TOPOLOGICAL READOUTS OF LEARNED POSITIONAL FILTRATIONS FOR HYPERGRAPH CLASSIFICATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Hypergraph neural networks perform hypergraph classification by message passing, which on graphs has been combined with positional encodings and persistent homology. Hypergraphs admit several expansions; a clique expansion can map hypergraphs whose star expansions differ in cycle rank to the same graph. On a widely used public benchmark, reported accuracies on two social collections lie near majority-class rates. Descriptors that combine positional and topological information on the star expansion may capture label information that these methods miss. We propose HyLPF: positional states initialized with random-walk positional encodings propagate over the star expansion, defining learned filtrations whose persistent homology, including cycle rank and essential-vertex flags, alone feeds the classifier. HyLPF achieves the best overall accuracy on the benchmark's four collections, ranking first on three and second on the fourth, with its configuration fixed before evaluation on new splits of the benchmark and under its split protocol and metric. On one of the social collections, matched controls show that the cycle rank and flags, with the weights reading them, carry label information, while merge values and cycle values add no detectable accuracy beyond them.

1 INTRODUCTION

Hypergraphs model interactions among sets of entities, such as the co-directors of a movie or the games owned by one player (Zhou et al., 2006; Feng et al., 2024). Hypergraph classification labels an entire hypergraph. Hypergraph neural networks pass messages on hypergraph operators, as in HGNN (Feng et al., 2019), UniGNN (Huang & Yang, 2021), AllSet (Chien et al., 2022), ED-HNN (Wang et al., 2023a), and sheaf networks (Duta et al., 2023), and a readout applies them to this task.

Public benchmarks with a common evaluation protocol are widely used to compare such networks, and how a protocol splits data, selects models, and reports accuracies shapes the conclusions of a comparison (Errica et al., 2020; Bouthillier et al., 2021). Feng et al. (2024) introduced six real-world single-label collections for this task. DHG-Bench (Li et al., 2026) evaluates 18 methods on four of them under one protocol: 15 hypergraph networks, two graph networks on clique expansions, and a multilayer perceptron (MLP), with test accuracy over five random 80/10/10 splits.

The label of a hypergraph can depend on its incidence structure, that is, on which nodes share which hyperedges. Clique expansions lose part of this structure, because they can map different hypergraphs to the same graph, whereas the star expansion keeps it. Positional encodings and persistent homology each capture a different aspect of such structure: the position of each vertex relative to the others, and how the structure connects across scales. Whether these descriptors carry label information for hypergraph classification is open, and no prior method reads the persistent homology of positional filtrations on the star expansion. The accuracies that current methods reach on the benchmark suggest that the descriptors they use capture little label information on some collections: the best accuracies that the benchmark reports on its two binary social collections, 54.85% on Steam-Player and 60.15% on Twitter-Friend, lie within five points of the majority-class rates of 50.0% and 59.2% (Table 5; Appendix E), whereas two cross-validation studies with other protocols report higher ones (Section 2). We ask whether positional and topological descriptors of the incidence structure carry label information on these collections.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

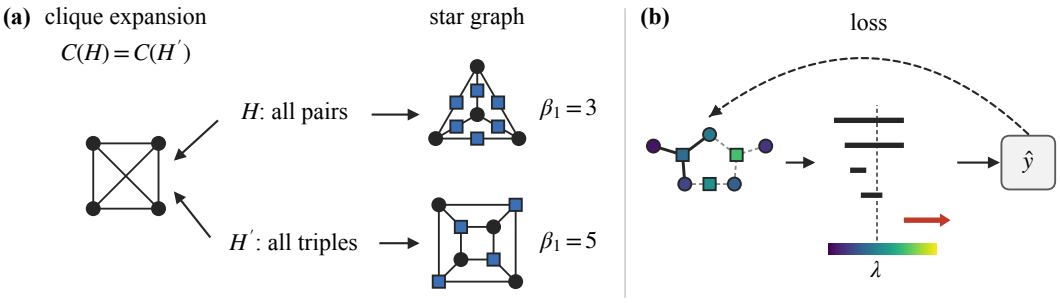

Figure 1: **Overview.** (a) Two hypergraphs with the same clique expansion but typed star graphs (squares: hyperedges) of different cycle rank (Proposition 6). (b) In HyLPF, positional states define learned filtrations of the star graph, and the classifier reads only their persistence.

The typed star graph, the star expansion or incidence graph with typed vertices, keeps every node-hyperedge membership. A clique expansion (Agarwal et al., 2006) can map different hypergraphs to the same graph (Figure 1a; Proposition 6). Random-walk positional encodings (RWPE) give each vertex a positional signature (Dwivedi et al., 2022), persistent homology (PH) of learned filtrations summarizes how a graph connects across scales (Edelsbrunner et al., 2002; Hofer et al., 2020; Horn et al., 2022), and PiPE combines the two on graphs (Verma et al., 2025). A reliable comparison needs a common protocol and a configuration frozen before evaluation (Errica et al., 2020; Cawley & Talbot, 2010).

We propose HyLPF, a hypergraph classifier built on one mechanism (Figure 1b). RWPE-initialized positional states of nodes and hyperedges are propagated along the incidences, and at every layer they define learned filtrations of the typed star graph, whose edges are these incidences. The classifier reads only a set readout of their persistence pairs, essential-vertex flags, and cycle values, whose number is the cycle rank β_1 (number of independent cycles) of the star graph, so propagation and readout share one structure and one set of states under one loss. We froze its configuration and re-evaluated it on two batches of five new random splits. Each batch contains a matched control for the persistence readout, and a later value-ablated control separates the cycle rank and the flags from the merge values and cycle values. We compare it with all 18 benchmark methods on the four real-world collections of the benchmark's hypergraph-classification track.

Our results support two findings. First, HyLPF achieves the best overall accuracy on the four collections under the benchmark's split protocol and metric, ranking first on three and second on the fourth (Section 5.1). Second, on Steam-Player, HyLPF is 3.71 points (95% interval 2.15 to 5.27) more accurate than a homology-matched twin without persistence-derived inputs, whose weights on the flags and cycle tokens stay inactive; a value-ablated twin that keeps only the cycle tokens and the flags shows that these inputs, with their weights, carry label information and that the merge values and cycle values add no detectable accuracy beyond them. Our contributions are as follows.

- HyLPF, one mechanism on the star expansion: positional states propagated over the incidences of a hypergraph define learned filtrations whose persistent homology, including the cycle rank and essential-vertex flags, is the only input of the classifier. We show that the readout receives the star-graph cycle rank, which clique expansions can lose, and we prove that it computes a tie-aware persistence descriptor that reduces to the elder-rule diagram for injective filtrations and that, at inference, isomorphic hypergraphs receive the same prediction.
- HyLPF achieves the best overall accuracy on four collections of a widely used public benchmark, ranking first on three and second on the fourth, under its split protocol and metric and with a configuration frozen before evaluation on new splits.
- With analysis plans fixed before their runs, matched controls show that, on Steam-Player, the cycle rank and essential-vertex flags, with the weights that read them, yield a measurable gain, beyond which the merge values and cycle values add no detectable accuracy. On the same collection, the random-walk initialization of the positional states raises accuracy over constant inputs.

2 RELATED WORK

Hypergraph neural networks and hypergraph-level classification. HGNN (Feng et al., 2019), HyperGCN (Yadati et al., 2019), HCHA (Bai et al., 2021), UniGNN (Huang & Yang, 2021), AllSet (Chien et al., 2022), ED-HNN (Wang et al., 2023a), sheaf hypergraph networks (Duta et al., 2023), and the training-free TF-HNN (Tang et al., 2025) define message passing on hypergraphs, and a common benchmark evaluates them on the collections of Feng et al. (2024) (Li et al., 2026). Recent hypergraph networks use directed sheaf Laplacians (Mule et al., 2026), particle-system dynamics (Ma et al., 2025), random-walk spectral convolutions (Zhang et al., 2025b), and tensor representations (Wang et al., 2025a). For hypergraph classification, Wang et al. (2025b) derive PAC-Bayes generalization bounds for four families of hypergraph networks. Zhang et al. (2025a) study the expressivity of hypergraph networks through generalized Weisfeiler-Leman tests, and Choi et al. (2026) propose categorical message passing on graded posets, with incidence posets as one instance (preprint). These two works and Feng et al. (2024) evaluate on the same collections with k-fold cross-validation instead of the benchmark's splits. Zhang et al. (2025a) report five-fold means with the best of one to three layers: 67.53% on Steam-Player (2-FHNN) and 64.12% on Twitter-Friend (2-OHNN), and 61.87% and 64.03% for AllSetTransformer. Choi et al. (2026) report the best mean validation accuracy of ten-fold cross-validation, 74.66% on Twitter-Friend for their incidence-poset network. These protocols are not directly comparable with the benchmark's 80/10/10 splits, so Table 1 compares only with methods run on such splits. HyLPF operates on the typed star graph, whose vertices are the nodes and the hyperedges, and adds a persistence readout.

Positional encodings and persistence readouts. Position-aware and distance-encoding networks use distances to sampled anchor sets (You et al., 2019) or random-walk landing probabilities from target nodes (Li et al., 2020), and GraphGPS treats Laplacian (Dwivedi et al., 2023) and random-walk encodings as modular inputs (Rampášek et al., 2022). Laplacian eigenvectors are defined only up to sign and basis, which SignNet, PEG, and SPE address (Lim et al., 2023; Wang et al., 2022; Huang et al., 2024). RWPE, the return probabilities of random walks, has no sign ambiguity and can initialize positional states that are updated alongside structural ones (Dwivedi et al., 2022). On hypergraphs, Zhang et al. (2025c) learn positional embeddings separately for nodes and hyperedges, HyperGT learns them from the incidence matrix (Liu et al., 2024), and a topology-guided hypergraph transformer adds structural and spatial encodings (Saifuddin et al., 2026). Pellegrin et al. (2025) define hypergraph-level encodings, including random-walk encodings, for graph networks on clique expansions. PersLay learns vectorizations of persistence diagrams (Carrière et al., 2020), Graph Filtration Learning learns the filtration itself (Hofer et al., 2020), TOGL injects PH of learned filtrations into message passing (Horn et al., 2022), and RePHINE augments persistence diagrams with vertex and edge information (Immonen et al., 2023). TopNets combines PH with higher-order message passing (Verma et al., 2024), CliquePH computes PH on clique graphs (Buffelli et al., 2025), and PiPE studies the interplay of positional encodings and PH on graphs (Verma et al., 2025). Aktas et al. (2026) use hand-designed filtrations for hypergraph classification, and DE-HNN combines Laplacian encodings with fixed persistence features for node- and hyperedge-level prediction (Luo et al., 2024). HyLPF carries this combination to hypergraph-level prediction as one mechanism on the star graph. Its RWPE-initialized node and hyperedge states, propagated along the incidences, define learned filtrations through separate networks per vertex type. The persistent homology of these filtrations, with a tie-aware merge map (Definition 3), is the classifier's only input.

Benchmark evaluation, duplicates, and preregistration. Errica et al. (2020) showed that inconsistent model selection undermines comparisons of graph classifiers. Model selection on the evaluation data biases performance estimates (Cawley & Talbot, 2010), results vary with splits and initializations (Bouthillier et al., 2021), and reported comparisons depend on the tuning effort (Dodge et al., 2019). In graph-classification benchmarks, isomorphic duplicates changed model rankings (Ivanov et al., 2024), duplicated nodes leak between training and test in heterophilous node-classification datasets (Platonov et al., 2023), and leakage and near-duplicates affect results in other domains (Kapoor & Narayanan, 2023; Barz & Denzler, 2020). Preregistration fixes analyses before results are seen (van Miltenburg et al., 2021), and Tynchenko et al. (2026) test the significance of rankings in the node-classification track of the benchmark. We apply a frozen configuration, new splits with analysis plans fixed before their runs, and matched controls to hypergraph classification, and Appendix D quantifies input duplicates in the benchmark's hypergraph-classification track.

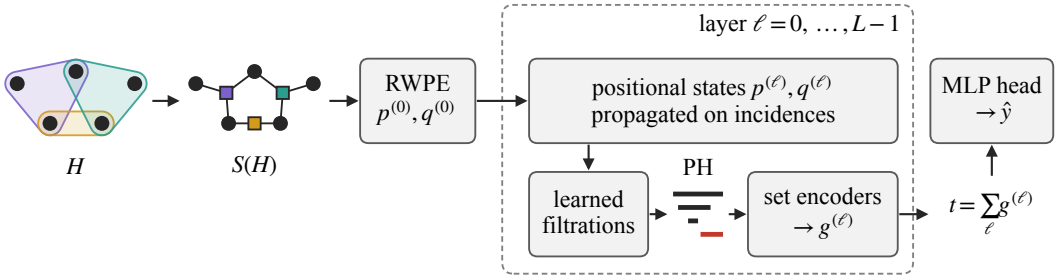

Figure 2: **HyLPF as one mechanism on the typed star graph $S(H)$:** positional states propagated over the incidences define learned filtrations at every layer, and the classifier reads only their persistence.

3 METHOD

HyLPF is one mechanism on the incidence structure of a hypergraph (Figure 2). Positional states propagated along the incidences define, at every layer, learned filtrations of the typed star graph, and the classifier reads only their persistent homology. Propagation and readout share one structure and one set of states, and one loss trains, end to end, all parameters on which the prediction depends.

3.1 INCIDENCE REPRESENTATION AND OBJECTIVE

Definition 1 (Input and typed star graph)**.** A hypergraph is a triple $H = (V, E, I)$ of finite sets with incidence relation $I \subseteq V \times E$, in which every node and every hyperedge has at least one incidence. Hyperedges are indexed, so two hyperedges may contain the same nodes. The incidence matrix $B \in \{0,1\}^{|V|\times|E|}$ has $B_{ve} = 1$ iff $(v, e) \in I$. Each node carries the one-hot encoding x_v of its degree over the degree vocabulary of the collection, and hyperedges carry no features. The typed star graph $S(H)$ has vertex set $V \sqcup E$, edge set $\{\{v, e\} : (v, e) \in I\}$, and type map κ with $\kappa(z) = 1$ iff $z \in E$.

All hypergraphs of the four collections satisfy these conditions (Appendix A), and the model realizes κ through separate parameters for the two vertex types. An isomorphism $(\mu, \nu) : H \to H'$ is a pair of bijections $\mu : V \to V'$, $\nu : E \to E'$ with $(v, e) \in I \Leftrightarrow (\mu(v), \nu(e)) \in I'$.

Definition 2 (RWPE)**.** With $D_V = \mathrm{diag}(B\mathbf{1})$ and $D_E = \mathrm{diag}(B^\top \mathbf{1})$, let $P_V = D_V^{-1} B D_E^{-1} B^\top$ and $P_E = D_E^{-1} B^\top D_V^{-1} B$. For walk length J, each node v and hyperedge e receives

$$\pi_v = \big((P_V^j)_{vv}\big)_{j=1}^{J} \in \mathbb{R}^J, \qquad \pi_e = \big((P_E^j)_{ee}\big)_{j=1}^{J} \in \mathbb{R}^J. \tag{1}$$

P_V moves from a node through a hyperedge to a node (Zhou et al., 2006; Chitra & Raphael, 2019), and P_E is its dual. The simple random walk $P_S = D_S^{-1} A_S$ on $S(H)$, with adjacency matrix A_S and degree matrix D_S, satisfies $P_S^2 = \mathrm{diag}(P_V, P_E)$, so π_z lists the return probabilities $(P_S^{2j})_{zz}$, the nonzero part of the RWPE of the bipartite graph $S(H)$ (Dwivedi et al., 2022). Separate affine maps give the initial positional states of nodes and hyperedges (cf. Zhang et al., 2025c), $p_v^{(0)} = W_V \pi_v + c_V$ and $q_e^{(0)} = W_E \pi_e + c_E$ in $\mathbb{R}^{d_p}$.

