THE WIDTH WALL: AN EXPRESSIVITY FRAMEWORK FOR HYPERGRAPH NEURAL NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Hypergraphs provide a natural framework for modeling higher-order interactions in scientific, social, and biological systems. Hypergraph neural networks (HGNNs) aim to learn from such data, yet it remains unclear which higher-order structural distinctions their representations preserve. We study this question through *homomorphism densities*, which measure how often finite patterns map into a hypergraph. Combining homomorphism-count completeness with invariant approximation, we show that finite pattern densities generate a dense algebra of continuous hypergraph invariants and organize them into a filtration indexed by generalized hypertree width. This yields the notion of a *Width Wall*: when an architecture is provably restricted to width-r information, no increase in hidden dimension, readout complexity, or optimization can recover distinctions erased beyond that representation. Our framework precisely characterizes information lost by clique expansion, with a Steiner–Pasch construction giving an explicit width-2 separator. We further study exact pattern-alignment invariants and introduce DENSNET-D, which augments an incidence-based backbone with Monte Carlo estimates of local pattern densities. Experiments on four real-world node-classification benchmarks show that these structural features can recover information lost by projection and improve a matched AllDeepSets backbone.

1 INTRODUCTION

Many real-world systems—such as co-authorship networks, protein complexes, and legislative coalitions—are organized by simultaneous interactions among multiple entities. Hypergraphs provide a natural language for modeling these higher-order relations: a single hyperedge can bind an entire group, rather than forcing interaction into pairwise links (Battiston et al., 2020; Benson et al., 2016). This modeling advantage has driven a rapid expansion of hypergraph neural networks (HGNNs) (Chien et al., 2022; Feng et al., 2019; Wang et al., 2023; Yadati et al., 2019).

However, preserving a hypergraph does not guarantee that an HGNN can represent all of its higher-order structure. Some methods, such as HGNN and HyperGCN, construct graph-based propagation operators from hypergraph data (Feng et al., 2019; Yadati et al., 2019); others, including HNHN, UniGNN, AllSet, and ED-HNN, retain the incidence structure and aggregate directly over nodes and hyperedges (Chien et al., 2022; Dong et al., 2020; Huang & Yang, 2021; Wang et al., 2023). Yet even incidence-based architectures may identify structurally distinct hypergraphs. This raises the central question: *when does an HGNN lose information that is relevant to a hypergraph invariant?*

We study this question through *information loss*. If a hypergraph is first replaced by its clique expansion, then any two hypergraphs with the same projected graph are indistinguishable to every downstream architecture that factors through that projection. We construct two non-isomorphic Steiner triple systems with identical clique expansions but different counts of a Pasch configuration, giving a concrete example in which pairwise projection erases recoverable higher-order information.

To formalize such distinctions, we use *homomorphism densities*, which measure how frequently finite patterns map into the input hypergraph. Homomorphism counts are classical structural invariants (Lovász, 1967); on the fixed-size compact tensor domain considered here, we show that finite pattern densities generate a dense algebra of continuous permutation-invariant hypergraph functions. This connects naturally with broader universality results for invariant architectures (Maron et al.,

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.

2019a; Keriven & Peyré, 2019). Thus, finite patterns provide a principled basis for describing which structural information a representation retains or discards.

We organize these invariants by generalized hypertree width. Let $\mathcal{H}_r$ denote the closed algebra generated by pattern densities with generalized hypertree width at most r. Motivated by connections between bounded-width homomorphism counts and guarded counting logics (Gottlob et al., 2002; Scheidt & Schweikardt, 2023), we obtain the filtration $\mathcal{H}_1 \subseteq \mathcal{H}_2 \subseteq \cdots \subseteq \mathcal{I}_{\mathrm{all}}, \qquad \overline{\bigcup_{r\geq 1} \mathcal{H}_r} = \mathcal{I}_{\mathrm{all}}$. An architecture faces a *Width Wall* at level r only when an independent factorization or indistinguishability argument shows that its outputs lie within $\mathcal{H}_r$. The Width Wall is therefore an information-theoretic limitation of a certified representation, not a consequence of message-passing depth alone. This viewpoint complements recent generalized Weisfeiler–Leman analyses of hypergraph expressivity (Zhang et al., 2025). Experiments on four real-world node-classification benchmarks show that these structural features can recover information lost by projection and improve a matched AllDeepSets backbone.

2 Homomorphism Features and the Expressivity Framework

Notation and Background. We write $[n] = \{1, \ldots, n\}$. A k-uniform hypergraph is $\mathcal{H} = ([n], \mathcal{E})$, where $\mathcal{E} \subseteq \binom{[n]}{k}$ is a set of unordered k-node hyperedges. Its adjacency tensor is the symmetric order-k array $\boldsymbol{A}$ with $\boldsymbol{A}_{i_1,\ldots,i_k} = 1$ if $\{i_1, \ldots, i_k\} \in \mathcal{E}$, and 0 otherwise. Entries with repeated indices are zero. Let $\mathbf{S}^{k,n}$ denote the space of order-k, dimension-n symmetric tensors. Thus, the adjacency tensor of a k-uniform hypergraph lies in the subspace of $\mathbf{S}^{k,n}$ with zero repeated-index entries. The case $k = 2$ recovers ordinary loopless graphs.

For the weighted setting, it is convenient to index the independent hyperedge coordinates directly. We therefore write $\mathcal{X}_{k,n} := [0,1]^{\binom{[n]}{k}}$, which may equivalently be embedded in $\mathbf{S}^{k,n}$ as the symmetric tensors with entries in $[0,1]$ and zero repeated-index entries. For non-uniform hypergraphs with maximum edge size K, we use $\mathcal{X}_{\leq K,n} := \prod_{j=2}^{K} [0,1]^{\binom{[n]}{j}}$, with $\boldsymbol{A} = (A^{(2)}, \ldots, A^{(K)})$, where $A^{(j)}$ records hyperedges of size j. Finally, $\mathrm{Sym}(n)$ denotes the group of vertex permutations, acting by simultaneously relabeling all hyperedge coordinates. A hypergraph invariant is a function unchanged under this action.

2.1 Tensor Domain and Hypergraph Invariants

Overview. In graph theory, homomorphism counts provide a canonical coordinate system for finite graphs. They characterize graphs up to isomorphism, underlie dense graph limits, and connect to logical and WL-style expressivity (Dell et al., 2018; Lovász, 2012; Lovász & Szegedy, 2006). We extend this viewpoint to hypergraphs and use *hypergraph homomorphism densities* as invariant structural coordinates. These densities measure how often a finite structural pattern appears in the input hypergraph (Elek & Szegedy, 2012; Lovász, 1967; 2012; Zhao, 2015). Each pattern defines one label-independent coordinate, while collections of such coordinates provide increasingly rich descriptions of higher-order structure. Further algebraic structure appears in Appendix A.1; concrete examples of the resulting hypergraph invariants are in Appendix H.

