}$, with ones on the diagonal to account for the reflexivity property of posets. Among the different methods to compute $\mathbf{W}$ from $\mathbf{A}$ [29], we focus on $\mathbf{W} = (\mathbf{I} - \mathbf{A})^{-1}$. This is equivalent to a linear SEM for $\mathbf{x}$, with exogenous inputs $\mathbf{c}$ [16, Section IV-D].

Given this model, every node $k \in \mathcal{V}$ induces a causal GSO on the signal $\mathbf{x}$ given by the matrix $\mathbf{T}_k \in \mathbb{R}^{N \times N}$, such that

$$[\mathbf{T}_k \mathbf{x}]_i = \sum_{j \leq i \text{ and } j \leq k} W_{ij} c_j. \tag{3}$$

Intuitively, the signal value at node i after shifting $\mathbf{x}$ with respect to node k results in a signal containing only causes from common predecessors to both i and k. This can be compactly written in matrix form as

$$\mathbf{T}_k \mathbf{x} = \mathbf{W}\mathbf{D}_k \mathbf{c} = \mathbf{W}\mathbf{D}_k \mathbf{W}^{-1} \mathbf{x}, \tag{4}$$

where $\mathbf{D}_k$ is a diagonal matrix whose entry $[\mathbf{D}_k]_{ii} = 1$ if $i \leq k$ (i.e., if node i is a predecessor of node k) and 0 otherwise. The inverse $\mathbf{W}^{-1}$ always exists since $\mathbf{W}$ is a full-rank matrix and it can be efficiently computed using the weighted Moebius inversion [30]. It follows that the columns of $\mathbf{W}$ and the

diagonal entries of $\mathbf{D}_k$ are, respectively, the eigenvectors and eigenvalues of the GSO $\mathbf{T}_k$. Matrix $\mathbf{W}$ is thus a Fourier basis for DAG signals, while $\mathbf{W}^{-1}$ is the corresponding Fourier transform. Because $\mathbf{x} = \mathbf{W}\mathbf{c}$, for this DAG signal model the causes $\mathbf{c}$ are also spectral coefficients.

In GSP, a graph filter is a polynomial of the (single) GSO. In contrast, now we have a GSO $\mathbf{T}_k$ for every node in $\mathcal{V}$ and, moreover, every $\mathbf{T}_k$ is an idempotent operator. Consequently, the most general shift-invariant DAG filter $\mathbf{H}$ is given by

$$\mathbf{H} = \sum_{k\in\mathcal{V}} h_k \mathbf{T}_k = \mathbf{W} \sum_{k\in\mathcal{V}} h_k \mathbf{D}_k \mathbf{W}^{-1}, \tag{5}$$

resulting in a convolution operation $\mathbf{h} * \mathbf{x} = \mathbf{H}\mathbf{x}$. Here, $\mathbf{h} \in \mathbb{R}^N$ collects the filter coefficients h_k, and the frequency response of $\mathbf{H}$ is given by the diagonal of $\sum_{k\in\mathcal{V}} h_k \mathbf{D}_k$. This definition of DAG filter plays a central role in our convolutional architecture, the subject dealt with next.

B. DAG Convolutional Network (DCN)

Now, we are ready to introduce our DAG-aware convolutional network, which we refer to as DCN. The proposed DCN harnesses the definition of causal graph filters in (5) to implement convolutions, which we compose with a point-wise nonlinearity to process DAG signals. In the simplest case, the output of layer $\ell = 1, \ldots, L$ can be computed as

$$\mathbf{x}^{(\ell+1)} = \sigma\left(\sum_{k\in\mathcal{V}} h_k^{(\ell)} \mathbf{T}_k \mathbf{x}^{(\ell)}\right), \tag{6}$$

where $h_k^{(\ell)}$ are the learnable filter coefficients.

We refer to (6) as a DAG perceptron, and we can gain intuition from two different standpoints. In the spectral domain, recall that $\mathbf{T}_k\mathbf{x}^{(\ell)} = \mathbf{W}\mathbf{D}_k\mathbf{c}^{(\ell)}$. In words, the k-th GSO selects the causes (i.e., the exogenous input in an SEM) of predecessors of the node k, and diffuses them across the reachability graph via $\mathbf{W}$. Clearly, this accounts for the partial ordering imposed by the DAG. Alternatively, one could look at (6) in the context of message-passing networks. For every node $i \in \mathcal{V}$, each $\mathbf{T}_k$ forms a message combining the features from predecessors common to i and k, to finally yield $x_i^{(\ell+1)}$ as a non-linear weighted combination of these messages. A complete example and schematic illustrating these layer operations will be included in the extended journal version.

