PANDA: Prior-guided Attentional Dual-path Architecture

Rishabh Jain
University of Cambridge, UK
rj412@cst.cam.ac.uk

Federico Belotti
University of Bergamo, Italy
federico.belotti@unibg.it

Stefano Coniglio
University of Bergamo, Italy
stefano.coniglio@unibg.it

Pietro Liò
University of Cambridge, UK
pl219@cst.cam.ac.uk

Stefano Fiorini
Independent Researcher
s.fiorini1994@gmail.com

Abstract

Hypergraph Neural Networks (HNNs), designed to model higher-order relations beyond pairwise interactions, typically rely either on fixed structural weights or on normalised attention. Fixed weights can propagate misleading information in heterophilic settings, while normalisation makes attention coefficients competitive, in the sense that, in a message-passing framework, increasing the weight of one sender must reduce the weight of others in the same aggregation set. Our main contribution is the introduction of PANDA (Prior-guided AttentioNal Dual-path Architecture), a plug-in for two-stage message passing HNNs applicable to both undirected and directed settings. The *main path* interpolates between learned receiver-normalised attention and structural incidence priors, while a non-competitive (not normalised) *auxiliary path* lets each sender contribute its own transformed information. We also show that, with fixed structural coefficients, PANDA recovers a certain complex-valued convolution operator adopted in previous works. Across four HNNs spanning spatial and spectral methods in both directed and undirected settings, plugging PANDA in yields an average improvement of 5.18 percentage points across eight real-world benchmark datasets. The code is available at: `https://github.com/jainris/panda`.

1 Introduction

Learning from structured, non-Euclidean data has been largely driven by Graph Neural Networks (GNNs), which propagate and aggregate features along pairwise edges [Scarselli et al., 2009, Asif et al., 2021]. Many real-world systems, however, involve higher-order relations that cannot be faithfully represented by pairwise links alone, including social networks [Benson et al., 2016, 2018], biological systems [Traversa et al., 2023], protein interactions [Murgas et al., 2022], and multi-robot systems [Jain et al., 2026]. Hypergraph Neural Networks (HNNs) address this limitation by modelling hyperedges as multi-node relations, enabling message passing over sets of interacting entities [Murgas et al., 2022, Chen et al., 2022].

A large class of HNNs can be understood as a two-stage node→hyperedge→node message passing in which features are first aggregated within each hyperedge and then redistributed to its member nodes [Telyatnikov et al., 2023]. Despite this common structure, many incidence-aggregation HNNs retain a smoothing bias inherited from fixed structural propagation. Fixed coefficients, such as degree normalisations or incidence weights, provide stable architecture-specific priors, but they can also propagate misleading information when a hyperedge connects nodes with different labels, roles, or feature distributions. This issue is especially relevant in heterophilic hypergraphs, where the same

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

higher-order relation may contain endpoints that should not share information in the same feature space [Zhu et al., 2020, Chen et al., 2022].

Attention mechanisms provide one way to adapt aggregation to the data. GAT [Veličković et al., 2017] showed that learned neighbour weights can improve node classification, with GATv2 [Brody et al., 2022] later addressing expressivity limitations of the scoring function. Hypergraph attention methods such as HCHA [Bai et al., 2021] and HyperGAT [Ding et al., 2020] extend this idea to node–hyperedge interactions, yet on standard benchmarks their gains over non-attentive baselines remain modest and often inconsistent across datasets, suggesting that adapting attention to the hypergraph setting is more subtle than a direct port from the pairwise case. A central reason is that these methods rely on receiver-wise normalisation, which forces attention coefficients to be *competitive*: increasing the weight on one sender necessarily reduces the mass available to other senders in the same aggregation set [Richter and Wattenhofer, 2020]. This is reasonable when neighbours genuinely compete to explain a target, but it precludes assigning high relevance to multiple senders simultaneously, an outcome that is often desirable in higher-order relations [Yan et al., 2025].

Directed hypergraphs add another layer of structure. In this setting, each hyperedge has head and tail roles, and aggregation should distinguish how information flows through these roles. Yet, most directed hypergraph operators still rely on fixed structural coefficients to weigh head and tail contributions [Ma et al., 2024, Fiorini et al., 2024]. The result is a propagation rule whose directional pattern is locked at architecture-design time, independent of the features being propagated. A further open question concerns the spectral operator of Fiorini et al. [2024]: Does it admit an equivalent two-stage message-passing realisation?

To address these issues, we introduce PANDA (Prior-guided AttentioNal Dual-path Architecture), a plug-in for two-stage hypergraph message-passing layers. Our approach applies to architectures whose propagation exposes node-to-hyperedge and hyperedge-to-node message passing, or to incidence aggregations that can accept node–hyperedge weights. At each aggregation stage, PANDA combines a main path, consisting of a structural prior interpolated with learned receiver-normalised scores, with a non-competitive, gated auxiliary path that lets each incidence contribute transformed information without requiring the auxiliary scores to sum to one across the receiver neighbourhood. In the directed setting, PANDA further factorises its attention along head/tail roles, so that the same plug-in applies to both undirected and directed HNNs. Additionally, focusing on GeDi-HNN [Fiorini et al., 2024], we show that it admits a two-stage message–passing realisation over complex-valued representations and that this realisation is not equivalent to transforming the directed hypergraph to a graph via a directed star expansion. We evaluate the integration of PANDA into four hypergraph architectures spanning spatial and spectral methods in both directed and undirected settings across eight real-world datasets. Across all models, PANDA consistently improves performance, yielding an average gain of 5.18 percentage points and improvements of up to 31.01 points on Telegram [Sen et al., 2008].

Our contributions can be summarised as follows:

- We introduce PANDA, a plug-in for two-stage message-passing architectures, combining a main path that interpolates per-incidence between receiver-normalised attention and the backbone's structural prior with a non-competitive gated auxiliary path. We frame it on directed hypergraphs as they subsume undirected hypergraphs and ordinary graphs as special cases; the mechanism is not hypergraph-specific (we demonstrate its effectiveness on a GCN backbone as well).
- We prove that GeDi-HNN's complex Hermitian convolution admits a two-stage node–hyperedge–node realisation through the complex incidence matrix and its conjugate (Theorem 4.1), and that this realisation is not recovered by applying the same operator after a directed star expansion, which differs in phase (Theorem 4.2).
- Across 4 backbones spanning spectral/spatial and directed/undirected architectures on eight benchmarks, PANDA yields an average gain of +5.18 pp, with gains of up to +31.01 pp.
- We show that non-competitiveness is what drives the gains: against a normalised auxiliary path with identical parameter count and identical separate transformation, PANDA wins on 27 of 32 model-dataset pairs (+2.60 pp average).
- We provide a mechanistic analysis of the learned routing, separating the auxiliary path's contribution (train-time ablation, Table 1) from the trained model's dependence on it (inference-

time interventions), and show its predictive effect is a broad margin improvement rather than only neighbour-majority correction.

2 Related Work

Undirected and Directed Hypergraph Neural Networks. On the *undirected* side, HNNs split into spectral and spatial families. Spectral methods build convolution operators from the eigendecomposition of a hypergraph Laplacian, drawing on graph signal processing and the graph Fourier transform; examples include HGNN [Feng et al., 2019], HCHA [Bai et al., 2021] and HGNN$^+$ [Gao et al., 2022], all of which can be viewed as GNNs operating on the clique expansion of a hypergraph. Spatial methods instead frame convolution as a localised aggregation over hyperedge neighbourhoods. HNHN [Dong et al., 2020], UniGCNII [Huang and Yang, 2021], HyperDN [Tudisco et al., 2021], and LEGCN [Yang et al., 2022], together with set-based models such as AllDeepSets and AllSetTransformer [Chien et al., 2022], propagate hyperedge features within a message-passing framework and can also be interpreted as GNNs on the star expansion. PhenomNN [Wang et al., 2023b] derives its architecture from hypergraph-regularised energy functions, combining both the clique and star expansions.

On the *directed* side, work is comparatively sparse. Fiorini et al. [2023] propose SigMaNet for directed graphs. DHRL [Ma et al., 2024] approximates a directed hypergraph Laplacian and defines convolutions on top of it, while GeDi-HNN [Fiorini et al., 2024] introduces a graph-convolution operator based on the Generalized Directed Laplacian, a complex-valued Hermitian matrix shown to subsume many existing directed and undirected (hyper)graph convolutions as special cases.

Attention in GNNs and HNNs. GAT [Veličković et al., 2017] first showed that learned similarity-based weighting of neighbour contributions improves node classification, and GATv2 [Brody et al., 2022] later introduced a strictly more expressive variant. In the HNN domain, early proposals such as HCHA [Bai et al., 2021] and HyperGAT [Ding et al., 2020] incorporate hyperedge-level attention but remain undirected and adopt a purely competitive, receiver-wise normalisation: increasing the weight on one sender necessarily reduces the mass available to others in the same receiver neighbourhood.

Gated Attention. Gated attention mechanisms augment standard attention with an additional learned gate that modulates the information passed by an attention head, feature channel, message, or residual path. While attention weights select where to aggregate from, gates control how much of the resulting signal should be admitted. This idea follows the broader use of gates for adaptive information flow in deep networks [Srivastava et al., 2015, Dauphin et al., 2017, Shazeer, 2020], and has been instantiated in attention models at different granularities: over attention heads, over value/output features, or over residual paths [Parisotto et al., 2020, Hua et al., 2022, Yang et al., 2024]. In graph learning, GaAN introduced gates over multi-head graph attention, using a lightweight subnetwork to control the importance of each attention head [Zhang et al., 2018]. More recently, Qiu et al. [2025] showed that a sigmoid gate applied after attention improves performance, training stability, and scaling behaviour, while inducing sparse modulation of the attention output and enhancing long-context extrapolation. Notably, all these existing methods apply gating in conjunction with competitive normalisation (usually after it), unlike our proposed work. Furthermore, these works do not explore applications to hypergraphs.

3 PANDA: Prior-guided AttentioNal Dual-path Architecture

Let $\mathcal{H} = (\mathcal{V}, \mathcal{E})$ be a directed hypergraph with node features $\mathbf{x}_u^{(\ell)} \in \mathbb{R}^d$ for node $u \in \mathcal{V}$ and hyperedge features $\mathbf{h}_e \in \mathbb{R}^d$, $e \in \mathcal{E}$. The hyperedge weights are stored in the diagonal matrix $W \in \mathbb{R}^{m \times m}$. The vertex and hyperedge degrees are defined as $d_u = \sum_{e \in \mathcal{E}: u \in e} |w_e|$ for $u \in \mathcal{V}$ and $\delta_e = |e|$ for $e \in \mathcal{E}$ and are stored in two diagonal matrices $D_v \in \mathbb{R}^{n \times n}$ and $D_e \in \mathbb{R}^{m \times m}$. Following Gallo et al. [1993], the direction of a hyperpedge $e \in \mathcal{E}$ is determined by the partition in tail set $T(e) \subseteq \mathcal{V}$ and head set $H(e) \subseteq \mathcal{V}$. For each $u \in \mathcal{V}$, $\Gamma_H(u)$ and $\Gamma_T(u)$ denote the sets of hyperedges in which u appears as a head or tail node, respectively, while $\Gamma(u)$ denotes the set of all hyperedges incident to u, i.e. $\Gamma(u) = \Gamma_H(u) \cup \Gamma_T(u)$. For undirected hypergraphs, we use the same notation by placing every incident node in the head set: $H(e) = e$ and $T(e) = \emptyset$, so $\Gamma_H(u)$ contains all hyperedges incident to u and $\Gamma_T(u) = \emptyset$.

The following framework describes a general *two-stage* node–hyperedge–node message-passing abstraction [Chien et al., 2022, Telyatnikov et al., 2023]:

$$\mathbf{h}_e^{(\ell)} = \mathrm{Agg}_{\mathcal{E}}^{(\ell)}\left(\left\{\mathrm{Trans}_{\mathcal{V}}^{(\ell)}\left(\mathbf{x}_v^{(\ell-1)}\right)\right\}_{v\in e}\right) \qquad e \in \mathcal{E} \tag{1}$$

$$\mathbf{x}_u'^{(\ell)} = \mathrm{Agg}_{\mathcal{V}}^{(\ell)}\left(\left\{\mathrm{Trans}_{\mathcal{E}}^{(\ell)}\left(\mathbf{h}_e^{(\ell)}\right)\right\}_{e\in\Gamma(u)}\right) \qquad u \in \mathcal{V} \tag{2}$$

$$\mathbf{x}_u^{(\ell)} = \mathrm{Upd}_{\mathcal{V}}^{(\ell)}\left(\mathbf{x}_u^{(\ell-1)}, \mathbf{x}_u'^{(\ell)}\right) \qquad u \in \mathcal{V}. \tag{3}$$

Equation 1 carries out the first stage of message passing, aggregating transformed embeddings of the relevant nodes, while Equations 2 and 3 carry out the second stage, updating the node embeddings using the aggregated hyperedge representations. $\mathrm{Trans}_{\cdot}^{(\ell)}$, $\mathrm{Agg}_{\cdot}^{(\ell)}$, and $\mathrm{Upd}_{\cdot}^{(\ell)}$ are learnable functions; their specific instantiation defines different model architectures. For instance, $\mathrm{Agg}_{\mathcal{E}}$ and $\mathrm{Agg}_{\mathcal{V}}$ may be weighted sums over incident nodes and hyperedges, as in GeDi-HNN Fiorini et al. [2024]; learned permutation-invariant multiset functions, as in AllSet and AllSetTransformer [Chien et al., 2022]; or attention-weighted sums, as in hypergraph attention models such as HCHA and HyperGAT [Bai et al., 2021, Ding et al., 2020].

PANDA is an attention plug-in that can be applied to virtually any two-stage message passing architecture by intercepting the node-to-hyperedge and hyperedge-to-node incidence messages used in Equations (1) and (2), and manipulating them. It consists of two paths: the main path interpolates between learned receiver-normalised attention and structural incidence priors, while an auxiliary path adds locally complement-bounded but non-normalised transformed messages. We describe them in detail in the following.

3.1 PANDA's Main Path

After intercepting the node-to-hyperedge and hyperedge-to-node incidence messages used in Equations (1) and (2), PANDA's main path re-weights them by the main-path attention coefficients $\hat{\alpha}_{e,v}^{\mathrm{main},(\ell)}$ and $\check{\alpha}_{u,e}^{\mathrm{main},(\ell)}$. The updated equations for the target layer reweighted by the new attention scores read:

$$\mathbf{h}_e^{(\ell)} = \mathrm{Agg}_{\mathcal{E}}^{(\ell)}\left(\left\{\hat{\alpha}_{e,v}^{\mathrm{main},(\ell)}\mathbf{x}_v^{\mathrm{main},(\ell)}\right\}_{v\in e}\right) \qquad e \in \mathcal{E} \tag{4}$$

$$\mathbf{x}_u'^{(\ell)} = \mathrm{Agg}_{\mathcal{V}}^{(\ell)}\left(\left\{\check{\alpha}_{u,e}^{\mathrm{main},(\ell)}\mathbf{h}_e^{\mathrm{main},(\ell)}\right\}_{e\in\Gamma(u)}\right) \qquad u \in \mathcal{V}, \tag{5}$$

where we define $\mathbf{x}_v^{\mathrm{main},(\ell)}$ and $\mathbf{h}_e^{\mathrm{main},(\ell)}$ as below for better readability:

$$\mathbf{x}_v^{\mathrm{main},(\ell)} = \mathrm{Trans}_{\mathcal{V}}^{(\ell)}\left(\mathbf{x}_v^{(\ell-1)}\right) \qquad v \in \mathcal{V} \tag{6}$$

$$\mathbf{h}_e^{\mathrm{main},(\ell)} = \mathrm{Trans}_{\mathcal{E}}^{(\ell)}\left(\mathbf{h}_e^{(\ell)}\right) \qquad e \in \mathcal{E}. \tag{7}$$

PANDA's main-path attention mechanism extends GATv2's [Brody et al., 2022] by combining GATv2's coefficients $\hat{\alpha}_{e,v}^{\mathrm{gat},(\ell)}$ and $\check{\alpha}_{u,e}^{\mathrm{gat},(\ell)}$ with the structural prior coefficients $\hat{\alpha}_{e,v}^{\mathrm{prior}}$ and $\check{\alpha}_{u,e}^{\mathrm{prior}}$. These are the fixed coefficients used by the target architecture in the node-to-hyperedge message from v to e and the hyperedge-to-node message from e to u. These coefficients may come from degree normalisation, hyperedge weights, incidence signs or phases, or model-specific hyperedge reduction scores. Setting the β coefficients to the prior-only regime, $\hat{\beta}_{e,v}^{(\ell)} = \check{\beta}_{u,e}^{(\ell)} = 0$, recovers these fixed coefficients in the main routing path.

Following Brody et al. [2022], the scores $\hat{\alpha}_{e,v}^{\mathrm{gat},(\ell)}$ and $\check{\alpha}_{u,e}^{\mathrm{gat},(\ell)}$ are defined as follows:

$$\hat{\alpha}_{e,v}^{\mathrm{gat},(\ell)} = \frac{\exp(\hat{s}_{e,v})}{\sum_{v'\in H(e)\cup T(e)}\exp(\hat{s}_{e,v'})} \quad \hat{s}_{e,v} = (\hat{\mathbf{a}}^{(\ell)})^{\top}\,\mathrm{LeakyReLU}\left(\hat{\mathbf{W}}^{(\ell)}\left[\mathbf{h}_e^{(\ell-1)} \,\|\, \mathbf{x}_v^{(\ell-1)}\right]\right) \tag{8}$$

$$\check{\alpha}_{u,e}^{\mathrm{gat},(\ell)} = \frac{\exp(\check{s}_{u,e})}{\sum_{e'\in\Gamma_H(u)\cup\Gamma_T(u)}\exp(\check{s}_{u,e'})} \quad \check{s}_{u,e} = (\check{\mathbf{a}}^{(\ell)})^{\top}\,\mathrm{LeakyReLU}\left(\check{\mathbf{W}}^{(\ell)}\left[\mathbf{x}_u^{(\ell-1)} \,\|\, \mathbf{h}_e^{(\ell)}\right]\right), \tag{9}$$

with learnable parameters $\hat{\mathbf{a}}^{(\ell)}, \check{\mathbf{a}}^{(\ell)} \in \mathbb{R}^d$, $\hat{\mathbf{W}}^{(\ell)}, \check{\mathbf{W}}^{(\ell)} \in \mathbb{R}^{d \times 2d}$ and $\|$ is the concatenation operation.

In PANDA's main path, the coefficients $\hat{\alpha}^{\mathrm{gat},(\ell)}_{e,v}, \check{\alpha}^{\mathrm{gat},(\ell)}_{u,e}$ and $\hat{\alpha}^{\mathrm{prior}}_{e,v}, \check{\alpha}^{\mathrm{prior}}_{u,e}$ are combined via the following per-representation convex combination with $\hat{\beta}^{(\ell)}_{e,v}, \check{\beta}^{(\ell)}_{u,e} \in [0,1]$:

$$\hat{\alpha}^{\mathrm{main},(\ell)}_{e,v} = \hat{\beta}^{(\ell)}_{e,v}\, \hat{\alpha}^{\mathrm{gat},(\ell)}_{e,v} + \left(1 - \hat{\beta}^{(\ell)}_{e,v}\right) \hat{\alpha}^{\mathrm{prior}}_{e,v}, \tag{10}$$

$$\check{\alpha}^{\mathrm{main},(\ell)}_{u,e} = \check{\beta}^{(\ell)}_{u,e}\, \check{\alpha}^{\mathrm{gat},(\ell)}_{u,e} + \left(1 - \check{\beta}^{(\ell)}_{u,e}\right) \check{\alpha}^{\mathrm{prior}}_{u,e}. \tag{11}$$

Notice that, since the interpolation gates $\hat{\beta}^{(\ell)}_{e,v}, \check{\beta}^{(\ell)}_{u,e}$ are used incidence-wise, the resulting $\hat{\alpha}^{\mathrm{main},(\ell)}_{e,v}, \check{\alpha}^{\mathrm{main},(\ell)}_{u,e}$ coefficients are not necessarily receiver-wise normalised, even when the GAT coefficients and the chosen structural prior are each separately normalised over the receiver neighbourhood. The learned gating coefficients $\hat{\beta}^{(\ell)}_{e,v}, \check{\beta}^{(\ell)}_{u,e} \in [0,1]$ are calculated as follows:

$$\hat{\beta}^{(\ell)}_{e,v} = \sigma_s\left(\left(\mathbf{u}^{(\ell)}_{\beta,\mathcal{V}}\right)^\top \mathrm{LeakyReLU}\left(W^{(\ell)}_{\beta,\mathcal{V}}\left[\mathbf{h}^{(\ell-1)}_e \,\|\, \mathbf{x}^{(\ell-1)}_v\right]\right)\right) \tag{12}$$

$$\check{\beta}^{(\ell)}_{u,e} = \sigma_s\left(\left(\mathbf{u}^{(\ell)}_{\beta,\mathcal{E}}\right)^\top \mathrm{LeakyReLU}\left(W^{(\ell)}_{\beta,\mathcal{E}}\left[\mathbf{x}^{(\ell-1)}_u \,\|\, \mathbf{h}^{(\ell)}_e\right]\right)\right), \tag{13}$$

with learnable $\mathbf{W}^{(\ell)}_{\beta,\cdot} \in \mathbb{R}^{c \times 2d}$ and $\mathbf{u}^{(\ell)}_{\beta,\cdot} \in \mathbb{R}^c$, and where $\sigma_s(\cdot)$ is the sigmoid function. Setting $\hat{\beta}^{(\ell)}_{e,v} = \check{\beta}^{(\ell)}_{u,e} = 1$ recovers the GATv2 attentional coefficients, while setting $\hat{\beta}^{(\ell)}_{e,v} = \check{\beta}^{(\ell)}_{u,e} = 0$ recovers the structural prior coefficients (in the main routing path).