Our discussion proceeds in three steps. First, we represent hypergraphs on a compact weighted domain with a natural vertex-relabeling action. Second, we introduce pattern densities and their exact aligned-template counterparts. Finally, we show that finite collections of pattern densities can approximate arbitrary continuous invariant functions for fixed-n. Under this view, the full pattern-density family provides an information-theoretic reference class. Sec. 3 then asks which portions of this invariant information can be accessed by restricted representations and neural architectures.

Tensor Domain and Invariance. To analyze structural information independently of arbitrary vertex labels, we let a permutation $\Pi \in \mathrm{Sym}(n)$ act on $\boldsymbol{A} \in \mathcal{X}_{k,n}$ by

$$(\Pi \cdot \boldsymbol{A})_{\{i_1,\ldots,i_k\}} = \boldsymbol{A}_{\{\Pi^{-1}(i_1),\ldots,\Pi^{-1}(i_k)\}}.$$

Equivalently, under the symmetric-tensor embedding this action simultaneously relabels every tensor index. The compact cube $\mathcal{X}_{k,n}$ contains ordinary unweighted hypergraphs as its $\{0,1\}$-valued points and weighted hypergraphs as its interior points. Accordingly, a continuous hypergraph invariant is a

continuous function on this domain that is constant on $\mathrm{Sym}(n)$-orbits. For non-uniform hypergraphs, the same construction applies to $\mathcal{X}_{\leq K,n}$ under the diagonal action of $\mathrm{Sym}(n)$ on all arity components.

Definition 2.1 (Hypergraph invariant)**.** A function $f : \mathcal{X}_{k,n} \to \mathbb{R}$ is a *hypergraph invariant* if $f(\Pi \cdot \boldsymbol{A}) = f(\boldsymbol{A})$ for all $\boldsymbol{A} \in \mathcal{X}_{k,n}$ and all $\Pi \in \mathrm{Sym}(n)$.

2.2 Homomorphism Densities and Pattern Alignment

We use two complementary families of invariants. The first is the family of *homomorphism densities*, which average a finite structural pattern over all vertex maps into the target and provide the count-based algebra underlying our completeness result. The second is the family of *pattern alignment scores*, which compare a target hypergraph to a template under optimal vertex relabeling and provide optimization-based template invariants.

Let $\mathcal{F}_k$ denote the set of finite k-uniform hypergraph patterns. When working on the weighted domain, we allow patterns in $\mathcal{F}_k$ to contain finite hyperedge multiplicities; this distinction is immaterial for unweighted hypergraphs, since $\boldsymbol{A}_e^m = \boldsymbol{A}_e$ for $\boldsymbol{A}_e \in \{0, 1\}$.

Let F be a finite k-uniform pattern and H a target k-uniform hypergraph. A map $\phi : V(F) \to V(H)$ places vertices of the pattern into the target, where $V(\cdot)$ denotes the vertex set.

Definition 2.2 (Hypergraph homomorphism (Lovász, 1967; 2012))**.** Let $F = ([p], E_F)$ and $H = ([n], E_H)$ be k-uniform hypergraphs. A *homomorphism* $\phi : V(F) \to V(H)$ is a map such that $\{\phi(i_1), \ldots, \phi(i_k)\} \in E_H$ whenever $\{i_1, \ldots, i_k\} \in E_F$. The *homomorphism count* is

$$\hom(F, H) = \big|\{\phi : \phi \text{ is a homomorphism from } F \text{ to } H\}\big|.$$

The *homomorphism density* is the fraction of all maps from $[p]$ to $[n]$ that are homomorphisms:

$$t(F, H) = \frac{\hom(F, H)}{n^p}.$$

For weighted target hypergraphs with adjacency representation $\boldsymbol{A} \in \mathcal{X}_{k,n}$ and a pattern $F = ([p], E_F)$, the same construction gives the polynomial homomorphism density (Elek & Szegedy, 2012; Lovász & Szegedy, 2006; Zhao, 2015)

$$t(F, \boldsymbol{A}) = \frac{1}{n^p} \sum_{\phi:[p]\to[n]} \prod_{e \in E_F} \boldsymbol{A}_{\phi(e)}, \tag{1}$$

where $\boldsymbol{A}_{\phi(e)} = 0$ whenever ϕ identifies two vertices belonging to the same hyperedge. For a simple pattern, $t(F, \boldsymbol{A})$ is a polynomial of degree $|E_F|$ in the hyperedge coordinates. For a multi-pattern, repeated hyperedges allow repeated powers of the same coordinate. In either case, $t(F, \cdot)$ is continuous and invariant under vertex relabeling.

These features encode the higher-order interaction specified by the query pattern. A single edge measures edge density, a star measures local overlap, and more entangled patterns measure increasingly joint higher-order structure. Thus, the family $\{t(F, \cdot) : F \in \mathcal{F}_k\}$ should be viewed not merely as a collection of statistics, but as a family of invariant structural coordinates.

Homomorphism densities form an algebra under pointwise addition and multiplication. In particular, multiplying two densities is equivalent to counting the disjoint union of their query patterns (Lovász, 2012, Ch. 5):$t(F_1, \boldsymbol{A})\, t(F_2, \boldsymbol{A}) = t(F_1 \sqcup F_2, \boldsymbol{A})$. We write $\mathfrak{A}_{k,n}$ for the unital algebra generated by these density functions, with constants supplied by the empty pattern $t(\varnothing, \boldsymbol{A}) = 1$. Further details on this algebraic structure appear in Appendix A.1.

For the architectural construction introduced later in Section 4, we also introduce an exact template-based analogue of pattern counting. Rather than averaging over all vertex maps, this feature compares a template with the target under the best vertex relabeling.

Definition 2.3 (Exact pattern alignment score)**.** For any template P in the set of real symmetric order-k tensors with zero repeated-index entries,

$$\Theta_P(\boldsymbol{A}) = \max_{\Pi \in \mathrm{Sym}(n)} \langle P, \Pi \cdot \boldsymbol{A} \rangle. \tag{2}$$

The score Θ_P is invariant under vertex relabeling of $\boldsymbol{A}$ by construction. Homomorphism densities provide the count-based algebra used in the completeness result, while exact alignment scores provide optimization-based template invariants.

2.3 Completeness and Density

The preceding constructions provide concrete invariant coordinates. We now ask whether the count-based coordinates contain enough information to approximate an arbitrary continuous invariant.

Finite densities model structural information. We prove that finite collections of pattern densities can approximate arbitrary continuous invariant targets on the fixed-n weighted domain.