While (6) is intuitive, a single filter may not be expressive enough to learn complex DAG signal representations. Furthermore, input signals typically comprise several features. We address these limitations following a rationale similar to that of (1). At each layer, the single filter is replaced with a bank of learnable DAG filters, resulting in the recursion [cf. (1)]

$$\mathbf{X}^{(\ell+1)} = \sigma\left(\sum_{k\in\mathcal{V}} \mathbf{T}_k \mathbf{X}^{(\ell)} \mathbf{\Theta}_k^{(\ell)}\right). \tag{7}$$

Here, the matrices $\mathbf{\Theta}_k^{(\ell)} \in \mathbb{R}^{F_i \times F_o}$ collect the learnable filter coefficients with F_i and F_o denoting the number of input and output features. Then, for a DCN with L layers, the prediction $\hat{\mathbf{y}}_m = \mathbf{X}^{(L)} = f_{\mathbf{\Theta}}(\mathbf{X}_m|\mathcal{D})$ will be the output of the final layer.

	Network Diffusion		**Source Identification**	
	MNSE	Time (s)	Accuracy	Time (s)
DCN	$\mathbf{0.016 \pm 0.014}$	3.6	0.052 ± 0.014	7.5
DCN-30	$\mathbf{0.029 \pm 0.017}$	3.5	0.052 ± 0.016	7.4
DCN-10	0.058 ± 0.021	3.5	0.055 ± 0.015	7.2
DCN-T	0.098 ± 0.024	4.1	$\mathbf{0.991 \pm 0.018}$	8.2
DCN-30-T	0.199 ± 0.030	3.7	$\mathbf{0.983 \pm 0.032}$	7.64
DCN-10-T	0.229 ± 0.030	3.5	0.865 ± 0.141	7.38
LS	0.050 ± 0.022	0.4	0.05 ± 0.016	0.36
FB-GCNN	0.091 ± 0.028	3.4	0.739 ± 0.172	7.4
GCN	0.167 ± 0.037	3.3	0.155 ± 0.216	7.1
GAT	0.649 ± 0.089	13.8	0.044 ± 0.081	28.4
GraphSAGE	0.359 ± 0.039	5.9	0.676 ± 0.163	12.5
GIN	0.402 ± 0.079	6.0	0.19 ± 0.163	12.5
MLP	0.353 ± 0.039	2.2	0.050 ± 0.016	4.7

TABLE I: Normalized MSE and accuracy when respectively solving the network diffusion learning and source identification tasks. We report the mean performance and standard deviation across 25 realizations, along with the training time.

Remark 1 *The DCN whose output is given by the recursion* (7) *is permutation equivariant. Due to space limitations, the proof is deferred to the extended journal version of this work.*

Discussion. The DCN convolutional layer (7) is closely related to GCNN architectures for general (cyclic) graphs. However, relying on causal graph filters is imperative to implement formal convolutions, and offers some key advantages. First, the spectrum of $\mathbf{T}_k$ is well defined, endowing the DCN with a spectral representation that is fundamental to analyze properties such as stability, transferability, or denoising capability [9], [31], an interesting future research direction. Furthermore, since the eigenvalues of $\mathbf{T}_k$ are the binary matrices $\mathbf{D}_k$, no issues stemming from numerical instability are expected when concatenating several layers.