Thanks to the β coefficients, the attention mechanism can shift towards the structural prior on representations where it is reliable (e.g., homophilic neighbourhoods in which smoothing is desirable and, thus, the prior is beneficial), while retaining the learned GATv2 scores where the prior is misleading (e.g., heterophilic settings).

3.2 Auxiliary Path

PANDA adds an *auxiliary aggregation mechanism* to the one already featured in the target architecture (after reweighting it with the coefficients $\hat{\alpha}^{(\ell)}_{e,v}$ and $\check{\alpha}^{(\ell)}_{u,e}$). We introduce this auxiliary aggregation mechanism by carrying out the following transformations

$$\mathbf{x}^{\mathrm{aux},(\ell)}_v = \mathrm{Trans}^{\mathrm{aux},(\ell)}_{\mathcal{V}}\left(\mathbf{x}^{(\ell-1)}_v\right) \qquad v \in \mathcal{V} \tag{14}$$

$$\mathbf{h}^{\mathrm{aux},(\ell)}_e = \mathrm{Trans}^{\mathrm{aux},(\ell)}_{\mathcal{E}}\left(\mathbf{h}^{(\ell-1)}_e\right) \qquad e \in \mathcal{E}. \tag{15}$$

We update the aggregation operations of Equations (4)–(5) by additively combining the transformed representations $\mathbf{h}^{\mathrm{aux},(\ell)}_e$ and $\mathbf{x}^{\mathrm{aux},(\ell)}_v$ to the original, reweighted ones:

$$\mathbf{h}^{(\ell)}_e = \mathrm{Agg}^{(\ell)}_{\mathcal{E}}\left(\left\{\hat{\alpha}^{\mathrm{main},(\ell)}_{e,v}\mathbf{x}^{\mathrm{main},(\ell)}_v\right\}_{v\in e} \oplus \left\{\hat{\alpha}^{\mathrm{aux},(\ell)}_{e,v}\mathbf{x}^{\mathrm{aux},(\ell)}_v\right\}_{v\in e}\right) \qquad e \in \mathcal{E} \tag{16}$$

$$\mathbf{x}'^{(\ell)}_u = \mathrm{Agg}^{(\ell)}_{\mathcal{V}}\left(\left\{\check{\alpha}^{\mathrm{main},(\ell)}_{u,e}\mathbf{h}^{\mathrm{main},(\ell)}_e\right\}_{e\in\Gamma(u)} \oplus \left\{\check{\alpha}^{\mathrm{aux},(\ell)}_{u,e}\mathrm{h}^{\mathrm{aux},(\ell)}_e\right\}_{e\in\Gamma(u)}\right) \qquad u \in \mathcal{V}, \tag{17}$$

where $\oplus$ indicates the per-index sum of two indexed sets that share the same indices.

As weights, we use a gated complement of the main path scores:

$$\hat{\alpha}^{\mathrm{aux},(\ell)}_{e,v} = \left(1 - \hat{\alpha}^{\mathrm{main},(\ell)}_{e,v}\right)\hat{g}^{(\ell)}_{e,v}, \qquad \hat{g}^{(\ell)}_{e,v} := \sigma_s\left(\mathrm{aux}^{(\ell)}_{\mathcal{E}}\left(\mathbf{h}^{(\ell-1)}_e, \mathbf{x}^{(\ell-1)}_v\right)\right) \tag{18}$$

$$\check{\alpha}^{\mathrm{aux},(\ell)}_{u,e} = \left(1 - \check{\alpha}^{\mathrm{main},(\ell)}_{u,e}\right)\check{g}^{(\ell)}_{u,e}, \qquad \check{g}^{(\ell)}_{u,e} := \sigma_s\left(\mathrm{aux}^{(\ell)}_{\mathcal{V}}\left(\mathbf{x}^{(\ell-1)}_u, \mathbf{h}^{(\ell)}_e\right)\right), \tag{19}$$

where $\sigma_s(\cdot)$ is the sigmoid function, the per-incidence values $\hat{g}^{(\ell)}_{e,v} \in [0,1]$ and $\check{g}^{(\ell)}_{u,e} \in [0,1]$ are the post-sigmoid auxiliary gates, and the functions $\mathrm{aux}^{(\ell)}_{\mathcal{E}}, \mathrm{aux}^{(\ell)}_{\mathcal{V}}$ are learned scalar scoring functions:

$$\mathrm{aux}^{(\ell)}_{\mathcal{E}}\left(\mathbf{h}^{(\ell-1)}_e, \mathbf{x}^{(\ell-1)}_v\right) = (\mathbf{a}^{(\ell)}_{\hat{\alpha}^{\mathrm{aux}}})^\top \mathrm{ReLU}\left(\mathbf{W}^{(\ell)}_{\hat{\alpha}^{\mathrm{aux}}}\left[\mathbf{h}^{(\ell-1)}_e \,\|\, \mathbf{x}^{(\ell-1)}_v\right]\right) \tag{20}$$

$$\mathrm{aux}^{(\ell)}_{\mathcal{V}}\left(\mathbf{x}^{(\ell-1)}_u, \mathbf{h}^{(\ell)}_e\right) = (\mathbf{a}^{(\ell)}_{\check{\alpha}^{\mathrm{aux}}})^\top \mathrm{ReLU}\left(\mathbf{W}^{(\ell)}_{\check{\alpha}^{\mathrm{aux}}}\left[\mathbf{x}^{(\ell-1)}_u \,\|\, \mathbf{h}^{(\ell)}_e\right]\right). \tag{21}$$

with learned $\mathbf{W}^{(\ell)}_{\hat{\alpha}^{\text{aux}}}, \mathbf{W}^{(\ell)}_{\check{\alpha}^{\text{aux}}} \in \mathbb{R}^{d\times 2d}$ and $\mathbf{a}^{(\ell)}_{\hat{\alpha}^{\text{aux}}}, \mathbf{a}^{(\ell)}_{\check{\alpha}^{\text{aux}}} \in \mathbb{R}^{d}$.

Assume that the structural priors are scaled so that the resulting main coefficients lie in $[0, 1]$. Then the auxiliary gates satisfy

$$0 \le \hat{\alpha}^{\text{aux},(\ell)}_{e,v} \le 1 - \hat{\alpha}^{\text{main},(\ell)}_{e,v}, \qquad 0 \le \check{\alpha}^{\text{aux},(\ell)}_{u,e} \le 1 - \check{\alpha}^{\text{main},(\ell)}_{u,e}.$$

Setting the auxiliary coefficients, or the auxiliary transformations, to zero recovers the prior-guided main-path reweighting of the target layer.

Notice that these are sender–receiver-wise bounds: $1 - \alpha^{\text{main}}$ is the local complement of the corresponding sender–receiver coefficient. In particular, the auxiliary coefficients are not normalised across $v \in e$ or across $e \in \Gamma(u)$. The main path blends the receiver-normalised GATv2 component with the structural prior, while the auxiliary path adds a separately transformed message whose coefficient is locally complement-bounded but non-competitive across the receiver's receptive field. Thus, interactions assigned a small main-path coefficient can still contribute after transformation, which is useful in heterophilic or role-sensitive hyperedges where a sender may be poor as a smoothing neighbour but informative after a separate transformation.

4 Generalised Directed Spectral Convolution as Two-Stage Message Passing

In this section, we analyse the relationship between two-stage message-passing and the only, to our knowledge, Hermitian convolutional operators defined for directed hypergraphs, namely, the operator used in GeDi-HNN and introduced by Fiorini et al. [2024]. The analysis is crucial to understand how to plug PANDA in such an architecture.

The relation between spectral graph convolutions and spatial/message-passing graph convolutions is well established for ordinary graphs, see [Balcilar et al., 2020, 2021, Chen et al., 2023]. For hypergraphs, the two-stage node–hyperedge–node message-passing abstraction is also standard in the undirected setting. HGNN and HCHA re-cast the Zhou–Huang–Schölkopf operator as a node–hyperedge–node message-passing scheme, and [Zhou et al., 2006, Feng et al., 2019, Bai et al., 2021] and, more broadly, AllSet formalises HNN layers as compositions of permutation-invariant multiset functions [Chien et al., 2022], while Telyatnikov et al. [2023] place HGNN, HCHA, HNHN, HyperGCN, UniGNN, and AllSet-type models under the same two-stage viewpoint. What is not immediate from these existing results is the directed, complex-Hermitian case considered by models such as GeDi-HNN [Fiorini et al., 2024].

In this section, we show that, to recover GeDi's convolutional operator, one needs a complex-valued message-passing scheme relying on the same complex incidence matrix and its conjugate used to define GeDi's Laplacian. We also show that the directed, complex-valued GeDi operator implemented by GeDi's Laplacian admits a node–hyperedge–node two-stage equivalent, but not by (as one may intuit) first converting the hypergraph into an ordinary directed star expansion and then applying a Hermitian graph operator.

4.1 A complex-valued message-passing scheme

To compare our two-stage template with GeDi's spectral convolution, we instantiate the two aggregation functions $\text{Agg}^{(\ell)}_{\mathcal{E}}$, $\text{Agg}^{(\ell)}_{\mathcal{V}}$ as follows:

$$\mathbf{h}^{(\ell)}_e = \sum_{v\in H(e)\cup T(e)} \vec{B}^{*}_{e,v} \left(\hat{\alpha}^{\text{main},(\ell)}_{e,v}\, \mathbf{x}^{\text{main},(\ell)}_v + \hat{\alpha}^{\text{aux},(\ell)}_{e,v}\, \mathbf{x}^{\text{aux},(\ell)}_v \right) \qquad e \in \mathcal{E} \tag{22}$$

$$\mathbf{x}'^{(\ell)}_u = \sum_{e\in\Gamma_H(u)\cup\Gamma_T(u)} \vec{B}_{u,e} \left(\check{\alpha}^{\text{main},(\ell)}_{u,e}\, \mathbf{h}^{\text{main},(\ell)}_e + \check{\alpha}^{\text{aux},(\ell+1)}_{u,e}\, \mathbf{h}^{\text{aux},(\ell)}_e \right) \qquad u \in \mathcal{V}, \tag{23}$$

where $\vec{B}^{*} \in \mathbb{C}^{n\times m}$ is the conjugate-transpose of $\vec{B} \in \mathbb{C}^{n\times m}$, which is Fiorini et al. [2024]'s complex vertex–edge incidence matrix

$$\vec{B}_{v,e} = \begin{cases} 1 & \text{if } v \in H(e) \\ -i & \text{if } v \in T(e) \\ 0 & \text{otherwise} \end{cases} \qquad v \in \mathcal{V},\ e \in \mathcal{E}. \tag{24}$$

The role of $\vec{B}$ is to encode the hyperedge directions and is central to the following theorem:

Theorem 4.1. *PANDA with additive aggregations* $\mathrm{Agg}_{\cdot}$, $\mathrm{Trans}_{\cdot}^{(\ell)} = \mathrm{id}$, $\mathrm{Trans}_{\cdot}^{\mathrm{aux},(\ell)} = \mathbf{0}$, $\mathrm{Upd}^{(\ell)}(X, X') = \phi(X\Theta_0 + X'\Theta_1)$, *with* $\hat{\beta}_{e,v}^{(\ell)} = \check{\beta}_{u,e}^{(\ell)} = 0$ *(so that* $\hat{\alpha}_{e,v}^{\mathrm{main},(\ell)} = \hat{\alpha}_{e,v}^{\mathrm{prior}}$ *and* $\check{\alpha}_{u,e}^{\mathrm{main},(\ell)} = \check{\alpha}_{u,e}^{\mathrm{prior}}$*),* $\hat{\alpha}_{e,v}^{\mathrm{aux},(\ell)} = \check{\alpha}_{u,e}^{\mathrm{aux},(\ell)} = 0$*, and where the structural priors are*

$$\hat{\alpha}_{e,v}^{\mathrm{prior}} = \sqrt{\frac{w_e}{\delta_e\, d_v}}, \qquad \check{\alpha}_{u,e}^{\mathrm{prior}} = \sqrt{\frac{w_e}{d_u\, \delta_e}}.$$

recovers GeDi's convolutional layer defined on top of the Generalized Directed (GeDi) Laplacian $\vec{L}_N = I - \vec{Q}_N$*, with* $\vec{Q}_N := D_v^{-1/2}\, \vec{B}\, W\, D_e^{-1}\, \vec{B}^*\, D_v^{-1/2}$.

Notice that, when the coefficients are learned, PANDA does not necessarily preserve the Hermitianicity of the spectral operator which otherwise holds in the frozen-prior case.

In the undirected hypergraph case, a spectral convolution based on the Zhou–Huang–Schölkopf Laplacian can be recovered as a two-stage message-passing operator applied to the star expansion of the hyperpgraph. This is not the case in the directed setting, where the phases encoded in the Hermitian matrix do not line up, as we now show:

Theorem 4.2. *Let* $\mathcal{G}_\star$ *be the directed star expansion obtained from a directed hypergraph* $\mathcal{H}$ *by replacing each hyperedge* e *with an auxiliary node* h_e*, adding arcs* $t \to h_e$ *for every* $t \in T(e)$ *and arcs* $h_e \to r$ *for every* $r \in H(e)$*. Let* $\vec{Q}_\star$ *and* $\vec{L}_\star$ *be the GeDi Signless and Signed Laplacian operators on this directed graph, and let* $\vec{Q}_N$ *be the GeDi normalised Signless Laplacian on* $\mathcal{H}$*. The block* $(\vec{Q}_\star^2)_{\mathcal{V},\mathcal{V}}$ *is the two-hop node–auxiliary–node propagation obtained by applying GeDi twice on the star-expanded graph. In general, however, neither the one-step operator* $\vec{Q}_\star$ *nor this two-hop star-expanded propagation equals* $\vec{Q}_N$ *on the original vertices. Thus even the two-hop message passing induced by the ordinary star expansion already differs from* $\vec{Q}_N$ *in phase.*

5 Experimental Evaluation

5.1 Experimental Setup

Baseline. We evaluate our PANDA modular plug-in applied to four HNNs on real-world datasets for the node classification task. The selected models span complementary regions of the spectral/spatial and directed/undirected design space. From the *undirected* hypergraph-learning literature, we include the spectral method HGNN [Feng et al., 2019] and the spatial methods HyperGCN [Yadati et al., 2019] and PhenomNN [Wang et al., 2023b]. From the *directed* hypergraph-learning literature, we consider the spectral method GeDi-HNN [Fiorini et al., 2024]. For each HNN, we compare its original formulation (VANILLA) against the same backbone augmented with PANDA, isolating the benefit of integrating our plug-in without changing the underlying architecture.

Datasets. For evaluation, we use eight real-world datasets and follow the pre-processing procedure introduced by Tran and Tran [2022] and Fiorini et al. [2024] to obtain (directed) hypergraph datasets for node classification. The considered datasets include: `Telegram` [Bovet and Grindrod, 2020], `Citeseer`, `PubMed` [Sen et al., 2008], `Cornell`, `Texas` [Pei et al., 2020], `Squirrel`, `Roman-empire` [Platonov et al., 2023a] and `email-Enron` [Benson et al., 2018]. Model performance is measured in terms of classification accuracy. Following the standard practice in the literature [Chien et al., 2022, Wang et al., 2023a, Fiorini et al., 2024], we adopt a 50%/25%/25% split for training, validation, and testing, respectively, and, for each model, we report the average test accuracy and the standard deviation over 10 independent runs. Details on the baselines, hyperparameter tuning, and the experimental setup are provided in Appendix B.

Integrating PANDA into existing architectures. PANDA is designed as a modular plug-in that can be integrated into different backbones. We detail each integration below, starting from GeDi-HNN, which follows the two-stage realisation derived in Section 4.1.

HGNN [Feng et al., 2019] can be written as a degree-normalised node→hyperedge→node propagation through the incidence matrix, so we apply the main and auxiliary paths directly to the two incidence aggregations: nodes contribute to hyperedges through the $\hat{\alpha}_{e,v}^{\mathrm{main},\ell}$ and $\hat{\alpha}_{e,v}^{\mathrm{aux},\ell}$ coefficients, and hyperedges return information to nodes through the $\check{\alpha}_{e,v}^{\mathrm{main},\ell}$ and $\check{\alpha}_{e,v}^{\mathrm{aux},\ell}$ coefficients, while the

original degree and hyperedge-weight normalisations are retained inside the HGNN aggregation. Note that, since PANDA W/O PRIOR,AUX is simply the backbone architecture, but with the vanilla aggregation replaced by an attentional one, PANDA W/O PRIOR,AUX on the HGNN backbones has an identical architecture to HCHA w/ attention [Bai et al., 2021].

HyperGCN [Yadati et al., 2019] replaces each hyperedge by a sparse graph built from representative node pairs and mediators before applying a GCN. We keep the hyperedge-to-graph reduction fixed and apply PANDA to the messages on the selected support: the main path re-weights the selected messages, and the auxiliary path adds parallel messages without altering the mediator construction.

Lastly, PhenomNN [Wang et al., 2023b] is derived from hypergraph-regularised energy functions rather than from a plain convolutional layer. It can be viewed as running message passing through the clique-expanded and star-expanded graphs, separately, and then applying energy function inspired-regularisations to form the new node embeddings. We apply the same rule to the hyperedge-local messages used by its iterative update, treating the original message as the main path and adding a separately transformed incidence-wise message as the auxiliary path before the model's original regularisation and update.

5.2 Results

Table 1: Node classification results reporting mean accuracies ± standard deviation across all datasets. We compare our PANDA plug-in and its ablations with different backbone architectures.