Theorem 2.4 (Pattern-density universality with fixed n)**.** *Fix $n \geq 1$ and $k \geq 2$. Let $\mathcal{K} = \mathcal{X}_{k,n} = [0,1]^{\binom{[n]}{k}}$, and let $C(\mathcal{K}/\mathrm{Sym}(n))$ denote the Banach space of continuous $\mathrm{Sym}(n)$-invariant functions on $\mathcal{K}$ with the supremum norm. Then the algebra generated by finite k-uniform pattern densities is uniformly dense in $C(\mathcal{K}/\mathrm{Sym}(n))$. Equivalently, for every continuous hypergraph invariant f and every $\epsilon > 0$, there exist patterns $F_1, \ldots, F_m \in \mathcal{F}_k$ and a polynomial $q : \mathbb{R}^m \to \mathbb{R}$ such that*

$$\sup_{\boldsymbol{A} \in \mathcal{K}} \left| f(\boldsymbol{A}) - q\big(t(F_1, \boldsymbol{A}), \ldots, t(F_m, \boldsymbol{A})\big) \right| < \epsilon.$$

The proof is given in Appendix A.2. For ordinary unweighted hypergraphs, the underlying separation principle is the classical finite-structure homomorphism theorem (Lovász, 1967; 2012). For the weighted domain used here, the proof proceeds by expressing invariant polynomial orbit sums as finite linear combinations of weighted homomorphism counts and then applying the Stone–Weierstrass theorem on the compact orbit space.

Theorem 2.4 shows that any fixed-size structural property that is continuous and invariant to vertex relabeling can be approximated by a polynomial in finitely many pattern densities. In this sense, pattern densities provide a complete coordinate system for continuous invariants on the fixed weighted domain: once the full family of pattern coordinates is available, no additional continuous label-independent information is required.

Importantly, Theorem 2.4 is an information-theoretic completeness result rather than an architectural realization result. It does not imply that a particular HGNN can compute all pattern densities. Section 3 instead asks which portions of this complete invariant information remain accessible under restricted representations and neural architectures.

For non-uniform hypergraphs, the corresponding result requires *mixed-arity patterns* whose relations are evaluated under a common vertex map. This is the natural relational-structure extension of homomorphism counting (Lovász, 1967). The resulting mixed-arity densities retain cross-order coupling that cannot be recovered from independently invariant per-order summaries. The corresponding fixed-n density result is proved in Appendix B.2. Extensions to dense hypergraph-limit settings are discussed in (Elek & Szegedy, 2012; Zhao, 2015).

3 Pattern Width and the Width Wall

We have shown that pattern densities provide a complete coordinate system for continuous invariants on the fixed-n domain. We now organize these coordinates by their structural complexity and ask which portions of this invariant information remain accessible under restricted representations and neural architectures. The key distinction is between *reference information classes*, defined by structural coordinates they preserve, and function classes realized by particular HGNN architectures.

3.1 Architecture Scope

Before comparing architectures, we fix the scope of our claims. Throughout, we consider structure-based hypergraph neural architectures satisfying the following conditions whenever the corresponding architectural bound is invoked.

Assumption 3.1 (Architecture scope)**.** The architecture satisfies: **(C1)** permutation equivariance, with no node identifiers or order-dependent operations; **(C2)** locality, when a local message-passing bound is invoked; **(C3)** symmetric aggregation, such as sum, mean, max, DeepSets, or attention over a multiset; **(C4)** no symmetry-breaking features, such as random IDs or externally supplied positional encodings; and **(C5)** structure-dependent preprocessing is treated as part of the architecture, so a projection or refinement bound is invoked only when the full preprocessing-and-network pipeline satisfies that bound.

For an HGNN architecture class $\mathcal{F}$, $\mathcal{I}(\mathcal{F})$ is the set of continuous hypergraph invariants that functions in $\mathcal{F}$ can approximate. Thus $\mathcal{I}(\mathcal{F})$ records the invariant information representable by the architecture.

Projection-based information loss. Projection-based methods first map a hypergraph to a reduced structural representation before learning. Let $\phi : \mathcal{X}_{k,n} \to \mathcal{X}_{2,n}$ denote a permutation-equivariant pairwise projection, such as a standard clique expansion. We define the corresponding reference information class $\mathcal{I}_{\mathrm{CE}} := \{g \circ \phi : g \in C(\phi(\mathcal{X}_{k,n})/\mathrm{Sym}(n))\}$. Thus $\mathcal{I}_{\mathrm{CE}}$ contains all continuous invariant information that can, in principle, be recovered from the projected representation. It should not be identified with the function class of a particular downstream graph neural network.

Methods whose full pipeline factors through such a pairwise projection, including standard clique-expansion formulations such as HGNN (Feng et al., 2019), inherit the following limitation.

Proposition 3.2 (Clique-expansion limitation)**.** *Suppose an architecture class $\mathcal{F}_{\mathrm{CE}}$ has the form $f_\theta(\boldsymbol{A}) = g_\theta(\phi(\boldsymbol{A}))$ for every parameter choice θ. Then $\mathcal{I}(\mathcal{F}_{\mathrm{CE}}) \subseteq \mathcal{I}_{\mathrm{CE}}$. In particular, if two hypergraphs have the same clique expansion, $\phi(\boldsymbol{A}_1) = \phi(\boldsymbol{A}_2)$, then $f_\theta(\boldsymbol{A}_1) = f_\theta(\boldsymbol{A}_2)$ for every parameter choice θ.*

Prop. 3.2 isolates information lost before a downstream learner is used. Native hyperedge architectures such as HNHN (Dong et al., 2020), UniGNN (Huang & Yang, 2021), AllSet (Chien et al., 2022), and ED-HNN (Wang et al., 2023) instead retain incidence structure. Their precise location in the pattern-width hierarchy, however, requires an architecture-specific refinement or indistinguishability argument; incidence preservation alone does not assign a generalized hypertree-width level.

3.2 Width, Invariant Classes, and the Width Wall

We next investigate how hyperedge structure must be coordinated jointly in order to count a pattern. We measure this coordination complexity using *generalized hypertree width* (Gottlob et al., 2002).

Definition 3.3 (Generalized hypertree width (Gottlob et al., 2002))**.** A *generalized hypertree decomposition* of a hypergraph F consists of a tree T whose nodes t are assigned bags $B_t \subseteq V(F)$ together with sets $\lambda_t \subseteq E(F)$ such that: (i) every hyperedge of F is contained in some bag; (ii) for every $v \in V(F)$, the bags containing v form a connected subtree; and (iii) $B_t \subseteq \bigcup_{e \in \lambda_t} e$ for every t. The *generalized hypertree width* is $W_{\mathrm{gh}}(F) = \min_{(T,B,\lambda)} \max_{t \in V(T)} |\lambda_t|$.

Intuitively, $W_{\mathrm{gh}}(F)$ measures how many hyperedges must be coordinated to cover the bags needed to organize the pattern. Acyclic patterns have $W_{\mathrm{gh}} = 1$, while more strongly coupled patterns may require larger width. For the weighted-pattern convention of Section 2.2, the width of a pattern with hyperedge multiplicities is the width of its underlying simple support.

Generalized hypertree width therefore induces a filtration of the pattern-density coordinates. For $r \geq 1$, define $\boldsymbol{A} \equiv_r \boldsymbol{B} \iff t(F, \boldsymbol{A}) = t(F, \boldsymbol{B})$ for every F with $W_{\mathrm{gh}}(F) \leq r$. Thus $\equiv_r$ records indistinguishability by all width-r pattern coordinates.