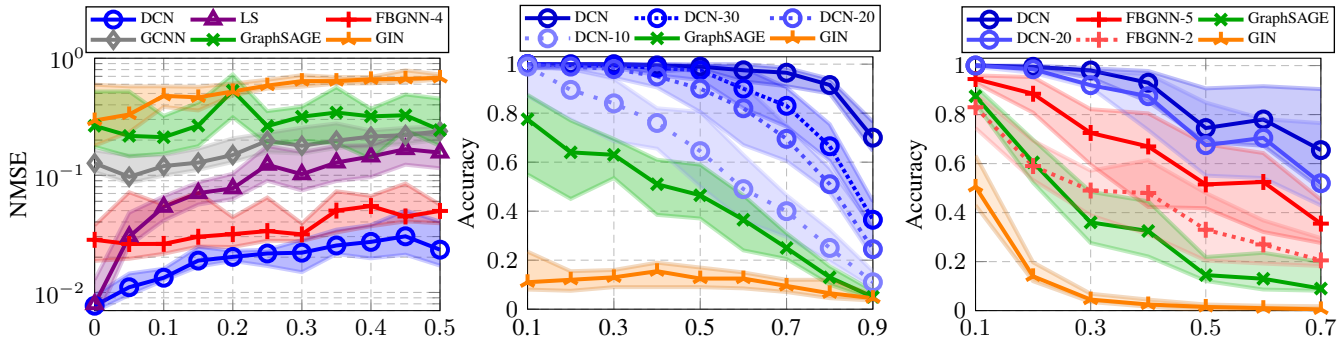
On the flip side, the number of GSOs potentially involved in the convolution – hence the number of learnable parameters – grows with the size of the graph. This may lead to computational and memory limitations. As an engineering workaround, one can approximate and simplify the convolution in (7) as $\sum_{k\in\mathcal{U}} h_k \mathbf{T}_k$, where $\mathcal{U} \subset \mathcal{V}$. In principle, this comes at the expense of reducing the expressiveness of the DCN. Still, this limitation can be alleviated by concatenating several layers since the cross-product of different $\mathbf{T}_k$ and $\mathbf{T}_l$ can give rise to a new GSO. As we illustrate in the upcoming numerical evaluation, in practice one can randomly select a small number of matrices $\mathbf{T}_k$ to define the convolution and maintain a competitive performance. Developing a principled approach to determine the subset of nodes $\mathcal{U}$ constitutes an interesting problem that is considered a future research direction.

IV. Numerical Evaluation

We assess the performance of DCN in different settings and compare it with several baselines. Code to reproduce all results is available on GitHub[1], and the interested reader is referred there for additional experiments and implementation details.

Experiment setup. The numerical evaluation is conducted over synthetic data. Unless specified otherwise, graphs are

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



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

Fig. 1: (a) reports the MNSE in the network diffusion task as the noise in the observations increases; For the source identification task, (b) and (c) depict the influence of increasing respectively the proportion of unobserved nodes and the edge probability. We report the median performance across 25 realizations and values between the first and third quartile in the shaded area.

sampled from an Erdős-Rényi random DAG model with $N = 100$ nodes and an edge probability $p = 0.2$. The input-output signals are related as $\mathbf{y} = \mathbf{H}\mathbf{x}$. Here, $\mathbf{x}$ is a sparse input signal whose non-zero entries are restricted to the first nodes, $\mathbf{H}$ is a DAG filter as in (5) composed of 25 causal GSOs selected uniformly at random, and $\mathbf{y}$ is the output of the diffusion of $\mathbf{x}$ across the DAG. We create $M = 2000$ input-output pairs of DAG signals and split them as 70% for training, 20% for validation, and 10% for testing.

We consider two learning problems. First, we address the task of learning a *network diffusion* process where, given a new sparse input $\mathbf{x}_{test}$, we aim at estimating the associated output $\mathbf{y}_{test}$. The observed input and output signals are corrupted with zero-mean additive white Gaussian noise with a normalized power of 0.05. This task amounts to solving a regression problem where we consider the MSE as the loss function $\mathcal{L}$, and report the normalized MSE (NMSE), computed as $\frac{1}{T}\sum_{t=1}^{T} \|\mathbf{y}_t - \hat{\mathbf{y}}_t\|_2^2 / \|\mathbf{y}_t\|_2^2$ over T test signals. The second task tackles the *source identification* problem, where we are given a partially observed output signal $\mathbf{y}_{test}$ and the goal is to predict which node was the source originating the observation. Now, the input $\mathbf{x}$ contains a single non-zero entry constrained to be located within the first 20 nodes. Since the diffusion $\mathbf{y} = \mathbf{H}\mathbf{x}$ does not alter the signal value at the source nodes, we mask the output $\mathbf{y}$ so observations from possibly seeding nodes are not available. The task is cast as a classification problem where the graph label is the index of the source node. We adopt the cross-entropy loss and report the mean accuracy. The reported results comprise 25 independent realizations.