	Variant	Telegram	Citeseer	Pubmed	Roman-empire	Cornell	Texas	Squirrel	email-Enron
GeDi	VANILLA	80.96 ± 4.506	76.47 ± 1.315	89.80 ± 0.408	84.10 ± 0.399	74.32 ± 5.019	82.97 ± 2.717	42.35 ± 2.059	53.24 ± 7.649
	PANDA w/o Prior,Aux	73.27 ± 6.631	75.89 ± 0.973	89.68 ± 0.415	88.65 ± 0.753	78.65 ± 3.908	83.51 ± 4.432	38.62 ± 1.851	58.11 ± 7.376
	PANDA w/o Aux	80.00 ± 4.485	76.20 ± 0.768	90.04 ± 0.316	87.98 ± 0.758	76.22 ± 3.585	84.05 ± 6.216	39.62 ± 2.573	60.81 ± 9.145
	PANDA	91.15 ± 3.567	76.12 ± 0.921	89.32 ± 0.416	90.11 ± 0.699	77.03 ± 4.230	84.05 ± 4.595	42.76 ± 1.742	60.27 ± 8.466
	Δ(PANDA - Vanilla)	+10.19	-0.35	-0.48	+6.01	+2.71	+1.08	+0.41	+7.03
HGNN	VANILLA	58.08 ± 6.760	77.49 ± 1.226	90.08 ± 0.318	78.84 ± 0.650	71.35 ± 4.713	81.89 ± 5.135	39.08 ± 2.861	42.43 ± 4.366
	PANDA w/o Prior,Aux	58.08 ± 8.104	77.08 ± 1.394	89.88 ± 0.595	76.82 ± 0.405	73.78 ± 5.135	82.43 ± 4.872	38.01 ± 3.601	48.11 ± 5.771
	PANDA w/o Aux	71.73 ± 3.224	77.56 ± 1.070	90.02 ± 0.365	82.66 ± 0.871	72.97 ± 4.009	83.24 ± 4.490	39.68 ± 2.830	44.05 ± 6.954
	PANDA	72.12 ± 2.885	77.17 ± 1.125	89.77 ± 0.368	82.37 ± 0.511	74.59 ± 2.162	83.24 ± 5.644	40.43 ± 0.999	54.59 ± 8.090
	Δ(PANDA - Vanilla)	+14.04	-0.32	-0.21	+3.53	+3.24	+1.31	+1.35	+12.16
HyperGCN	VANILLA	61.92 ± 4.106	75.59 ± 0.776	88.94 ± 0.326	65.04 ± 0.702	57.57 ± 7.257	65.41 ± 6.373	38.28 ± 1.628	47.30 ± 8.653
	PANDA w/o Prior,Aux	62.50 ± 7.410	75.46 ± 0.732	87.84 ± 0.244	77.12 ± 0.486	67.03 ± 9.269	76.49 ± 8.967	34.51 ± 2.694	45.75 ± 2.936
	PANDA w/o Aux	68.65 ± 5.374	75.41 ± 0.753	88.68 ± 0.499	77.94 ± 0.419	67.03 ± 6.139	73.62 ± 6.397	34.67 ± 2.126	55.68 ± 7.852
	PANDA	70.58 ± 5.165	75.63 ± 0.894	88.59 ± 0.568	80.16 ± 0.661	64.86 ± 6.620	78.65 ± 5.976	42.46 ± 2.923	54.86 ± 6.402
	Δ(PANDA - Vanilla)	+8.66	+0.04	-0.25	+15.12	+7.29	+13.24	+4.18	+7.56
PhenomNN	VANILLA	44.81 ± 4.472	77.40 ± 1.105	90.24 ± 0.365	80.33 ± 0.423	74.05 ± 4.713	82.97 ± 4.196	37.63 ± 2.451	39.43 ± 3.148
	PANDA w/o Prior,Aux	52.12 ± 8.088	77.52 ± 1.112	90.10 ± 0.392	78.72 ± 0.492	76.22 ± 4.151	80.81 ± 6.886	38.28 ± 2.841	34.05 ± 8.470
	PANDA w/o Aux	75.58 ± 6.084	77.20 ± 1.075	87.87 ± 0.666	76.63 ± 0.376	75.14 ± 2.648	82.43 ± 4.230	38.49 ± 2.593	48.11 ± 6.817
	PANDA	75.82 ± 4.343	77.08 ± 1.113	87.51 ± 0.581	85.08 ± 0.752	75.68 ± 2.961	82.70 ± 5.565	41.01 ± 1.999	50.27 ± 6.304
	Δ(PANDA - Vanilla)	+31.01	-0.32	-2.73	+4.75	+1.63	-0.27	+3.38	+10.84

Table 1 reports node classification accuracy across eight benchmark datasets for PANDA integrated with four HNN models: GeDi-HNN, HGNN, HyperGCN, and PhenomNN. For each model, we compare the original formulation (VANILLA), two PANDA ablations, and the full PANDA module. Overall, PANDA achieves an average improvement of 5.18 percentage points over the corresponding VANILLA models. The gains concentrate on heterophilic benchmarks, while performance remains largely unchanged on homophilic citation datasets. The largest improvements occur on Telegram, where PANDA lifts all four models by near double-digit margins, reaching 31.01 percentage points on PhenomNN and exceeding 10 points on GeDi-HNN and HGNN. Heterophilic hypergraph datasets follow the same pattern: HyperGCN benefits the most, with gains of over 13 percentage points on both Roman-empire and Texas, while the other models show consistent improvements. On email-Enron, PANDA improves all four models by an average of 9.40 percentage points, suggesting that the gains are driven by the underlying interaction structure rather than by a single model family. On the homophilic citation datasets Citeseer and Pubmed, where the structural prior of the VANILLA models is already well aligned with the data, the difference between full PANDA and VANILLA remains within $[-2.73, +0.04]$ percentage points, with most changes below one point in absolute value. PANDA therefore acts as a corrective plug-in when fixed structural propagation is misaligned with the data, while preserving performance when the original propagation is already appropriate.

Ablations. The ablation rows isolate the effect of PANDA's two main components. PANDA w/o Prior,Aux keeps only the learned receiver-normalised attention. This variant is sometimes unable to recover the VANILLA accuracy and does not consistently outperform it, suggesting that learned

Table 2: Node classification results reporting mean accuracies ± standard deviation across all datasets for GCN, GAT and PANDA on GCN.

	Variant	Telegram	Citeseer	Pubmed	Roman-empire	Cornell	Texas	Squirrel	email-Enron
GCN	VANILLA	90.38 ± 2.852	77.00 ± 1.197	89.28 ± 0.272	75.18 ± 0.683	52.43 ± 9.300	67.57 ± 6.396	43.70 ± 1.947	62.16 ± 5.539
	GAT (PANDA w/o Prior,Aux)	82.88 ± 3.488	76.75 ± 0.919	89.92 ± 0.247	85.90 ± 0.707	53.78 ± 10.429	68.92 ± 6.648	39.73 ± 1.634	68.92 ± 6.310
	PANDA	91.73 ± 3.851	76.30 ± 1.070	88.97 ± 0.466	86.74 ± 0.711	73.78 ± 6.049	80.54 ± 4.804	42.84 ± 1.322	64.86 ± 7.151
	Δ(PANDA - Vanilla)	+1.37	-0.70	-0.31	+11.55	+21.35	+12.97	-0.86	+2.70

Table 3: Main auxiliary-path analysis across five datasets. $\mathcal{H}_{\mathrm{CE}}$ is clique-expansion homophily from Wang et al. [2023b]. ΔAcc. is the accuracy change after shutting off both auxiliary gates, $\hat{g}^{\ell}_{e,v}$ and $\check{g}^{\ell}_{u,e}$. q_{mis} is the fraction of test nodes whose clique-expansion neighbour-majority class differs from the true label. q_{help} and q_{hurt} are the fractions of nodes flipped onto or off the true class by the auxiliary path. $q_{\text{nbr-rescue}}$ is the fraction of test nodes rescued specifically from the misleading neighbour-majority class. $\overline{\Delta}^{\mathrm{nbr}}_{\neq}$ is the true-vs-neighbour-majority logit-margin shift on misleading-neighbour nodes. All rates are percentages; full definitions are in Appendix F.

Dataset	$\mathcal{H}_{\mathrm{CE}}$	ΔAcc.	q_{mis}	$q_{\mathrm{help}}/q_{\mathrm{hurt}}$	$q_{\text{nbr-rescue}}$	$\overline{\Delta}^{\mathrm{nbr}}_{\neq}$
Roman-empire	0.236	−27.7	95.68	29.44 / 1.77	2.20	+5.29
Telegram	0.285	−54.0	81.73	56.54 / 2.50	16.35	+3.58
Cornell	0.420	−4.6	57.84	5.95 / 1.35	1.08	+0.09
Texas	0.505	+0.0	47.03	3.24 / 3.24	1.35	+0.09
Citeseer	0.795	−1.0	23.97	1.84 / 0.82	0.22	−0.28

attention alone is insufficient to replace the fixed structural propagation of existing HNN models. Adding the structural prior (PANDA w/o Aux) recovers much of the VANILLA performance, while also allowing the model to outperform the baseline on some datasets, highlighting the importance of both attention and the structural priors. The full PANDA module further adds the gated, non-competitive auxiliary path, which is decisive in several cases where the main path alone remains limited. This indicates that the auxiliary path captures complementary information that the receiver-normalised main path cannot express on its own.

PANDA on GNNs. We have presented PANDA with a focus on directed hypergraphs as they subsume undirected hypergraphs and ordinary graphs. Table 2 shows that PANDA's gains translate to ordinary GNNs as well. We see that PANDA consistently improves performance for GCN across the datasets, with the exceptions where the differences are primarily small. PANDA + GCN also consistently outperforms GAT, with exceptions on the homophilic datasets where the difference is small, and the email-Enron dataset, where the difference is greater.

Auxiliary-Path Analysis. We further analyse the learned auxiliary path of the GeDi-PANDA on five datasets using post-hoc inference-time interventions on fixed trained checkpoints; full definitions, routing statistics, and additional geometric diagnostics are reported in Appendix F. These interventions measure the predictive dependence of the trained model on the auxiliary gates at test time. Table 3 summarises the main findings. Disabling both $\hat{g}^{\ell}_{e,v}$ and $\check{g}^{\ell}_{u,e}$ auxiliary gates has a large effect on Telegram and Roman-empire, decreasing test accuracy by 54.0 and 27.7 percentage points, respectively, with a smaller effect on Citeseer, Cornell, and Texas.

To understand whether this effect corresponds to correcting misleading neighbourhood information, we measure $\overline{\Delta}^{\mathrm{nbr}}_{\neq}$, i.e., how much the auxiliary path shifts the logit margin between the true class and the neighbour-majority class, when the two are different. On the two low-homophily datasets, as reported by Wang et al. [2023b], Roman-empire and Telegram, the auxiliary path strongly increases the true-vs-neighbour-majority logit margin on misleading-neighbour nodes ($+5.29$ and $+3.58$). Thus, when information flowing from neighbourhoods is unreliable, the auxiliary path behaves as a counter-neighbourhood signal. However, strict neighbour-majority rescues explain only part of the total helped flips: 2.20% out of 29.44% on Roman-empire and 16.35% out of 56.54% on Telegram. This indicates that the auxiliary path is not merely correcting predictions that follow the wrong neighbour-majority class; rather, it improves margins against a broader set of competing classes.

The cosine analysis in Appendix F.3 provides a complementary geometric view. We measure the source- and receiver-wise cosine similarity between the main-path representation and the auxiliary representation in each pass. In the $\mathcal{V}\to\mathcal{E}$ pass, the cosine is positive across datasets, indicating that

Table 4: Node classification results reporting mean accuracies ± standard deviation across all datasets. We compare our PANDA plug-in with the Norm Aux variant on different backbone architectures.

Model	Telegram	Citeseer	Pubmed	Roman-empire	Cornell	Texas	Squirrel	email-Enron
GeDi + Norm Aux	75.96 ± 5.319	76.56 ± 0.862	89.25 ± 0.498	89.42 ± 0.466	75.14 ± 3.784	80.27 ± 6.401	40.07 ± 4.317	52.16 ± 7.649
GeDi + PANDA	91.15 ± 3.567	76.12 ± 0.921	89.32 ± 0.416	90.11 ± 0.699	77.03 ± 4.230	84.05 ± 4.595	42.76 ± 1.742	60.27 ± 8.466
Δ(PANDA - Norm Aux)	+15.19	-0.44	+0.07	+0.69	+1.89	+3.78	+2.69	+8.11
HGNN + Norm Aux	69.62 ± 3.922	77.38 ± 0.648	89.66 ± 0.379	79.83 ± 0.621	70.54 ± 4.265	80.54 ± 2.911	38.78 ± 2.902	51.62 ± 3.354
HGNN + PANDA	72.12 ± 2.885	77.17 ± 1.125	89.77 ± 0.368	82.37 ± 0.511	74.59 ± 2.162	83.24 ± 5.644	40.43 ± 0.999	54.59 ± 8.090
Δ(PANDA - Norm Aux)	+2.50	-0.21	+0.11	+2.54	+4.05	+2.70	+1.65	+2.97
HyperGCN + Norm Aux	67.31 ± 5.160	75.87 ± 1.118	88.28 ± 0.218	78.67 ± 0.581	66.49 ± 5.822	77.57 ± 7.257	37.83 ± 3.124	45.59 ± 3.713
HyperGCN + PANDA	70.58 ± 5.165	75.63 ± 0.894	88.59 ± 0.568	80.16 ± 0.661	64.86 ± 6.620	78.65 ± 5.976	42.46 ± 2.923	54.86 ± 6.402
Δ(PANDA - Norm Aux)	+3.27	-0.24	+0.31	+1.49	-1.63	+1.08	+4.63	+9.27
PhenomNN + Norm Aux	72.88 ± 3.889	77.04 ± 1.259	88.56 ± 0.378	79.75 ± 0.419	74.95 ± 4.090	81.97 ± 4.018	39.57 ± 3.157	43.76 ± 8.683
PhenomNN + PANDA	75.82 ± 4.343	77.08 ± 1.113	87.51 ± 0.581	85.08 ± 0.752	75.68 ± 2.961	82.70 ± 5.565	41.01 ± 1.999	50.27 ± 6.304
Δ(PANDA - Norm Aux)	+2.94	+0.04	-1.05	+5.33	+0.73	+0.73	+1.44	+6.51

the auxiliary node representation usually retains part of the main node direction. On the other hand, in the $\mathcal{E}\to\mathcal{V}$ pass, the cosine is instead close to zero, showing that the auxiliary-transformed hyperedge representation is not a rescaled copy of the main one but a distinct, approximately orthogonal direction. This geometric evidence does not by itself explain the margin gains, but it supports the interpretation that the auxiliary branch provides a non-redundant message channel rather than simply amplifying the main path.

Non-Competitiveness Ablation. We further investigate whether the performance gains come from the non-competitive nature of the auxiliary path or from the additional parameters introduced in the auxiliary path. Towards this end, we compare the performance of PANDA to a variant where we normalise the auxiliary path's scores, while maintaining identical parameter counts. We dub this variant Norm Aux. Table 4 shows that PANDA consistently outperforms the Norm Aux ablation, with a few small exceptions where the gap is primarily small. This strongly supports our hypothesis that non-competitiveness is important.

Additional Analysis. We perform parameter and inference time comparisons for PANDA and the vanilla models in Appendix C, and these demonstrate that PANDA can improve parameter efficiency and, in several settings, inference speed. We also extend our analysis on additional datasets in Appendix D and show that similar trends follow on these additional datasets. We include a large scale hypergraph dataset and show how PANDA helps with scalability, by allowing for more parameter efficient and faster models. We also perform norm stability analyses in Appendix E to ensure there are no stability related issues with the introduction of the auxiliary paths in PANDA.

6 Conclusions

HNNs typically rely either on fixed structural weights, which can propagate misleading information in heterophilic settings, or on receiver-wise normalised attention, whose coefficients are competitive by construction. We introduced PANDA, a plug-in for two-stage hypergraph message-passing layers, applicable to both undirected and directed HNNs. Its main path interpolates between learned receiver-normalised attention and structural incidence priors, while a non-competitive auxiliary path supplies an additional channel that the receiver-normalised path cannot express on its own. We further showed that GeDi-HNN's complex Hermitian convolution admits a two-stage realisation through the original complex incidence matrix and its conjugate, but that this realisation is not equivalent to applying a Hermitian graph operator after a directed star expansion of the hypergraph. Across four HNNs, PANDA yields consistent improvements on heterophilic benchmarks without degrading performance on homophilic ones, acting as a corrective plug-in where the structural propagation is misaligned with the data. Our post-hoc analysis makes this behaviour explicit. Removing the auxiliary branch at test time sharply reduces accuracy, showing that it is informative for the trained model. On heterophilic datasets, the auxiliary branch shifts predictions away from misleading neighbourhood signals towards the correct class, and its representations, especially in the hyperedge-to-node stage, are nearly orthogonal to those of the main path, indicating a distinct message channel rather than a rescaling. Future work will address the main architectural limitation of PANDA, i.e., being restricted to two-stage message-passing HNNs, by extending it to more general information-propagation paradigms rooted in algebraic topology, such as sheaf neural networks.

Acknowledgments and Disclosure of Funding

This work is partially funded by Trinity College, Cambridge.

References

Nurul A. Asif, Yeahia Sarker, Ripon K. Chakrabortty, Michael J. Ryan, Md. Hafiz Ahamed, Dip K. Saha, Faisal R. Badal, Sajal K. Das, Md. Firoz Ali, Sumaya I. Moyeen, Md. Robiul Islam, and Zinat Tasneem. Graph neural network: A comprehensive review on non-euclidean space. *IEEE Access*, 9:60588–60606, 2021. doi: 10.1109/ACCESS.2021.3071274.

Song Bai, Feihu Zhang, and Philip H. S. Torr. Hypergraph convolution and hypergraph attention. *Pattern Recognition*, 110:107637, 2021. doi: 10.1016/j.patcog.2020.107637.

Muhammet Balcilar, Guillaume Renton, Pierre Heroux, Benoit Gauzere, Sebastien Adam, and Paul Honeine. Bridging the gap between spectral and spatial domains in graph neural networks, 2020. URL `https://arxiv.org/abs/2003.11702`.

Muhammet Balcilar, Guillaume Renton, Pierre Heroux, Benoit Gauzere, Sebastien Adam, and Paul Honeine. Analyzing the expressive power of graph neural networks in a spectral perspective. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=-qh0M9XWxnv`.

Austin R. Benson, David F. Gleich, and Jure Leskovec. Higher-order organization of complex networks. *Science*, 353(6295):163–166, July 2016. ISSN 1095-9203. doi: 10.1126/science.aad9029. URL `http://dx.doi.org/10.1126/science.aad9029`.

Austin R. Benson, Rediet Abebe, Michael T. Schaub, Ali Jadbabaie, and Jon Kleinberg. Simplicial closure and higher-order link prediction. *Proceedings of the National Academy of Sciences*, 115 (48):E11221–E11230, 2018. doi: 10.1073/pnas.1800683115. URL `https://www.pnas.org/doi/abs/10.1073/pnas.1800683115`.

Alexandre Bovet and Peter Grindrod. The activity of the far right on Telegram. *ResearchGate preprint*, DOI: 10.13140/RG.2.2.16700.05764:1–19, 2020.

Shaked Brody, Uri Alon, and Eran Yahav. How attentive are graph attention networks? In *International Conference on Learning Representations*, 2022.

Guanzi Chen, Jiying Zhang, Xi Xiao, and Yang Li. Preventing over-smoothing for hypergraph neural networks. *arXiv preprint arXiv:2203.17159*, 2022.

Zhiqian Chen, Fanglan Chen, Lei Zhang, Taoran Ji, Kaiqun Fu, Liang Zhao, Feng Chen, Lingfei Wu, Charu Aggarwal, and Chang-Tien Lu. Bridging the gap between spatial and spectral domains: A unified framework for graph neural networks. *ACM Computing Surveys*, 56(5):1–42, 2023. doi: 10.1145/3627816.

Eli Chien, Chao Pan, Jianhao Peng, and Olgica Milenkovic. You are allset: A multiset function framework for hypergraph neural networks. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=hpBTIv2uy_E`.

Yann N. Dauphin, Angela Fan, Michael Auli, and David Grangier. Language modeling with gated convolutional networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 933–941. PMLR, 2017.

Kaize Ding, Jianling Wang, Jundong Li, Dingcheng Li, and Huan Liu. Be more with less: Hypergraph attention networks for inductive text classification. In *Proceedings of the 2020 conference on empirical methods in natural language processing (EMNLP)*, pages 4927–4936, 2020.

Yihe Dong, Will Sawin, and Yoshua Bengio. Hnhn: Hypergraph networks with hyperedge neurons. *arXiv preprint arXiv:2006.12278*, 2020.

Yifan Feng, Haoxuan You, Zizhao Zhang, Rongrong Ji, and Yue Gao. Hypergraph neural networks. In *AAAI*, 2019.

Stefano Fiorini, Stefano Coniglio, Michele Ciavotta, and Enza Messina. Sigmanet: One laplacian to rule them all. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 7568–7576, 2023.

Stefano Fiorini, Stefano Coniglio, Michele Ciavotta, and Alessio Del Bue. Let there be direction in hypergraph neural networks. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=h48Ri6pmvi.

Giorgio Gallo, Giustino Longo, Stefano Pallottino, and Sang Nguyen. Directed hypergraphs and applications. *Discrete applied mathematics*, 42(2-3):177–201, 1993.

Yue Gao, Yifan Feng, Shuyi Ji, and Rongrong Ji. Hgnn+: General hypergraph neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(3):3181–3199, 2022.

Weizhe Hua, Zihang Dai, Hanxiao Liu, and Quoc Le. Transformer quality in linear time. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 9099–9117. PMLR, 2022.

Jing Huang and Jie Yang. Unignn: a unified framework for graph and hypergraph neural networks. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, 2021.

Rishabh Jain, Keisuke Okumura, Michael Amir, Pietro Lio, and Amanda Prorok. Pairwise is not enough: Hypergraph neural networks for multi-agent pathfinding. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust, editors, *International Conference on Learning Representations*, volume 2026, pages 131604–131627, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/d5ae1c5167be330b90f4c0a1eed7f8f0-Paper-Conference.pdf.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. URL https://arxiv.org/abs/1412.6980.

Ming Li, Yongchun Gu, Yi Wang, Yujie Fang, Lu Bai, Xiaosheng Zhuang, and Pietro Lio. When hypergraph meets heterophily: New benchmark datasets and baseline. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 18377–18384, 2025.