Definition 3.4 (Invariant classes via pattern complexity)**.** For $r \geq 1$, we define $\mathcal{H}_r := \overline{\mathrm{alg}\,\{t(F,\cdot) : W_{\mathrm{gh}}(F) \leq r\}}^{\|\cdot\|_\infty}$ to be the class generated by pattern densities of generalized hypertree width at most r. Further, $\mathcal{I}_{\mathrm{all}} := C(\mathcal{X}_{k,n}/\mathrm{Sym}(n))$ is defined to be the full continuous invariant class from Theorem 2.4.

Equivalently, $\mathcal{H}_r$ consists of the continuous invariants that are constant on the equivalence classes of $\equiv_r$; this characterization is proved in Appendix A.4. The classes $\mathcal{H}_r$ are reference information classes. They do not, by definition, coincide with the function class of any particular HGNN.

Definition 3.5 (Width Wall)**.** Let $\mathcal{F}$ be an architecture class. If $\mathcal{I}(\mathcal{F}) \subseteq \mathcal{H}_r$, we say that $\mathcal{F}$ faces a *Width Wall at level r*. The Width Wall is the information boundary induced by width-r indistinguishability: an invariant $f \in \mathcal{I}_{\mathrm{all}} \setminus \mathcal{H}_r$ lies beyond the wall and cannot be represented by an architecture confined to $\mathcal{H}_r$.

Thus, the Width Wall is a structural rather than parametric barrier. Once an architecture has been shown to be constant on width-r indistinguishability classes, increasing hidden dimension, changing the invariant readout, or further optimizing parameters cannot recover an invariant that separates two hypergraphs within the same $\equiv_r$-class. Importantly, the wall is defined by indistinguishability rather

than by the syntactic width of a single pattern: a particular density lies beyond the level-r wall only when it is not determined by the width-r coordinates.

3.3 Pattern-Width Filtration and Architectural Consequences

The classes in Definition 3.4 form a nested filtration of the complete invariant space.

Theorem 3.6 (Pattern-width filtration and projection separation)**.** *For every $r \geq 1$, $\mathcal{H}_r \subseteq \mathcal{H}_{r+1} \subseteq \mathcal{I}_{\mathrm{all}}$, and $\overline{\bigcup_{r\geq 1} \mathcal{H}_r}^{\|\cdot\|_\infty} = \mathcal{I}_{\mathrm{all}}$. Moreover, standard clique expansion loses information already visible at width 2: there exist two non-isomorphic 3-uniform hypergraphs H_1, H_2 on 13 vertices with identical clique expansions and a pattern F_{P} satisfying $W_{\mathrm{gh}}(F_{\mathrm{P}}) = 2, t(F_{\mathrm{P}}, H_1) \neq t(F_{\mathrm{P}}, H_2)$. Consequently, $t(F_{\mathrm{P}}, \cdot) \in \mathcal{H}_2 \setminus \mathcal{I}_{\mathrm{CE}}$.*

The proof is given in Appendix A.4, with the explicit Steiner construction in Appendix E. The first part of Theorem 3.6 organizes the complete pattern-density algebra by structural complexity. The second makes the projection loss concrete: pairwise reduction can discard a width-2 invariant even when all pairwise co-occurrences are preserved.

We do not require adjacent strictness $\mathcal{H}_r \subsetneq \mathcal{H}_{r+1}$ at every level. Indeed, the present fixed-n formulation eventually saturates as the available structural information becomes complete. The role of the filtration is instead to identify which distinctions are already determined at a given width and which remain unresolved. A Width Wall becomes an architectural statement once a model is proved to factor through one of these bounded-width indistinguishability relations.

The connection between generalized hypertree width and logical indistinguishability is known exactly. (Scheidt & Schweikardt, 2023) introduce a two-sorted guarded counting logic with a bounded number of variables for hyperedges and an unbounded number of guarded vertex variables. For finite unweighted hypergraphs H_1, H_2, they show that $\hom(F, H_1) = \hom(F, H_2)$; for every F with $W_{\mathrm{gh}}(F) \leq r$ if and only if H_1 and H_2 satisfy the same sentences of the corresponding width-r guarded counting logic (Scheidt & Schweikardt, 2023). This is the appropriate bounded-width logical characterization for hypergraphs; it should not be identified with ordinary $(r+1)$-variable counting logic.

We use this correspondence as a reference characterization of width-bounded structural information. The remaining step for a neural architecture is architecture-specific: one must first prove that its outputs are constant on the corresponding indistinguishability classes.

Proposition 3.7 (Architecture-to-width correspondence)**.** *Let $\mathcal{F}$ be an architecture class and fix $r \geq 1$. Suppose that $\boldsymbol{A} \equiv_r \boldsymbol{B} \implies f(\boldsymbol{A}) = f(\boldsymbol{B})$ for every invariant f realizable by $\mathcal{F}$. Then $\mathcal{I}(\mathcal{F}) \subseteq \mathcal{H}_r$, and $\mathcal{F}$ faces a Width Wall at level r in the sense of Definition 3.5.*

Proposition 3.7 separates message-passing *depth* from refinement *dimension*. Additional propagation rounds can enlarge the receptive field of a local architecture without establishing access to a higher generalized hypertree-width level. Conversely, an upper bound $\mathcal{I}(\mathcal{F}) \subseteq \mathcal{H}_r$ does not imply that $\mathcal{F}$ realizes every invariant in $\mathcal{H}_r$. Appendix C revisits modern HGNN architectures under this distinction, separating exact projection bounds from refinement-based upper bounds and from cases where an exact generalized-hypertree-width assignment has not been established.

Quantitative estimation. Theorem 2.4 is an approximation result and does not specify the cost of estimating individual pattern coordinates. For any fixed pattern F, however, its density $t(F, H)$ can be estimated by sampling random vertex maps. Appendix B shows that additive accuracy ϵ with failure probability δ requires $O(\log(1/\delta)/\epsilon^2)$ independent samples (Proposition B.1).

4 InvNet: An Invariant-Theoretic Architecture

We propose InvNet, an HGNN that augments a standard local hypergraph backbone with global pattern-alignment features. The model combines two complementary sources of structural information: local incidence-based messages and global template scores that compare the input hypergraph against learned higher-order patterns. InvNet serves primarily as a proof-of-concept architecture for injecting invariant structural coordinates beyond those available to a chosen local backbone.

The exact pattern-alignment score from Definition 2.3 is $\Theta_P(\boldsymbol{A}) = \max_{\Pi \in \mathrm{Sym}(n)} \langle P, \Pi \cdot \boldsymbol{A} \rangle$, where P belongs to the ambient real tensor space. This exact score is permutation invariant but requires combinatorial optimization over vertex relabelings. A natural continuous relaxation replaces permutation matrices by the Birkhoff polytope $\mathcal{B}_n$:

$$\Theta_P^{\mathcal{B}}(\boldsymbol{A}) := \max_{D \in \mathcal{B}_n} \langle P, D^{\otimes k} \cdot \boldsymbol{A} \rangle. \tag{3}$$

Since permutation matrices are contained in $\mathcal{B}_n$, $\Theta_P(\boldsymbol{A}) \le \Theta_P^{\mathcal{B}}(\boldsymbol{A})$. The relaxed optimum remains a continuous permutation invariant, but it need not coincide with the exact alignment score.