Baselines. The performance of the DCN is compared with several baselines. The least-squares (LS) model captures the true linear model, computes an LS estimate of $\mathbf{h}$ from the training data and then predicts test outputs. As non-linear baselines, we include FB-CGNN, the convolutional GNN based on graph filters from [9] [cf. (1)], GCN [7], GAT [10], GIN [8], GraphSAGE [32], and the MLP. We do not include D-VAE [25] and DAGNN [26] here (although they are GNNs for DAG-based data), because their training time is prohibitive for moderately large graphs (as here where $N = 100$).

Preliminary results. We start by assessing the overall performance of the models considered in the network diffusion and the source identification problems. The results collected in Table I showcase the superior performance of the proposed DCN in both tasks. In the network diffusion setting, we observe how using a subset of 30 or 10 GSOs randomly selected (DCN-30 and DCN-10) decouples the DCN from the number of nodes without significantly hindering the performance; DCN-30 is the second-best alternative. Comparing DCN and LS, the superior performance of DCN is due to a major resilience to the noise in the inputs, as will be evident in the upcoming experiments. As expected, when using the transposed matrices $\mathbf{T}_k^\top$ in DCN-T (i.e., when following the reverse ordering of the DAG), the performance drops significantly, especially when using only a subset of the GSOs. In contrast, moving on to the source identification task we observe that while the DCN variants are incapable of identifying the source node, the alternatives DCN-T have an almost perfect accuracy. Intuitively, this reflects how for identifying the source nodes from the output one needs to navigate the DAGs in the reverse order as facilitated by the transposed GSOs. In conclusion, this highlights the importance of properly harnessing the directionality of the DAG. Note that for the baselines we use the GSO with the best performance.

Test case 1. The results depicted in Fig. 1a reflect how increasing the power of the noise in the observed signals impacts the performance of the network diffusion problem. In the absence of noise, the LS model outperforms the alternatives, but its performance deteriorates swiftly as the noise power increases. In contrast, the proposed DCN attains a similar NMSE to that of the LS in the noiseless setting and its performance is more robust and in the presence of noise. This experiment showcases that i) the proposed DCN obtains a performance comparable to that of the optimal solution (LS); and ii) assuming a non-linear model even though the true generative model is linear provides a solution more robust to noise.

Test Case 2. Moving on to the source identification problem, Fig. 1b evaluates the accuracy of the DCN when a different number of GSOs is employed as the proportion of unobserved

nodes increases. Based on the results of Table I, all the methods use the transposed of their respective GSO. It is worth mentioning that higher values on the x-axis render a more challenging setting since: i) fewer nodal observations are available; and ii) every unobserved node is allowed to be a source node, hence increasing the complexity of the classification problem. First, we note that the accuracy of DCN remains high even for large fractions of unobserved nodes, demonstrating its robustness. From the performance of DCN-30, DCN-20, and DCN-10, the trade-off between reducing the number of GSOs and maintaining a high accuracy and low variability is evident. Nonetheless, DCN-30 shows a similar performance to that of DCN up to observing 50% of the nodes, and DCN-10 outperforms the non-DAG-based alternatives.

Test Case 3. Finally, we evaluate the impact of the density of the graph on the source identification problem. To that end, we measure the accuracy as the edge probability increases (Fig. 1c), where we observe that a denser graph poses a greater challenge to identify the source. Intuitively, as the graph gets closer to a fully connected DAG, the signal generated from different nodes becomes more homogeneous, with the discrepancies being due only to the distance between seeding nodes. In addition, this experiment shows that both DCN and DCN-20 consistently outperform FBGNN using filters of order 2 and 5, emphasizing the benefits stemming from replacing GSP filters with DAG filters.

V. Conclusions

In this work, we proposed DCN, a GCNN tailored to the particular structure encoded by DAGs. At its core lies a novel convolutional layer, which is built upon recent results to process signals defined over DAGs and posets, has ties to linear SEMs, is permutation equivariant, endows the architecture with a spectral representation, and exploits the partial order imposed by the DAG. Moreover, our preliminary numerical evaluation demonstrated a promising performance and showed that the convolution operation can be detached from the number of nodes, thus addressing a potential limitation.

References

[1] 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.

[2] X. Dong, D. Thanou, L. Toni, M. Bronstein, and P. Frossard, "Graph signal processing for machine learning: A review and new perspectives," *IEEE Signal Process. Mag.*, vol. 37, no. 6, pp. 117–127, 2020.