Zitong Ma, Wenbo Zhao, and Zhe Yang. Directed hypergraph representation learning for link prediction. In Sanjoy Dasgupta, Stephan Mandt, and Yingzhen Li, editors, *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, volume 238 of *Proceedings of Machine Learning Research*, pages 3268–3276. PMLR, 02–04 May 2024. URL https://proceedings.mlr.press/v238/ma24b.html.

Katarina A. Murgas, Emil Saucan, and Romeil Sandhu. Hypergraph geometry reflects higher-order dynamics in protein interaction networks. *Scientific Reports*, 12(1):20879, 2022. doi: 10.1038/s41598-022-24584-w. URL https://doi.org/10.1038/s41598-022-24584-w.

Emilio Parisotto, Francis Song, Jack Rae, Razvan Pascanu, Caglar Gulcehre, Siddhant Jayakumar, Max Jaderberg, Raphaël Lopez Kaufman, Aidan Clark, Seb Noury, Matthew Botvinick, Nicolas Heess, and Raia Hadsell. Stabilizing transformers for reinforcement learning. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 7487–7498. PMLR, 2020.

Hongbin Pei, Bingzhe Wei, Kevin Chen Chuan Chang, Yu Lei, and Bo Yang. Geom-gcn: Geometric graph convolutional networks, 2020.

Oleg Platonov, Denis Kuznedelev, Michael Diskin, Artem Babenko, and Liudmila Prokhorenkova. A critical look at the evaluation of GNNs under heterophily: Are we really making progress? In *The Eleventh International Conference on Learning Representations*, 2023a. URL https://openreview.net/forum?id=tJbbQfw-5wv.

Oleg Platonov, Denis Kuznedelev, Michael Diskin, Artem Babenko, and Liudmila Prokhorenkova. A critical look at the evaluation of gnns under heterophily: Are we really making progress? In *The Eleventh International Conference on Learning Representations*, 2023b.

Zihan Qiu, Zekun Wang, Bo Zheng, Zeyu Huang, Kaiyue Wen, Songlin Yang, Rui Men, Le Yu, Fei Huang, Suozhi Huang, Dayiheng Liu, Jingren Zhou, and Junyang Lin. Gated attention for large language models: Non-linearity, sparsity, and attention-sink-free. In *Advances in Neural Information Processing Systems*, 2025. NeurIPS 2025 oral.

Jörg Reichardt and Stefan Bornholdt. Statistical mechanics of community detection. *Physical Review E*, 74(1), July 2006. ISSN 1550-2376. doi: 10.1103/physreve.74.016110. URL http://dx.doi.org/10.1103/PhysRevE.74.016110.

Oliver Richter and Roger Wattenhofer. Normalized attention without probability cage. *arXiv preprint arXiv:2005.09561*, 2020.

Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2009. doi: 10.1109/TNN.2008.2005605.

Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. *AI magazine*, 29(3):93–93, 2008.

Noam Shazeer. GLU variants improve transformer, 2020.

Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. Highway networks, 2015.

Lev Telyatnikov, Maria Sofia Bucarelli, Guillermo Bernardez, Olga Zaghen, Simone Scardapane, and Pietro Lio. Hypergraph neural networks through the lens of message passing: A common perspective to homophily and architecture design. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL https://openreview.net/forum?id=8rxtL0kZnX.

Loc Hoang Tran and Linh Hoang Tran. Directed hypergraph neural network, 2022. URL https://arxiv.org/abs/2008.03626.

Pietro Traversa, Guilherme Ferraz de Arruda, Alexei Vazquez, and Yamir Moreno. Robustness and complexity of directed and weighted metabolic hypergraphs, 2023.

Francesco Tudisco, Austin R Benson, and Konstantin Prokopchik. Nonlinear higher-order label spreading. In *Proceedings of the Web Conference 2021*, pages 2402–2413, 2021.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

Peihao Wang, Shenghao Yang, Yunyu Liu, Zhangyang Wang, and Pan Li. Equivariant hypergraph diffusion neural operators. In *International Conference on Learning Representations (ICLR)*, 2023a.

Yuxin Wang, Quan Gan, Xipeng Qiu, Xuanjing Huang, and David Wipf. From hypergraph energy functions to hypergraph neural networks. In *Proceedings of the 40th International Conference on Machine Learning*, pages 35605–35623, 2023b.

Naganand Yadati, Madhav Nimishakavi, Prateek Yadav, Vikram Nitin, Anand Louis, and Partha Talukdar. Hypergcn: A new method for training graph convolutional networks on hypergraphs, 2019.

Fanqi Yan, Huy Nguyen, Pedram Akbarian, Nhat Ho, and Alessandro Rinaldo. Sigmoid self-attention has lower sample complexity than softmax self-attention: A mixture-of-experts perspective. *arXiv preprint arXiv:2502.00281*, 2025.

Chaoqi Yang, Ruijie Wang, Shuochao Yao, and Tarek Abdelzaher. Semi-supervised hypergraph node classification on hypergraph line expansion. In *Proceedings of the 31st ACM International Conference on Information and Knowledge Management, CIKM 2022*, 2022.

Songlin Yang, Bailin Wang, Yikang Shen, Rameswar Panda, and Yoon Kim. Gated linear attention transformers with hardware-efficient training. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 56501–56523. PMLR, 2024.

Jiani Zhang, Xingjian Shi, Junyuan Xie, Hao Ma, Irwin King, and Dit-Yan Yeung. GaAN: Gated attention networks for learning on large and spatiotemporal graphs. In *Proceedings of the Thirty-Fourth Conference on Uncertainty in Artificial Intelligence*, pages 339–349, 2018.

Jiying Zhang, Fuyang Li, Xi Xiao, Tingyang Xu, Yu Rong, Junzhou Huang, and Yatao Bian. Hypergraph convolutional networks via equivalency between hypergraphs and undirected graphs, 2022. URL https://arxiv.org/abs/2203.16939.

Dengyong Zhou, Jiayuan Huang, and Bernhard Schölkopf. Learning with hypergraphs: Clustering, classification, and embedding. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems*, volume 19. MIT Press, 2006. URL https://proceedings.neurips.cc/paper_files/paper/2006/file/dff8e9c2ac33381546d96deea9922999-Paper.pdf.

Jiong Zhu, Yujun Yan, Lingxiao Zhao, Mark Heimann, Leman Akoglu, and Danai Koutra. Beyond homophily in graph neural networks: Current limitations and effective designs. *Advances in neural information processing systems*, 33:7793–7804, 2020.

A Proof of our theoretical results

Theorem A.1. *PANDA with additive aggregations* $\mathrm{Agg.}$, $\mathrm{Trans}_{\cdot}^{(\ell)} = \mathrm{id}$, $\mathrm{Trans}_{\cdot}^{\mathrm{aux},(\ell)} = \mathbf{0}$, $\mathrm{Upd}^{(\ell)}(X, X') = \phi(X\Theta_0 + X'\Theta_1)$, *with* $\hat{\beta}_{e,v}^{(\ell)} = \check{\beta}_{u,e}^{(\ell)} = 0$ *(so that* $\hat{\alpha}_{e,v}^{\mathrm{main},(\ell)} = \hat{\alpha}_{e,v}^{\mathrm{prior}}$ *and* $\check{\alpha}_{u,e}^{\mathrm{main},(\ell)} = \check{\alpha}_{u,e}^{\mathrm{prior}}$*),* $\hat{\alpha}_{e,v}^{\mathrm{aux},(\ell)} = \check{\alpha}_{u,e}^{\mathrm{aux},(\ell)} = 0$*, and where the structural priors are*

$$\hat{\alpha}_{e,v}^{\mathrm{prior}} = \sqrt{\frac{w_e}{\delta_e\, d_v}}, \qquad \check{\alpha}_{u,e}^{\mathrm{prior}} = \sqrt{\frac{w_e}{d_u\, \delta_e}}.$$

recovers GeDi's convolutional layer defined on top of the Generalized Directed (GeDi) Laplacian $\vec{L}_N = I - \vec{Q}_N$*, with* $\vec{Q}_N := D_v^{-1/2}\, \vec{B}\, W\, D_e^{-1}\, \vec{B}^*\, D_v^{-1/2}$.

Proof. GeDi defines the Signless Laplacian and (Signed) Laplacian operators as

$$\vec{Q}_N = D_v^{-1/2}\, \vec{B}\, W\, D_e^{-1}\, \vec{B}^*\, D_v^{-1/2}, \qquad \vec{L}_N = I - \vec{Q}_N, \tag{25}$$

with $\vec{B}$ defined as in Equation (24). Its convolutional layer maps $X^{(\ell)} \in \mathbb{C}^{n\times c}$ to

$$X_{\mathrm{prior}}^{(\ell+1)} = \phi\Big(X^{(\ell)}\Theta_0^G + \vec{L}_N X^{(\ell)}\Theta_1^G\Big), \tag{26}$$

for learnable matrices Θ_0^G, Θ_1^G.

Given that $*$ denotes the complex conjugate-adjoint operator, under the hypotheses of the theorem, $\hat{\alpha}^{\mathrm{main}} = \hat{\alpha}^{\mathrm{prior}}$, $\check{\alpha}^{\mathrm{main}} = \check{\alpha}^{\mathrm{prior}}$, and the auxiliary path is switched off. Therefore the node-to-hyperedge aggregation in Eq. (22) becomes, for every $e \in \mathcal{E}$,

$$\mathbf{h}_e^{(\ell+1)} = \sum_{v\in H(e)} \sqrt{\tfrac{w_e}{\delta_e d_v}}\, \mathbf{x}_v^{(\ell)} \;+\; i \sum_{v\in T(e)} \sqrt{\tfrac{w_e}{\delta_e d_v}}\, \mathbf{x}_v^{(\ell)} = \sum_{v\in\mathcal{V}} \sqrt{\tfrac{w_e}{\delta_e d_v}}\, \vec{B}_{e,v}^*\, \mathbf{x}_v^{(\ell)}.$$

Stacking across hyperedges into $H^{(\ell+1)} \in \mathbb{C}^{m\times c}$,

$$H^{(\ell+1)} \;=\; D_e^{-1/2}\, W^{1/2}\, \vec{B}^*\, D_v^{-1/2}\, X^{(\ell)}. \tag{27}$$

The hyperedge-to-node aggregation in Eq. (23) similarly gives

$$\mathbf{x}'^{(\ell+1)}_u = \sum_{e\in\Gamma_H(u)} \sqrt{\tfrac{w_e}{d_u \delta_e}}\, \mathbf{h}_e^{(\ell+1)} \;-\; i \sum_{e\in\Gamma_T(u)} \sqrt{\tfrac{w_e}{d_u \delta_e}}\, \mathbf{h}_e^{(\ell+1)} = \sum_{e\in\mathcal{E}} \sqrt{\tfrac{w_e}{d_u \delta_e}}\, \vec{B}_{u,e}\, \mathbf{h}_e^{(\ell+1)},$$

which in matrix form reads

$$X'^{(\ell+1)} \;=\; D_v^{-1/2}\, \vec{B}\, W^{1/2}\, D_e^{-1/2}\, H^{(\ell+1)}. \tag{28}$$

Composing Eqs. (27) and (28) and using that W, D_e, D_v are real diagonal,

$$X'^{(\ell+1)} = D_v^{-1/2}\, \vec{B}\, W^{1/2}\, D_e^{-1/2}\, D_e^{-1/2}\, W^{1/2}\, \vec{B}^*\, D_v^{-1/2}\, X^{(\ell)}$$

$$= D_v^{-1/2}\,\vec{B}\,W\,D_e^{-1}\,\vec{B}^*\,D_v^{-1/2}\,X^{(\ell)} \;=\; \vec{Q}_N\,X^{(\ell)},$$

where the last equality is exactly the definition of $\vec{Q}_N$ in Eq. (25). Hence the proposed aggregation mechanism recovers the GeDi convolution operator $\vec{Q}_N$. Combining this with the update in (iii) yields

$$X^{(\ell+1)} \;=\; \phi\Big(X^{(\ell)}\Theta_0 \;+\; \vec{Q}_N\,X^{(\ell)}\Theta_1\Big).$$

Given any GeDi parameters (Θ_0^G, Θ_1^G), choose $\Theta_0 = \Theta_0^G + \Theta_1^G$ and $\Theta_1 = -\Theta_1^G$. Then

$$\begin{aligned}\phi\Big(X^{(\ell)}\Theta_0 + \vec{Q}_N X^{(\ell)}\Theta_1\Big) &= \phi\Big(X^{(\ell)}(\Theta_0^G + \Theta_1^G) - \vec{Q}_N X^{(\ell)}\Theta_1^G\Big)\\ &= \phi\Big(X^{(\ell)}\Theta_0^G + (I - \vec{Q}_N)X^{(\ell)}\Theta_1^G\Big)\\ &= \phi\Big(X^{(\ell)}\Theta_0^G + \vec{L}_N X^{(\ell)}\Theta_1^G\Big),\end{aligned}$$

which is the GeDi-HNN layer in Eq. (26). □

Theorem A.2. *Let $\mathcal{G}_\star$ be the directed star expansion obtained from a directed hypergraph $\mathcal{H}$ by replacing each hyperedge e with an auxiliary node h_e, adding arcs $t \to h_e$ for every $t \in T(e)$ and arcs $h_e \to r$ for every $r \in H(e)$. Let $\vec{Q}_\star$ and $\vec{L}_\star$ be the GeDi Signless and Signed Laplacian operators on this directed graph, and let $\vec{Q}_N$ be the GeDi normalised Signless Laplacian on $\mathcal{H}$. The block $(\vec{Q}_\star^2)_{\mathcal{V},\mathcal{V}}$ is the two-hop node–auxiliary–node propagation obtained by applying GeDi twice on the star-expanded graph. In general, however, neither the one-step operator $\vec{Q}_\star$ nor this two-hop star-expanded propagation equals $\vec{Q}_N$ on the original vertices. Thus even the two-hop message passing induced by the ordinary star expansion already differs from $\vec{Q}_N$ in phase.*

Proof. One may ask whether the same operator can be obtained by replacing each directed hyperedge e with an auxiliary node h_e, connecting every tail node to h_e and then connecting h_e to every head node. Formally, define the star-expanded directed graph

$$\mathcal{G}_\star = (\mathcal{V}_\star, \mathcal{A}_\star), \qquad \mathcal{V}_\star = \mathcal{V} \cup \{h_e : e \in \mathcal{E}\},$$

with ordinary directed edges

$$\mathcal{A}_\star = \{a_{t,e} = (t, h_e) : t \in T(e)\} \cup \{a_{e,r} = (h_e, r) : r \in H(e)\}.$$

Each $a = (p, q) \in \mathcal{A}_\star$ is now an ordinary directed edge, i.e., $T_\star(a) = \{p\}$ and $H_\star(a) = \{q\}$. Let $\vec{B}_\star$ be the corresponding GeDi incidence matrix, with $(\vec{B}_\star)_{p,a} = -i$ and $(\vec{B}_\star)_{q,a} = 1$, and let $W_\star$, $D_{\mathcal{V}_\star}$, and $D_{\mathcal{A}_\star}$ be the edge weights, vertex degrees, and edge degrees of this ordinary directed graph. The GeDi adjacency and Laplacian on the star expansion are

$$\vec{Q}_\star = D_{\mathcal{V}_\star}^{-1/2}\vec{B}_\star W_\star D_{\mathcal{A}_\star}^{-1}\vec{B}_\star^* D_{\mathcal{V}_\star}^{-1/2}, \qquad \vec{L}_\star = I - \vec{Q}_\star.$$

Using $\vec{L}_\star$ or $\vec{Q}_\star$ on this expanded graph is not equivalent to the hypergraph GeDi operator $\vec{Q}_N$. A single GeDi step on the star-expanded graph only mixes original nodes with auxiliary nodes, so it does not create the off-diagonal interactions between distinct original nodes that lie in the same hyperedge. If one instead compares the two-step adjacency operator $\vec{Q}_\star^2$, the complex phases still do not match.

Ignoring positive degree and weight normalisations, the phase of the hypergraph GeDi contribution from input node v to output node u through hyperedge e is

$$c_e(u, v) = \vec{B}_{u,e}\vec{B}^*_{e,v} = \begin{cases} 1, & u, v \in H(e) \text{ or } u, v \in T(e),\\ i, & u \in H(e),\ v \in T(e),\\ -i, & u \in T(e),\ v \in H(e).\end{cases}$$

In the star-expanded graph, a two-hop interaction from a tail node $v \in T(e)$ to a head node $u \in H(e)$ follows $v \to h_e \to u$. The two ordinary directed edges contribute phases i and i, giving $i \cdot i = -1$ instead of i. Similarly, the reverse cross-role path gives $(-i) \cdot (-i) = -1$ instead of $-i$. Thus the star expansion changes the cross-role phases.

The smallest counterexample has two vertices and one directed hyperedge:

$$\mathcal{V} = \{t, r\}, \qquad T(e) = \{t\}, \qquad H(e) = \{r\}, \qquad w_e = 1.$$

Then $\delta_e = 2$, $d_t = d_r = 1$, and the original hypergraph operator has

$$(\vec{Q}_N)_{r,t} = \frac{1}{2}\vec{B}_{r,e}\vec{B}^*_{e,t} = \frac{i}{2}.$$

Replace e by the directed path $t \to h_e \to r$, with ordinary directed edges $a = (t, h_e)$ and $b = (h_e, r)$. A single GeDi step on this expanded graph has zero direct coefficient from t to r. The two-step contribution through h_e has phase

$$\left(\vec{B}_{r,b}\vec{B}^*_{b,h_e}\right)\left(\vec{B}_{h_e,a}\vec{B}^*_{a,t}\right) = (1 \cdot i)(1 \cdot i) = -1.$$

With the same GeDi normalisations on the expanded graph, $d_t = d_r = 1$, $d_{h_e} = 2$, and $\delta_a = \delta_b = 2$, hence

$$(\vec{Q}_\star)_{h_e,t} = \frac{i}{2\sqrt{2}}, \qquad (\vec{Q}_\star)_{r,h_e} = \frac{i}{2\sqrt{2}}, \qquad (\vec{Q}_\star)_{r,h_e}(\vec{Q}_\star)_{h_e,t} = -\frac{1}{8}.$$

This differs from the hypergraph coefficient $i/2$ already in phase.

The correct equivalence is instead the two-stage computation of the original hypergraph operator $\vec{Q}_N$. The symbol h_e can be used as a temporary hyperedge variable, but it should not be interpreted as a new vertex of an ordinary directed graph. The two stages must use the conjugate incidence $\vec{B}^*$ from nodes to hyperedges and the incidence $\vec{B}$ from hyperedges back to nodes:

$$m_e = \sum_{v \in e} \vec{B}^*_{e,v} \frac{x_v}{\sqrt{d_v}}, \qquad (\vec{Q}_N x)_u = \frac{1}{\sqrt{d_u}} \sum_{e \ni u} \frac{w_e}{\delta_e} \vec{B}_{u,e} m_e.$$

This is exactly the $\vec{B}^*$-then-$\vec{B}$ factorisation of $\vec{Q}_N$. Hence the attentive message-passing reduction is equivalent to GeDi when the intermediate hyperedge variables are computed with these frozen complex coefficients, but not when one first converts the hypergraph into an ordinary directed graph and then applies GeDi to that graph. □

B Experimental Setup

B.1 Code Repository

The code written for this research work is attached as supplementary material and will be freely distributed under the Apache 2.0 licence once accepted.[1]

The baseline models used in this paper were taken from the publicly available implementations from GeDi-HNN `https://github.com/Stefa1994/GeDi-HNN` with Apache License 2.0.

B.2 Hardware Details

All experiments are carried out on two different workstations: one equipped with four NVIDIA RTX 2080 GPUs (11 GB each) and Intel Xeon Gold 6248R processors, and another featuring an Intel Xeon CPU E5-2630 and two NVIDIA TITAN Xp GPUs with 12 GB of VRAM each.

The runtime required for a given hyperparameter setting, which involves training and validation over 10 different dataset splits, depends on both the dataset and the backbone architecture. For example, on the `Telegram` dataset, each hyperparameter setting of GeDi+PANDA requires on average 14.2 mins, while Phenomnn+PANDA requires on average 21.3 mins. On the other hand, GeDi+PANDA requires only 7.2 mins on average on the `Cornell` dataset, while it requires on average 40.4 mins on the `Roman-empire` dataset. On average, each setting for GeDi+PANDA requires 12.5 mins.