The finite implementation used by INVNET is distinct from the global optimization problem in Eqn. (3). For each template P_j, a learned score map $S_{\psi_j}(\boldsymbol{A}) \in \mathbb{R}^{n \times n}$ produces an alignment matrix $D_{j,T}(\boldsymbol{A}) = \mathrm{Sinkhorn}_T\left(S_{\psi_j}(\boldsymbol{A})/\tau\right)$, where T is the number of Sinkhorn iterations and $\tau > 0$ is the temperature. The resulting finite-Sinkhorn feature is

$$\widetilde{\Theta}_{P_j,\psi_j,T}(\boldsymbol{A}) := \langle P_j, D_{j,T}(\boldsymbol{A})^{\otimes k} \cdot \boldsymbol{A} \rangle. \tag{4}$$

Thus, the implemented layer should be viewed as a learned continuous alignment map rather than as an exact solver for Eqn. (3).

Concretely, INVNET computes

$$f_\theta(\boldsymbol{A}) = \mathrm{MLP}_\theta\Big(\underbrace{\mathrm{ReadOut}\big(\boldsymbol{h}^{(L)}\big)}_{\text{local branch}} \;\Big\|\; \underbrace{\big[\widetilde{\Theta}_{P_1,\psi_1,T}(\boldsymbol{A}) - \alpha_1, \ldots, \widetilde{\Theta}_{P_J,\psi_J,T}(\boldsymbol{A}) - \alpha_J\big]}_{\text{alignment branch}}\Big), \tag{5}$$

where $\boldsymbol{h}^{(L)}$ denotes the node representation produced by the local hypergraph backbone. Appendix F gives the full implementation.

The distinction between the exact, globally relaxed, and finite implementations is important for the invariance guarantee. For the finite Sinkhorn layer, suppose the score map is continuous and satisfies the right-equivariance condition

$$S_\psi(\Pi \cdot \boldsymbol{A}) = S_\psi(\boldsymbol{A})\Pi^\top \qquad \text{for every } \Pi \in \mathrm{Sym}(n). \tag{6}$$

Row and column normalization commute with permutations, so Eqn. (6) implies $D_T(\Pi \cdot \boldsymbol{A}) = D_T(\boldsymbol{A})\Pi^\top$ for every finite number of Sinkhorn iterations.

Theorem 4.1 (Alignment invariance and separation)**.** *The alignment constructions above satisfy:* ***(i)*** *For every template P, the exact score Θ_P is a continuous permutation invariant, and the family $\{\Theta_P\}_P$ separates distinct permutation orbits;* ***(ii)*** *The global Birkhoff relaxation $\Theta_P^{\mathcal{B}}$ is a continuous permutation invariant and satisfies $\Theta_P(\boldsymbol{A}) \le \Theta_P^{\mathcal{B}}(\boldsymbol{A})$;* ***(iii)*** *If S_ψ is continuous and satisfies Eqn.* (6)*, then the finite-Sinkhorn feature $\widetilde{\Theta}_{P,\psi,T}$ is a continuous permutation invariant for all $T < \infty, \tau > 0$.*

The proof is given in Appendix A.5. The first statement gives the information-theoretic motivation for the alignment branch: exact template alignments are sufficiently rich to distinguish non-isomorphic labeled-tensor orbits. The second and third statements separate this ideal invariant family from the two relaxations used to make alignment computationally tractable.

This viewpoint also clarifies the relation to the Width Wall. Suppose a local backbone has a certified width-r bound $\mathcal{I}(\mathcal{F}_{\mathrm{local}}) \subseteq \mathcal{H}_r$. If an alignment feature distinguishes a pair $\boldsymbol{A} \equiv_r \boldsymbol{B}$ that the local backbone cannot distinguish, then augmenting the backbone with that feature strictly enlarges its representable invariant information. Thus the alignment branch provides a mechanism for crossing a *certified* Width Wall; the existence of such a wall is not inferred from message-passing depth alone.

The Steiner pair from Appendix E.1 provides a concrete example of the distinction between pairwise projection and global alignment. Its two hypergraphs have identical clique expansions but are non-isomorphic, so an exact alignment template separates them.

5 EXPERIMENTS

Our experiments examine the practical relevance of information restrictions developed in Sec. 3–4. The theoretical results identify structural distinctions that can be lost under particular representations, but they do not imply that benchmark accuracy should increase monotonically with generalized hypertree width or with any simple dataset statistic. Accordingly, we address two empirical

questions: (i) how do graph-reduction and incidence-based HGNNs compare on real hypergraph node-classification tasks; and (ii) when can explicitly estimated pattern-density features improve a matched incidence-based backbone?

5.1 Experimental Setup

Models. We compare three broad model families. The *graph-reduction* baselines contain HGNN (Feng et al., 2019) and HyperGCN (Yadati et al., 2019), which construct graph-based propagation operators from hypergraph data. The *native-incidence* baselines include AllDeepSets, HNHN (Dong et al., 2020), and UniGNN (Huang & Yang, 2021), which operate on the incidence structure through message passing or permutation-invariant set aggregation. For node classification we additionally report MLP, ED-HNN (Wang et al., 2023), and SheafHyperGNN (Duta et al., 2023) where applicable. Our density-aware model, DensNet-D, derived from the principles guiding InvNet, augments an AllDeepSets backbone with explicitly estimated local pattern features. For each node, DensNet-D estimates pattern densities over its star neighborhood by Monte Carlo sampling and combines the resulting features with the backbone representation. The full architecture is specified in Appendix G.

Task and datasets. The Application Node Classification Suite (ANCS) evaluates transductive node classification on four empirical hypergraph benchmarks: Cora, Citeseer, PubMed, and Cora-CA. Nodes use the benchmark-provided input features, hyperedges encode co-citation or co-authorship relations, and the model predicts node labels under the random-split protocol described in Appendix D. Dataset statistics and preprocessing details are given in Appendices D.1 and J.

For compact presentation, we group datasets by mean hyperedge size: $\bar{k} < 4$ and $4 \leq \bar{k} < 7$. These bins are descriptive summaries only; mean hyperedge size is not a proxy for generalized hypertree width and does not determine an architecture's Width Wall.

- **Cora** ($\bar{k} = 3.03$): a co-citation hypergraph with papers as nodes and hyperedges induced by common citing papers; node labels are paper topics and features are bag-of-words vectors (Chien et al., 2022; Yadati et al., 2019).
- **Citeseer** ($\bar{k} = 3.20$): a co-citation hypergraph constructed analogously to Cora, with document-topic labels and sparse text features (Chien et al., 2022; Yadati et al., 2019).
- **PubMed** ($\bar{k} = 4.35$): a larger co-citation hypergraph over biomedical papers, using the benchmark input features and labels (Chien et al., 2022; Yadati et al., 2019).
- **Cora-CA** ($\bar{k} = 4.28$): co-authorship hypergraph from (Chien et al., 2022).