[3] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst, "Geometric deep learning: Going beyond euclidean data," *IEEE Signal Process. Mag.*, vol. 34, no. 4, pp. 18–42, July 2017.

[4] S. Rey, A. G. Marques, and S. Segarra, "An underparametrized decoder architecture for graph signals," in *IEEE Intl. Wrksp. Computat. Advances Multi-Sensor Adaptive Process. (CAMSAP)*. IEEE, 2019, pp. 231–235.

[5] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip, "A comprehensive survey on graph neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, 2020.

[6] V. M. Tenorio, S. Rey, F. Gama, S. Segarra, and A. G. Marques, "A robust alternative for graph convolutional neural networks via graph neighborhood filters," in *Conf. Signals, Syst., Computers (Asilomar)*. IEEE, 2021, pp. 1573–1578.

[7] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[8] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?," *arXiv preprint arXiv:1810.00826*, 2018.

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

[10] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *Intl. Conf. Learn. Repr. (ICLR)*, 2018.

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

[12] 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 *European Signal Process. Conf. (EUSIPCO)*. IEEE, 2021, pp. 855–859.

[13] R. Shafipour, A. Khodabakhsh, G. Mateos, and E. Nikolova, "Digraph Fourier transform via spectral dispersion minimization," in *IEEE Int. Conf. Acoustics, Speech Signal Process. (ICASSP)*, 2018, pp. 6284–6288.

[14] R. Shafipour, S. Segarra, A. G. Marques, and G. Mateos, "Directed network topology inference via graph filter identification," in *IEEE Data Science Wrksp. (DSW)*, 2018, pp. 210–214.

[15] 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," *IEEE Signal Process. Mag.*, vol. 37, no. 6, pp. 99–116, 2020.

[16] 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.

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

[18] J. Peters and P. Bühlmann, "Identifiability of Gaussian structural equation models with equal error variances," *Biometrika*, vol. 101, no. 1, pp. 219–228, 2014.

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

[20] D'Acunto G., Di Lorenzo P., and Barbarossa S., "Multiscale causal structure learning," *Trans. Mach. Learning Res.*, 2023.

[21] S. S. Saboksayr, G. Mateos, and M. Tepper, "CoLiDE: Concomitant linear DAG estimation," in *Intl. Conf. Learn. Repr. (ICLR)*, 2024.

[22] 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.

[23] C. Zhang, M. Ren, and R. Urtasun, "Graph hypernetworks for neural architecture search," *arXiv preprint arXiv:1810.05749*, 2018.

[24] M. Püschel, B. Seifert, and C. Wendler, "Discrete signal processing on meet/join lattices," *IEEE Trans. Signal Process.*, vol. 69, pp. 3571–3584, 2021.

[25] M. Zhang, S. Jiang, Z. Cui, R. Garnett, and Y. Chen, "D-VAE: A variational autoencoder for directed acyclic graphs," in *Advances Neural Inf. Process. Syst.*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. 2019, vol. 32, Curran Associates, Inc.

[26] V. Thost and J. Chen, "Directed acyclic graph neural networks," in *Intl. Conf. Learn. Repr. (ICLR)*, 2021.

[27] Y. Luo, V. Thost, and L. Shi, "Transformers over directed acyclic graphs," in *Advances Neural Inf. Process. Syst.*, A. Oh, T. Neumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, Eds. 2023, vol. 36, pp. 47764–47782, Curran Associates, Inc.

[28] S. Segarra, A. G. Marques, and A. Ribeiro, "Optimal graph-filter design and applications to distributed linear network operators," *IEEE Trans. Signal Process.*, vol. 65, no. 15, pp. 4117–4131, 2017.

[29] Daniel J Lehmann, "Algebraic structures for transitive closure," *Theoretical Comput. Sci.*, vol. 4, no. 1, pp. 59–76, 1977.

[30] G.-C. Rota, "On the foundations of combinatorial theory: I. Theory of Möbius functions," in *Classic Papers Combinatorics*, pp. 332–360. Springer, 1964.

[31] S. Rey, S. Segarra, R. Heckel, and A. G. Marques, "Untrained graph neural networks for denoising," *IEEE Trans. Signal Process.*, vol. 70, pp. 5708–5723, 2022.

[32] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," *Advances Neural Inf. Process. Syst.*, vol. 30, 2017.