[1] `https://www.apache.org/licenses/LICENSE-2.0`

B.3 Hyperparameter tuning

We utilise the WandB platform to monitor training procedures and to carry out hyperparameter tuning. All models are trained for up to 500 epochs with early stopping set to 200 epochs. We employ Adam [Kingma and Ba, 2017] for optimising the model parameters with lr $\in [0.001, 0.02]$, wd $\in \{0,\ 5\times10^{-5}, 5\times10^{-4}, 5\times10^{-3}\}$. For all the models, we adopt a dropout $\in \{0.1, 0.2, \ldots, 0.9\}$, and for each model that has a selectable number of layers for the final classifier we fix it to 2. For each baseline, we select a range of parameters consistent with those investigated in their respective original works:

- HGNN: basic blocks $\{1, 2, 3, 4, 5\}$; MLP hidden width $\{64, 128, 256, 512\}$, classifier width $\{64, 128, 256\}$.
- HyperGCN: basic blocks $\{1, 2, 3, 4, 5\}$; HyperGCN-Fast $\in \{\text{True}, \text{False}\}$; Using-Mediators $\in \{\text{True}, \text{False}\}$.
- PhenomNN: hidden width $\{64, 128, 256, 512\}$; $\lambda_0 \in \{0, 0.1, 0.2, \ldots, 100.0\}$; $\lambda_1 \in \{0, 0.1, 0.2, \ldots, 100.0\}$; propagation steps $\{8, 16\}$; (H, HisI, twoHgamma) $\in \{(\text{False}, \text{False}, \text{False}), (\text{True}, \text{False}, \text{False}), (\text{True}, \text{True}, \text{False}), (\text{True}, \text{False}, \text{True})\}$.
- GeDi-HNN: convolutional layers $\{1, 2, 3, 4, 5\}$; MLP hidden width $\{64, 128, 256, 512\}$; classifier width $\{64, 128, 256\}$; other complex $\in \{\text{True}, \text{False}\}$.

For our plug-in, PANDA, and its ablations, we use the following additional hyperparameters on top of the backbone's hyperparameters:

- GAT attention version $\in \{\text{v1}, \text{v2}\}$;
- Attention heads $\in \{1, 2, \ldots, 8\}$;
- Distinct attention weights for source and destination $\in \{\text{True}, \text{False}\}$.

Calculating the attention scores, as described in Equation (8), requires knowledge of the previous layer's hyperedge representation. Since all the considered datasets do not start with any initial hyperedge representation, we calculate the initial hyperedge representation using the backbone's vanilla scores. For consistency, we use the same strategy in subsequent layers.

B.4 Dataset Descriptions

The statistics of the 8 real-world datasets are provided in Table 5. The datasets used for the experiments are:

- `Citeseer`, `Pubmed` Standard citation benchmarks in which vertices represent research papers and directed edges encode citation links. Node attributes are constructed from text using bag-of-words representations of the documents.
- `email-Enron` A corporate email communication network built from Enron's message logs. Nodes correspond to email accounts and edges record sender interactions. As ground-truth labels are unavailable, we derive node classes via the Spinglass community detection method Reichardt and Bornholdt [2006].
- `Texas, Cornell` WebKB datasets collected from university computer science departments. Each node is a webpage, hyperlinks are edges, and features are bag-of-words over page content. Pages are annotated into five categories: student, project, course, staff, and faculty.
- `Telegram` An interaction network extracted from Telegram, capturing exchanges among users who propagate political content.
- `Squirrel` The Squirrel dataset consists of articles from the English Wikipedia (December 2018). Nodes represent articles, and edges represent mutual links between them. Node features indicate the presence of specific nouns in the articles. Nodes are grouped into five categories based on the original regression targets.
- `Roman-empire` The dataset is based on the *Roman Empire* article from English Wikipedia, which was selected since it is one of the longest articles on Wikipedia and it follows the construction proposed by Platonov et al. [2023a]. Each node in the graph corresponds to one (non-unique) word in the text.

Table 5: Statistics of the datasets used in our experiments. Reported are the number of nodes, features, hyperedges, and classes, as well as the average hyperedge size ($|e|$), the average node degree ($|v|$), the clique-expansion (CE) homophily computed as in Wang et al. [2023a], and the node and hyperedge homophily as calculated in Li et al. [2025].

Dataset	# Nodes	# Features	# Hyperedges	# Classes	avg $\lvert e\rvert$	avg $\lvert v\rvert$	CE Hom.	Node Hom.	Hyperedge Hom.
Roman-empire	22,662	300	22,662	18	2.73	2.73	0.2363	0.0582	0.0582
Squirrel	2,223	2,089	2,060	5	23.81	22.07	0.2448	0.2081	0.2152
Telegram	245	1	183	4	49.70	37.12	0.2854	0.3459	0.3784
email-Enron	143	–	139	7	19.58	19.03	0.3251	0.4163	0.4314
Cornell	183	1,703	96	5	4.07	2.14	0.4200	0.4674	0.4534
Texas	183	1,703	110	5	3.81	2.29	0.5049	0.5076	0.3172
Citeseer	3,312	3,703	1,951	6	3.35	1.98	0.7947	0.5771	0.6814
Pubmed	19,717	500	7,963	3	4.35	1.76	0.8012	0.5499	0.7765

Table 6: Parameter counts for PANDA and the vanilla models across different datasets and different backbone architectures.

Model	Telegram	Citeseer	Pubmed	Roman-empire	Cornell	Texas	Squirrel	email-Enron
GeDi (Vanilla)	8.922 M	**30.864 M**	**0.578 M**	5.383 M	**1.461** M	4.015 M	11.533 M	26.884 M
GeDi + PANDA	**0.765 M**	35.682 M	6.714 M	**0.421 M**	8.470 M	**0.638 M**	**1.902 M**	**5.880 M**
HGNN (Vanilla)	**0.134 M**	**13.734 M**	1.932 M	**1.761 M**	**10.861 M**	**3.168 M**	3.860 M	**0.035 M**
HGNN + PANDA	0.163 M	15.720 M	**1.400 M**	2.590 M	12.358 M	6.638 M	**3.366 M**	0.139 M
HyperGCN (Vanilla)	0.695 M	3.154 M	**0.059 M**	**0.046 M**	**0.051** M	**0.026 M**	0.579 M	0.396 M
HyperGCN + PANDA	**0.027 M**	**1.933 M**	0.060 M	0.138 M	0.225 M	0.060 M	**0.275 M**	**0.052 M**
PhenomNN (Vanilla)	**0.001 M**	1.662 M	0.516 M	**0.135 M**	5.464 M	**0.439 M**	1.597 M	0.407 M
PhenomNN + PANDA	1.852 M	**1.534 M**	**0.066 M**	0.150 M	**1.190 M**	0.974 M	**0.345 M**	**0.230 M**

Table 7: Wall-clock inference times (mean $\pm$ standard deviation) for PANDA and the vanilla models across different datasets and different backbone architectures.

Model	Telegram	Citeseer	Pubmed	Roman-empire	Cornell	Texas	Squirrel	email-Enron
GeDi (Vanilla)	20.3 $\pm$ 0.1 ms	50.9 $\pm$ 1.9 ms	24.8 $\pm$ 0.2 ms	203.8 $\pm$ 1.1 ms	7.2 $\pm$ 0.0 ms	5.5 $\pm$ 0.1 ms	41.2 $\pm$ 0.9 ms	12.8 $\pm$ 0.1 ms
GeDi + PANDA	10.9 $\pm$ 0.0 ms	122.6 $\pm$ 0.7 ms	171.7 $\pm$ 1.0 ms	115.9 $\pm$ 1.9 ms	9.7 $\pm$ 0.0 ms	8.2 $\pm$ 0.1 ms	28.4 $\pm$ 0.1 ms	33.2 $\pm$ 0.1 ms
HGNN (Vanilla)	1.2 $\pm$ 0.0 ms	10.5 $\pm$ 0.0 ms	16.8 $\pm$ 0.0 ms	22.1 $\pm$ 0.0 ms	1.9 $\pm$ 0.0 ms	1.3 $\pm$ 0.0 ms	5.9 $\pm$ 0.0 ms	1.4 $\pm$ 0.0 ms
HGNN + PANDA	5.7 $\pm$ 0.2 ms	22.5 $\pm$ 1.4 ms	42.1 $\pm$ 0.4 ms	120.7 $\pm$ 0.5 ms	5.5 $\pm$ 0.0 ms	3.9 $\pm$ 0.0 ms	23.6 $\pm$ 1.1 ms	20.2 $\pm$ 0.0 ms
HyperGCN (Vanilla)	360.9 $\pm$ 69.2 ms	74.3 $\pm$ 1.8 ms	2.5 $\pm$ 0.0 ms	1.3 $\pm$ 0.0 ms	2.8 $\pm$ 0.2 ms	2.7 $\pm$ 0.1 ms	3.5 $\pm$ 0.0 ms	5.9 $\pm$ 0.0 ms
HyperGCN + PANDA	239.0 $\pm$ 5.2 ms	50.8 $\pm$ 0.7 ms	20.4 $\pm$ 0.0 ms	31.1 $\pm$ 0.2 ms	6.8 $\pm$ 0.0 ms	3.6 $\pm$ 0.1 ms	8.5 $\pm$ 0.0 ms	17.1 $\pm$ 0.0 ms
PhenomNN (Vanilla)	25.1 $\pm$ 0.4 ms	26.8 $\pm$ 0.1 ms	178.3 $\pm$ 0.0 ms	105.3 $\pm$ 7.5 ms	15.5 $\pm$ 0.0 ms	15.2 $\pm$ 0.1 ms	132.4 $\pm$ 0.1 ms	200.2 $\pm$ 17.9 ms
PhenomNN + PANDA	103.4 $\pm$ 0.1 ms	116.8 $\pm$ 0.2 ms	75.4 $\pm$ 3.1 ms	143.9 $\pm$ 5.4 ms	99.7 $\pm$ 1.5 ms	102.6 $\pm$ 0.6 ms	94.9 $\pm$ 0.3 ms	355.1 $\pm$ 7.9 ms

C Computational and Parameter Comparison

We report the computational and parameter overhead of the plug-in here. We note that the plug-in introduces various parameters to the model, leading to a theoretical increase in computational and parameter overhead. However, since we run a hyperparameter search, the improved architecture can allow for faster and more parameter efficient models.

Tables 6 and 7 report the parameter and computational overheads of PANDA and the vanilla models for the different backbone architectures across different datasets. We see that despite PANDA itself adding various components, 50% of the time, our hyperparameter search leads us to a more parameter efficient model. We also see that 20% of the time, the PANDA variant has faster inference speed than the baseline model.

D Results on Additional Datasets

Due to space constraints, we only report the results of eight datasets in Table 1. We provide results for additional datasets here. We include results over `Wisconsin` [Pei et al., 2020], `email-EU` [Benson et al., 2018] and `Questions` [Platonov et al., 2023b]. Questions consists of 48,921 nodes and acts as a scalability test.

Table 8 shows the results on these additional datasets. We see that for `Wisconsin` and `email-EU`, HyperGCN benefits massively from PANDA, while the other three backbones see small but consis-

Table 8: Node classification results reporting mean accuracies ± standard deviation for `Wisconsin` and `email-EU`. For `Questions`, we report the percentage AUC-ROC for each method, as this is a highly unbalanced dataset. We compare our PANDA plug-in and the vanilla model with different backbone architectures.

	Variant	Wisconsin	email-EU	Questions
GeDi	Vanilla	87.45 ± 5.203	53.48 ± 4.53	78.29 ± 1.026
	PANDA	87.84 ± 4.454	52.87 ± 5.60	79.69 ± 1.21
	Δ(PANDA - Vanilla)	+0.39	-0.61	+1.40
HGNN	Vanilla	85.88 ± 5.173	52.88 ± 1.83	78.25 ± 1.122
	PANDA	85.88 ± 5.462	52.83 ± 6.80	79.16 ± 1.126
	Δ(PANDA - Vanilla)	+0.00	-0.05	+0.91
HyperGCN	Vanilla	60.59 ± 3.965	32.51 ± 2.99	72.97 ± 0.955
	PANDA	80.78 ± 6.188	43.32 ± 5.02	74.05 ± 0.659
	Δ(PANDA - Vanilla)	+20.19	+10.81	+1.08
PhenomNN	Vanilla	85.49 ± 5.203	42.55 ± 3.40	76.02 ± 0.938
	PANDA	85.88 ± 4.094	47.73 ± 6.83	76.15 ± 2.849
	Δ(PANDA - Vanilla)	+0.39	+5.18	+0.13

Table 9: Parameter counts and wall-clock inference times (mean ± standard deviation) for PANDA and the vanilla models across different backbone architectures on `Questions` dataset.

Model	Parameter Count	Inference Time
GeDi (Vanilla)	0.511 M	195.7 ± 0.8 ms
GeDi + PANDA	**0.427 M**	**72.8 ± 0.2 ms**
HGNN (Vanilla)	1.463 M	53.4 ± 1.3 ms
HGNN + PANDA	**0.108 M**	**22.2 ± 0.0 ms**
HyperGCN (Vanilla)	0.213 M	**9.3 ± 0.0 ms**
HyperGCN + PANDA	**0.032 M**	13.1 ± 0.0 ms
PhenomNN (Vanilla)	0.559 M	1282.9 ± 83.7 ms
PhenomNN + PANDA	**0.041 M**	**431.4 ± 40.3 ms**

tently positive gains or almost matching performance. For `Questions` dataset, PANDA improves the performance across the board.

Table 9 shows the parameter and computational overhead for PANDA and the vanilla models for different backbones on the `Questions` dataset. We notice that PANDA is consistently more parameter efficient than the vanilla models across the backbones. We also find that PANDA leads to much faster inference for all the backbones, except for HyperGCN, where it is 1.4 × slower. In fact, it is more than 2 × faster than the vanilla model for all the other backbones.

E Embedding Stability and Degree Sensitivity

The introduction of auxiliary paths in PANDA can lead to explosion or contraction of the node embedding norms. Since this auxiliary path is not normalised, it means the model may be degree-sensitive, which in turn may lead to disproportionate growth of the node embedding norms. We perform experiments to investigate whether PANDA suffers from these phenomena.

Table 10 shows the average L2 norm growth of the node embeddings for GeDi and GeDi + PANDA across layers for different datasets. We do notice any clear trend of increased instability due to the PANDA plug-in: PANDA leads to comparable norms on Citeseer, Roman-empire and Cornell, larger norms on Telegram, Pubmed and email-Enron, and smaller ones on Texas and Squirrel.

Table 11 shows the average coefficient of variation for the L2 norm growth of the node embeddings for GeDi and GeDi + PANDA across layers for different datasets. We find that PANDA does not lead to disproportionate growth of node embedding norms. In fact, for almost half of the datasets, it exhibits a smaller coefficient of variation than the vanilla model.

Table 10: Average L2 norm growth of the node embeddings across layers for different datasets. A value of 1 means the L2 norm does not change across layers, <1 means the embeddings are contracting, while >1 means that they are expanding.

Model	`Telegram`	`Citeseer`	`Pubmed`	`Roman-empire`	`Cornell`	`Texas`	`Squirrel`	`email-Enron`
GeDi (Vanilla)	29.6782	1.0956	0.1784	1.2821	1.1428	1.0881	11.038	1.1938
GeDi + PANDA	61.9474	0.8936	0.9324	1.4452	1.5472	0.4988	1.8974	6.2111

Table 11: Average coefficient of variation for the L2 norm growth of the node embeddings across layers for different datasets. A value of 0 means that each node's L2 norm grows equally, while a high value means that the increase is disproportionate.

Model	`Telegram`	`Citeseer`	`Pubmed`	`Roman-empire`	`Cornell`	`Texas`	`Squirrel`	`email-Enron`
GeDi (Vanilla)	1.711	1.848	1.430	1.124	1.143	1.462	1.174	1.178
GeDi + PANDA	1.226	1.956	1.663	0.990	1.236	2.324	1.761	0.961

F Extended analysis

We analyse the learned GeDi-PANDA routing variables on Citeseer, Cornell, Roman-empire, Telegram, and Texas. For each dataset, we reload the ten saved checkpoints corresponding to the ten data splits and evaluate all quantities in inference mode. Reported accuracy values are means $\pm$ standard deviations over the ten checkpoints. Routing statistics are also reported as means $\pm$ standard deviations over the ten checkpoints. All standard deviations use the unbiased (sample) estimator. For the multi-layer Roman-empire model, routing statistics are first computed at each PANDA layer and then summarised over all layer–checkpoint instances, so the reported standard deviations mix layer-to-layer variation with checkpoint-to-checkpoint variation and are not directly comparable to the single-layer datasets; when this pooling hides substantial layer structure, we state the layerwise behaviour explicitly.

Table 12 collects the symbols used throughout this section. Routing variables describe how each PANDA layer combines learned attention, the structural prior, and the auxiliary path; auxiliary-path quantities are computed at inference time on the test split, with both per-incidence auxiliary gates forced to zero in the aux-off baseline.

F.1 Routing statistics and gate interventions

The model has two routing stages. In the node-to-hyperedge stage, denoted $\mathcal{V}\to\mathcal{E}$, hyperedges receive messages from incident nodes, with main and auxiliary coefficients $\hat{\alpha}^{\text{main}}_{e,v}$ and $\hat{\alpha}^{\text{aux}}_{e,v}$. In the hyperedge-to-node stage, denoted $\mathcal{E}\to\mathcal{V}$, nodes receive messages from incident hyperedges, with coefficients $\check{\alpha}^{\text{main}}_{u,e}$ and $\check{\alpha}^{\text{aux}}_{u,e}$. In both stages, the main coefficient interpolates between learned attention and the structural prior, while the auxiliary coefficient is bounded by the remaining local capacity, $1-\alpha^{\text{main}}$, and modulated by a learned sigmoid gate.

Table 13 first reports basic hypergraph statistics: the number of nodes $|\mathcal{V}|$, the number of hyperedges $|\mathcal{E}|$, the average hyperedge size $\overline{|e|}$, and the average node degree $\overline{|\Gamma(v)|}$. Then, it summarises six routing quantities. We denote by $\eta^{\text{gat}}_{\text{unif}}$ the fraction of hyperedges (in the $\mathcal{V}\to\mathcal{E}$ stage) or nodes (in the $\mathcal{E}\to\mathcal{V}$ stage) for which the learned attention component is effectively uniform over the same set used in the corresponding softmax. To avoid trivially-uniform singleton receiver groups, define the multi-element eligible sets $\mathcal{E}^{\geq 2}=\{e\in\mathcal{E}:|H(e)\cup T(e)|\geq 2\}$ and $\mathcal{V}^{\geq 2}=\{u\in\mathcal{V}:|\Gamma_H(u)\cup\Gamma_T(u)|\geq$

Table 12: Symbols used in the "Extended Analysis" section.

Routing variables	
α^{main}, α^{aux}	Main and auxiliary incidence coefficients ($\hat{\alpha}$ in $\mathcal{V}\to\mathcal{E}$, $\check{\alpha}$ in $\mathcal{E}\to\mathcal{V}$).
α^{gat}	Learned GAT component of the main coefficient.
β^{prior}	Per-incidence prior gate; large value $\Rightarrow$ structural prior dominates main path.
g	Post-sigmoid auxiliary gate.
$\eta^{\text{gat}}_{\text{unif}}$	Fraction of non-singleton receiver groups with effectively uniform α^{gat}.
$\rho_{s,R}$, $\pi_{s,R}$	Aux-to-main norm ratio and aux share of total norm, in stage $s \in \{\mathcal{V}\to\mathcal{E}, \mathcal{E}\to\mathcal{V}\}$, role $R \in \{H, T\}$.
Auxiliary-path quantities	
Acc^{full}, $\text{Acc}^{\text{aux-off}}$	Test accuracy of the full model and with both per-incidence auxiliary gates forced to zero.
$\Delta\text{Acc}.$	$\text{Acc}^{\text{aux-off}} - \text{Acc}^{\text{full}}$ in percentage points (negative means closing the auxiliary path hurts).
$\mathcal{H}_{\text{CE}}$	Dataset-level clique-expansion homophily [Wang et al., 2023b].
y_u^{nbr}	Majority class among u's clique-expansion neighbours.
q_{mis}	Fraction of CE-valid test nodes $u \in \mathcal{T}_{\text{CE}}$ with $y_u^{\text{nbr}} \neq y_u$ (*misleading neighbour*).
q_{help}, q_{hurt}	Fraction of test nodes the auxiliary path flips onto / off the true class.
$q_{\text{nbr-rescue}}$	Fraction of all test nodes rescued from the misleading neighbour-majority class to the true class (aux-off predicts y_u^{nbr}, full predicts y_u); on the same $\lvert\mathcal{T}\rvert$ scale as q_{help}.
$\overline{\Delta}^{\text{nbr}}_{\neq}$	Mean true-vs-neighbour-majority logit-margin shift on misleading-neighbour nodes; positive means the auxiliary path pushes towards the true class.
$\overline{\Delta}^{\text{max}}$	Mean true-vs-best-other-class logit-margin shift over all test nodes.