Metrics. We report test classification accuracy in percent. For models run under our ANCS protocol, results are aggregated over 10 seeds (Appendix D). Results imported from prior work are marked separately in Table 1 and should be interpreted with their original training protocols.

Parameter setup. For the methods run under our ANCS protocol, we use Adam with learning rate 10^{-3}, weight decay 0, gradient clipping 1.0, label smoothing 0.1, cosine annealing, and 200 epochs. The standard baselines use hidden dimension 64 and dropout 0.2.

DensNet-D uses hidden dimension 128, dropout 0.5, and 200 Monte Carlo density samples. Its gate initialization is selected by the DatasetProfiler using the descriptive mean-edge-size bins: $b = -5.0$ for $\bar{k} < 4$ and $b = -2.0$ for $4 \leq \bar{k} < 7$. Capacity and sampling parameters are selected by validation grid search, while gate initialization and freezing schedule are determined from dataset statistics. Training, hyperparameter-selection, and compute-budget details are in Appendices K and L. Since DensNet-D and standard baselines use different model capacities and training hyperparameters, Table 1 is a broad benchmark comparison rather than a controlled ablation. The effect of the density branch is examined separately using a matched AllDeepSets recipe in Appendix D.4.

5.2 Experimental Results

Finding 1: performance depends on the structural representation and model choice. Table 1 shows substantial variation across model families and datasets. On each of the four benchmarks,

Table 1: Node classification accuracy (%). Best **bold**, second underlined. ‡Published results use architecture-specific training recipes; the remaining methods are run under the ANCS protocol. Standard baselines use hidden $= 64$; DENSNET-D uses hidden $= 128$.

	$\bar{k} < 4$		$4 \leq \bar{k} < 7$	
Method	**Cora**	**Citeseer**	**PubMed**	**Cora-CA**
MLP	74.1	72.3	87.1	74.6
Graph-reduction baselines				
HGNN	69.9	68.1	71.2	76.8
HyperGCN	71.0	68.1	78.3	74.3
Native-incidence / structured baselines				
HNHN	72.8	68.3	70.0	76.6
UniGNN	77.3	72.4	87.6	82.9
AllDeepSets	76.2	70.9	87.2	80.6
ED-HNN‡	80.3	73.7	**89.0**	84.0
SheafHyperGNN‡	**81.3**	**74.7**	87.7	**85.5**
Density-aware model (OURS)				
DENSNET-D	77.5	72.6	87.7	81.2

at least one native-incidence or structured model outperforms the graph-reduction baselines. For example, on PubMed, UniGNN reaches 87.6% and the published ED-HNN result reaches 89.0%, compared with 71.2% for HGNN and 78.3% for HyperGCN; on Cora-CA, UniGNN reaches 82.9% compared with 76.8% for HGNN and 74.3% for HyperGCN.

These comparisons are consistent with the practical value of retaining hypergraph incidence structure, but they do not isolate a single cause for the observed accuracy differences. Optimization, preprocessing, model capacity, input features, and architecture-specific inductive biases also vary across methods. Accordingly, Table 1 should not be interpreted as an empirical assignment of the architectures to generalized-hypertree-width levels or as direct verification of a Width Wall.

Finding 2: density features improve a matched AllDeepSets backbone. To isolate contribution of the density-aware branch, Appendix D.4 compares DENSNET-D with an AllDeepSets backbone using same hidden dimension, dropout, optimization settings, and data splits. Under this matched comparison, the density-aware variant improves mean test accuracy on all four benchmarks. Improvements are $+1.4$ points on Cora, $+1.8$ on Citeseer, $+0.6$ on PubMed, and $+0.8$ on Cora-CA. These gains provide evidence that explicitly estimated local pattern features can supply useful predictive information beyond the matched AllDeepSets representation.

Additional experimental results. Appendix D provides the full experimental protocol, dataset statistics, ablations, and results with error bars. Our experimental findings support the practical motivation for preserving and explicitly augmenting hypergraph structure.

6 CONCLUSION

We studied when hypergraph representations preserve genuinely higher-order structural information. Homomorphism densities provide a universal basis for continuous hypergraph invariants, while generalized hypertree width organizes these invariants into a graded filtration. This yields the *Width Wall*: when an architecture is provably restricted to width-r information, distinctions erased beyond that representation cannot be recovered by increasing hidden dimension or improving optimization. Our framework precisely identifies information lost by clique expansion, develops exact alignment invariants that separate hypergraph orbits, and motivates density-aware structural features. Experiments on real node-classification benchmarks illustrate both projection-level information loss and the practical value of augmenting an incidence-based backbone with local pattern densities. Characterizing Width Walls for broader HGNN families and quantifying the statistical and computational cost of richer structural representations remain important directions for future work.

AI USE STATEMENT

In this work, we used generative AI tools for implementing and running experiments, finding related work, generating synthetic data sets, cleaning and reformatting dataset, and improving the clarity of the manuscript. We have not used generative AI tools for research ideation, assisting with translation, supporting qualitative and thematic data analysis, or interpreting results. Developing theoretical models or conceptual frameworks, formulating mathematical claims, providing critical ingredients for proving mathematical claims, proposing or refining hypotheses are not applicable to this work. Additionally, we used generative AI tools for editing paper to improve readability, identifying relevant literature, formatting references, suggesting a structure for a research paper, proposing a title or keywords for a research paper. We have reviewed all AI-assisted work including code, figures, paper edit suggestions, search results. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work is a theoretical and benchmark study. It does not involve recruiting participants or collecting new personal data. The experiments use existing hypergraph benchmarks (see Appendix J). Improved modeling of higher-order relations may benefit scientific and biomedical analysis, but the same methods could be applied to sensitive relational data. In particular, predictions based on political affiliations, disease associations, or group memberships could expose private information, reinforce spurious associations, or lead to unfair decisions. Our benchmark accuracies do not establish suitability for decisions about individuals. Any such use requires appropriate data permissions, privacy safeguards, subgroup evaluation, and validation in the intended domain.

REPRODUCIBILITY STATEMENT

The mathematical claims are supported by proofs and explicit separation constructions in the appendices. For the empirical results, §5 and Appendices D, J, K, and L document dataset provenance and preprocessing, the reported train/validation/test protocol with seeds, model and optimizer settings, hyperparameter selection, and compute resources. The appendix also reports results with error bars and a controlled density-branch ablation. The datasets are publicly available through the cited benchmarks. We plan to release the implementation and experiment configurations upon acceptance; code is not yet publicly available.

REFERENCES

Devanshu Arya, Deepak Kumar Gupta, Stevan Rudinac, and Marcel Worring. HyperSAGE: Generalizing inductive representation learning on hypergraphs. In *arXiv preprint arXiv:2010.04558*, 2020.