Propagation. For $\ell = 1, \ldots, L$, two-stage sums over incidences update the states,

$$q_e^{(\ell)} = \phi_q^{(\ell)}\Big(\big[q_e^{(\ell-1)}, \textstyle\sum_{v\in e} p_v^{(\ell-1)}\big]\Big), \qquad p_v^{(\ell)} = \phi_p^{(\ell)}\Big(\big[p_v^{(\ell-1)}, \textstyle\sum_{e\ni v} q_e^{(\ell)}\big]\Big), \tag{2}$$

where every $\phi(z) = W_2\,\mathrm{ReLU}(\mathrm{BN}(W_1 z + c_1)) + c_2$ is a two-layer MLP with batch normalization and no residual connection. Sum aggregation keeps multiset information about the neighborhood (Xu et al., 2019). A feature channel $(x_v^{(\ell)}, \xi_e^{(\ell)})$ starts from an affine map of the degree features and is updated by two-stage sums that also receive $p^{(\ell)}$ and $q^{(\ell)}$ (Equation (6) in Appendix A).

Objective. The classifier $z_\theta(H) = W_2 \,\mathrm{Dropout}(\mathrm{ReLU}(W_1 t + c_1)) + c_2 \in \mathbb{R}^C$ reads only the readout t (Section 3.3). For labels $y_H \in \{1, \dots, C\}$, training minimizes the mean cross-entropy $\mathcal{L}(\theta; \mathcal{B}) = -|\mathcal{B}|^{-1} \sum_{H \in \mathcal{B}} \log \mathrm{softmax}(z_\theta(H))_{y_H}$ over mini-batches $\mathcal{B}$ of training hypergraphs, so one loss trains all parameters that enter t and reaches the states only through their filtration values.

3.2 Filtrations induced by the positional states

At each layer index $\ell \in \{0, \dots, L-1\}$, networks $\psi_V^{(\ell)}, \psi_E^{(\ell)}$ with one hidden ReLU layer and the sigmoid output σ map the positional states to K filtration values,

$$f^{(\ell)}(v) = \sigma\big(\psi_V^{(\ell)}(p_v^{(\ell)})\big) \in [0,1]^K, \qquad f^{(\ell)}(e) = \sigma\big(\psi_E^{(\ell)}(q_e^{(\ell)})\big) \in [0,1]^K. \tag{3}$$

Each $f_k = f_k^{(\ell)}$ is thus a function on the vertices of the star graph along which the states propagate.

Definition 3 (Star-graph persistence)**.** Let $f : V \sqcup E \to \mathbb{R}$. An incidence $\{v, e\}$ enters at $\max(f(v), f(e))$. For $\lambda \in \mathbb{R}$, let $S_{\le\lambda}$ and $S_{<\lambda}$ be the spanning subgraphs of $S(H)$ with the incidences that enter at most at λ and below λ, and write $u \sim_\lambda z$ if u and z are connected in $S_{\le\lambda}$. If some u with $f(u) < f(z)$ is connected to z in $S(H)$, the vertex z has merge value

$$d_f(z) = \min\{\lambda : \exists u \text{ with } f(u) < f(z) \text{ and } u \sim_\lambda z\}, \qquad \epsilon_f(z) = 0; \tag{4}$$

otherwise z is essential, with $\epsilon_f(z) = 1$ and $d_f(z) = \max_{u \in V \sqcup E} f(u)$. The one-dimensional diagram $\mathrm{Dgm}_1(f)$ is the multiset of cycle values in which each λ has multiplicity $\beta_1(S_{\le\lambda}) - \beta_1(S_{<\lambda})$, where the cycle rank β_1 is the number of edges minus vertices plus connected components.

Proposition 4 (Computation and elder rule; proof in Appendix A)**.** *The union-find procedure of HyLPF (Appendix A), run over the incidences in any order of non-decreasing entry value, returns d_f, ϵ_f, and $\mathrm{Dgm}_1(f)$ of Definition 3. Moreover, $\epsilon_f(z) = 1$ iff $f(z)$ is the minimum of f on the connected component of z in $S(H)$. If f is injective, the pairs $(f(z), d_f(z))$ with $\epsilon_f(z) = 0$ form the finite part of the elder-rule zero-dimensional persistence diagram of the sublevel filtration, including its diagonal points.*

Proposition 5 (Invariance; proof in Appendix A)**.** *Let batch normalization use running statistics and dropout be inactive. If $(\mu, \nu) : H \to H'$ is an isomorphism, every state, filtration value, merge value, and essential flag of H' equals that of H at the vertex mapped by (μ, ν), $\mathrm{Dgm}_1(f'_k) = \mathrm{Dgm}_1(f_k)$ for all layers and filtrations, and HyLPF gives H and H' the same logits.*

Learned filtration values can tie, systematically so: in exact arithmetic, vertices exchanged by an automorphism of H have equal states and filtration values (Proposition 5, $H' = H$). The readout then uses a tie-aware descriptor: every vertex that attains the minimum of f on its component has $\epsilon_f = 1$, and for injective f the pairs reduce to the standard diagram (Proposition 4). Stability results for standard persistence thus do not apply directly to the flags near ties (Appendix A).

3.3 Persistence readout on the same structure

Set readout. For each vertex z of $S(H)$, the token $\tau_z = \big(f_k(z), d_{f_k}(z), \epsilon_{f_k}(z)\big)_{k=1}^K \in \mathbb{R}^{3K}$ is encoded by ρ_V for nodes and ρ_E for hyperedges. A point $b \in \mathrm{Dgm}_1(f_k)$ becomes the tagged token $\mathbf{e}_k \otimes (b, 1) \in \mathbb{R}^{2K}$, where $\mathbf{e}_k$ is the k-th standard basis vector of $\mathbb{R}^K$, and is encoded by ρ_1. Following DeepSets (Zaheer et al., 2017), the layer descriptor is

$$g^{(\ell)} = \rho_{\mathrm{out}}\Big(\Big[\textstyle\sum_{v \in V} \rho_V(\tau_v),\ \sum_{e \in E} \rho_E(\tau_e),\ \sum_{k=1}^K \sum_{b \in \mathrm{Dgm}_1(f_k)} \rho_1\big(\mathbf{e}_k \otimes (b,1)\big)\Big]\Big) \in \mathbb{R}^{d_p}, \tag{5}$$

where the inner sum counts multiplicity and every ρ is a two-layer MLP with batch normalization. The readout of HyLPF is $t = \sum_{\ell=0}^{L-1} g^{(\ell)}$, with one descriptor per layer. Equation (5) describes inference mode; in training mode, the batch statistics of ρ_1 also include masked rows (Appendix A).

Coupling. Union-find returns only indices, and the persistence values are gathered from the differentiable filtration tensor (Appendix A), so $\mathcal{L}$ trains the filtration networks and, through them, the maps W_V, W_E and the positional updates of layers 1 to $L-1$; the flags and the number of cycle tokens are constant in the backward pass. The prediction depends on H only through $S(H)$, not on the degree features, and the feature channel and the last positional update do not enter t. The persistence computation sorts the incidences and runs union-find with path halving, at a cost of $O(|I| \log |I|)$ per filtration (Appendix A).

What the readout can represent. The clique graph $C(H)$ of a hypergraph H is the simple graph on V in which two distinct nodes are adjacent iff some hyperedge contains both (Agarwal et al., 2006).

Proposition 6 (Clique graph and star-graph cycle rank; proof in Appendix A)**.** *Let $V = \{1, 2, 3, 4\}$. Let H have the six hyperedges $\{u, v\} \subset V$ with $u \neq v$, and let H' have the four hyperedges $\{u, v, w\} \subset V$ with three distinct nodes. Then H and H' are not isomorphic, $C(H) = C(H')$, every node has degree 3 in both, $\beta_1(S(H)) = 3$, and $\beta_1(S(H')) = 5$. Moreover, for the star graph S of any hypergraph of Definition 1 and every real function f on the vertices of S, the multiset $\mathrm{Dgm}_1(f)$ of Definition 3 has $\beta_1(S)$ points, counted with multiplicity.*

A model reading $C(H)$ with the degree features thus gives H and H' the same output (Figure 1a). In HyLPF, each filtration contributes $\beta_1(S(H))$ cycle tokens, 3 for H and 5 for H', and its flags enter per vertex through ρ_V and ρ_E, marking in each component the vertices where f attains its minimum (Proposition 4). We claim no expressivity result relative to incidence-based networks.

3.4 Training and matched controls

HyLPF and its two twins are trained in the same way, with the loss $\mathcal{L}$ and Adam (Kingma & Ba, 2015), without weight decay, label smoothing, or class weights; dropout is the only regularizer. The learning rate is halved when the validation accuracy stops improving, and training stops a fixed number of epochs after the selected epoch, the one with the highest validation accuracy, ties broken by the lower mean validation cross-entropy and then the earlier epoch (Appendix B). The selected model is evaluated once on the test part, whose batches are built only after selection.

Definition 7 (Matched twins)**.** The homology-matched twin of HyLPF evaluates Equation (5) with $d_{f_k}(z)$ replaced by $f_k(z)$, $\epsilon_{f_k}(z)$ by 0, and $\mathrm{Dgm}_1(f_k)$ by the empty multiset, so the third block of the input of ρ_{out} is zero. The value-ablated twin of HyLPF evaluates Equation (5) with $d_{f_k}(z)$ replaced by $f_k(z)$, $\epsilon_{f_k}(z)$ kept, and every point $b \in \mathrm{Dgm}_1(f_k)$ replaced by $1/2$.

Both twins keep the configuration and the parameter tensors of HyLPF. The homology-matched twin withholds all persistence-derived inputs. The value-ablated twin keeps the $\beta_1(S(H))$ cycle tokens of each filtration (Proposition 6) and the flags, which mark its minima on each component, and withholds the persistence values, the merge values and cycle values. The three models are nested in their inputs but trained separately, so this nesting does not order their accuracies.

4 Evaluation protocol

Collections. We use four collections of Feng et al. (2024) that are part of DHG-Bench (Li et al., 2026). In Steam-Player (2,048 hypergraphs), a player's games are linked by shared tags, and the label is a preference for single-player or multiplayer games. In Twitter-Friend (1,310 hypergraphs), a user's friends form a hypergraph, and the label is one of two posted topics. IMDB-Dir-Form (1,869) and IMDB-Dir-Genre (3,393) are three-class tasks on director hypergraphs. Appendix E gives their statistics.

Splits and benchmark runs. The benchmark splits each collection 80/10/10 at random, selects the checkpoint with the best validation accuracy, and reports test accuracy and macro-F1 over five splits (Definition A.1). Our comparisons share the benchmark's split generator, split proportions, and test metrics, but not its selection rule, training recipe, or number of initializations: our models use the training of Section 3.4 with three initializations per split, and the benchmark methods we ran use the upstream selection and training with one initialization. Unless marked as reported, accuracies of the benchmark methods come from our runs of the benchmark's code with its published configurations (Appendices C and D).

Frozen configuration. The configuration of HyLPF was designated by an informal cross-collection criterion, which referred to accuracies reported by Choi et al. (2026), in a random search on cross-validation folds of the six collections of Feng et al. (2024) (Appendix B) and frozen before any re-evaluation run (Cawley & Talbot, 2010). The search used all graphs of the six collections, so the re-evaluation controls split seeds and configuration but not data exposure.

New splits and analysis plans. We draw new splits with the benchmark's generator, distinct from those behind the reported values and not used by any earlier run: ten for the social collections, in two batches of five, and five for the IMDB collections, which the first batch also covers (Appendix B). On these five IMDB splits, the 18 methods and DSHN were run after HyLPF's results there were known. The analysis plans, endpoints, and decision rules of the two batches and of the analyses named next were written down before their runs; not every endpoint of these plans is reported here. The second batch and the comparison over 48 sampled configurations were planned after the results of the first batch had been read, and the same-split run, the value-ablated contrast, the count-feature baseline, the sum-readout run, the DSHN comparison, the IMDB runs, and the extended ablation after those of both batches. These analyses use ordinary 95% t intervals. Random splits share training graphs, so these intervals do not account for the correlation between split-level values and are conditional on the collections analyzed. The corrected resampled t interval of Nadeau & Bengio (2003) with $n_{\text{test}}/n_{\text{train}} = |\text{Te}_s|/|\text{Tr}_s|$ widens each half-width over n splits by $\sqrt{1 + n\, n_{\text{test}}/n_{\text{train}}}$, about 1.50 for $n = 10$ and 1.28 for $n = 5$. Under it, every comparison of HyLPF on the two social collections with the same-split run of the 18 methods, the configurations selected among 48, and the reported values keeps its lower end above its threshold, as do, on Steam-Player, the gains of HyLPF and of the value-ablated twin over the homology-matched twin and of HyLPF over its variant without RWPE (Appendix C.1; Table 2).

Estimators and verdicts. The split-level value a_s averages the test accuracies (%) of the three initializations of split s, and a matched contrast uses $\Delta_s = a_s - a_s^{\text{twin}}$. Over a set of n splits, we report the mean and the two-sided 95% t confidence interval (CI) with $n - 1$ degrees of freedom (Definition A.2). A model exceeds a reference accuracy if the lower end of its interval exceeds it, and a matched contrast holds if the lower end for Δ exceeds zero. The leave-one-split-out check repeats the verdict with each split removed. Each batch has $n = 5$, and the pooled analysis over both batches has $n = 10$. Accuracy is the endpoint metric; macro-F1 uses the same estimators.

Computation. For one hypergraph and one filtration, the implementation sorts the incidences by entry value with a stable sort, and a union-find with path halving then processes them in this order. Each component stores the minimum of f on it and the list of its vertices that attain this minimum and have not merged. When an incidence joins two components with different minima, every listed vertex of the component with the larger minimum merges at the current entry value, and the merged component keeps the list of the other component; when the minima are equal, the two lists are concatenated and no vertex merges. An incidence whose endpoints are already connected records its entry value as a cycle value, and vertices without a merge value are essential. This computation runs on detached filtration values; the values $f(z)$, $d_f(z)$, $\max f$, and the cycle values are then read from the differentiable filtration tensor at the vertices that attain them (Appendix A).

5 Results

5.1 Comparison with the benchmark methods

HyLPF achieves the best overall accuracy on the four collections, ranking first on three and second on the fourth (Table 1). We ran all 18 benchmark methods with their released code and published configurations on our ten splits of the two social collections, one initialization each (Table 1; Appendix C). On these splits, 16 of the 18 methods on Steam-Player and 14 on Twitter-Friend reach a higher mean accuracy than DHG-Bench reports, so, with respect to the reference values, the same-split comparison is the stricter one. By mean accuracy, the best method is AllSetTransformer on Steam-Player (57.52%) and DPHGNN on Twitter-Friend (61.15%). In paired split-level differences, HyLPF exceeds that method by 8.88 points (95% interval 6.10 to 11.67) on Steam-Player and 11.78 points (7.91 to 15.65) on Twitter-Friend. HyLPF reaches 66.41% (64.65 to 68.16) and 72.93% (70.69 to 75.16), so on both collections the lower end exceeds the best method's ten-split mean.