$2\}$. With tolerance $\varepsilon = 10^{-6}$, we compute

$$\eta^{\text{gat},\mathcal{V}\to\mathcal{E}}_{\text{unif}} = \frac{1}{|\mathcal{E}^{\geq 2}|} \sum_{e\in\mathcal{E}^{\geq 2}} \mathbb{I}\left[\max_{v\in H(e)\cup T(e)} \left|\hat{\alpha}^{\text{gat}}_{e,v} - \frac{1}{|H(e)\cup T(e)|}\right| < \varepsilon\right],$$
$$\eta^{\text{gat},\mathcal{E}\to\mathcal{V}}_{\text{unif}} = \frac{1}{|\mathcal{V}^{\geq 2}|} \sum_{u\in\mathcal{V}^{\geq 2}} \mathbb{I}\left[\max_{e\in\Gamma_H(u)\cup\Gamma_T(u)} \left|\check{\alpha}^{\text{gat}}_{u,e} - \frac{1}{|\Gamma_H(u)\cup\Gamma_T(u)|}\right| < \varepsilon\right]. \tag{29}$$

For multi-head models, this same check is applied independently to each attention head and then averaged over heads; we suppress the head index in the notation. Singleton receiver groups ($|H(e)\cup T(e)| = 1$ in the $\mathcal{V}\to\mathcal{E}$ stage, or $|\Gamma_H(u)\cup\Gamma_T(u)| = 1$ in the $\mathcal{E}\to\mathcal{V}$ stage) are trivially uniform and are excluded from both the numerator and denominator of $\eta^{\text{gat}}_{\text{unif}}$ so that the reported fraction reflects non-trivial uniformity only.[2] The same $\eta^{\text{gat}}_{\text{unif}}$ value is reported for head and tail rows within a stage because learned GAT attention is normalised over the same set, i.e., $H(e)\cup T(e)$ in the $\mathcal{V}\to\mathcal{E}$ stage and $\Gamma_H(u)\cup\Gamma_T(u)$ in the $\mathcal{E}\to\mathcal{V}$ stage. The remaining columns are role-split at analysis time by masking incidences according to their role: the H row in the $\mathcal{V}\to\mathcal{E}$ stage averages over $\{(e,v) : v \in H(e)\}$, the T row over $\{(e,v) : v \in T(e)\}$, analogously in the $\mathcal{E}\to\mathcal{V}$ stage over $\{(u,e) : e \in \Gamma_H(u)\}$ and $\{(u,e) : e \in \Gamma_T(u)\}$. The model itself still computes α^{gat}, β^{prior}, the gates $\hat{g}, \check{g}$, and α^{aux} on the full incidence support; the table only separates their empirical values by head and tail role.

On these role-restricted incidence sets, the columns β^{prior}, g, and α^{aux} report, respectively, the mean prior-gate mass, the mean post-sigmoid auxiliary gate, and the mean auxiliary coefficient, averaged over heads. The auxiliary-gate column g refers to $\hat{g}_{e,v}$ in the $\mathcal{V}\to\mathcal{E}$ stage and to $\check{g}_{u,e}$ in the $\mathcal{E}\to\mathcal{V}$ stage, as specified in the table notes. The reported β^{prior} is the empirical incidence-level mean of the per-incidence prior mass $1-\hat{\beta}^{(\ell)}_{e,v}$ (resp. $1-\check{\beta}^{(\ell)}_{u,e}$) from the convex combination of Eq. (11), averaged over the H- or T-restricted incidence set, over heads, and over checkpoints; a high β^{prior} corresponds to the prior dominating the main path. Because the per-incidence gates $\hat{\beta}^{(\ell)}_{e,v}$, $\check{\beta}^{(\ell)}_{u,e}$ take different values across incidences, the H and T rows can therefore differ. The table then reports the mean $\pm$ standard

[2] On the $\mathcal{V}\to\mathcal{E}$ side, singleton groups are size-1 hyperedges (self-loops, either present in the source data or inserted during preprocessing for otherwise-isolated nodes). On the $\mathcal{E}\to\mathcal{V}$ side, they are predominantly *leaf nodes* – nodes incident to exactly one multi-node hyperedge – rather than isolated nodes; the leaf fraction is substantial on most datasets (Citeseer 53.8%, Cornell 48.1%, Texas 42.1%, Telegram 21.6%, Roman-empire 0%). The post-exclusion $\eta^{\text{gat},\mathcal{E}\to\mathcal{V}}_{\text{unif}}$ is therefore the uniformity rate among nodes for which attention has more than one source to choose between.

Table 13: Analysis summary of learned incidence routing and the auxiliary path. The accuracy column reports mean ± standard deviation test accuracy in percent over ten saved checkpoints. Dataset statistics are computed on the loaded hypergraphs. Routing values are means ± standard deviations over the same checkpoints, and over layer–checkpoint instances for Roman-empire. The fraction of uniform attention is computed over non-singleton receiver groups, namely $H(e) \cup T(e)$ with $|H(e) \cup T(e)| \geq 2$ in the $\mathcal{V}\to\mathcal{E}$ stage and $\Gamma_H(u) \cup \Gamma_T(u)$ with $|\Gamma_H(u) \cup \Gamma_T(u)| \geq 2$ in the $\mathcal{E}\to\mathcal{V}$ stage.

Dataset	Test acc. mean±std	$\lvert\mathcal{V}\rvert$	$\lvert\mathcal{E}\rvert$	$\overline{\lvert e\rvert}$	$\overline{\lvert\Gamma(v)\rvert}$	Stage	Role	$\eta^{\mathrm{gat}}_{\mathrm{unif}}$	β^{prior}	g	α^{aux}	$\rho_{s,R}$	$\pi_{s,R}$
Citeseer	75.6 ± 1.2	3312	1951	3.35	1.98	$\mathcal{V}\to\mathcal{E}$	H	0.000±0.000	0.863±0.045	0.378±0.041	0.243±0.027	0.320±0.047	0.241±0.027
							T		0.862±0.046	0.395±0.040	0.253±0.027	0.335±0.047	0.250±0.027
						$\mathcal{E}\to\mathcal{V}$	H	0.005±0.000	0.930±0.040	0.390±0.052	0.242±0.033	0.269±0.041	0.211±0.025
							T		0.844±0.042	0.399±0.050	0.247±0.032	0.293±0.042	0.226±0.025
Cornell	74.9 ± 4.8	183	96	4.07	2.14	$\mathcal{V}\to\mathcal{E}$	H	0.429±0.001	0.368±0.100	0.181±0.079	0.140±0.062	0.294±0.131	0.220±0.078
							T		0.379±0.093	0.207±0.090	0.117±0.051	0.124±0.056	0.108±0.044
						$\mathcal{E}\to\mathcal{V}$	H	0.018±0.057	0.832±0.055	0.292±0.083	0.215±0.066	0.204±0.089	0.166±0.062
							T		0.732±0.096	0.368±0.083	0.190±0.044	0.183±0.050	0.154±0.036
Roman-empire	89.2 ± 0.5	22662	22662	2.73	2.73	$\mathcal{V}\to\mathcal{E}$	H	0.184±0.270	0.831±0.079	0.497±0.140	0.329±0.093	0.959±0.262	0.480±0.074
							T		0.828±0.078	0.490±0.129	0.306±0.077	0.805±0.219	0.438±0.071
						$\mathcal{E}\to\mathcal{V}$	H	0.001±0.004	0.908±0.066	0.370±0.132	0.247±0.085	0.764±0.250	0.423±0.076
							T		0.874±0.075	0.413±0.134	0.268±0.082	0.830±0.328	0.440±0.082
Telegram	89.6 ± 5.2	245	183	49.70	37.12	$\mathcal{V}\to\mathcal{E}$	H	1.000±0.000	0.500±0.000	0.411±0.031	0.404±0.031	27.578±4.087	0.964±0.005
							T		0.500±0.000	0.410±0.031	0.382±0.029	6.423±0.934	0.864±0.016
						$\mathcal{E}\to\mathcal{V}$	H	0.000±0.000	0.551±0.292	0.123±0.025	0.120±0.025	3.153±0.507	0.756±0.029
							T		0.553±0.221	0.228±0.028	0.200±0.028	0.533±0.441	0.306±0.163
Texas	83.0 ± 5.1	183	110	3.81	2.29	$\mathcal{V}\to\mathcal{E}$	H	0.414±0.035	0.490±0.034	0.272±0.067	0.211±0.053	0.393±0.205	0.270±0.090
							T		0.490±0.031	0.300±0.078	0.166±0.043	0.136±0.038	0.119±0.030
						$\mathcal{E}\to\mathcal{V}$	H	0.001±0.002	0.785±0.106	0.304±0.106	0.231±0.086	0.226±0.141	0.175±0.093
							T		0.691±0.041	0.399±0.081	0.201±0.042	0.173±0.065	0.145±0.047

Notes. In the $\mathcal{V}\to\mathcal{E}$ stage, the routing variables are $\hat{\alpha}^{\mathrm{gat}}_{e,v}$, $1-\hat{\beta}^{(\ell)}_{e,v}$, $\hat{g}_{e,v}$, and $\hat{\alpha}^{\mathrm{aux}}_{e,v}$. In the $\mathcal{E}\to\mathcal{V}$ stage, they are $\check{\alpha}^{\mathrm{gat}}_{u,e}$, $1-\check{\beta}^{(\ell)}_{u,e}$, $\check{g}_{u,e}$, and $\check{\alpha}^{\mathrm{aux}}_{u,e}$. $\rho_{s,R}$ and $\pi_{s,R}$ are defined in Eq. (32), for stage $s \in \{\mathcal{V}\to\mathcal{E}, \mathcal{E}\to\mathcal{V}\}$ and role $R \in \{H, T\}$.

deviation of these checkpoint-level quantities. Accuracy is reported with the same mean ± standard deviation convention.

The last two columns compare the magnitudes of the main and auxiliary messages. For this purpose, we define the incidence-level messages

$$\begin{aligned} \hat{m}^{\mathrm{main}}_{e,v} &= \hat{\alpha}^{\mathrm{main}}_{e,v}\,\mathbf{x}^{\mathrm{main}}_{v}, \quad \hat{m}^{\mathrm{aux}}_{e,v} = \hat{\alpha}^{\mathrm{aux}}_{e,v}\,\mathbf{x}^{\mathrm{aux}}_{v}, \\ \check{m}^{\mathrm{main}}_{u,e} &= \check{\alpha}^{\mathrm{main}}_{u,e}\,\mathbf{h}^{\mathrm{main}}_{e}, \quad \check{m}^{\mathrm{aux}}_{u,e} = \check{\alpha}^{\mathrm{aux}}_{u,e}\,\mathbf{h}^{\mathrm{aux}}_{e}. \end{aligned} \tag{30}$$

For the head side, let

$$\begin{aligned} M^{\mathrm{main}}_{\mathcal{V}\to\mathcal{E},H} &= \sum_{e\in\mathcal{E}}\sum_{v\in H(e)} \|\hat{m}^{\mathrm{main}}_{e,v}\|_2, \quad M^{\mathrm{aux}}_{\mathcal{V}\to\mathcal{E},H} = \sum_{e\in\mathcal{E}}\sum_{v\in H(e)} \|\hat{m}^{\mathrm{aux}}_{e,v}\|_2, \\ M^{\mathrm{main}}_{\mathcal{E}\to\mathcal{V},H} &= \sum_{u\in\mathcal{V}}\sum_{e\in\Gamma_H(u)} \|\check{m}^{\mathrm{main}}_{u,e}\|_2, \quad M^{\mathrm{aux}}_{\mathcal{E}\to\mathcal{V},H} = \sum_{u\in\mathcal{V}}\sum_{e\in\Gamma_H(u)} \|\check{m}^{\mathrm{aux}}_{u,e}\|_2. \end{aligned} \tag{31}$$

The corresponding tail-side quantities are obtained by replacing $H(e)$ with $T(e)$ in the $\mathcal{V}\to\mathcal{E}$ sums and $\Gamma_H(u)$ with $\Gamma_T(u)$ in the $\mathcal{E}\to\mathcal{V}$ sums. From these aggregate magnitudes, we report

$$\rho_{s,R} = \frac{M^{\mathrm{aux}}_{s,R}}{M^{\mathrm{main}}_{s,R}}, \qquad \pi_{s,R} = \frac{M^{\mathrm{aux}}_{s,R}}{M^{\mathrm{main}}_{s,R} + M^{\mathrm{aux}}_{s,R}}, \tag{32}$$

where $s \in \{\mathcal{V}\to\mathcal{E}, \mathcal{E}\to\mathcal{V}\}$ and $R \in \{H, T\}$. Thus, $\rho > 1$ means that the auxiliary path has larger total norm than the main path, while $\pi = 0$ means that the auxiliary path is inactive and $\pi = 1$ means that almost all message magnitude comes from the auxiliary path. These are magnitude statistics, not causal tests: the auxiliary representation includes learned linear maps, so the functional role of a large norm must be checked by intervention.

Table 13 shows that PANDA realises different routing regimes across datasets. Telegram is the clearest high-auxiliary-magnitude regime. Its learned $\mathcal{V}\to\mathcal{E}$ attention component is uniform in all eligible non-singleton receiver groups and the empirical prior mass is fixed at $\beta^{\mathrm{prior}} = 0.500 \pm 0.000$. Therefore, in the $\mathcal{V}\to\mathcal{E}$ stage, the learned GAT component does not provide non-uniform routing; any non-uniformity in the main path must therefore come from the structural prior rather than from

Table 14: Norms of weight matrices and bias vectors in each stage of message passing for the single layer GeDi + PANDA model for Telegram dataset.

Stage	norm(W_β)	norm(u_β)
$V \to E$	0.00272 ± 0.00436	0.07674 ± 0.14079
$E \to V$	0.54175 ± 0.32409	0.41068 ± 0.23789

learned GAT attention, while the auxiliary path carries most of the message magnitude. Specifically, $\pi_{\mathcal{V}\to\mathcal{E},H} = 0.964 \pm 0.005$ on head-side incidences and $\pi_{\mathcal{V}\to\mathcal{E},T} = 0.864 \pm 0.016$ on tail-side incidences. Equivalently, $\rho_{\mathcal{V}\to\mathcal{E},H} = 27.578 \pm 4.087$ and $\rho_{\mathcal{V}\to\mathcal{E},T} = 6.423 \pm 0.934$, so the auxiliary norm is much larger than the main-path norm in the $\mathcal{V}\to\mathcal{E}$ stage. Because these are norm ratios, the interventions below are needed to establish that this high-magnitude auxiliary path is also predictive.

We note that the $\beta^{prior} = 0.500 \pm 0.000$ pinning is specific to Telegram's $V \to E$ stage. This is likely due to Telegram's degenerate node features, rather than being a general failure of the interpolation mechanism. Indeed, on Telegram each node carries a single scalar feature $x_v \sim \mathcal{N}(0, 1)$ which is drawn independently of labels and topology. This feature is clearly poorly informative in the $V \to E$ layer, where all the nodes belonging to a hyperedge are scored based on embeddings calculated after a linear layer from these uninformative 1-dimensional init features (and also against the same, shared hyperedge representation h_e). Thus, the value of β^{prior} calculated in Eqs. (12) and (13) at the $V \to E$ stage is entirely uninformative. It follows that the data-dependent component of the CE part of the loss likely provides no informative gradient signal. Thus, the L2 penalty term of the loss becomes dominant, leading to a choice of weights W_β close to zero. This behaviour is observed in all ten independent runs, with the trained W_β, u_β are both extremely small (of the order of 10^{-3} in both cases, in Frobenius norm). As a result, the sigmoid gate in both equations collapses due to the learned scores W being close to zero for every node-hyperedge incidence, yielding $\beta^{prior} = 0.500 \pm 0.000$.

In contrast, the $E \to V$ stage of the two-stage message-passing protocol scores the nodes against aggregated hyperedge representations. There, the scoring parameters retain norms that are one to two orders of magnitude larger and produce non-uniform attention, confirming that the observed degeneracy is caused by the input embedding of the $V \to E$ stage rather than by the attention mechanism itself. Table 14 supports this hypothesis.

Roman-empire also uses the auxiliary path substantially, but with a different profile. Its $\mathcal{V}\to\mathcal{E}$ learned-attention uniformity is 0.184 ± 0.270 after pooling over the four PANDA layers, which reflects the layer structure: the layerwise means are 0.548, 0.088, 0.090, and 0.010 from the first to the fourth layer. Its $\mathcal{E}\to\mathcal{V}$ learned attention is almost never uniform (0.001 ± 0.004). The prior gate remains high ($\beta^{\text{prior}} \approx 0.83$–$0.91$), and the auxiliary path accounts for roughly 42–48% of the message norm across stages and roles. Citeseer, Cornell, and Texas have smaller auxiliary shares: across all four (s, R) combinations, $\pi_{s,R} \in [0.21, 0.25]$ on Citeseer, $\pi_{s,R} \in [0.11, 0.22]$ on Cornell, and $\pi_{s,R} \in [0.12, 0.27]$ on Texas. In all three cases $\rho_{s,R} < 1$ on average, so the auxiliary messages are active but do not dominate the main-path norm. The prior gate also changes across stages: Citeseer and Roman-empire rely strongly on the prior in both stages, while Cornell and Texas move more strongly towards the GeDi prior in the $\mathcal{E}\to\mathcal{V}$ stage than in the $\mathcal{V}\to\mathcal{E}$ stage.

To test whether the auxiliary-path statistics correspond to predictive behaviour, Table 15 reports controlled interventions on the same checkpoints. We close the auxiliary gate by forcing the post-sigmoid value $\hat{g}_{e,v}^{(\ell)}$ (in the $\mathcal{V}\to\mathcal{E}$ stage) or $\check{g}_{u,e}^{(\ell)}$ (in the $\mathcal{E}\to\mathcal{V}$ stage) to zero for every incidence and every layer, and evaluate the model again on the same test split. We report the signed change in accuracy, ΔAcc., defined as the intervened accuracy minus baseline accuracy, in percentage points; negative values therefore mean that closing the auxiliary gate hurts performance.

Table 15 confirms that the auxiliary path has different predictive roles across datasets. On Telegram, closing either gate, or both, causes large drops in test accuracy. Roman-empire is also strongly dependent on the auxiliary path: closing both gates reduces accuracy by 27.7 percentage points on average. These results show that the large auxiliary shares in Table 13 correspond to predictive behaviour on Telegram and Roman-empire, not merely to representational magnitude. Citeseer, Cornell, and Texas show much smaller mean effects. In these datasets, the auxiliary path is available and sometimes active in magnitude, but the final predictions are less dependent on it. The intervention

Table 15: Auxiliary-gate shut-off interventions. The intervention column specifies which post-sigmoid auxiliary gate value is forced to zero. Values are mean ± standard deviation test-accuracy changes in percentage points over ten saved checkpoints; baseline accuracy is reported with the same convention. Negative values mean that the intervention decreases accuracy. Note that this measures the dependence of the model on the auxiliary path, not the marginal contribution of this path.

Dataset	Test acc. mean±std	Intervention	ΔAcc. after gate shut-off
Citeseer	75.6 ± 1.2	$\hat{g}_{e,v}^{(\ell)} = 0$	-0.5 ± 0.5
		$\check{g}_{u,e}^{(\ell)} = 0$	-0.5 ± 0.3
		both gates $= 0$	-1.0 ± 0.6
Cornell	74.9 ± 4.8	$\hat{g}_{e,v}^{(\ell)} = 0$	-4.1 ± 7.9
		$\check{g}_{u,e}^{(\ell)} = 0$	-0.5 ± 1.1
		both gates $= 0$	-4.6 ± 9.7
Roman-empire	89.2 ± 0.5	$\hat{g}_{e,v}^{(\ell)} = 0$	-19.4 ± 3.7
		$\check{g}_{u,e}^{(\ell)} = 0$	-21.0 ± 2.6
		both gates $= 0$	-27.7 ± 3.3
Telegram	89.6 ± 5.2	$\hat{g}_{e,v}^{(\ell)} = 0$	-53.8 ± 12.7
		$\check{g}_{u,e}^{(\ell)} = 0$	-41.0 ± 9.2
		both gates $= 0$	-54.0 ± 5.2
Texas	83.0 ± 5.1	$\hat{g}_{e,v}^{(\ell)} = 0$	-1.4 ± 5.6
		$\check{g}_{u,e}^{(\ell)} = 0$	$+0.5 \pm 2.1$
		both gates $= 0$	$+0.0 \pm 4.0$

Table 16: Per-layer auxiliary-gate interventions on Roman-empire (four PANDA layers, ten saved checkpoints). LOO shuts off both per-incidence auxiliary gates only at the named layer; cumulative shuts off both gates at layers $1, \ldots, k$ jointly. Values are mean test accuracy ± standard deviation in percent; ΔAcc. is the corresponding accuracy change vs. the full model (89.2 ± 0.5) in percentage points.