Chaewoon Bae, Doyun Choi, Jaehyun Lee, and Jaemin Yoo. Parameter-free hypergraph neural network for few-shot node classification. *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Federico Battiston, Giulia Cencetti, Iacopo Iacopini, Vito Latora, Maxime Lucas, Alice Patania, Jean-Gabriel Young, and Giovanni Petri. Networks beyond pairwise interactions: Structure and dynamics. *Physics Reports*, 874:1–92, 2020.

Austin R. Benson, David F. Gleich, and Jure Leskovec. Higher-order organization of complex networks. *Science*, 353(6295):163–166, 2016.

Cristian Bodnar, Fabrizio Frasca, Yu Guang Wang, Nina Otter, Guido Montufar, Pietro Liò, and Michael Bronstein. Weisfeiler and Lehman go topological: Message passing simplicial networks. In *International Conference on Machine Learning (ICML)*, 2021.

Jan Böker. Color refinement, homomorphisms, and hypergraphs. In *Graph-Theoretic Concepts in Computer Science (WG)*, volume 11789 of *LNCS*, pp. 338–350, 2019. doi: 10.1007/978-3-030-30786-8_26.

Michael M. Bronstein, Joan Bruna, Taco Cohen, and Petar Veličković. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *arXiv preprint arXiv:2104.13478*, 2021.

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.

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

Charles J. Colbourn and Jeffrey H. Dinitz. *Handbook of Combinatorial Designs*. CRC Press, 2nd edition, 2007.

Holger Dell, Martin Grohe, and Gaurav Rattan. Lovász meets Weisfeiler and Leman. In *International Colloquium on Automata, Languages, and Programming (ICALP)*, 2018.

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

Iulia Duta, Giulia Cassarà, Fabrizio Silvestri, and Pietro Liò. Sheaf hypergraph networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Gábor Elek and Balázs Szegedy. A measure-theoretic approach to the theory of dense hypergraphs. *Advances in Mathematics*, 231(3–4):1731–1772, 2012.

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

Georg Gottlob, Nicola Leone, and Francesco Scarcello. Hypertree decompositions and tractable queries. *Journal of Computer and System Sciences*, 64(3):579–627, 2002.

Mustafa Hajij, Ghada Zamzmi, Theodore Papamarkou, Nina Miolane, Aldo Guzmán-Sáenz, Karthikeyan Natesan Ramamurthy, et al. Topological deep learning: Going beyond graph data. *arXiv preprint arXiv:2206.00606*, 2022.

Jing Huang and Jie Yang. UniGNN: a unified framework for graph and hypergraph neural networks. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2021.

Nicolas Keriven and Gabriel Peyré. Universal invariant and equivariant graph neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Jinwoo Kim, Saeyoon Oh, Sungjun Cho, and Seunghoon Hong. Equivariant hypergraph neural networks. In *European Conference on Computer Vision (ECCV)*, 2022.

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

Tamara G. Kolda and Brett W. Bader. Tensor decompositions and applications. *SIAM Review*, 51(3): 455–500, 2009.

Xiaoyu Li, Guangyu Tang, and Jiaojiao Jiang. Implicit hypergraph neural networks: A stable framework for higher-order relational learning with provable guarantees. *arXiv preprint arXiv:2508.09427*, 2025.

László Lovász. Operations with structures. *Acta Mathematica Hungarica*, 18(3–4):321–328, 1967.

László Lovász. *Large Networks and Graph Limits*, volume 60 of *Colloquium Publications*. American Mathematical Society, 2012.

László Lovász and Balázs Szegedy. Limits of dense graph sequences. *Journal of Combinatorial Theory, Series B*, 96(6):933–957, 2006.

Haggai Maron, Heli Ben-Hamu, Hadar Serviansky, and Yaron Lipman. Provably powerful graph networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019a.

Haggai Maron, Heli Ben-Hamu, Nadav Shamir, and Yaron Lipman. Invariant and equivariant graph networks. In *International Conference on Learning Representations (ICLR)*, 2019b.

Christopher Morris, Martin Ritzert, Matthias Fey, William L. Hamilton, Jan Eric Lenssen, Gaurav Rattan, and Martin Grohe. Weisfeiler and leman go neural: Higher-order graph neural networks. In *AAAI Conference on Artificial Intelligence*, 2019.

Benjamin Scheidt. On homomorphism indistinguishability and hypertree depth. In *International Colloquium on Automata, Languages, and Programming (ICALP)*, 2024.

Benjamin Scheidt and Nicole Schweikardt. Counting homomorphisms from hypergraphs of bounded generalised hypertree width: A logical characterisation. In *International Symposium on Mathematical Foundations of Computer Science (MFCS)*, 2023.

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.

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

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 Niber, Anand Louis, and Partha Talukdar. HyperGCN: A new method for training graph convolutional networks on hypergraphs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabás Póczos, Ruslan Salakhutdinov, and Alexander J. Smola. Deep sets. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

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

Yufei Zhao. Hypergraph limits: A regularity approach. *Random Structures & Algorithms*, 47(2): 205–226, 2015.

APPENDIX

A PROOFS

This appendix collects the proofs of the main theoretical results. The algebraic structure of pattern densities and the density theorem of Section 2 are established in Appendices A.1–A.2. Appendix A.3 proves the clique-expansion limitation of Proposition 3.2, while Appendix A.4 establishes the generalized-hypertree-width filtration of Theorem 3.6 and the architecture-to-width transfer result of Proposition 3.7. Finally, Appendix A.5 proves the alignment invariance and separation results underlying INVNET in Theorem 4.1.

A.1 ALGEBRA STRUCTURE OF PATTERN DENSITIES

We first record the algebraic property of homomorphism densities used throughout the paper. Let $\mathcal{F}_k$ denote the set of finite k-uniform patterns, with finite hyperedge multiplicities permitted when evaluating densities on the weighted domain. For ordinary unweighted hypergraphs, these multiplicities are immaterial.

Theorem A.1 (Algebra structure (cf. Lovász, 2012, Ch. 5))**.** *The collection of functions* $\{t(F,\cdot) : F \in \mathcal{F}_k^{\mathrm{multi}}\}$ *generates a unital algebra* $\mathfrak{A}_{k,n}$ *of continuous* $\mathrm{Sym}(n)$*-invariant functions on* $\mathcal{X}_{k,n} = [0,1]^{\binom{[n]}{k}}$*. In particular,*

$$t(F_1, \boldsymbol{A})\, t(F_2, \boldsymbol{A}) = t(F_1 \sqcup F_2, \boldsymbol{A}), \tag{7}$$

where $F_1 \sqcup F_2$ *denotes the disjoint union. The algebra contains constants through the empty pattern:* $t(\varnothing, \boldsymbol{A}) = 1$.

Proof. For every finite pattern F, the map $\boldsymbol{A} \mapsto t(F, \boldsymbol{A})$ is a finite polynomial in the hyperedge coordinates of $\boldsymbol{A}$, and is therefore continuous. It is invariant under vertex relabeling because, for any $\Pi \in \mathrm{Sym}(n)$, relabeling the target induces a bijection on the maps $\phi : V(F) \to [n]$ appearing in the definition of $t(F, \boldsymbol{A})$.