As a secondary reference, both lower ends also exceed the best reported accuracies (Table 5), 54.85% (TF-HNN) and 60.15% (AllSetTransformer), in each batch and when any one split is left out (Table 4), and in mean macro-F1 on these ten splits HyLPF is above all 18 methods (Tables 5 and 10): it reaches 66.06% on Steam-Player and 70.53% on Twitter-Friend, against at most 56.56% and 51.67% for the 18 methods. DSHN (Mule et al., 2026), a 2026 hypergraph network, has no pub-

Table 1: **Test accuracy (%) on four collections.** Same splits for all methods. The methods available in a public benchmark are run with their public code and configurations, one initialization per split (reported values: Table 5). Mean ± SD over splits; methods by year; bold: best; underlined: second best; shaded: our model.

Method	Steam-Player	Twitter-Friend	IMDB-Dir-Form	IMDB-Dir-Genre
MLP	55.39 ± 2.57	59.85 ± 3.69	64.68 ± 2.54	75.65 ± 2.18
CEGCN (Kipf & Welling, 2017)	53.59 ± 4.15	60.00 ± 5.84	63.62 ± 3.33	76.06 ± 1.63
CEGAT (Veličković et al., 2018)	52.82 ± 3.86	58.63 ± 6.82	64.68 ± 2.98	74.24 ± 1.83
HGNN (Feng et al., 2019)	56.46 ± 4.62	58.17 ± 6.73	64.79 ± 2.85	76.53 ± 1.71
HyperGCN (Yadati et al., 2019)	56.70 ± 4.13	57.25 ± 5.75	64.04 ± 2.73	76.88 ± 3.24
HNHN (Dong et al., 2020)	55.97 ± 4.31	59.92 ± 6.02	$\mathbf{66.49 \pm 3.21}$	74.24 ± 1.61
HCHA (Bai et al., 2021)	55.87 ± 3.87	58.09 ± 6.17	65.11 ± 2.73	76.82 ± 2.16
UniGCNII (Huang & Yang, 2021)	55.10 ± 3.52	58.47 ± 8.36	61.81 ± 2.42	77.59 ± 2.96
LEGCN (Yang et al., 2022)	56.94 ± 3.82	58.32 ± 3.30	63.72 ± 2.39	$\underline{79.12 \pm 1.56}$
HyperND (Prokopchik et al., 2022)	57.38 ± 3.57	57.86 ± 5.48	63.62 ± 1.67	75.65 ± 1.11
AllSetTransformer (Chien et al., 2022)	$\underline{57.52 \pm 3.94}$	59.54 ± 5.92	62.66 ± 3.48	76.41 ± 2.52
PhenomNN (Wang et al., 2023b)	55.87 ± 4.13	58.85 ± 6.07	64.68 ± 3.79	77.00 ± 2.78
SheafHyperGNN (Duta et al., 2023)	54.90 ± 3.64	55.57 ± 5.71	62.34 ± 2.17	77.12 ± 1.35
ED-HNN (Wang et al., 2023a)	54.27 ± 4.56	58.09 ± 5.20	62.66 ± 2.27	77.18 ± 2.16
HJRL (Yan et al., 2024)	56.55 ± 4.43	59.62 ± 5.36	64.89 ± 1.60	77.35 ± 1.47
DPHGNN (Saxena et al., 2024)	56.80 ± 2.44	$\underline{61.15 \pm 5.34}$	64.57 ± 2.40	75.82 ± 1.97
HyperGT (Liu et al., 2024)	53.83 ± 3.17	58.24 ± 5.62	63.72 ± 1.90	75.47 ± 1.39
TF-HNN (Tang et al., 2025)	54.61 ± 3.61	54.27 ± 5.04	62.66 ± 2.53	77.94 ± 2.50
DSHN (Mule et al., 2026)	55.87 ± 2.90	55.95 ± 4.33	63.94 ± 1.74	79.00 ± 1.89
HyLPF (ours)	$\mathbf{66.41 \pm 2.45}$	$\mathbf{72.93 \pm 3.13}$	$\underline{66.31 \pm 2.42}$	$\mathbf{79.73 \pm 1.50}$

lished hypergraph-classification results, so we add a graph readout as in the benchmark's graph predictor (Appendix C). It reaches 55.87% and 55.95% on the two social collections, where HyLPF exceeds it by 10.53 and 16.97 points on the same splits, and 63.94% and 79.00% on the two IMDB collections. On Twitter-Friend, every accuracy of the 18 methods and of DSHN lies within five points of the majority-class rate of 59.2% (Appendix E).

IMDB collections. On the IMDB collections, the benchmark's random splits place isomorphic copies of training or validation inputs in about 28% of the test graphs; all methods, including HyLPF, are evaluated under the same benchmark split protocol, and Appendix D quantifies the effect on the benchmark's own splits. We also ran the 18 methods on our five first-batch splits of both IMDB collections (Table 1; Appendix C). On IMDB-Dir-Genre, HyLPF's mean accuracy, 79.73%, is the highest in Table 1, above 79.12% for LEGCN, the best of the 18 methods, and 79.00% for DSHN. As a secondary reference, it is also above the best reported accuracy, 78.12% (Table 5).

Three methods with the highest reported accuracy. On Steam-Player the methods are TF-HNN, HyperGT, and HyperND, and on Twitter-Friend they are AllSetTransformer, DPHGNN, and HNHN. The 48 sampled configurations of each method (39 to 47 distinct) include its published configuration and vary the learning rate, hidden width, number of layers, dropout, weight decay, and mean, max, or sum pooling, as far as the method reads these hyperparameters. When each of these methods is selected among 48 sampled configurations, including the published one and sum pooling, by validation accuracy (Dodge et al., 2019), the 95% lower ends of HyLPF on the first batch exceed, on both social collections, the mean accuracy of every selected and every published configuration (Table 6 in Appendix C). For AllSetTransformer and HyperGCN, a sum readout in place of the published mean readout changes their ten-split mean accuracy by $+0.39$ and -0.05 points on Steam-Player and -1.91 and -0.76 points on Twitter-Friend (Appendix C); this leaves most of the gap to HyLPF for all four pairs. Apart from the readout, both methods keep their published configurations, not retuned.

5.2 Ablation study

Persistence readout. On Steam-Player, HyLPF is 3.71 points (95% interval 2.15 to 5.27) more accurate than its homology-matched twin (Definition 7; Table 2), with every split-level difference

Table 2: **Ablation study** (mean ± SD over splits, ten for the social collections and five for the IMDB collections; gain: paired split-level difference HyLPF − variant with the 95% t interval and the overlap-corrected (NB) interval).

Model	Steam-Player	Twitter-Friend	IMDB-Dir-Form	IMDB-Dir-Genre
Homology-matched twin	62.70 ± 2.54	70.71 ± 2.47	66.81 ± 3.54	79.65 ± 1.20
Gain, 95% t CI	+3.71 [+2.15, +5.27]	+2.21 [−0.11, +4.54]	−0.50 [−2.37, +1.38]	+0.08 [−0.78, +0.94]
NB 95% CI	[+1.36, +6.05]	[−1.28, +5.70]	[−2.89, +1.90]	[−1.02, +1.18]
Value-ablated twin	66.39 ± 3.39	72.62 ± 2.84	65.89 ± 3.71	79.49 ± 2.20
Gain, 95% t CI	+0.02 [−1.27, +1.30]	+0.31 [−2.06, +2.67]	+0.43 [−1.35, +2.20]	+0.24 [−1.70, +2.17]
NB 95% CI	[−1.91, +1.94]	[−3.24, +3.85]	[−1.84, +2.69]	[−2.24, +2.71]
No RWPE	61.29 ± 4.24	72.42 ± 3.30	67.45 ± 2.84	79.61 ± 2.14
Gain, 95% t CI	+5.11 [+2.89, +7.33]	+0.51 [−2.62, +3.64]	−1.13 [−3.11, +0.84]	+0.12 [−1.78, +2.02]
NB 95% CI	[+1.78, +8.45]	[−4.18, +5.20]	[−3.65, +1.38]	[−2.31, +2.54]
No propagation ($L = 1$)	65.73 ± 3.15	70.46 ± 2.13	66.06 ± 2.87	80.04 ± 1.69
Gain, 95% t CI	+0.68 [−1.13, +2.49]	+2.47 [−0.05, +4.98]	+0.25 [−1.45, +1.94]	−0.31 [−2.03, +1.40]
NB 95% CI	[−2.04, +3.40]	[−1.30, +6.24]	[−1.92, +2.41]	[−2.50, +1.88]
HyLPF (ours)	66.41 ± 2.45	72.93 ± 3.13	66.31 ± 2.42	79.73 ± 1.50

positive (Table 8). The twin has the architecture and total parameter count of HyLPF, but its constant inputs leave 188,610 active parameters against 230,594 (Appendix B), so this gain reflects both the persistence-derived inputs and the weights that read them. The value-ablated twin (Definition 7) has the 230,594 active parameters of HyLPF. On Steam-Player, it exceeds the homology-matched twin by 3.69 points (2.54 to 4.83; Table 13), with every split-level difference positive, so the cycle tokens and the essential flags carry label information together with the weights that read them. HyLPF differs from this parameter-matched twin by $+0.02$ points (-1.27 to $+1.30$), so the persistence values (merge values and cycle values) add no detectable accuracy beyond the cycle rank and the essential flags. On Steam-Player, HyLPF exceeds one logistic regression on 22 graph-level count features by 6.65 points (4.57 to 8.73), also under the overlap correction, so this count baseline does not account for HyLPF's accuracy there (Appendix C). The regression reads 22 label-free graph-level features of the star graph: $|V|$, $|E|$, $|I|$, the number β_0 of connected components, the cycle rank β_1, the mean, maximum, and standard deviation of the node degrees and of the hyperedge sizes, and $\log(1 + x)$ of these 11 values.

Variants. The variant "No RWPE" replaces the random-walk positional encodings that initialize the positional states by constant inputs and keeps everything else. The variant "No propagation ($L = 1$)" reads only the descriptor $g^{(0)}$ of the initial positional states, so it also drops the other three layer descriptors and has about a quarter of the total parameters of HyLPF. Unlike the twins, these two variants do not keep the configuration and the parameter tensors of HyLPF.

Positional initialization and propagation. On Steam-Player, the random-walk initialization of the positional states raises accuracy by 5.11 points (95% interval 2.89 to 7.33) over constant inputs, also under the overlap correction. All contrasts in Table 2 other than this gain and the gain over the homology-matched twin on Steam-Player, including those of the homology-matched twin on Twitter-Friend (Table 9), have 95% intervals that contain zero, so they are inconclusive.

6 Conclusion

We presented HyLPF, a hypergraph classifier that reads only the persistent homology of learned filtrations induced by positional states propagated over hypergraph incidences. On the four collections of a public benchmark, under its split protocol and metric, the approach achieves the best overall accuracy, ranking first on three and second on the fourth. On one of the two social collections, the cycle rank and the essential-vertex flags of these filtrations, with the weights that read them, carry label information beyond a homology-matched control, while the merge values and cycle values add no detectable accuracy beyond them. On the same collection, the random-walk initialization of the positional states raises accuracy over constant inputs.

REPRODUCIBILITY STATEMENT

Section 3 and Appendices A and B specify the models and every hyperparameter, and Appendix A proves the stated properties. Section 4 and Appendices C and D specify the splits, training, selection, the one-time test, and the analyses. Result tables and data figures are generated by scripts from the saved per-fit results; the values of the dataset-statistics table were computed by a script from the raw collection files and entered by hand. The code is provided in the anonymous supplementary material.

AI-USE DISCLOSURE

As guided by the conference AI disclosure policy, in this work, we used generative AI tools to assist in the writing of proofs, design or provide feedback on research methodology or experiments, implement methods, assist with translation, clean and reformat datasets, formulate mathematical claims and support qualitative and thematic data analysis. We have not used generative AI tools to generate synthetic datasets, help develop theoretical models or conceptual frameworks, provide critical ingredients for proving mathematical claims, interpret results, propose or refine hypotheses. Additionally, we also used generative AI tools to search related works, formulate questions for surveys or interviews, help authors create or modify scientific figures or images, help authors build experiment pipelines, help improve the writing quality and clarity of the human-drafted manuscript, edit a research paper to improve readability, sourcing/searching for information and check for potential plagiarism, brainstorming, identify relevant literature, format references. All work is first drafted by human authors and subsequently reviewed and edited by human authors, with the help of generative AI tools. And all AI-assisted work has been reviewed and verified by human authors. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Sameer Agarwal, Kristin Branson, and Serge Belongie. Higher order learning with graphs. In *Proceedings of the 23rd International Conference on Machine Learning (ICML)*, pp. 17–24, 2006.

Mehmet Emin Aktas, Thu Nguyen, Rakin Riza, Muhammad Ifte Islam, Ahmed Guler, and Esra Akbas. Persistent homology of hypergraphs via barycentric subdivision. *Journal of Complex Networks*, 14(3):cnag013, 2026.

Song Bai, Feihu Zhang, and Philip H. S. Torr. Hypergraph convolution and hypergraph attention. *Pattern Recognition*, 110:107637, 2021.

Björn Barz and Joachim Denzler. Do we train on test data? Purging CIFAR of near-duplicates. *Journal of Imaging*, 6(6):41, 2020.

Xavier Bouthillier, Pierre Delaunay, Mirko Bronzi, Assya Trofimov, Brennan Nichyporuk, Justin Szeto, Nazanin Mohammadi Sepahvand, Edward Raff, Kanika Madan, Vikram Voleti, Samira Ebrahimi Kahou, Vincent Michalski, Tal Arbel, Chris Pal, Gael Varoquaux, and Pascal Vincent. Accounting for variance in machine learning benchmarks. In *Proceedings of Machine Learning and Systems*, volume 3, pp. 747–769, 2021.

Davide Buffelli, Farzin Soleymani, and Bastian Rieck. CliquePH: Higher-order information for graph neural networks through persistent homology on clique graphs. In *Proceedings of the Third Learning on Graphs Conference (LoG)*, volume 269 of *Proceedings of Machine Learning Research*, pp. 45:1–45:17, 2025.

Mathieu Carrière, Frédéric Chazal, Yuichi Ike, Théo Lacombe, Martin Royer, and Yuhei Umeda. PersLay: A neural network layer for persistence diagrams and new graph topological signatures. In *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 108 of *Proceedings of Machine Learning Research*, pp. 2786–2796, 2020.

Gavin C. Cawley and Nicola L. C. Talbot. On over-fitting in model selection and subsequent selection bias in performance evaluation. *Journal of Machine Learning Research*, 11(70): 2079–2107, 2010.

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 (ICLR)*, 2022.

Uthsav Chitra and Benjamin Raphael. Random walks on hypergraphs with edge-dependent vertex weights. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, volume 97 of *Proceedings of Machine Learning Research*, pp. 1172–1181, 2019.

Seongjin Choi, Gahee Kim, and Se-Young Yun. Weisfeiler and Lehman go categorical. *arXiv preprint arXiv:2602.06787*, 2026.

Jesse Dodge, Suchin Gururangan, Dallas Card, Roy Schwartz, and Noah A. Smith. Show your work: Improved reporting of experimental results. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 2185–2194, 2019.

Yihe Dong, Will Sawin, and Yoshua Bengio. HNHN: Hypergraph networks with hyperedge neurons. Graph Representation Learning and Beyond (GRL+) Workshop at ICML 2020, arXiv preprint arXiv:2006.12278, 2020.