Intervention	Layer(s) off	Test acc. (%)	ΔAcc. (pp)
LOO	$\{1\}$	81.8 ± 3.0	-7.4 ± 2.8
LOO	$\{2\}$	83.3 ± 1.8	-5.8 ± 1.5
LOO	$\{3\}$	85.1 ± 2.8	-4.1 ± 2.4
LOO	$\{4\}$	84.9 ± 3.2	-4.3 ± 3.4
Cumulative	$\{1\}$	81.8 ± 3.0	-7.4 ± 2.8
Cumulative	$\{1, 2\}$	72.5 ± 4.7	-16.7 ± 4.7
Cumulative	$\{1, 2, 3\}$	67.3 ± 4.3	-21.9 ± 4.1
Cumulative	$\{1, 2, 3, 4\}$	61.5 ± 3.2	-27.7 ± 3.3

tells us *whether* the auxiliary path matters but not *what kind of predictive effect* it produces; we probe the latter below.

Per-layer interventions on Roman-empire. Because Roman-empire is the only dataset analysed here that uses a multi-layer PANDA configuration (four layers), the all-aux-off intervention of Table 15 aggregates four layerwise contributions into a single number. The layerwise GAT-uniformity profile reported earlier already shows that the four layers behave differently ($\eta_{\text{unif}}^{\text{gat},\mathcal{V}\to\mathcal{E}} = 0.548,\ 0.088,\ 0.090,\ 0.010$ from the first to the fourth layer), so we ask which layers' auxiliary paths carry the predictive load. We run two complementary inference-time interventions on the same ten checkpoints. *Leave-one-out* (LOO) intervention shuts off both per-incidence auxiliary gates only at one chosen layer ℓ and leaves the other three layers' auxiliary gates active; *cumulative* intervention shuts off both per-incidence auxiliary gates at layers $1, \ldots, k$ jointly.

The per-layer intervention in Table 16 reveals three things. First, the predictive role of the auxiliary path is distributed across all four layers: every LOO drop is non-trivial and no single layer's removal

reproduces the full -27.7 point all-off drop. Second, the first layer carries the largest LOO load (-7.4 percentage points), which lines up with its having the highest learned-attention uniformity among the four layers ($\eta_{\text{unif}}^{\text{gat},\mathcal{V}\to\mathcal{E}} = 0.548$ at layer 1). This is consistent with the auxiliary path being especially important in layers where learned GAT attention contributes less non-uniform routing. Third, the single-layer LOO accuracy losses sum to 21.6 percentage points but the joint all-off accuracy loss is 27.7 points, so the intervention effects are non-additive: the joint loss exceeds the sum of the single-layer losses. This non-additivity is consistent with cross-layer interaction among auxiliary paths and with partial compensation when only one layer's auxiliary path is removed. Reading the cumulative column makes the same point: removing layer 2's auxiliary gates *after* layer 1's auxiliary gates are already off costs an additional 9.3 points, well above its standalone LOO cost of 5.8, so layer 2's auxiliary path becomes substantially more critical once layer 1's is unavailable.

F.2 Auxiliary Path Analysis

Table 15 already establishes that the auxiliary path is predictively necessary on Telegram and Roman-empire (-54.0 and -27.7 percentage-point accuracy drops when both per-incidence auxiliary gates are shut off), and only weakly on Citeseer, Cornell, and Texas. The analysis in this subsection therefore asks *along which axis* it acts: whether its predictive dependence is concentrated along the misleading neighbour-majority axis, or whether it reflects a broader margin-improvement effect spanning many class boundaries, and whether the auxiliary path mainly rescues aux-off-wrong nodes or reinforces nodes that aux-off already predicts correctly. All interventions in this subsection operate at inference time on a fixed trained model; they therefore measure the trained model's predictive dependence on the auxiliary path, not the architectural necessity of having an auxiliary path at training time. For a test node u, let

$$\mathcal{N}_{\text{CE}}(u) = \{v \in \mathcal{V} \setminus \{u\} : \exists e \in \mathcal{E} \text{ such that } u, v \in H(e) \cup T(e)\} \tag{33}$$

be its unique role-agnostic clique-expansion neighbours in the loaded hypergraph. We use this only as a diagnostic neighbourhood, following the common reduction of hyperedges to pairwise node neighbourhoods [Zhou et al., 2006, Yadati et al., 2019, Zhang et al., 2022]; it does not change the directed incidence support used by PANDA. Let $y_u \in \{1, \dots, C\}$ denote the ground-truth class label of node u, with C the number of classes. We define the neighbour-majority class as

$$y_u^{\text{nbr}} = \arg\max_c \sum_{v \in \mathcal{N}_{\text{CE}}(u)} \mathbb{I}[y_v = c]\,. \tag{34}$$

Ties in the $\arg\max$ are broken by the fixed integer ordering of the class labels. The neighbour-majority class y_u^{nbr} is defined only on nodes with $\mathcal{N}_{\text{CE}}(u) \neq \emptyset$; on the analysed datasets, every test node has at least one CE neighbour except for Citeseer, where roughly 1.4% of test nodes per run are isolated and are excluded from the y_u^{nbr}-referencing rates below.

Let $o_{u,c}^{\text{full}}$ be the output logit assigned by the original model to class c for node u, and let $o_{u,c}^{\text{aux-off}}$ be the corresponding output logit obtained by forcing both per-incidence auxiliary gates to zero, i.e., $\hat{g}_{e,v}^{(\ell)} = 0$ and $\check{g}_{u,e}^{(\ell)} = 0$ at every layer and every incidence (the same intervention used in Table 15). The corresponding per-node predictions are $\hat{y}_u^{\text{full}} = \arg\max_c o_{u,c}^{\text{full}}$ and $\hat{y}_u^{\text{aux-off}} = \arg\max_c o_{u,c}^{\text{aux-off}}$ (ties broken by the same fixed class ordering used for y_u^{nbr}). Let $\mathcal{T}$ denote the set of test nodes for the run, and let $\mathcal{T}_{\text{CE}} = \{u \in \mathcal{T} : \mathcal{N}_{\text{CE}}(u) \neq \emptyset\}$ be the subset on which y_u^{nbr} is defined. All quantities below are sample statistics computed over $\mathcal{T}$ or $\mathcal{T}_{\text{CE}}$ depending on whether they reference y_u^{nbr}. For any node-level event $E(u)$ and any subset $\mathcal{S} \subseteq \mathcal{T}$ we write

$$\widehat{\Pr}_{u\in\mathcal{S}}[E(u)] = \frac{1}{|\mathcal{S}|} \sum_{u \in \mathcal{S}} \mathbb{I}[E(u)]\,,$$

and define the test accuracies

$$\text{Acc}^{\text{full}} = \widehat{\Pr}_{u\in\mathcal{T}}\big[\hat{y}_u^{\text{full}} = y_u\big]\,, \qquad \text{Acc}^{\text{aux-off}} = \widehat{\Pr}_{u\in\mathcal{T}}\big[\hat{y}_u^{\text{aux-off}} = y_u\big]\,.$$

Table 17 reports the following quantities. The accuracy-gap column is

$$\Delta\text{Acc.} = \text{Acc}^{\text{aux-off}} - \text{Acc}^{\text{full}},$$

Table 17: Neighbour-majority analysis of the auxiliary path. $\mathcal{H}_{\mathrm{CE}}$ is the dataset-level clique-expansion homophily reported by Wang et al. [2023b]. $q_{\text{nbr-rescue}}$ is the unconditional fraction of test nodes (over $|\mathcal{T}|$) for which aux-off follows the misleading neighbour-majority class and the full model predicts the true class. $\overline{\Delta}_{\neq}^{\mathrm{nbr}}$ is the true-vs-neighbour-majority logit-margin shift on nodes with $y_u^{\mathrm{nbr}} \neq y_u$.

Dataset	$\mathcal{H}_{\mathrm{CE}}$	ΔAcc.	q_{mis}	q_{help}	q_{hurt}	$q_{\text{nbr-rescue}}$	$\overline{\Delta}_{\neq}^{\mathrm{nbr}}$
Citeseer	0.795	-1.0	23.97	1.84	0.82	0.22	-0.28
Cornell	0.420	-4.6	57.84	5.95	1.35	1.08	$+0.09$
Roman-empire	0.236	-27.7	95.68	29.44	1.77	2.20	$+5.29$
Telegram	0.285	-54.0	81.73	56.54	2.50	16.35	$+3.58$
Texas	0.505	0.0	47.03	3.24	3.24	1.35	$+0.09$

expressed in percentage points, so that negative values mean that shutting off the auxiliary path hurts accuracy, matching the sign convention of Table 15. The full-model accuracy is reported in Table 15, and $\mathrm{Acc}^{\text{aux-off}}$ is obtained by applying the "both gates $= 0$" accuracy change to the corresponding full-model accuracy. The column q_{mis} is the fraction of test nodes for which the neighbour-majority class is misleading:

$$q_{\mathrm{mis}} = \widehat{\Pr}_{u \in \mathcal{T}_{\mathrm{CE}}}\left[y_u^{\mathrm{nbr}} \neq y_u\right].$$

The class-agnostic flip rates are

$$q_{\mathrm{help}} = \widehat{\Pr}_{u \in \mathcal{T}}\left[\hat{y}_u^{\text{aux-off}} \neq y_u \ \wedge\ \hat{y}_u^{\text{full}} = y_u\right] \quad q_{\mathrm{hurt}} = \widehat{\Pr}_{u \in \mathcal{T}}\left[\hat{y}_u^{\text{aux-off}} = y_u \ \wedge\ \hat{y}_u^{\text{full}} \neq y_u\right],$$

which measure the fractions of test nodes where the auxiliary path flips the prediction onto or off of the true class, regardless of the neighbour-majority class. By construction, $\Delta\mathrm{Acc.} = q_{\mathrm{hurt}} - q_{\mathrm{help}}$, so these two rates decompose the accuracy gap into a gross-help and gross-hurt component. The column $q_{\text{nbr-rescue}}$ counts the test nodes for which the aux-off model follows the misleading neighbour-majority class while the full model predicts the true class, normalised over the full test set. Writing $\mathcal{T}_{\neq} = \{u \in \mathcal{T}_{\mathrm{CE}} : y_u^{\mathrm{nbr}} \neq y_u\}$,

$$q_{\text{nbr-rescue}} = \widehat{\Pr}_{u \in \mathcal{T}}\left[u \in \mathcal{T}_{\neq} \ \wedge\ \hat{y}_u^{\text{aux-off}} = y_u^{\mathrm{nbr}} \ \wedge\ \hat{y}_u^{\text{full}} = y_u\right].$$

The ratio between $q_{\text{nbr-rescue}}$ and q_{help} measures the share of helped flips attributable to strict neighbour-majority rescue.

Finally, we measure the logit-margin shift from the neighbour-majority class to the true one induced by the auxiliary path:

$$\Delta_u^{\mathrm{nbr}} = \left(o_{u,y_u}^{\text{full}} - o_{u,y_u^{\mathrm{nbr}}}^{\text{full}}\right) - \left(o_{u,y_u}^{\text{aux-off}} - o_{u,y_u^{\mathrm{nbr}}}^{\text{aux-off}}\right). \tag{35}$$

Δ_u^{nbr} is identically zero when $y_u = y_u^{\mathrm{nbr}}$ (both terms compare a class logit with itself), so we report only its mean on $\mathcal{T}_{\neq}$:

$$\overline{\Delta}_{\neq}^{\mathrm{nbr}} = \frac{1}{|\mathcal{T}_{\neq}|} \sum_{u \in \mathcal{T}_{\neq}} \Delta_u^{\mathrm{nbr}}.$$

A positive $\overline{\Delta}_{\neq}^{\mathrm{nbr}}$ means that, on average, the auxiliary path shifts the true-vs-neighbour-majority logit difference in favour of the true class. This is a directional statement, not an absolute one: the difference may still be negative under both models (neighbour-majority still wins, by a smaller gap), flip sign, or grow from positive to more positive — the shift is detectable even when the final argmax does not change.

Table 17 separates a margin-level neighbour-majority effect ($\overline{\Delta}_{\neq}^{\mathrm{nbr}}$) from a strict argmax-level rescue effect ($q_{\text{nbr-rescue}}$), and shows that the auxiliary path's behaviour on the neighbour axis is dataset-dependent and qualitatively aligned with clique-expansion homophily. On Roman-empire and Telegram, where $\mathcal{H}_{\mathrm{CE}}$ is low and q_{mis} is high (95.7% and 81.7%), the auxiliary path strongly increases the true-vs-neighbour-majority margin ($\overline{\Delta}_{\neq}^{\mathrm{nbr}} = +5.29$ and $+3.58$). On these low-homophily datasets, where the clique-expansion neighbourhood is often unreliable, the auxiliary path therefore acts as a counter-neighbourhood signal.

Table 18: Conditional split of $\Delta_u^{\max}$ by aux-off correctness. n is the mean subset size per checkpoint; $\% > 0$ is the fraction of nodes in the subset with positive $\Delta_u^{\max}$. n_{hurt} counts aux-off-correct nodes flipped to wrong by the full model, while n_{help} counts aux-off-wrong nodes flipped to correct. All columns are means over ten saved checkpoints.

	$u \in \mathcal{T}_{\checkmark}^{\text{aux-off}}$ (aux-off correct)				$u \in \mathcal{T}_{\times}^{\text{aux-off}}$ (aux-off wrong)			
Dataset	n	$\overline{\Delta}^{\max}$	$\% > 0$	n_{hurt}	n	$\overline{\Delta}^{\max}$	$\% > 0$	n_{help}
Citeseer	618	+0.19	69.6	6.8	210	+0.09	41.9	15.2
Cornell	26	+0.75	71.7	0.5	11	+0.58	55.3	2.2
Roman-empire	3484	+2.99	83.3	100.5	2182	+13.97	91.1	1668.0
Telegram	18	+0.14	61.5	1.3	34	+9.39	98.5	29.4
Texas	31	+0.03	50.8	1.2	6	+0.42	50.4	1.2

Strict neighbour-majority rescue, however, explains only part of that gain. On Roman-empire $q_{\text{nbr-rescue}} = 2.20\%$ is far below $q_{\text{help}} = 29.44\%$ ($2.20/29.44 \approx 7.5\%$; about 125 of 5666 test nodes per run), and on Telegram the corresponding values are 16.35% and 56.54% ($\approx 28.9\%$). The auxiliary path therefore does not merely correct predictions that follow the misleading neighbour-majority class; its main predictive role, especially on Roman-empire, is broader margin improvement against many competing classes.

Citeseer shows the opposite pattern. It is the most homophilic dataset ($\mathcal{H}_{\text{CE}} = 0.795$), the auxiliary intervention has only a small effect ($\Delta\text{Acc.} = -1.0$), and $\overline{\Delta}_{\neq}^{\text{nbr}} = -0.28$: on the rare misleading-neighbour nodes, the auxiliary path mildly leans *with* the neighbour-majority, consistent with the fact that the neighbour-majority is usually informative on homophilic graphs. The auxiliary path is therefore not universally anti-neighbour-majority; in these experiments it pushes against the neighbour-majority mainly on the datasets where the neighbourhood is unreliable. Cornell and Texas sit between these regimes with mid-range $\mathcal{H}_{\text{CE}}$ and small intervention effects, so their $\overline{\Delta}_{\neq}^{\text{nbr}}$ values are close to zero.

Generalised margin shift $\Delta_u^{\max}$ and conditional split by aux-off correctness

The fact that $q_{\text{nbr-rescue}}$ is much smaller than q_{help}, especially on Roman-empire, motivates a generalised margin analysis: if the auxiliary path is not only correcting neighbour-majority mistakes, then it should also improve the true-class margin against non-neighbour-majority competitors. The neighbour-majority margin $\overline{\Delta}_{\neq}^{\text{nbr}}$ focuses on one specific axis, the true class versus the neighbour-majority class. To also cover corrections whose strongest competitor is not the neighbour-majority class, we report a second margin shift defined against the best non-true class at each node:

$$\Delta_u^{\max} = \left(o_{u,y_u}^{\text{full}} - \max_{c \neq y_u} o_{u,c}^{\text{full}} \right) - \left(o_{u,y_u}^{\text{aux-off}} - \max_{c \neq y_u} o_{u,c}^{\text{aux-off}} \right). \tag{36}$$

The competitor class is chosen independently of the neighbour-majority and independently in the full and aux-off logits, so $\Delta_u^{\max}$ is defined for every test node and is positive whenever the auxiliary path shifts the logit difference between the true class and its strongest competitor in favour of the true class. We report its sample mean over all test nodes:

$$\overline{\Delta}^{\max} = \frac{1}{|\mathcal{T}|} \sum_{u \in \mathcal{T}} \Delta_u^{\max}.$$

We focus on two distinct services the auxiliary path can render: *reinforcing* the true-class margin on nodes the aux-off model already predicts correctly, and *rescuing* nodes the aux-off model gets wrong by closing (or flipping) their true-vs-best-other gap. Splitting the test set by the aux-off prediction's correctness separates the two. Let $\mathcal{T}_{\checkmark}^{\text{aux-off}} = \{u \in \mathcal{T} : \hat{y}_u^{\text{aux-off}} = y_u\}$ and $\mathcal{T}_{\times}^{\text{aux-off}} = \mathcal{T} \setminus \mathcal{T}_{\checkmark}^{\text{aux-off}}$, and write $n_{\checkmark} = |\mathcal{T}_{\checkmark}^{\text{aux-off}}|$ and $n_{\times} = |\mathcal{T}_{\times}^{\text{aux-off}}|$ for the corresponding per-checkpoint subset sizes (so $n_{\checkmark} + n_{\times} = |\mathcal{T}|$ by construction). Table 18 reports the mean of $\Delta_u^{\max}$ and the fraction of positive $\Delta_u^{\max}$ on each subset, separately for each dataset; the n columns of that table report the checkpoint-mean of $n_{\checkmark}$ and $n_{\times}$.

Two interpretive notes apply throughout. $\Delta_u^{\max}$ is a continuous logit-margin shift, not an argmax-flip indicator: on the aux-off-wrong subset a positive $\Delta_u^{\max}$ is necessary for a wrong-to-right flip (the

true-vs-best-other gap must cross zero) but not sufficient – the gap can shrink without crossing it, leaving the prediction wrong with a smaller margin. The $\%>0$ column on the aux-off-wrong subset is therefore an upper bound on the actual flip-to-correct rate, which is given by $n_{\mathrm{help}}/n_{\times}$. On the aux-off-correct subset there is no such ambiguity for the reinforcement reading: aux-off-correct implies that the true-class logit is maximal under the tie-breaking rule, which (ignoring measure-zero ties) corresponds to $(o_{u,y_u} - \max_{c \neq y_u} o_{u,c})^{\text{aux-off}} > 0$, so a positive $\Delta_u^{\max}$ means the gap is even larger and the prediction stays correct; $n_{\mathrm{hurt}}/n_{\checkmark}$ gives the opposite-direction flip rate (correct to wrong).

Table 18 is intended as a rescue-versus-reinforcement diagnostic rather than additional evidence that the auxiliary path is useful, since that case is already made by Table 15. The two subsets answer different questions: the aux-off-correct subset shows whether the auxiliary path widens already-positive gaps (reinforcement), while the aux-off-wrong subset shows whether it can flip negative gaps to positive (rescue). Both effects can coexist, and their relative size is what we read off the table.

Roman-empire is the clearest case where both mechanisms are active. On the aux-off-correct subset ($n_{\checkmark} = 3484$), the per-node mean shift is $\overline{\Delta}^{\max} = +2.99$ with 83.3% positive and only ~ 100 hurt nodes per checkpoint, so reinforcement is real but moderate in magnitude. On the aux-off-wrong subset ($n_{\times} = 2182$), the per-node mean shift is much larger ($+13.97$), 91.1% positive, and 1668 of those 2182 nodes flip to correct ($n_{\mathrm{help}}/n_{\times} \approx 76\%$). The total accuracy contribution is therefore dominated by rescue: the per-node shift is roughly $5\times$ larger on aux-off-wrong than on aux-off-correct, and 1668 rescued nodes vastly outnumber the $\sim$100 hurt by reinforcement.

Telegram is sharper: rescue is essentially the only mechanism. The aux-off-correct subset has only $n_{\checkmark} = 18$ nodes per checkpoint with a small mean shift ($+0.14$) and a near-coin-flip 61.5% positive – there is little reinforcement to do because there are few aux-off-correct nodes to begin with. The aux-off-wrong subset has $n_{\times} = 34$ nodes per checkpoint with mean shift $+9.39$, 98.5% positive, and 29.4 of those 34 nodes flip to correct ($\approx 86\%$). The auxiliary path is acting almost exclusively as a rescue mechanism on Telegram.

Citeseer does not show a consistent rescue signal. On its aux-off-wrong subset ($n_{\times} = 210$), only 41.9% of nodes have a positive shift – below 50%, so on average the auxiliary path shrinks the gap on as many wrong nodes as it widens it. The mean shift ($+0.09$) is small on the logit scale, and only 15.2 of 210 wrong nodes flip to correct ($\approx$7%). This is consistent with Citeseer's small intervention effect in Table 15: the auxiliary path is active in magnitude but does not translate into coherent rescue behaviour.

Cornell and Texas have too few aux-off-wrong nodes per checkpoint ($n_{\times} \approx 11$ and $n_{\times} \approx 6$ respectively) for the rescue percentages to be statistically meaningful – a single misclassified node moves the rate by $\sim$10–17 percentage points. The Cornell numbers are qualitatively in line with Roman-empire (positive shift in both subsets) but we treat them as suggestive at best. Texas is closer to noise: $\%>0$ hovers around 50% on both subsets and per-node shifts are an order of magnitude smaller than on the larger datasets.

F.3 Pass-wise geometry of the auxiliary path

The previous analyses characterise *whether* the auxiliary path matters and *what kind* of predictive effect it produces. We next ask a simpler geometric question: within each message-passing pass, is the auxiliary representation aligned with the corresponding main-path representation, or does it provide a distinct direction? We measure this from two complementary angles. The *source-side* cosine compares the main and auxiliary representations of each source before message passing. The *receiver-side* cosine compares the aggregated main and auxiliary contributions arriving at each destination after attention-weighted summation. The two answer different questions: the source-side characterises the intrinsic geometry of the auxiliary representation relative to the main one at each source, while the receiver-side characterises the geometry of the contributions actually mixed at the destination, where per-incidence weights and cross-source interference can either reinforce or wash out the per-source angle.

Source-side cosine. For the $\mathcal{V}\to\mathcal{E}$ stage at layer ℓ, the source is a node v, with main and auxiliary representations $\mathbf{x}_v^{\mathrm{main},(\ell)}$ and $\mathbf{x}_v^{\mathrm{aux},(\ell)}$. For the $\mathcal{E}\to\mathcal{V}$ stage, the source is a hyperedge e, with $\mathbf{h}_e^{\mathrm{main},(\ell)}$

Table 19: Source-side and receiver-side cosines between the main-path and auxiliary representations, Eqs. (37), (39). Source-side means are computed across all sources of the pass (nodes for $\mathcal{V}\to\mathcal{E}$, hyperedges for $\mathcal{E}\to\mathcal{V}$) and ten saved checkpoints; receiver-side means are computed across all receivers of the pass (hyperedges for $\mathcal{V}\to\mathcal{E}$, nodes for $\mathcal{E}\to\mathcal{V}$) and ten checkpoints. Standard deviations across pooled source–checkpoint or receiver–checkpoint instances are shown in parentheses. Roman-empire is the only multi-layer configuration in this analysis.

		$\mathcal{V}\to\mathcal{E}$		$\mathcal{E}\to\mathcal{V}$	
Dataset	Layer	Avg. c_s^{src} (std)	Avg. c_r^{recv} (std)	Avg. c_s^{src} (std)	Avg. c_r^{recv} (std)
Citeseer	0	$+0.375\,(0.099)$	$+0.300\,(0.094)$	$+0.006\,(0.124)$	$+0.025\,(0.114)$
Cornell	0	$+0.546\,(0.130)$	$+0.190\,(0.091)$	$+0.027\,(0.131)$	$+0.090\,(0.119)$
Texas	0	$+0.490\,(0.145)$	$+0.211\,(0.114)$	$+0.035\,(0.129)$	$+0.086\,(0.094)$
Telegram	0	$+0.138\,(0.113)$	$+0.036\,(0.101)$	$-0.023\,(0.171)$	$-0.044\,(0.166)$
Roman-empire	0	$+0.529\,(0.106)$	$+0.389\,(0.077)$	$+0.125\,(0.114)$	$+0.006\,(0.103)$
	1	$+0.324\,(0.140)$	$+0.217\,(0.108)$	$-0.017\,(0.137)$	$-0.035\,(0.092)$
	2	$+0.213\,(0.149)$	$+0.172\,(0.113)$	$-0.160\,(0.174)$	$-0.087\,(0.133)$
	3	$+0.146\,(0.188)$	$+0.130\,(0.165)$	$-0.082\,(0.178)$	$-0.057\,(0.128)$

and $\mathbf{h}_e^{\mathrm{aux},(\ell)}$. The source-side cosine is

$$c_s^{\mathrm{src},(\ell,p)} = \begin{cases} \cos\left(\mathbf{x}_v^{\mathrm{main},(\ell)}, \mathbf{x}_v^{\mathrm{aux},(\ell)}\right) & \text{if } p = \mathcal{V}\to\mathcal{E},\ s = v \in \mathcal{V}, \\ \cos\left(\mathbf{h}_e^{\mathrm{main},(\ell)}, \mathbf{h}_e^{\mathrm{aux},(\ell)}\right) & \text{if } p = \mathcal{E}\to\mathcal{V},\ s = e \in \mathcal{E}. \end{cases} \tag{37}$$

Receiver-side cosine. Using the per-incidence main and auxiliary contributions of Eq. (30), define the per-receiver aggregates

$$\begin{aligned} \hat{m}_e^{\mathrm{main},(\ell)} &= \sum_{v\in H(e)\cup T(e)} \hat{\alpha}_{e,v}^{\mathrm{main}}\, \mathbf{x}_v^{\mathrm{main},(\ell)}, & \hat{m}_e^{\mathrm{aux},(\ell)} &= \sum_{v\in H(e)\cup T(e)} \hat{\alpha}_{e,v}^{\mathrm{aux}}\, \mathbf{x}_v^{\mathrm{aux},(\ell)}, \\ \check{m}_u^{\mathrm{main},(\ell)} &= \sum_{e\in\Gamma(u)} \check{\alpha}_{u,e}^{\mathrm{main}}\, \mathbf{h}_e^{\mathrm{main},(\ell)}, & \check{m}_u^{\mathrm{aux},(\ell)} &= \sum_{e\in\Gamma(u)} \check{\alpha}_{u,e}^{\mathrm{aux}}\, \mathbf{h}_e^{\mathrm{aux},(\ell)}, \end{aligned} \tag{38}$$

The receiver-side cosine is then

$$c_r^{\mathrm{recv},(\ell,p)} = \begin{cases} \cos\left(\hat{m}_e^{\mathrm{main},(\ell)}, \hat{m}_e^{\mathrm{aux},(\ell)}\right) & \text{if } p = \mathcal{V}\to\mathcal{E},\ r = e \in \mathcal{E}, \\ \cos\left(\check{m}_u^{\mathrm{main},(\ell)}, \check{m}_u^{\mathrm{aux},(\ell)}\right) & \text{if } p = \mathcal{E}\to\mathcal{V},\ r = u \in \mathcal{V}. \end{cases} \tag{39}$$

Table 19 reports the mean of both cosines across all sources or receivers and ten saved checkpoints, grouped by dataset, layer, and pass; Figures 1 and 2 reports the distribution of the source- and receiver-side cosine, respectively.

$\mathcal{V}\to\mathcal{E}$ keeps a positive component of the main direction. The node-to-hyperedge pass has positive mean source-side cosine in every dataset and layer. On Citeseer, Cornell, Texas, and the first Roman-empire layer, the source-side cosine is moderately positive (0.38–0.55), suggesting that the auxiliary node representation preserves a sizeable component of the corresponding main-path direction. The receiver-side cosine is also positive throughout but smaller in magnitude (e.g. Cornell $0.55 \to 0.19$, Texas $0.49 \to 0.21$, Roman-empire layer 0 $0.53 \to 0.39$). Aggregation thus dilutes but does not invert the alignment: the auxiliary contributions still arrive at hyperedges with a positive component along the main aggregate, just less strongly than the per-source geometry would suggest.

Telegram and the deeper Roman-empire layers are different. Telegram has a much smaller source-side $\mathcal{V}\to\mathcal{E}$ cosine ($+0.138$), and Roman-empire decreases from $+0.529$ at layer 0 to $+0.146$ at layer 3. These are also the settings where the auxiliary path is most predictively important. Thus, when the auxiliary path carries substantial predictive load, its node-to-hyperedge representation becomes less aligned with the main path and more geometrically distinct.

$\mathcal{E}\to\mathcal{V}$ is nearly orthogonal at both ends. The hyperedge-to-node pass is centred near zero on the single-layer datasets at the source side and stays close to zero at the receiver side: Citeseer, Cornell,

Texas, and Telegram all have receiver-side cosines in $[-0.05, +0.10]$. The auxiliary hyperedge representation is therefore neither a rescaled copy of the main one at the source, nor does aggregation reintroduce alignment at the receiver: at both ends of the pass the auxiliary contribution provides a distinct feature direction.

Roman-empire makes the picture sharper across layers. The source-side $\mathcal{E}\to\mathcal{V}$ cosine moves from mildly positive at layer 0 ($+0.125$) to near-zero at layer 1, and becomes negative in layers 2 and 3 (-0.160 and -0.082). The receiver-side cosine consistently sits closer to zero than the source-side: layer 0 collapses from $+0.125$ to $+0.006$, and layer 2 moves from -0.160 to -0.087. This matches the intervention results: Roman-empire relies on the auxiliary path across multiple layers, and the auxiliary contribution arrives at each node along a direction that is essentially orthogonal to the main aggregate, mildly tilted away from it in the deeper layers.

Interpretation. These cosines support the view that the auxiliary path is not merely a duplicate message stream. In the $\mathcal{V}\to\mathcal{E}$ pass, the auxiliary representation preserves part of the main node direction at the source and at the receiver, but this alignment weakens in Telegram and in deeper Roman-empire layers, where the auxiliary path is most important. In the $\mathcal{E}\to\mathcal{V}$ pass, the auxiliary representation is nearly orthogonal at the source, and aggregation generally tightens rather than loosens this orthogonality, so the auxiliary contribution at each node provides a geometrically distinct channel that cannot be obtained by rescaling the main aggregate.

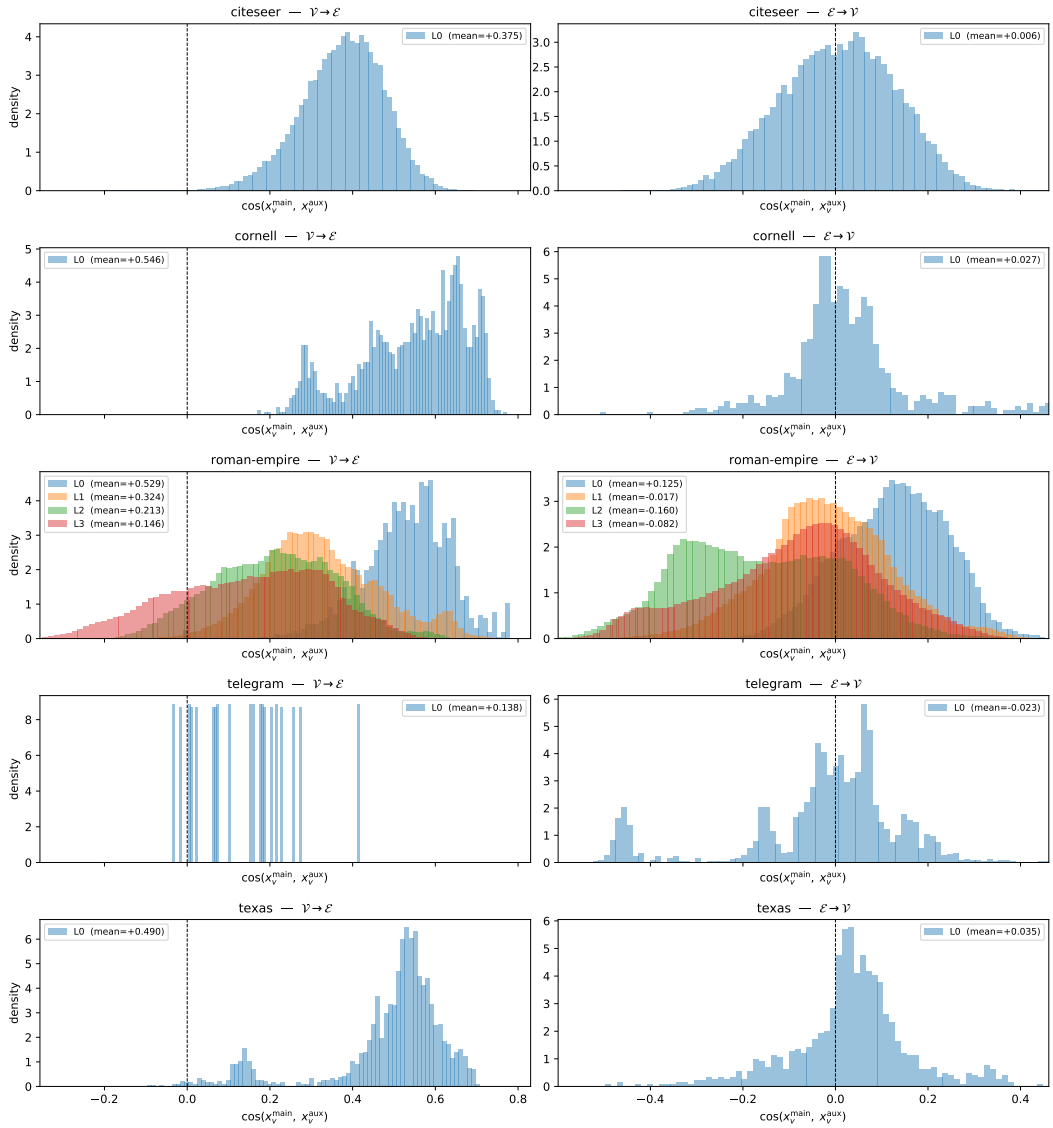


Figure 1: Source-wise distribution of $c_s^{(\ell,p)}$ (Eq. (37)) across all sources and ten saved checkpoints, overlaid by layer for the multi-layer Roman-empire family. The vertical dashed line marks $c = 0$. The $\mathcal{V}\to\mathcal{E}$ cosine is uniformly positive across families and layers; the $\mathcal{E}\to\mathcal{V}$ cosine is concentrated near zero on Citeseer, Cornell, Texas, and Telegram, and spreads more widely on Roman-empire (with layer 2 mass clearly to the left of zero).

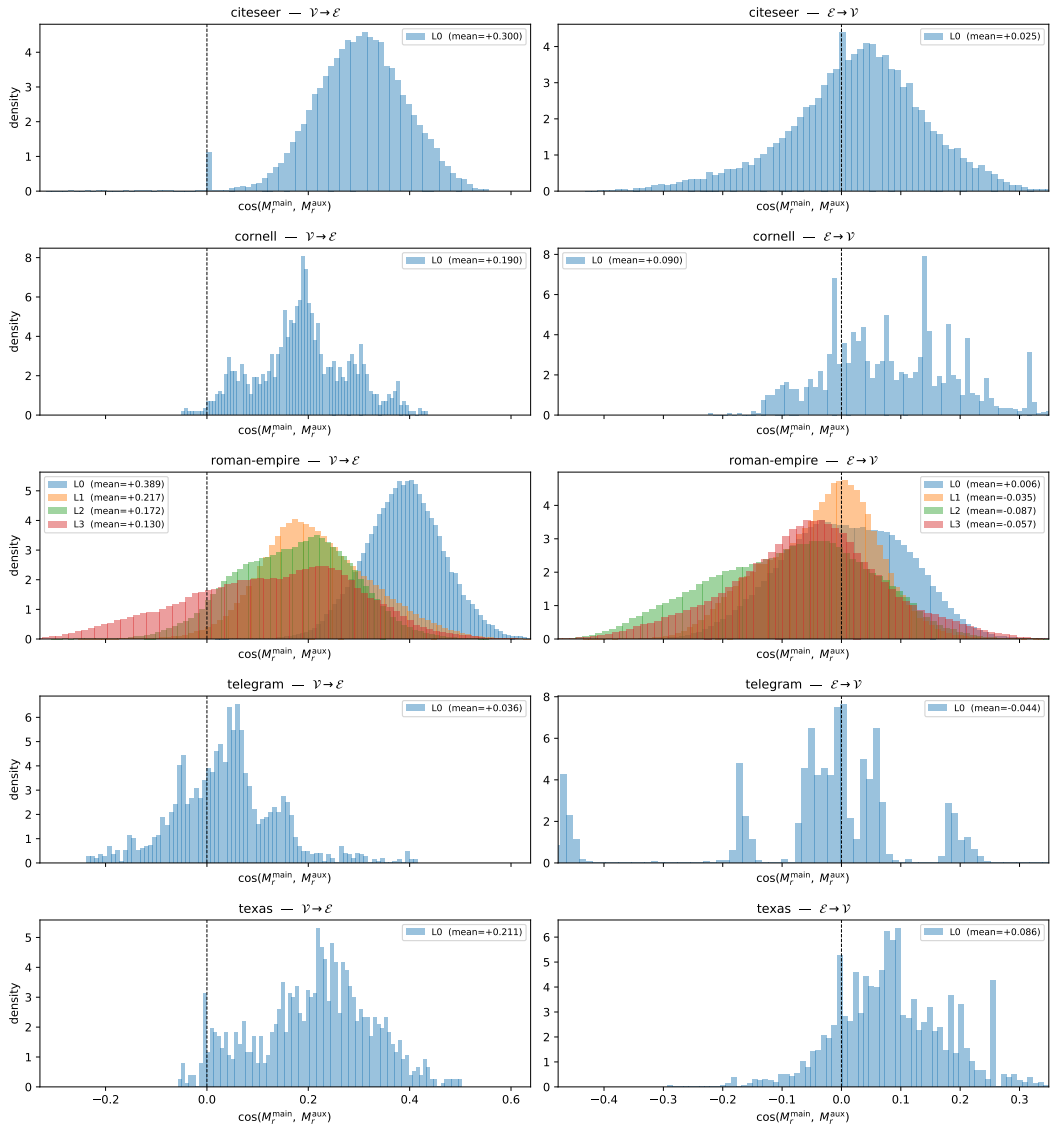


Figure 2: Receiver-side distribution of $c_r^{\mathrm{recv},(\ell,p)}$ (Eq. (39)) across all receivers and ten saved checkpoints, overlaid by layer for the multi-layer Roman-empire family. The vertical dashed line marks $c = 0$. The $\mathcal{V}\to\mathcal{E}$ receiver-side cosine is positive throughout but smaller in magnitude than its source-side counterpart; the $\mathcal{E}\to\mathcal{V}$ receiver-side cosine is concentrated near zero on every dataset and layer, including Roman-empire's deeper layers, where aggregation tightens the source-side near-orthogonality further.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: Section 5.2 and Table 1 show our quantitative evaluation results, supporting our performance claims. We also provide Theorems 4.1 and 4.2, with proofs in Appendix A.

   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: [Yes]

   Justification: We discuss the limitations within our result analysis in Section 5.2 and the conclusion in Section 6.

   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: [Yes]

Justification: We provide proofs for our theorems in Appendix A.

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: [Yes]

Justification: We provide complete details about our plug-in model in Sections 3 and 4. We provide the information about the datasets, baseline models and hyperparameter tuning in Section 5.1 and Appendix B. Furthermore, we provide our codebase at `https://github.com/jainris/panda`.

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: [Yes]

   Justification: We provide our codebase at `https://github.com/jainris/panda`, which includes scripts to download the relevant datasets. All the datasets used in our submission are publicly available, as noted in Section 5.1.

   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: [Yes]

   Justification: We provide complete details about our plug-in model in Sections 3 and 4. We provide the information about the datasets, baseline models and hyperparameter tuning in Section 5.1 and Appendix B. Furthermore, we provide our codebase at `https://github.com/jainris/panda`.

   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: [Yes]

   Justification: Our results in Section 5.2 includes the standard deviations and appropriate information.

   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: [Yes]

   Justification: We provide information about our computer resources in Appendix B.

   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: [Yes]

   Justification: Our research conforms with NeurIPS Code of Ethics.

   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: [N/A]

Justification: This paper presents work whose goal is to advance the field of machine learning. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

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: [N/A]

Justification: The paper poses no such risks.

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: [Yes]

Justification: We have included relevant citations and licences for the code and data we use. The relevant licence is listed in Appendix B.

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: [Yes]

    Justification: We provide our code at `https://github.com/jainris/panda`.

    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: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    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: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    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: [N/A]

    Justification: The core method development in this research does not involve LLMs.

    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.