Iulia Duta, Giulia Cassarà, Fabrizio Silvestri, and Pietro Lió. Sheaf hypergraph networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, pp. 12087–12099, 2023.

Vijay Prakash Dwivedi, Anh Tuan Luu, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. Graph neural networks with learnable structural and positional representations. In *International Conference on Learning Representations (ICLR)*, 2022.

Vijay Prakash Dwivedi, Chaitanya K. Joshi, Anh Tuan Luu, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. Benchmarking graph neural networks. *Journal of Machine Learning Research*, 24(43):1–48, 2023.

Herbert Edelsbrunner, David Letscher, and Afra Zomorodian. Topological persistence and simplification. *Discrete & Computational Geometry*, 28(4):511–533, 2002.

Federico Errica, Marco Podda, Davide Bacciu, and Alessio Micheli. A fair comparison of graph neural networks for graph classification. In *International Conference on Learning Representations (ICLR)*, 2020.

Yifan Feng, Haoxuan You, Zizhao Zhang, Rongrong Ji, and Yue Gao. Hypergraph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 3558–3565, 2019.

Yifan Feng, Jiashu Han, Shihui Ying, and Yue Gao. Hypergraph isomorphism computation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(5):3880–3896, 2024.

Christoph Hofer, Florian Graf, Bastian Rieck, Marc Niethammer, and Roland Kwitt. Graph filtration learning. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, volume 119 of *Proceedings of Machine Learning Research*, pp. 4314–4323, 2020.

Max Horn, Edward De Brouwer, Michael Moor, Yves Moreau, Bastian Rieck, and Karsten Borgwardt. Topological graph neural networks. In *International Conference on Learning Representations (ICLR)*, 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)*, pp. 2563–2569, 2021.

Yinan Huang, William Lu, Joshua Robinson, Yu Yang, Muhan Zhang, Stefanie Jegelka, and Pan Li. On the stability of expressive positional encodings for graphs. In *International Conference on Learning Representations (ICLR)*, 2024.

Johanna Immonen, Amauri H. Souza, and Vikas Garg. Going beyond persistent homology using persistent homology. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, pp. 63150–63173, 2023.

Sergei Ivanov, Sergei Sviridov, and Evgeny Burnaev. Rethinking graph classification problem in presence of isomorphism. *Doklady Mathematics*, 110(Suppl. 1):S312–S331, 2024.

Sayash Kapoor and Arvind Narayanan. Leakage and the reproducibility crisis in machine-learning-based science. *Patterns*, 4(9):100804, 2023.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2017.

Fan Li, Xiaoyang Wang, Wenjie Zhang, Ying Zhang, and Xuemin Lin. DHG-Bench: A comprehensive benchmark for deep hypergraph learning. In *International Conference on Learning Representations (ICLR)*, pp. 47693–47727, 2026.

Pan Li, Yanbang Wang, Hongwei Wang, and Jure Leskovec. Distance encoding: Design provably more powerful neural networks for graph representation learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pp. 4465–4478, 2020.

Derek Lim, Joshua Robinson, Lingxiao Zhao, Tess Smidt, Suvrit Sra, Haggai Maron, and Stefanie Jegelka. Sign and basis invariant networks for spectral graph representation learning. In *International Conference on Learning Representations (ICLR)*, 2023.

Zexi Liu, Bohan Tang, Ziyuan Ye, Xiaowen Dong, Siheng Chen, and Yanfeng Wang. Hypergraph transformer for semi-supervised classification. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 7515–7519, 2024.

Zhishang Luo, Truong Son Hy, Puoya Tabaghi, Michaël Defferrard, Elahe Rezaei, Ryan M. Carey, Rhett Davis, Rajeev Jain, and Yusu Wang. DE-HNN: An effective neural model for Circuit Netlist representation. In *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 238 of *Proceedings of Machine Learning Research*, pp. 4258–4266, 2024.

Yixuan Ma, Kai Yi, Pietro Liò, Shi Jin, and Yu Guang Wang. How particle system theory enhances hypergraph message passing. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 38, 2025.

Emanuele Mule, Stefano Fiorini, Antonio Purificato, Federico Siciliano, Stefano Coniglio, and Fabrizio Silvestri. Directional sheaf hypergraph networks: Unifying learning on directed and undirected hypergraphs. In *International Conference on Learning Representations (ICLR)*, pp. 81483–81512, 2026.

Claude Nadeau and Yoshua Bengio. Inference for the generalization error. *Machine Learning*, 52 (3):239–281, 2003.

Raphaël Pellegrin, Lukas Fesser, and Melanie Weber. Higher-order learning with graph neural networks via hypergraph encodings. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 38, 2025.

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 *International Conference on Learning Representations (ICLR)*, 2023.

Konstantin Prokopchik, Austin R. Benson, and Francesco Tudisco. Nonlinear feature diffusion on hypergraphs. In *Proceedings of the 39th International Conference on Machine Learning (ICML)*, volume 162 of *Proceedings of Machine Learning Research*, pp. 17945–17958, 2022.

Ladislav Rampášek, Mikhail Galkin, Vijay Prakash Dwivedi, Anh Tuan Luu, Guy Wolf, and Dominique Beaini. Recipe for a general, powerful, scalable graph transformer. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, pp. 14501–14515, 2022.

Khaled Mohammed Saifuddin, Mehmet Emin Aktas, and Esra Akbas. Topology-guided hypergraph transformer network: Unveiling structural insights. In *PRICAI 2025: Trends in Artificial Intelligence*, Lecture Notes in Computer Science, pp. 346–362, 2026.

Siddhant Saxena, Shounak Ghatak, Raghu Kolla, Debashis Mukherjee, and Tanmoy Chakraborty. DPHGNN: A dual perspective hypergraph neural networks. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pp. 2548–2559, 2024.

Bohan Tang, Zexi Liu, Keyue Jiang, Siheng Chen, and Xiaowen Dong. Training-free message passing for learning on hypergraphs. In *International Conference on Learning Representations (ICLR)*, 2025.

Valeriya V. Tynchenko, Sergei O. Kurashkin, Aleksei S. Borodulin, Tee Connie, Ahmad Hammoud, and Vadim S. Tynchenko. Is there a best hypergraph neural network? A significance-aware recomputation and statistical audit of DHG-Bench. *Machine Learning and Knowledge Extraction*, 8(8):224, 2026.

Emiel van Miltenburg, Chris van der Lee, and Emiel Krahmer. Preregistering NLP research. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pp. 613–623, 2021.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations (ICLR)*, 2018.

Yogesh Verma, Amauri H. Souza, and Vikas Garg. Topological neural networks go persistent, equivariant, and continuous. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 49388–49407, 2024.

Yogesh Verma, Amauri H. Souza, and Vikas K. Garg. Positional encoding meets persistent homology on graphs. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, volume 267 of *Proceedings of Machine Learning Research*, pp. 61323–61343, 2025.

Fuli Wang, Karelia Pena-Pena, Wei Qian, and Gonzalo R. Arce. T-HyperGNNs: Hypergraph neural networks via tensor representations. *IEEE Transactions on Neural Networks and Learning Systems*, 36(3):5044–5058, 2025a.

Haorui Wang, Haoteng Yin, Muhan Zhang, and Pan Li. Equivariant and stable positional encoding for more powerful graph neural networks. In *International Conference on Learning Representations (ICLR)*, 2022.

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

Yifan Wang, Gonzalo R. Arce, and Guangmo Tong. Generalization performance of hypergraph neural networks. In *Proceedings of the ACM on Web Conference 2025 (WWW)*, pp. 1273–1291, 2025b.

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 (ICML)*, volume 202 of *Proceedings of Machine Learning Research*, pp. 35605–35623, 2023b.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations (ICLR)*, 2019.

Naganand Yadati, Madhav Nimishakavi, Prateek Yadav, Vikram Nitin, Anand Louis, and Partha Talukdar. HyperGCN: A new method for training graph convolutional networks on hypergraphs. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32, 2019.

Yuguang Yan, Yuanlin Chen, Shibo Wang, Hanrui Wu, and Ruichu Cai. Hypergraph joint representation learning for hypervertices and hyperedges via cross expansion. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 9232–9240, 2024.

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 & Knowledge Management (CIKM)*, pp. 2352–2361, 2022.

Jiaxuan You, Rex Ying, and Jure Leskovec. Position-aware graph neural networks. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, volume 97 of *Proceedings of Machine Learning Research*, pp. 7134–7143, 2019.

Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Ruslan Salakhutdinov, and Alexander Smola. Deep sets. In *Advances in Neural Information Processing Systems*, 2017.

Detian Zhang, Chengqiang Zhang, Yanghui Rao, Li Qing, and Chunjiang Zhu. Improved expressivity of hypergraph neural networks through high-dimensional generalized Weisfeiler-Leman algorithms. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, volume 267 of *Proceedings of Machine Learning Research*, pp. 76880–76908, 2025a.

Jiying Zhang, Fuyang Li, Xi Xiao, Guanzi Chen, Tingyang Xu, Yu Rong, Junzhou Huang, and Yatao Bian. A unified random walk, its induced Laplacians and spectral convolutions for deep hypergraph learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(11): 10129–10141, 2025b.

Xinyu Zhang, Qize Jiang, Hanyuan Zhang, and Weiwei Sun. Position-aware hypergraph message-passing neural network. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5, 2025c.

Dengyong Zhou, Jiayuan Huang, and Bernhard Schölkopf. Learning with hypergraphs: Clustering, classification, and embedding. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2006.

A FORMAL DEFINITIONS AND PROOFS

Input conditions. An isomorphism (μ, ν) of Definition 1 maps V to V' and E to E' and preserves incidence, so it is exactly a type-preserving isomorphism $S(H) \to S(H')$. In Steam-Player, Twitter-Friend, IMDB-Dir-Form, and IMDB-Dir-Genre, every node and every hyperedge has at least one incidence, and no incidence is repeated, so D_V and D_E in Definition 2 are invertible. Some hyperedges have identical node sets in 401 hypergraphs of Twitter-Friend, 210 of IMDB-Dir-Form, and 476 of IMDB-Dir-Genre. Definition 1 and the loader keep them as distinct hyperedges. The loader removes repeated incidences and renumbers the hyperedges consecutively. For general inputs, the implementation sets the inverse of a zero degree to zero.

Proof of Proposition 6. An isomorphism contains a bijection $E \to E'$, and $|E| = 6 \neq 4 = |E'|$. Every pair of distinct nodes lies in one hyperedge of H and in two hyperedges of H', so $C(H)$ and $C(H')$ are both the complete graph on V. Every node lies in three pairs and in three triples. The star graph $S(H)$ is connected with 10 vertices and 12 edges, and $S(H')$ is connected with 8 vertices and 12 edges, so their cycle ranks are 3 and 5. For the last claim, let $\lambda_1 < \cdots < \lambda_r$ be the distinct entry values of the incidences of S. The multiplicity $\beta_1(S_{\leq\lambda}) - \beta_1(S_{<\lambda})$ is zero unless λ is an entry value. The graph $S_{<\lambda_1}$ has no edges and cycle rank 0, $S_{<\lambda_{i+1}} = S_{\leq\lambda_i}$ for $i < r$, and $S_{\leq\lambda_r} = S$. The multiplicities therefore sum to $\beta_1(S)$. □

A model whose input is $C(H)$ together with the degree features x_v therefore gives H and H' the same output. The same holds for a weighted expansion that gives each node pair the sum of $w(|e|) > 0$ over the hyperedges containing it and is normalized symmetrically without added self-loops. Its two weighted adjacency matrices are $w(2)A_{K_4}$ and $2w(3)A_{K_4}$, where A_{K_4} is the adjacency matrix of the complete graph on V, so their symmetric normalizations coincide. The readout of Equation (5) sums ρ_1 over the points of $\mathrm{Dgm}_1(f_k)$, whose number differs between H and H' for every filtration. If ρ_1 has a constant nonzero output coordinate, the third block of the input of ρ_{out} therefore differs between H and H' at every layer. The two hypergraphs also differ in $|E|$, so a readout that sums over hyperedges can separate them as well. The proposition concerns what the readout can represent. Among the 18 DHG-Bench methods, CEGCN and CEGAT take the clique expansion as input; the other methods operate on the hypergraph or on other expansions of it, and the proposition makes no statement about them.

Evaluation definitions. Section 4 and Appendix D describe the splits, the estimators, and the leakage audit in words; the following definitions state them exactly.

Definition A.1 (Splits and model selection)**.** For a collection $\mathcal{D}$ of hypergraphs in file order and a split index s, let ϖ_s be the permutation `torch.randperm`$(|\mathcal{D}|)$ drawn from a generator seeded with $42 + s$. The training, validation, and test parts $\mathrm{Tr}_s, \mathrm{Va}_s, \mathrm{Te}_s$ are the first $\lfloor 0.8|\mathcal{D}| \rfloor$ entries of ϖ_s, the next $\lfloor 0.1|\mathcal{D}| \rfloor$, and the remaining ones. A fit with initialization index r seeds its generators with $100r + s$. After epoch η it records the validation accuracy α_η and mean cross-entropy $\mathcal{L}_\eta$, keeps the earliest epoch maximizing $(\alpha_\eta, -\mathcal{L}_\eta)$ lexicographically, and evaluates it once on Te_s.

Definition A.2 (Estimators and verdicts)**.** Let $A_{s,r}$ be the test accuracy (%) of the fit with split index s and initialization index $r \in \{0, 1, 2\}$. The split-level value is $a_s = \frac{1}{3}\sum_{r=0}^{2} A_{s,r}$, and the matched difference to a twin is $\Delta_s = a_s - a_s^{\mathrm{twin}}$. For values $(u_s)_{s\in\mathcal{S}}$ on $n = |\mathcal{S}|$ splits, let $\bar{u} = \frac{1}{n}\sum_s u_s$ and $\hat{\sigma}^2 = \frac{1}{n-1}\sum_s (u_s - \bar{u})^2$. The two-sided 95% confidence interval is $\mathrm{CI}(\mathcal{S}) = \bar{u} \pm \omega_{0.975,\,n-1}\,\hat{\sigma}/\sqrt{n}$, where $\omega_{0.975,\,n-1}$ is the 0.975 quantile of Student's t distribution with $n - 1$ degrees of freedom. A model exceeds a reference accuracy a^* if the lower end of $\mathrm{CI}(\mathcal{S})$ for $u = a$ exceeds a^*, and a matched contrast holds if the lower end for $u = \Delta$ exceeds zero. The leave-one-split-out check counts the $s \in \mathcal{S}$ for which the verdict persists on $\mathcal{S} \setminus \{s\}$. In the comparison with the reported values, a^* is the best accuracy that the benchmark reports for the collection.

Definition A.3 (Duplicates and evaluation gap)**.** Let $h(H)$ be the SHA-256 digest of a canonical form of the typed star graph of H whose vertices are colored by their type and input attributes (paragraph "Input digest" below). Barring hash collisions, $h(H) = h(H')$ iff H and H' are isomorphic. The duplicated test graphs of split s are $T_s = \{H \in \mathrm{Te}_s : h(H) \in h(\mathrm{Tr}_s \cup \mathrm{Va}_s)\}$. For a trained method M and a subset T, $\mathrm{acc}_M(T)$ is its accuracy on T and $\mathrm{BA}_M(T) = \frac{1}{|Y_T|}\sum_{c\in Y_T} \mathrm{acc}_M(\{H \in T : y_H = c\})$, with Y_T the set of classes present in T. The evaluation gap is $\iota_M(s) = \mathrm{acc}_M(\mathrm{Te}_s) - \mathrm{acc}_M(\mathrm{Te}_s \setminus T_s)$, and likewise for BA_M. For accuracy,

$\iota_M(s) = \frac{|T_s|}{|\mathrm{Te}_s|}\big(\mathrm{acc}_M(T_s) - \mathrm{acc}_M(\mathrm{Te}_s \setminus T_s)\big)$. The endpoint is the mean $\bar{\iota}$ over $s = 0, \ldots, 4$ of the method average $\iota(s) = \frac{1}{18}\sum_M \iota_M(s)$, with the interval of Definition A.2.

Feature channel. With $x_v^{(0)} = W_x x_v + c_x$ and the positional states of Equation (2), the feature channel is, for $\ell = 1, \ldots, L$,

$$\begin{aligned} \xi_e^{(\ell)} &= \phi_\xi^{(\ell)}\Big(\big[\xi_e^{(\ell-1)},\ \textstyle\sum_{v\in e}[x_v^{(\ell-1)}, p_v^{(\ell)}]\big]\Big), \qquad \text{with } \xi_e^{(0)} \text{ omitted at } \ell = 1, \\ x_v^{(\ell)} &= \mathrm{Dropout}\Big(\phi_x^{(\ell)}\Big(\big[x_v^{(\ell-1)},\ p_v^{(\ell)},\ \textstyle\sum_{e\ni v}[\xi_e^{(\ell)}, q_e^{(\ell)}]\big]\Big)\Big). \end{aligned} \tag{6}$$

The feature-channel hyperedge update of layer ℓ uses the updated node states $p^{(\ell)}$. HyLPF computes this channel, but its readout t depends only on $p^{(\ell)}, q^{(\ell)}$ for $\ell \le L-1$.

Gradients and cost. Persistence is computed on the CPU by a compiled union-find, which runs on detached filtration values. The values $f(z)$, $d_f(z)$, $\max f$, and the cycle values are then read from the differentiable filtration tensor at the vertices that attain them. The loss therefore trains the filtrations and the positional channel (Hofer et al., 2020; Horn et al., 2022). Sorting and union-find with path halving cost $O(|I| \log |I|)$ per filtration. The mean star graph has 210 edges on Steam-Player and 411 on Twitter-Friend (Appendix E).

Union-find procedure. For one hypergraph and one filtration, the implementation computes the entry values and sorts the incidences by entry value. The sort is stable, so ties keep the input order. A union-find with path halving then processes the incidences in this order. Each component stores the minimum of f on it and the list of its vertices that attain this minimum and have not merged. When an incidence joins two components with different minima, every listed vertex of the component with the larger minimum merges at the current entry value. The merged component keeps the list of the other component. When the minima are equal, the two lists are concatenated and no vertex merges. An incidence whose endpoints are already connected records its entry value as a cycle value. Vertices without a merge value are essential. The merge value is read at the endpoint that attains the entry value, and the value $\max f$ over the hypergraph is assigned to essential vertices.

Readout in the implementation. For every incidence i and filtration k, the implementation forms the token $\mathbf{e}_k \otimes (c\, w_k(i), c)$ with $c = 1$ if i closes a cycle and $c = 0$ otherwise, where $w_k(i)$ is the entry value, and multiplies the output of ρ_1 by c. By Proposition 4, the resulting sum equals the inner double sum of Equation (5). In training mode, the batch statistics of ρ_1 are computed over all rows of the batch, one row per incidence and filtration, including the masked rows with $c = 0$. Equation (5) therefore describes the model in inference mode, in which batch normalization uses running statistics; during training, the statistics of ρ_1 also depend on the masked rows. Proposition 5 concerns inference mode and is unaffected.

Ties and the standard diagram. Let $V = \{v_1, v_2\}$ and $E = \{e\}$ with $e = \{v_1, v_2\}$, so $S(H)$ is the path (v_1, e, v_2), and let $f(v_1) = f(v_2) = 0.2$ and $f(e) = 0.8$. Definition 3 gives $\epsilon_f(v_1) = \epsilon_f(v_2) = 1$, whereas the standard zero-dimensional diagram has one essential class, the finite interval $[0.2, 0.8)$, and a diagonal point for e. If $f(v_2) = 0.2 + \delta$ for any $\delta \in (0, 0.6)$, then $\epsilon_f(v_2) = 0$ and $d_f(v_2) = 0.8$. The flags are therefore discontinuous in f at ties, and the stability of standard persistence diagrams does not apply to them directly.

Proof of Proposition 4. Let $i_1, \ldots, i_{|I|}$ be the incidences in the processing order, with entry values $w(i_1) \le \cdots \le w(i_{|I|})$, and let G_m be the spanning subgraph of $S(H)$ with the incidences $i_1, \ldots, i_m$. By induction on m, after step m: (a) the union-find components are the connected components of G_m; (b) the stored value of each component is the minimum of f on it; (c) a vertex z has no merge value iff $f(z)$ equals the minimum of f on its component. Statements (a) and (b) are standard. For (c), when components U and U' with $\min_U f < \min_{U'} f$ merge, the new minimum is $\min_U f$; the vertices of U' that satisfied (c) have $f(z) = \min_{U'} f > \min_U f$ and merge, and those of U keep (c). When the minima are equal, the minimum and all flags are unchanged. Hence z merges at the first step m after which its component in G_m contains some u with $f(u) < f(z)$, and the recorded value is $\lambda = w(i_m)$. Since $G_m \subseteq S_{\le\lambda}$, $d_f(z) \le \lambda$. For

$\lambda' < \lambda$, all incidences of $S_{\leq\lambda'}$ precede i_m, so $S_{\leq\lambda'} \subseteq G_{m-1}$, whose component of z contains no such u; hence $d_f(z) = \lambda$. A vertex never merges iff no u with $f(u) < f(z)$ lies in its component of $S(H)$, that is, iff $\epsilon_f(z) = 1$, and its value $\max f$ is assigned afterwards. Within the block I_λ of incidences that enter at λ, each incidence either joins two components, lowering the number β_0 of connected components by one, or closes a cycle. The number of cycle values at λ is therefore $|I_\lambda| - \big(\beta_0(S_{<\lambda}) - \beta_0(S_{\leq\lambda})\big) = \beta_1(S_{\leq\lambda}) - \beta_1(S_{<\lambda})$ for every order within the block. If f is injective, every component has a unique minimum, no equal-minimum merge occurs, and each merge kills the minimum of the younger component at the merge value, which is the elder rule. A vertex z with a neighbor u with $f(u) < f(z)$ is isolated in $S_{<f(z)}$ and joined to u at $f(z)$, so $d_f(z) = f(z)$, a diagonal point. □

Proof of Proposition 5. Let Π_V and Π_E be the permutation matrices of μ and ν. Then $B' = \Pi_V B \Pi_E^\top$, $P'_V = \Pi_V P_V \Pi_V^\top$, and $P'_E = \Pi_E P_E \Pi_E^\top$, so the diagonals of their powers, and hence π, are transported by (μ, ν). Degrees are preserved by isomorphisms, so the degree features are transported as well. Affine maps, batch normalization with running statistics, ReLU, and the sigmoid act on each row separately, and the sums in Equations (2) and (6) run over neighborhoods that (μ, ν) maps onto each other. By induction over layers, all states and all filtration values are transported. Definition 3 uses only $S(H)$, f, the order of values, connectivity, and $\max f$, and the type-preserving isomorphism of star graphs preserves all of them. Hence d_f and ϵ_f are transported, and the sublevel graphs of H and H' are isomorphic, which gives equal cycle ranks and $\mathrm{Dgm}_1(f'_k) = \mathrm{Dgm}_1(f_k)$. The sums in Equation (5) and the readout t are therefore equal for H and H'. Equalities hold in exact arithmetic; the order of floating-point summation can change results by rounding. □

Input digest. The canonical form colors each vertex of $S(H)$ by its type and the bytes of its attribute rows. It computes a canonical labeling of the colored graph with bliss in igraph 0.11.9, renumbers nodes and hyperedges accordingly, and sorts the incidences lexicographically. The digest $h(H)$ is the SHA-256 hash of $(|V|, |E|)$ and of the incidence, node-feature, and positional-attribute arrays of this form, with their types and shapes. In the audit, the positional attributes are fixed spectral encodings computed on the canonical form of the input with degree features, which is then canonicalized again with these attributes. Isomorphic inputs therefore have equal canonical forms, equal attributes, and equal digests. Conversely, barring hash collisions, equal digests imply equal canonical forms, and each input is isomorphic to its canonical form. Labels, graph indices, and split membership do not enter h.

Metrics. Macro-F1 is the unweighted mean of the per-class F1 scores over the classes that occur in the labels or predictions of the evaluated part. The per-method interval of Table 16 applies Definition A.2 to $\iota_M(s)$ over $s = 0, \ldots, 4$.

B Model and training details

Table 3: **Configurations of HyLPF and its two twins** (Definition 7). Parameter counts are given for Steam-Player / Twitter-Friend. The twins differ from HyLPF only in the persistence-derived inputs of the readout.

	HyLPF	Homology-matched twin	Value-ablated twin
Readout inputs	f, d_f, ϵ_f, Dgm_1	f	f, ϵ_f, β_1 constant tokens
Classifier input	t	t	t
Positional input	RWPE, $J = 16$	RWPE, $J = 16$	RWPE, $J = 16$
Hidden / positional width	128 / 64	128 / 64	128 / 64
Layers L	4	4	4
Filtrations K (hidden width)	8 (16)	8 (16)	8 (16)
Batch size	32	32	32
Parameters (total)	737,602 / 754,242	737,602 / 754,242	737,602 / 754,242
Parameters (active)	230,594 / 230,594	188,610 / 188,610	230,594 / 230,594

Shared settings. All three models use dropout 0.2, sum pooling, cross-entropy, and Adam (Kingma & Ba, 2015) with learning rate 10^{-3} and no weight decay. The learning rate is halved

when validation accuracy has not improved by a relative 10^{-4} for more than 15 epochs, with a floor of 10^{-5} (PyTorch `ReduceLROnPlateau`). Training runs for at most 200 epochs and stops once more than 50 epochs have passed since the selected epoch of Definition A.1. Evaluation uses batches of 256 hypergraphs. Test batches are built only after the selected epoch is fixed. Node features are degree one-hot vectors over the degree vocabulary of each collection, which is built from the degrees in all of its hypergraphs, test graphs included; the readout of HyLPF does not use them. Hyperedges carry no input features. The MLPs ϕ and ρ have hidden and output width equal to their channel width. The filtration networks ψ have hidden width 16. The parameter counts include the feature channel and the last positional update $\phi_q^{(L)}, \phi_p^{(L)}$, which the readout t of HyLPF does not use. The remaining 230,594 parameters of HyLPF have a path to the loss, and this count is the same on both collections. The homology-matched twin has the same parameters. In the twin, ρ_1 receives only zero tokens and its outputs are masked, so ρ_1 receives no gradient. The essential flags and the third block of the input of ρ_{out} are constant zeros, and weights whose inputs are constant zeros receive no gradient: the first-layer weights of ρ_V and ρ_E on the flags and the first-layer weights of ρ_{out} on the third block. Hence 188,610 parameters of the twin are active (Table 3). In the value-ablated twin, the flags and the cycle tokens are kept, so all 230,594 parameters of HyLPF are active.

Configuration search. The configuration of HyLPF comes from a random search on the six collections of Feng et al. (2024), each partitioned into ten folds. Each fit trained on nine folds, and the held-out fold both selected its best epoch and scored it. The development metric, the mean over folds of the held-out accuracy at each fold's best epoch, is therefore optimistic and served only to rank configurations and in the designation criterion below. The first round ran 24 configurations with ten-fold cross-validation on the six collections, and 16 of the 144 configuration–collection pairs did not complete all ten folds. It drew the hidden width from $\{64, 128, 256\}$, the positional width from $\{32, 64\}$, L from $\{2, 3, 4\}$, and dropout from $\{0, 0.2, 0.5\}$. It also drew the learning rate from $\{10^{-3}, 3 \cdot 10^{-3}\}$, weight decay from $\{0, 10^{-4}, 10^{-3}\}$, sum or mean pooling, K from $\{8, 16\}$, J from $\{8, 16\}$, the batch size from $\{32, 64, 128\}$, and a classifier input of t alone or t with the pooled states. The second round drew 24 configurations around the best region, with the hidden width from $\{128, 256, 384\}$, the positional width from $\{32, 64, 96\}$, L from $\{3, 4, 5\}$, and dropout from $\{0, 0.2, 0.5\}$. It also drew the learning rate from $\{5 \cdot 10^{-4}, 10^{-3}, 2 \cdot 10^{-3}\}$, weight decay from $\{0, 10^{-4}\}$, sum or mean pooling, K from $\{8, 16, 32\}$, the filtration hidden width from $\{16, 32\}$, J from $\{8, 16, 24\}$, and the batch size from $\{32, 64, 128\}$, with t alone as the classifier input. It screened these configurations on at most five of the ten folds and completed the three highest-ranked per collection in this screen to ten folds. The frozen configuration is a first-round configuration. After the first round, it was designated as the single configuration for all collections by an informal cross-collection criterion, not by a formal aggregation rule: under the development metric with one initialization, it exceeded the best cross-validation accuracy reported by Choi et al. (2026) on each of the six collections and ranked among the four best first-round configurations on each. It was retained after the second round and frozen before the re-evaluation. A third round of 16 configurations on one of the two collections not used in this paper, run after the designation, did not change the frozen configuration.

Initializations and splits. Each split s is trained with the initialization indices $r = 0, 1, 2$, which seed the generators with $100r + s$ (Definition A.1). The training set is partitioned into shuffled batches. Eight such partitions are drawn once and reused cyclically, and the batch order is permuted in every epoch. A final batch of one hypergraph is merged into the previous batch. Each fit records the split indices it used. The benchmark's reported values come from the split indices $s = 0, \ldots, 4$ of Definition A.1, that is, split seeds 42 to 46. The re-evaluation uses $s = 10, \ldots, 19$, split seeds 52 to 61: the first batch $\mathcal{S}_1 = \{10, \ldots, 14\}$ (seeds 52 to 56), which also covers the two IMDB collections, and the second batch $\mathcal{S}_2 = \{15, \ldots, 19\}$ (seeds 57 to 61). No earlier run had used these split seeds.

Experimental environment and runs. The re-evaluation runs used RTX 5090 GPUs with PyTorch 2.11 and CUDA 12.8 under Python 3.10. We report the median wall-clock time per fit over the 30 fits of each model and collection in the two batches, for training and the single test evaluation without preprocessing. It was 335 s for HyLPF and 264 s for the homology-matched twin on Steam-Player, and 260 and 171 s on Twitter-Friend. The GPUs were shared with other jobs.

C RE-EVALUATION DETAILS

Table 4: **External comparisons per batch.** Mean test accuracy (%) with the 95% t interval over five splits per batch and over the ten pooled splits. LOO: number of the ten leave-one-split-out intervals whose lower end exceeds the best DHG-Bench accuracy.

Collection	Model	Best DHG-Bench (reported)	Batch 1 (seeds 52–56)	Batch 2 (seeds 57–61)	Ten splits	LOO
Steam-Player	HyLPF (ours)	54.85	68.45 [66.88, 70.01]	64.37 [62.82, 65.91]	66.41 [64.65, 68.16]	10/10
Twitter-Friend	HyLPF (ours)	60.15	72.82 [69.90, 75.75]	73.03 [67.99, 78.06]	72.93 [70.69, 75.16]	10/10

External comparisons. The initialization seeds $100r + s$ of Definition A.1 differ across splits, so each split-level value a_s of Definition A.2 contains the variation of its three initializations. Table 4 gives each batch separately. In both batches, the lower end of the interval of HyLPF exceeds the best DHG-Bench accuracy on both collections. In the pooled analysis, all ten leave-one-split-out intervals keep their lower ends above it.

Same-split run of the DHG-Bench methods. The 18 methods ran on CPU with the upstream DHG-Bench code, their published per-dataset configurations, and one initialization per split, on the splits of Definition A.1 with $s = 10, \ldots, 19$. Table 5 lists every method. On these splits, 16 of the 18 methods on Steam-Player and 14 on Twitter-Friend reach a higher mean accuracy than DHG-Bench reports, so, with respect to the reference values, the same-split comparison is the stricter one. In macro-F1, HyLPF reaches 66.06% on Steam-Player and 70.53% on Twitter-Friend, against at most 56.56% and 51.67% for the 18 methods.

Same-split run on the IMDB collections. The 18 methods were run on the five first-batch splits of IMDB-Dir-Form and IMDB-Dir-Genre ($s = 10, \ldots, 14$) with the upstream code, their published configurations, and one initialization per split, on CPU. Only hosts whose outputs matched those of the main host bit for bit on a gate job were used, and all 180 fits completed. The split files equal those of HyLPF's runs element by element. HyLPF's IMDB values were known before this run was designed. On these splits, 17 of the 18 methods on IMDB-Dir-Form and 11 on IMDB-Dir-Genre reach a higher mean accuracy than DHG-Bench reports (Table 5). The best methods are HNHN (66.49%) on IMDB-Dir-Form and LEGCN (79.12%) on IMDB-Dir-Genre, and the paired split-level differences of HyLPF to them are -0.18 points (95% interval -3.72 to 3.36) and $+0.61$ (-1.41 to 2.63). In mean macro-F1, HyLPF reaches 62.40% and 76.00%, against at most 61.16% and 74.61% for the 18 methods.

Three methods with the highest reported accuracy, over 48 sampled configurations. On Steam-Player the methods are TF-HNN, HyperGT, and HyperND, and on Twitter-Friend they are AllSetTransformer, DPHGNN, and HNHN. The methods were chosen by reported accuracy, so AllSetTransformer, the best same-split method on Steam-Player, is not among the three Steam-Player methods. The 48 sampled configurations of each method (39 to 47 distinct) include its published configuration and vary the learning rate, hidden width, number of layers, dropout, weight decay, and mean, max, or sum pooling, as far as the method reads these hyperparameters. HyperND does not read the number of layers, and DPHGNN reads neither the number of layers nor dropout and reads the hidden width only in its embedding; a method is unaffected by a hyperparameter it does not read, so some draws coincide. The selection used only validation accuracy on the DHG-Bench splits and was fixed before the evaluation fits. The selected configurations use max pooling (TF-HNN, HyperGT, HyperND, DPHGNN), mean pooling (AllSetTransformer), and sum pooling (HNHN). Table 6 lists the selected and the published configuration of every method on split seeds 52 to 56. On this batch, the 95% lower ends of HyLPF, 66.88 on Steam-Player and 69.90 on Twitter-Friend, exceed the best configurations selected on validation accuracy, HyperGT (55.57%) and DPHGNN (59.13%), and the strongest evaluated configuration, the published one of HyperND (58.80%) and of DPHGNN (60.05%). The paired split-level differences of HyLPF to the accuracy-selected configurations of HyperGT and DPHGNN are $+12.88$ $[+9.39, +16.37]$ on Steam-Player and $+13.69$ $[+7.14, +20.24]$ on Twitter-Friend in accuracy, and $+13.70$ $[+10.79, +16.60]$ and $+24.04$ $[+15.21, +32.87]$ in macro-F1.

Table 5: **The 18 DHG-Bench methods on our splits** (split seeds 52 to 61 for the social collections and 52 to 56 for the IMDB collections, one initialization per split): test accuracy mean ± SD, mean macro-F1 (%), and the accuracy reported by DHG-Bench (split seeds 42 to 46; UniGCNII is named UniGNN there).

	Steam-Player (ten splits, seeds 52–61)			Twitter-Friend (ten splits, seeds 52–61)		
Method	Accuracy	Macro-F1	Reported acc.	Accuracy	Macro-F1	Reported acc.
MLP	55.39 ± 2.57	54.77	52.34	59.85 ± 3.69	50.39	57.25
CEGCN	53.59 ± 4.15	51.73	48.16	60.00 ± 5.84	40.91	54.66
CEGAT	52.82 ± 3.86	50.33	49.51	58.63 ± 6.82	39.96	57.32
HGNN	56.46 ± 4.62	55.99	51.65	58.17 ± 6.73	45.55	55.42
HyperGCN	56.70 ± 4.13	56.30	51.17	57.25 ± 5.75	45.76	56.95
HCHA	55.87 ± 3.87	55.58	52.43	58.09 ± 6.17	44.97	58.17
LEGCN	56.94 ± 3.82	56.30	53.11	58.32 ± 3.30	51.67	56.64
HyperND	57.38 ± 3.57	56.38	53.88	57.86 ± 5.48	40.68	55.27
PhenomNN	55.87 ± 4.13	54.81	51.65	58.85 ± 6.07	46.98	57.40
SheafHyperGNN	54.90 ± 3.64	52.83	53.11	55.57 ± 5.71	43.99	56.49
HJRL	56.55 ± 4.43	55.95	51.84	59.62 ± 5.36	46.08	57.10
DPHGNN	56.80 ± 2.44	56.01	51.36	61.15 ± 5.34	43.39	59.24
TF-HNN	54.61 ± 3.61	53.88	54.85	54.27 ± 5.04	50.28	56.18
HNHN	55.97 ± 4.31	55.40	52.82	59.92 ± 6.02	40.11	58.47
UniGCNII	55.10 ± 3.52	52.20	51.46	58.47 ± 8.36	42.88	55.88
AllSetTransformer	57.52 ± 3.94	56.56	53.01	59.54 ± 5.92	46.33	60.15
ED-HNN	54.27 ± 4.56	50.25	52.82	58.09 ± 5.20	42.47	57.40
HyperGT	53.83 ± 3.17	53.02	54.47	58.24 ± 5.62	45.45	54.35
	IMDB-Dir-Form (five splits, seeds 52–56)			IMDB-Dir-Genre (five splits, seeds 52–56)		
Method	Accuracy	Macro-F1	Reported acc.	Accuracy	Macro-F1	Reported acc.
MLP	64.68 ± 2.54	59.36	63.62	75.65 ± 2.18	70.97	75.12
CEGCN	63.62 ± 3.33	57.38	62.66	76.06 ± 1.63	70.68	75.06
CEGAT	64.68 ± 2.98	59.43	63.51	74.24 ± 1.83	68.36	74.12
HGNN	64.79 ± 2.85	60.25	63.72	76.53 ± 1.71	72.25	76.76
HyperGCN	64.04 ± 2.73	59.50	62.87	76.88 ± 3.24	72.33	77.53
HCHA	65.11 ± 2.73	60.00	61.60	76.82 ± 2.16	72.36	78.12
LEGCN	63.72 ± 2.39	58.97	61.81	79.12 ± 1.56	74.61	76.38
HyperND	63.62 ± 1.67	58.35	60.74	75.65 ± 1.11	70.88	75.65
PhenomNN	64.68 ± 3.79	59.41	61.28	77.00 ± 2.78	72.54	74.59
SheafHyperGNN	62.34 ± 2.17	57.42	62.34	77.12 ± 1.35	72.27	77.00
HJRL	64.89 ± 1.60	59.62	63.09	77.35 ± 1.47	72.65	77.82
DPHGNN	64.57 ± 2.40	59.57	64.04	75.82 ± 1.97	71.42	76.18
TF-HNN	62.66 ± 2.53	57.92	62.34	77.94 ± 2.50	73.19	76.41
HNHN	66.49 ± 3.21	61.16	62.34	74.24 ± 1.61	69.77	73.65
UniGCNII	61.81 ± 2.42	56.41	61.06	77.59 ± 2.96	72.91	77.12
AllSetTransformer	62.66 ± 3.48	58.29	62.23	76.41 ± 2.52	70.55	76.47
ED-HNN	62.66 ± 2.27	59.00	62.13	77.18 ± 2.16	72.39	77.12
HyperGT	63.72 ± 1.90	58.42	61.49	75.47 ± 1.39	70.92	73.82

Sum-readout run. This run was designed after the results of all earlier analyses and further internal reviews of an earlier draft had been read, and its analysis plan was fixed before any of its fits. Its single factor is the graph readout of AllSetTransformer and HyperGCN: a sum readout replaces the published mean readout, and everything else is the published configuration, not retuned. Each method was trained once per split on split seeds 52 to 61, with one initialization, and each run is paired with the published run on the same split (Table 5). Both arms use `fix_seed(0)`, and pooling consumes no random numbers, so their initial weights are identical and the paired difference contains only the pooling change and the divergence of the training trajectory that it causes. An environment check reproduced the published runs of both methods bitwise on split seed 52, for AllSetTransformer before the formal run and for HyperGCN alongside it. None of the four paired changes is detected: their ordinary and corrected intervals all contain zero. The share of the gap to HyLPF that the sum readout closes is below one half for all four pairs (Table 7).

Table 6: **The three DHG-Bench methods with the highest reported accuracy on each social collection**, with the configuration selected among 48 on validation accuracy (split seeds 42 to 46) and the published one (#0), on the five splits of the first batch with three initializations per split (test accuracy, %).

				Seeds				
Collection	Method	Configuration	Mean [95% CI]	52	53	54	55	56
Steam-Player	TF-HNN	selected (#33)	54.17 [50.53, 57.82]	54.05	49.51	57.61	54.69	55.02
Steam-Player	TF-HNN	published (#0)	55.50 [51.04, 59.96]	51.62	51.78	57.44	57.12	59.55
Steam-Player	HyperGT	selected (#8)	55.57 [51.52, 59.61]	51.46	53.07	58.41	55.99	58.90
Steam-Player	HyperGT	published (#0)	54.30 [51.54, 57.07]	52.59	51.78	56.15	54.05	56.96
Steam-Player	HyperND	selected (#24)	54.66 [48.92, 60.40]	55.02	47.57	56.63	53.88	60.19
Steam-Player	HyperND	published (#0)	58.80 [55.32, 62.28]	56.80	60.84	54.85	60.68	60.84
Steam-Player	HyLPF (ours)	frozen	**68.45** [66.88, 70.01]	67.31	68.93	68.12	67.48	70.39
Twitter-Friend	AllSetTransformer	selected (#6)	56.69 [50.48, 62.91]	53.94	64.89	52.16	57.76	54.71
Twitter-Friend	AllSetTransformer	published (#0)	58.42 [54.59, 62.26]	54.45	60.31	61.07	60.56	55.73
Twitter-Friend	DPHGNN	selected (#8)	59.13 [51.80, 66.47]	50.38	65.90	59.80	62.60	57.00
Twitter-Friend	DPHGNN	published (#0)	60.05 [51.47, 68.63]	49.11	65.14	64.12	64.63	57.25
Twitter-Friend	HNHN	selected (#9)	58.37 [49.76, 66.99]	47.58	61.07	63.87	63.87	55.47
Twitter-Friend	HNHN	published (#0)	58.42 [49.93, 66.92]	47.33	62.85	62.34	63.36	56.23
Twitter-Friend	HyLPF (ours)	frozen	**72.82** [69.90, 75.75]	73.03	76.08	69.97	73.79	71.25

Table 7: **Sum-readout run** (split seeds 52 to 61, one initialization per split; test accuracy). Published: the same-split runs of Table 5 with the published mean readout; sum: the same configuration with a sum readout. Differences are paired over the ten splits, with the ordinary 95% t interval (9 degrees of freedom) and the Nadeau–Bengio corrected interval (Appendix C.1). Share of gap closed: $(g_{\mathrm{pub}} - g_{\mathrm{sum}})/g_{\mathrm{pub}}$, where g is the ten-split mean accuracy of HyLPF minus that of the method; negative values mean that the gap widens.

		Mean accuracy (%)		Sum − published (points)		Share of	HyLPF − sum (points)	
Collection	Method	Published (mean)	Sum	Mean [95% t]	Corrected (NB)	gap closed	Mean [95% t]	Corrected (NB)
Steam-Player	AllSetTransformer	57.52	57.91	$+0.39$ $[-1.29, +2.06]$	$[-2.13, +2.91]$	$+0.044$	$+8.50$ $[+6.46, +10.53]$	$[+5.44, +11.55]$
Steam-Player	HyperGCN	56.70	56.65	-0.05 $[-1.79, +1.70]$	$[-2.67, +2.57]$	-0.005	$+9.76$ $[+7.69, +11.82]$	$[+6.65, +12.86]$
Twitter-Friend	AllSetTransformer	59.54	57.63	-1.91 $[-6.52, +2.71]$	$[-8.83, +5.01]$	-0.143	$+15.29$ $[+11.34, +19.25]$	$[+9.36, +21.23]$
Twitter-Friend	HyperGCN	57.25	56.49	-0.76 $[-3.55, +2.02]$	$[-4.94, +3.41]$	-0.049	$+16.44$ $[+11.83, +21.04]$	$[+9.53, +23.34]$

DSHN. DSHN (Mule et al., 2026) has no published hypergraph-classification results, so we added a graph readout as in the benchmark's graph predictor, in three stages: the DSHN node network, whose node classifier MLP outputs 64 channels instead of class scores; mean pooling over the nodes of each hypergraph; and an MLP from 64 to 128 channels and then to the classes, with LayerNorm and dropout 0.2. The 16 configurations are the two published configurations for Cora and Citeseer and 14 random draws from the authors' search space, all with input normalization off, so that no test graph affects another through batch statistics. On each collection, the configuration with the highest mean validation accuracy on split seeds 42 to 46 was selected: on Steam-Player hidden width 512, stalk dimension 1, one sheaf layer, learning rate 0.02 and dropout 0.1, and on Twitter-Friend hidden width 256, stalk dimension 5, five sheaf layers, learning rate 0.001 and dropout 0.4. It was trained once per split on split seeds 52 to 61. DSHN reaches $55.87 \pm 2.90\%$ and $55.95 \pm 4.33\%$ (ten-split mean $\pm$ SD) with mean macro-F1 53.82 and 49.83, and predicting the training-majority class scores 48.69% and 60.38% on the test parts. The paired split-level differences HyLPF − DSHN are $+10.53$ $[+8.66, +12.41]$ on Steam-Player and $+16.97$ $[+13.02, +20.93]$ on Twitter-Friend, with corrected intervals in Table 13. On the IMDB collections, the readout outputs three class scores, and the configuration with the highest mean validation accuracy on split seeds 42 to 44 was selected among the first eight of the 16 configurations by index, the two published configurations and six random draws: on IMDB-Dir-Form hidden width 256, stalk dimension 1, three sheaf layers, learning rate 0.005 and dropout 0.1, and on IMDB-Dir-Genre hidden width 256, stalk dimension 2, five sheaf layers, learning rate 0.001 and dropout 0.3. It was trained once per split on the five IMDB splits of HyLPF, split seeds 52 to 56. On these splits, DSHN reaches 63.94% and 79.00% (mean macro-F1 59.38 and 74.14); the paired split-level differences HyLPF − DSHN are $+2.38$ (95% interval -1.89 to 6.64; corrected -3.07 to 7.82) and $+0.73$ (-0.72 to 2.17; corrected -1.12 to 2.57).

Split-level differences on Steam-Player. Table 8 lists the matched split-level differences Δ_s of Definition A.2 on Steam-Player. Every entry is positive.

Table 8: **Split-level matched differences on Steam-Player** (test accuracy, points; each value averages three initializations).

Split seed	52	53	54	55	56	57	58	59	60	61
HyLPF − homology-matched twin	+7.61	+2.10	+1.46	+4.85	+6.31	+4.05	+4.21	+3.24	+0.49	+2.75

Matched contrasts on Twitter-Friend. Table 9 reports the matched contrast of HyLPF with its homology-matched twin on Twitter-Friend. In both batches and in the pooled analysis, every 95% interval contains zero.

Table 9: **Matched contrasts on Twitter-Friend** (test accuracy, points): mean paired split-level difference with the 95% t interval, taken from the endpoint files of the two batches.

Contrast on Twitter-Friend (accuracy, points)	Batch 1 (seeds 52–56)	Batch 2 (seeds 57–61)	Ten splits
HyLPF − homology-matched twin	+1.58 [−2.33, +5.49]	+2.85 [−1.60, +7.30]	+2.21 [−0.11, +4.54]

Execution. All fits of both batches completed. In the second batch, three fits, one of them a HyLPF fit, stopped with out-of-memory errors because other jobs shared the GPUs. The errors occurred while allocating GPU memory, before any result was produced. They were rerun with the same configuration, seed, and split before any test metric was read. The pooled analysis gives the ten split-level values of both batches equal weight.

Value-ablated twin. The value-ablated twin of Definition 7 has the configuration and the parameter tensors of HyLPF. It computes the persistence of every learned filtration and evaluates Equation (5) with $d_{f_k}(z)$ replaced by $f_k(z)$, as in the homology-matched twin, with the essential flags $\epsilon_{f_k}(z)$ kept, and with every point $b \in \mathrm{Dgm}_1(f_k)$ replaced by the constant $1/2$, so that each filtration contributes $\beta_1(S(H))$ identical cycle tokens (Proposition 6). Its readout inputs are thus functions of those of HyLPF, and the inputs of the homology-matched twin are functions of its inputs. The three models are therefore nested in their inputs, with the same hyperparameters and parameter tensors; they are trained separately, so the nesting does not order their accuracies. The value-ablated twin has the 230,594 active parameters of HyLPF, and the homology-matched twin has 188,610 (Table 3). On Steam-Player, the value-ablated twin exceeds the homology-matched twin by $+3.69$ $[+2.54, +4.83]$ points in accuracy, with a positive difference on all ten splits, and by $+4.72$ $[+3.53, +5.90]$ in macro-F1 (Table 10); on Twitter-Friend, the accuracy difference is $+1.91$ $[-0.01, +3.83]$, an interval that contains zero (Table 13). The paired difference between HyLPF and the value-ablated twin is $+0.02$ $[-1.27, +1.30]$ in accuracy and $+0.04$ $[-1.28, +1.35]$ in macro-F1 on Steam-Player, and $+0.31$ $[-2.06, +2.67]$ and $+0.97$ $[-1.38, +3.33]$ on Twitter-Friend. The rows of HyLPF and of the homology-matched twin are the fits of the two batches. The 60 fits of the value-ablated twin, on two collections with ten splits and three initializations each, used the same training code: between the two sets of runs, the code changed only by the value-ablated branch and by a split option that these runs do not use, and two HyLPF fits rerun with the new code reproduced the split manifest, dataset digest, parameter count, and model semantics of the reused rows. Its analysis plan was fixed after the results of the first and second batches had been read and before its runs.

Extended ablation. The two variants of Section 5.2, on all four collections, and the value-ablated twin on the two IMDB collections were trained with the configuration of HyLPF apart from the ablated component, on the same splits and with three initializations per split. The variant "No propagation ($L = 1$)" has about a quarter of the parameters of HyLPF (182,194 against 737,602 total parameters on Steam-Player). The remaining rows of Table 2 come from the runs of the two batches. The ablation plan was fixed before these runs. Table 11 lists the first-batch split-level accuracies of all arms.

Table 10: **Value-ablated twin on the ten splits** (split seeds 52 to 61, three initializations per split). Top: mean ± SD over splits of test accuracy and macro-F1 (%), the number of active parameters, and the paired split-level difference between HyLPF and the value-ablated twin with its 95% t interval (9 degrees of freedom). Bottom: split-level test accuracy (%), each value averaging three initializations. The rows of HyLPF and of the homology-matched twin come from the same fits as Tables 1 and 2.

Collection	Arm	Active parameters	Accuracy (%)	Macro-F1 (%)
Steam-Player	Homology-matched twin	188,610	62.70 ± 2.54	61.31 ± 3.00
Steam-Player	Value-ablated twin	230,594	66.39 ± 3.39	66.03 ± 3.32
Steam-Player	HyLPF (ours)	230,594	66.41 ± 2.45	66.06 ± 2.45
Twitter-Friend	Homology-matched twin	188,610	70.71 ± 2.47	68.05 ± 2.53
Twitter-Friend	Value-ablated twin	230,594	72.62 ± 2.84	69.56 ± 2.48
Twitter-Friend	HyLPF (ours)	230,594	72.93 ± 3.13	70.53 ± 2.97

Paired difference, points [95% CI]		Accuracy	Macro-F1
Steam-Player	HyLPF − value-ablated twin	$+0.02\ [-1.27, +1.30]$	$+0.04\ [-1.28, +1.35]$
Twitter-Friend	HyLPF − value-ablated twin	$+0.31\ [-2.06, +2.67]$	$+0.97\ [-1.38, +3.33]$

Collection	Arm	$s=10$	$s=11$	$s=12$	$s=13$	$s=14$	$s=15$	$s=16$	$s=17$	$s=18$	$s=19$
Steam-Player	Homology-matched twin	59.71	66.83	66.67	62.62	64.08	60.03	62.14	60.19	62.78	61.97
Steam-Player	Value-ablated twin	64.24	71.20	70.06	67.15	71.52	62.46	64.56	62.94	65.05	64.72
Steam-Player	HyLPF (ours)	67.31	68.93	68.12	67.48	70.39	64.08	66.34	63.43	63.27	64.72
Twitter-Friend	Homology-matched twin	67.18	72.26	69.72	75.57	71.50	69.47	72.26	68.45	72.01	68.70
Twitter-Friend	Value-ablated twin	70.23	74.81	75.06	74.55	67.43	71.25	75.83	72.01	75.06	69.97
Twitter-Friend	HyLPF (ours)	73.03	76.08	69.97	73.79	71.25	77.10	75.06	72.52	74.05	66.41

Table 11: **Split-level test accuracy of the ablation arms** on the five splits of the first batch ($s = 10, \ldots, 14$; %, each value averaging three initializations; split indices as in Appendix B). For the social collections, Table 2 averages these and the five splits of the second batch.

Collection	Arm	$s=10$	$s=11$	$s=12$	$s=13$	$s=14$
Steam-Player	Homology-matched twin	59.71	66.83	66.67	62.62	64.08
Steam-Player	Value-ablated twin	64.24	71.20	70.06	67.15	71.52
Steam-Player	No RWPE	62.46	64.24	63.92	62.62	66.50
Steam-Player	No propagation ($L=1$)	62.94	70.06	69.58	63.43	70.06
Steam-Player	HyLPF (ours)	67.31	68.93	68.12	67.48	70.39
Twitter-Friend	Homology-matched twin	67.18	72.26	69.72	75.57	71.50
Twitter-Friend	Value-ablated twin	70.23	74.81	75.06	74.55	67.43
Twitter-Friend	No RWPE	65.90	77.10	75.32	70.99	69.97
Twitter-Friend	No propagation ($L=1$)	68.45	71.50	70.74	72.26	68.70
Twitter-Friend	HyLPF (ours)	73.03	76.08	69.97	73.79	71.25
IMDB-Dir-Form	Homology-matched twin	64.18	64.18	71.28	70.04	64.36
IMDB-Dir-Form	Value-ablated twin	64.01	63.12	71.99	66.84	63.48
IMDB-Dir-Form	No RWPE	66.31	64.01	71.81	67.73	67.38
IMDB-Dir-Form	No propagation ($L=1$)	62.94	64.36	70.39	67.20	65.43
IMDB-Dir-Form	HyLPF (ours)	65.25	64.89	70.04	67.38	64.01
IMDB-Dir-Genre	Homology-matched twin	79.12	78.24	79.41	81.47	80.00
IMDB-Dir-Genre	Value-ablated twin	79.31	77.35	80.39	82.75	77.65
IMDB-Dir-Genre	No RWPE	78.14	78.24	79.31	83.33	79.02
IMDB-Dir-Genre	No propagation ($L=1$)	79.41	79.61	79.02	83.04	79.12
IMDB-Dir-Genre	HyLPF (ours)	80.29	77.55	79.12	81.57	80.10

Count-feature baseline. This analysis was designed after the results above and two internal reviews of an earlier draft had been read, and its analysis plan was fixed before any fit. On the ten splits of Table 1, a logistic regression with an L_2 penalty reads 22 label-free graph-level features of the star graph: $|V|$, $|E|$, $|I|$, the number β_0 of connected components, the cycle rank β_1, the mean, maximum, and standard deviation of the node degrees and of the hyperedge sizes, and $\log(1+x)$ of these 11 values. The features are standardized on the training part, the inverse regularization

strength C is selected from $\{0.01, 0.1, 1, 10, 100\}$ by validation accuracy, with ties broken toward the smaller value, the model is fitted on the training part only, and the selected model is evaluated once on the test part. The selected C reached the grid maximum (100) in 5 of 10 splits on each collection; this is one count model, and stronger count models may reach higher accuracy. Table 12 pairs the regression on each split with HyLPF and the best of the 18 methods on that split. In a descriptive comparison, the ten-split mean of the regression exceeds that of each of the 18 methods on both collections.

Table 12: **Count-feature baseline** (count LR; split seeds 52 to 61, one selected deterministic model per split). Test accuracy as mean ± SD over splits, mean macro-F1, and the test accuracy of predicting the class that is most frequent in the training part. The last row compares ten-split means. Paired-difference rows show ordinary 95% t intervals (9 degrees of freedom) and Nadeau–Bengio corrected intervals (Appendix C.1); †: the ordinary lower end is above zero, the corrected one is not.

	Steam-Player		Twitter-Friend	
	Value [95% t]	Corrected (NB)	Value [95% t]	Corrected (NB)
Count LR, test accuracy (%)	59.76 ± 2.16		68.55 ± 3.29	
Count LR, macro-F1 (%)	59.56		65.29	
Training-majority class, test accuracy (%)	48.69		60.38	
Splits with $C = 100$ selected	5 of 10		5 of 10	
Paired split-level difference in accuracy, points [95% t interval, 9 degrees of freedom]				
HyLPF − count LR	$+6.65$ $[+4.57, +8.73]$	$[+3.52, +9.78]$	$+4.38$ $[+1.02, +7.74]$	$[-0.66, +9.42]^{\dagger}$
Count LR − best of the 18 methods in the same split	-0.39 $[-1.95, +1.17]$	$[-2.73, +1.96]$	$+5.11$ $[+2.32, +7.91]$	$[+0.92, +9.31]$
Count LR − highest ten-split mean of the 18 methods	$+2.23$		$+7.40$	

C.1 Overlap-corrected intervals

Table 13 recomputes the intervals of the main comparisons with the corrected resampled t interval of Nadeau & Bengio (2003) (Section 4), as a post hoc sensitivity analysis; the analysis plans specified ordinary t intervals, which the paper reports throughout. Using $n_{\text{test}}/n_{\text{train}} = |\text{Te}_s|/(|\text{Tr}_s| + |\text{Va}_s|)$ instead gives the same conclusion for every row.

D Input duplicates in the benchmark's IMDB splits

Purpose and design. This audit checks whether random splits of the benchmark's IMDB collections place test graphs that are isomorphic to a training or validation input; such duplicates can inflate test accuracy. All methods compared in Table 1, including HyLPF, are evaluated under the same benchmark split protocol. We reconstruct the benchmark's IMDB-Dir-Form and IMDB-Dir-Genre splits for $s = 0, \ldots, 4$, that is, split seeds 42 to 46. A test graph is duplicated if the isomorphism-invariant digest of its typed star graph equals that of a training or validation graph. The evaluation gap of a method, its accuracy or balanced accuracy on all test graphs minus that on the non-duplicated ones, is a descriptive difference between scoring subsets of one trained model, not the effect of retraining without duplicates. The endpoint is its mean over the 18 methods, with the interval over five splits (Definition A.3).

Results. In the benchmark's splits, 28.5% of the IMDB-Dir-Form and 27.7% of the IMDB-Dir-Genre test graphs duplicate a training or validation input (Tables 14 and 15). On IMDB-Dir-Form, the 18 methods average 88.9% accuracy on these graphs and 51.8% on the others (Figure 3), and their mean evaluation gap, the accuracy on all test graphs minus that on the non-duplicated ones, is 10.61 points (95% interval 7.93 to 13.29); on IMDB-Dir-Genre, it is 2.36 points (1.40 to 3.31). The balanced-accuracy gaps are 12.45 and 12.25 points; on IMDB-Dir-Genre, where duplicates concentrate in one class, this gap mixes the class composition of the non-duplicated subset with any effect of the duplicates. All 18 per-method gaps are positive on both collections, with 95% intervals above zero on IMDB-Dir-Form (Table 16).

Split reconstruction and identity. The audit reconstructs DHG-Bench's graph-level splits exactly, using `random_split` with generator seed $42 + s$ for $s = 0, \ldots, 4$. The reconstructed splits are identical under two PyTorch versions. Duplicates are identified by the digest h of Definition A.3

Table 13: **Nadeau–Bengio corrected 95% intervals** (post hoc sensitivity analysis; $n_{\text{test}}/n_{\text{train}} = |\text{test}|/|\text{train}|$; n splits). Paired differences are compared with zero and means with the stated values, which are treated as fixed. The last block repeats the two intervals marked with † in Table 16. Throughout the paper, † marks an interval whose lower end is above its threshold under the ordinary t interval but not under the correction.

Collection	Comparison	n	Mean	95% t interval	Corrected interval
Same-split comparisons (18 benchmark methods and DSHN; IMDB: 18 methods, split seeds 52–56)					
Steam-Player	HyLPF − AllSetTransformer	10	8.88	[6.10, 11.67]	[4.70, 13.06]
Steam-Player	HyLPF − DSHN	10	10.53	[8.66, 12.41]	[7.72, 13.35]
Steam-Player	HyLPF mean vs. 57.52 (AllSetTransformer)	10	66.41	[64.65, 68.16]	[63.77, 69.04]
Twitter-Friend	HyLPF − DPHGNN	10	11.78	[7.91, 15.65]	[5.97, 17.59]
Twitter-Friend	HyLPF − DSHN	10	16.97	[13.02, 20.93]	[11.04, 22.91]
Twitter-Friend	HyLPF mean vs. 61.15 (DPHGNN)	10	72.93	[70.69, 75.16]	[69.57, 76.28]
IMDB-Dir-Form	HyLPF − HNHN	5	−0.18	[−3.72, 3.36]	[−4.69, 4.34]
IMDB-Dir-Form	HyLPF mean vs. 66.49 (HNHN)	5	66.31	[63.31, 69.32]	[62.48, 70.15]
IMDB-Dir-Genre	HyLPF − LEGCN	5	0.61	[−1.41, 2.63]	[−1.97, 3.18]
IMDB-Dir-Genre	HyLPF mean vs. 79.12 (LEGCN)	5	79.73	[77.87, 81.58]	[77.35, 82.10]
Reported values					
Steam-Player	HyLPF mean vs. reported value 54.85	10	66.41	[64.65, 68.16]	[63.77, 69.04]
Twitter-Friend	HyLPF mean vs. reported value 60.15	10	72.93	[70.69, 75.16]	[69.57, 76.28]
Configurations selected among 48 ($n = 5$)					
Steam-Player	HyLPF − HyperGT (selected)	5	12.88	[9.39, 16.37]	[8.42, 17.34]
Twitter-Friend	HyLPF − DPHGNN (selected)	5	13.69	[7.14, 20.24]	[5.34, 22.04]
Ablations					
Steam-Player	HyLPF − homology-matched twin	10	3.71	[2.15, 5.27]	[1.36, 6.05]
Steam-Player	Value-ablated twin − homology-matched twin	10	3.69	[2.54, 4.83]	[1.97, 5.41]
Steam-Player	HyLPF − value-ablated twin	10	0.02	[−1.27, 1.30]	[−1.91, 1.94]
Twitter-Friend	HyLPF − homology-matched twin	10	2.21	[−0.11, 4.54]	[−1.28, 5.70]
Twitter-Friend	Value-ablated twin − homology-matched twin	10	1.91	[−0.01, 3.83]	[−0.98, 4.79]
Twitter-Friend	HyLPF − value-ablated twin	10	0.31	[−2.06, 2.67]	[−3.24, 3.85]
Count LR					
Steam-Player	HyLPF − count LR	10	6.65	[4.57, 8.73]	[3.52, 9.78]
Steam-Player	Count LR − best of the 18 methods in the same split	10	−0.39	[−1.95, 1.17]	[−2.73, 1.96]
Steam-Player	Count LR mean vs. highest ten-split mean of the 18 methods (57.52)	10	59.76	[58.21, 61.31]	[57.43, 62.08]†
Twitter-Friend	HyLPF − count LR	10	4.38	[1.02, 7.74]	[−0.66, 9.42]†
Twitter-Friend	Count LR − best of the 18 methods in the same split	10	5.11	[2.32, 7.91]	[0.92, 9.31]
Twitter-Friend	Count LR mean vs. highest ten-split mean of the 18 methods (61.15)	10	68.55	[66.19, 70.91]	[65.01, 72.08]
Intervals marked † in other tables					
IMDB-Dir-Genre	Evaluation gap of HyperND	5	2.25	[0.09, 4.42]	[−0.51, 5.02]†
IMDB-Dir-Genre	Evaluation gap of CEGAT	5	2.19	[0.40, 3.98]	[−0.10, 4.47]†

Table 14: **Input duplicates in the benchmark's splits** (seeds 42–46; 18 methods; five splits): mean share of duplicated test graphs and gap $\bar{\imath}$ (95% t interval).

	Duplicated	Evaluation gap [95% CI]	
Collection	test graphs (%)	Accuracy	Balanced accuracy
IMDB-Dir-Form	28.5	10.61 [7.93, 13.29]	12.45 [8.31, 16.59]
IMDB-Dir-Genre	27.7	2.36 [1.40, 3.31]	12.25 [11.39, 13.11]

(Appendix A). A second identity, the canonical-form digest of the raw DHG-Bench input with degree features, gives identical groups. A test graph counts as duplicated when its input occurs in training or in validation. Some duplicated test graphs therefore have copies only in validation. The model was not trained on these copies, although validation was used for model selection.

Runs. All 18 methods ran with the upstream DHG-Bench code and its per-dataset configurations, with five seeds each, on the four DHG-Bench collections (Section 4). All 360 runs completed. The read-only hook writes the test predictions without consuming random numbers or changing any tensor, and the accuracies recomputed from them equal the upstream ones in all 360 runs.

Reproduction of the reported accuracies. Of the 72 cells of the reported table (18 methods, four collections), 57 of our five-seed means lie within one reported standard deviation of the reported mean and 70 within two (Table 17). For a perfect reproduction, a Monte-Carlo simulation draws two independent sets of five runs from one normal distribution, so both the means and the standard

Table 15: Duplicate census of the benchmark's splits (split seeds 42 to 46). Label-conflicting duplicates: mean number per split of duplicated test graphs with a training or validation copy of a different label.

Collection	Graphs	Unique inputs	Test size	Duplicated test graphs, % mean [min, max]	Label-conflicting duplicates
IMDB-Dir-Form	1869	1369	188	28.5 [26.1, 33.0]	4.4
IMDB-Dir-Genre	3393	2591	340	27.7 [25.3, 29.1]	24.0
Steam-Player	2048	2048	206	0.0 [0.0, 0.0]	0.0
Twitter-Friend	1310	1310	131	0.0 [0.0, 0.0]	0.0

Table 16: Per-method accuracy on duplicated and non-duplicated test graphs (%, five-seed means) and evaluation gap with the 95% t interval. Every one of the 18 methods has a positive lower end on both collections. † marks the two IMDB-Dir-Genre intervals whose lower end is not above zero under the overlap correction (Appendix C.1).

Method	Dir-Form: dup. / non-dup. acc.	Dir-Form gap [95% CI]	Dir-Genre: dup. / non-dup. acc.	Dir-Genre gap [95% CI]
HCHA	90.39 / 54.39	10.29 [7.25, 13.33]	85.60 / 74.53	3.05 [1.96, 4.15]
PhenomNN	90.40 / 52.92	10.70 [7.44, 13.95]	81.74 / 72.74	2.49 [1.08, 3.91]
DPHGNN	92.25 / 52.16	11.46 [8.13, 14.79]	81.08 / 74.84	1.75 [0.98, 2.53]
TF-HNN	89.58 / 52.70	10.49 [7.33, 13.66]	82.43 / 74.76	2.12 [1.07, 3.16]
CEGCN	87.58 / 53.37	9.71 [7.36, 12.07]	82.99 / 72.78	2.87 [1.52, 4.21]
SheafHyperGNN	86.82 / 53.18	9.69 [6.15, 13.23]	82.19 / 74.68	2.08 [0.75, 3.42]
AllSetTransformer	88.43 / 52.29	10.37 [6.89, 13.85]	81.60 / 74.54	1.93 [0.87, 2.98]
HyperND	88.40 / 52.39	10.27 [8.53, 12.02]	81.49 / 73.39	2.25 [0.09, 4.42]†
HNHN	86.99 / 52.78	9.78 [6.36, 13.19]	80.46 / 72.73	2.15 [0.67, 3.63]
MLP	87.61 / 52.43	10.12 [6.16, 14.08]	81.35 / 74.03	2.03 [0.67, 3.39]
HyperGCN	88.84 / 51.90	10.55 [7.05, 14.05]	84.38 / 74.51	2.73 [1.80, 3.66]
ED-HNN	88.07 / 51.80	10.33 [7.16, 13.50]	82.16 / 75.42	1.87 [1.04, 2.71]
HGNN	92.22 / 49.65	12.16 [9.05, 15.26]	84.44 / 74.96	2.57 [1.35, 3.79]
HJRL	89.68 / 50.70	11.11 [7.62, 14.60]	83.34 / 75.34	2.19 [1.07, 3.31]
HyperGT	87.10 / 51.43	10.17 [8.41, 11.92]	82.62 / 72.08	2.92 [1.55, 4.29]
LEGCN	89.38 / 49.71	11.25 [8.18, 14.32]	84.55 / 74.03	2.91 [1.95, 3.88]
CEGAT	88.50 / 49.66	11.08 [7.39, 14.78]	79.32 / 71.58	2.19 [0.40, 3.98]†
UniGCNII	88.72 / 48.21	11.47 [9.97, 12.97]	82.61 / 74.33	2.31 [0.50, 4.13]

deviations are simulated. A cell is within k standard deviations ($k = 1, 2$) when the two means differ by at most k times the ddof $= 0$ standard deviation of the set in the role of the reported values. The simulation gives an expected 55.4 and 68.6 cells. The observed agreement is therefore consistent with sampling variation, so the audit results above, which score these runs, refer to the benchmark's splits and published configurations. Under the combined rule $|\Delta| > 2\sqrt{s^2_{\text{rep}} + s^2_{\text{ours}}}$, with our standard deviation also computed with ddof $= 0$, one cell exceeds sampling variation: AllSetTransformer on Twitter-Friend reaches 53.74 ± 0.92 in our runs against the reported 60.15 ± 1.70. Where Section 5.1 compares with reported accuracies, it uses the reported value, 60.15%.

E Dataset statistics

Table 18 summarizes the four collections. Steam-Player and Twitter-Friend have a unique input for every hypergraph.

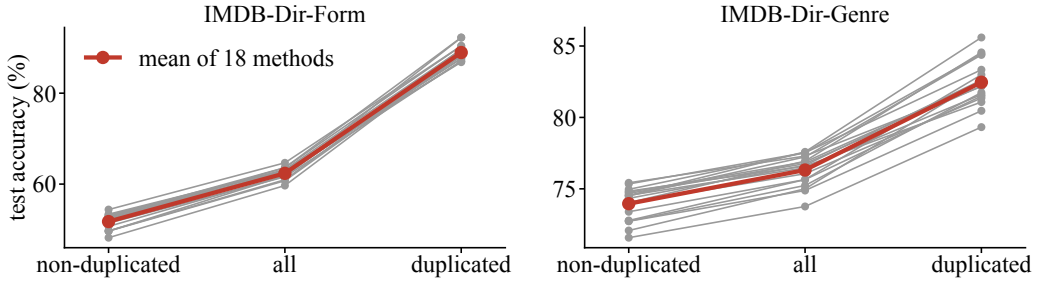

Figure 3: **Accuracy of the 18 benchmark methods on non-duplicated, all, and duplicated test graphs** (gray: one method each).

Table 17: Reproduction under the DHG-Bench protocol: our five-seed mean ± SD (ddof = 0) / reported mean ± SD (test accuracy, %). *: difference larger than one reported standard deviation (15 of 72 cells).

Method	IMDB-Dir-Form	IMDB-Dir-Genre	Steam-Player	Twitter-Friend
AllSetTransformer	62.66±1.70 / 62.23±1.01	76.47±1.15 / 76.47±1.38	53.11±1.81 / 53.01±2.77	53.74±0.92 / 60.15±1.70*
CEGAT	60.74±1.44 / 63.51±1.54*	73.76±2.78 / 74.12±2.69	49.03±2.86 / 49.51±4.71	58.63±3.49 / 57.32±2.59
CEGCN	63.09±1.37 / 62.66±1.82	75.65±2.07 / 75.06±0.76	50.68±4.19 / 48.16±3.87	58.17±2.07 / 54.66±5.66
DPHGNN	63.62±2.66 / 64.04±2.70	76.59±0.73 / 76.18±1.30	50.39±2.26 / 51.36±1.72	56.18±1.84 / 59.24±2.88*
ED-HNN	62.13±1.73 / 62.13±2.36	77.29±0.43 / 77.12±1.11	54.76±3.55 / 52.82±2.65	58.93±3.43 / 57.40±2.66
HCHA	64.68±1.29 / 61.60±2.16*	77.59±2.02 / 78.12±1.96	52.52±2.70 / 52.43±2.30	57.56±3.33 / 58.17±2.34
HGNN	61.81±1.40 / 63.72±0.62*	77.53±0.84 / 76.76±2.66	51.65±2.60 / 51.65±2.51	58.17±3.56 / 55.42±2.03*
HJRL	61.81±1.28 / 63.09±2.83	77.53±1.95 / 77.82±1.47	51.46±2.06 / 51.84±3.52	56.79±2.44 / 57.10±2.79
HNHN	62.55±2.94 / 62.34±2.98	74.88±1.35 / 73.65±1.47	53.69±3.79 / 52.82±1.61	58.78±3.95 / 58.47±4.65
HyperGCN	62.45±0.80 / 62.87±0.40*	77.24±1.27 / 77.53±0.99	50.78±3.14 / 51.17±3.32	57.71±3.76 / 56.95±4.17
HyperGT	61.60±1.70 / 61.49±4.32	75.00±1.25 / 73.82±1.27	52.91±1.37 / 54.47±1.33*	56.49±2.98 / 54.35±2.72
HyperND	62.66±3.40 / 60.74±3.25	75.65±1.66 / 75.65±0.51	51.55±1.52 / 53.88±2.15*	56.18±3.60 / 55.27±3.79
LEGCN	60.96±1.60 / 61.81±1.32	76.94±1.14 / 76.38±1.68	50.68±2.54 / 53.11±1.58*	54.05±4.41 / 56.64±3.72
MLP	62.55±1.60 / 63.62±1.69	76.06±1.64 / 75.12±0.70*	53.30±2.00 / 52.34±0.55*	57.10±5.82 / 57.25±1.81
PhenomNN	63.62±2.96 / 61.28±1.97*	75.24±1.62 / 74.59±0.61*	51.84±3.03 / 51.65±3.06	56.49±2.55 / 57.40±3.84
SheafHyperGNN	62.87±2.19 / 62.34±2.06	76.76±0.53 / 77.00±1.14	51.75±2.25 / 53.11±2.39	56.79±1.91 / 56.49±2.51
TF-HNN	63.19±2.19 / 62.34±1.76	76.88±1.23 / 76.41±1.31	52.62±1.46 / 54.85±1.82*	53.59±4.19 / 56.18±3.53
UniGCNII	59.68±1.48 / 61.06±2.88	76.65±1.32 / 77.12±0.88	51.94±2.21 / 51.46±2.48	57.56±2.49 / 55.88±4.14

Table 18: **Statistics of the four collections.** $|V|$ and $|E|$ are the per-hypergraph node and hyperedge counts of the raw files, summarized over hypergraphs; the hyperedge size is pooled over all hyperedges of a collection, not averaged per hypergraph. Star nodes and edges describe the typed star graph $S(H)$. Class sizes list the number of hypergraphs per label in label order, the majority share is that of the largest class, and the unique-input ratio is the number of distinct inputs divided by the number of hypergraphs.

	Steam-Player	Twitter-Friend	IMDB-Dir-Form	IMDB-Dir-Genre
Hypergraphs	2048	1310	1869	3393
Classes	2	2	3	3
Class sizes (label order)	1024/1024	776/534	798/688/383	1107/547/1739
Majority class (%)	50.0	59.2	42.7	51.3
Unique-input ratio	1.000	1.000	0.732	0.764
$\|V\|$ (mean / med. / max)	13.83 / 13 / 76	21.65 / 16 / 166	15.72 / 12 / 264	17.33 / 12 / 264
$\|E\|$ (mean / med. / max)	46.36 / 44 / 140	84.77 / 57 / 603	39.30 / 13 / 450	36.55 / 13 / 450
Hyperedge size (mean / max)	4.54 / 71	4.85 / 64	3.37 / 110	3.48 / 110
Star nodes (mean / med. / max)	60.19 / 56 / 204	106.42 / 75 / 765	55.02 / 27 / 537	53.88 / 28 / 551
Star edges (mean / med. / max)	210.47 / 188 / 1460	410.97 / 251 / 4589	132.31 / 49 / 2021	127.31 / 51 / 2021