For the product identity, let F_1 and F_2 have disjoint vertex sets. Then

$$\begin{aligned} t(F_1, \boldsymbol{A}) t(F_2, \boldsymbol{A}) &= \frac{1}{n^{|V(F_1)|+|V(F_2)|}} \sum_{\phi_1 : V(F_1) \to [n]} \sum_{\phi_2 : V(F_2) \to [n]} \prod_{e \in E(F_1)} \boldsymbol{A}_{\phi_1(e)} \prod_{e \in E(F_2)} \boldsymbol{A}_{\phi_2(e)} \\ &= t(F_1 \sqcup F_2, \boldsymbol{A}), \end{aligned}$$

since a map from $V(F_1 \sqcup F_2)$ to $[n]$ is equivalent to a pair of maps on the two components. Finally, the empty pattern has density one. □

A.2 PROOF OF PATTERN-DENSITY UNIVERSALITY

We now prove Theorem 2.4. The classical finite-structure result states that homomorphism counts from all finite patterns determine an unweighted finite structure up to isomorphism (Lovász, 1967; 2012). For the weighted compact domain used in Theorem 2.4, we require the corresponding polynomial-orbit argument. Let $V_{k,n} = \mathbb{R}^{\binom{[n]}{k}}$ with coordinate functions x_e, $e \in \binom{[n]}{k}$. The group $\mathrm{Sym}(n)$ acts linearly on $V_{k,n}$ by permuting these hyperedge coordinates.

Lemma A.2 (Invariant polynomials from pattern counts)**.** *Every* $\mathrm{Sym}(n)$*-invariant polynomial on* $V_{k,n}$ *is a finite linear combination of weighted homomorphism counts of finite* k*-uniform multi-patterns.*

Proof. Every polynomial on $V_{k,n}$ is a linear combination of coordinate monomials of the form $m(\boldsymbol{A}) = \prod_{e \in \binom{[n]}{k}} \boldsymbol{A}_e^{a_e}$, where only finitely many exponents a_e are nonzero. The exponent vector (a_e) determines a finite labeled k-uniform multi-pattern: the hyperedge e occurs with multiplicity a_e.

Averaging m over the finite group gives its Reynolds orbit average

$$\mathcal{R}m(\boldsymbol{A}) = \frac{1}{|\mathrm{Sym}(n)|} \sum_{\Pi \in \mathrm{Sym}(n)} m(\Pi \cdot \boldsymbol{A}).$$

Up to a constant depending only on the labeling and automorphisms of the associated pattern, this orbit average is its weighted injective homomorphism count.

Weighted injective homomorphism counts can be expressed as finite linear combinations of weighted homomorphism counts by Möbius inversion over the partition lattice of the pattern vertex set. If a quotient identifies two vertices belonging to the same hyperedge, the corresponding contribution is zero because repeated-index coordinates are fixed to zero. Other identifications may create parallel hyperedges, which accounts for the use of multi-patterns.

Hence every Reynolds orbit average lies in the linear span of weighted homomorphism counts. Since every invariant polynomial is a linear combination of orbit averages of monomials, the claim follows. □

Lemma A.3 (Invariant polynomials separate orbits)**.** *If $\boldsymbol{A}_1, \boldsymbol{A}_2 \in V_{k,n}$ lie in distinct $\mathrm{Sym}(n)$-orbits, then there exists a $\mathrm{Sym}(n)$-invariant polynomial p such that $p(\boldsymbol{A}_1) \neq p(\boldsymbol{A}_2)$.*

Proof. Because $\mathrm{Sym}(n)$ is finite, the two orbits $\mathcal{O}(\boldsymbol{A}_i) = \{\Pi \cdot \boldsymbol{A}_i : \Pi \in \mathrm{Sym}(n)\}, i = 1, 2$, are finite and disjoint. There therefore exists a polynomial q on $V_{k,n}$ that takes value 1 on $\mathcal{O}(\boldsymbol{A}_1)$ and value 0 on $\mathcal{O}(\boldsymbol{A}_2)$. For example, such a polynomial can be obtained by interpolation on the finite set $\mathcal{O}(\boldsymbol{A}_1) \cup \mathcal{O}(\boldsymbol{A}_2)$.

Averaging q over $\mathrm{Sym}(n)$ gives $p(\boldsymbol{A}) = \frac{1}{|\mathrm{Sym}(n)|} \sum_{\Pi \in \mathrm{Sym}(n)} q(\Pi \cdot \boldsymbol{A})$. The polynomial p is $\mathrm{Sym}(n)$-invariant and satisfies $p(\boldsymbol{A}_1) = 1$ and $p(\boldsymbol{A}_2) = 0$. □

Proof of Theorem 2.4. Let $\mathcal{K} = \mathcal{X}_{k,n} = [0,1]^{\binom{[n]}{k}}$. Since $\mathcal{K}$ is compact and $\mathrm{Sym}(n)$ is finite, the orbit space $\mathcal{K}/\mathrm{Sym}(n)$ is compact Hausdorff.

By Theorem A.1, $\mathfrak{A}_{k,n}$ is a unital algebra of continuous functions on the orbit space. By Lemma A.3, invariant polynomials separate distinct $\mathrm{Sym}(n)$-orbits. Lemma A.2 shows that these invariant polynomials belong to the linear span of weighted homomorphism counts, and hence to $\mathfrak{A}_{k,n}$. Therefore $\mathfrak{A}_{k,n}$ contains constants and separates points of $\mathcal{K}/\mathrm{Sym}(n)$.

The Stone–Weierstrass theorem then implies that $\mathfrak{A}_{k,n}$ is uniformly dense in $C(\mathcal{K}/\mathrm{Sym}(n))$. Thus, for every continuous invariant f and every $\epsilon > 0$, there exists a polynomial in finitely many pattern densities that approximates f uniformly to accuracy ϵ. □

Relation to finite homomorphism completeness. For ordinary unweighted hypergraphs, the preceding argument is consistent with the classical finite-structure characterization (Lovász, 1967; 2012): $H_1 \cong H_2 \iff \hom(F, H_1) = \hom(F, H_2)$ for every finite k-uniform pattern F. Since $\boldsymbol{A}_e^m = \boldsymbol{A}_e$ on $\{0,1\}$-valued targets, parallel-edge multiplicities do not add information in the unweighted setting. Their role above is only to generate the repeated coordinate powers needed for polynomial approximation on the weighted domain.

A.3 Proof of Proposition 3.2

Proof. By assumption, every function realized by the architecture factors through the clique-expansion map: $f_\theta(\boldsymbol{A}) = g_\theta(\phi(\boldsymbol{A}))$. Hence, if $\phi(\boldsymbol{A}_1) = \phi(\boldsymbol{A}_2)$, then $f_\theta(\boldsymbol{A}_1) = g_\theta(\phi(\boldsymbol{A}_1)) = g_\theta(\phi(\boldsymbol{A}_2)) = f_\theta(\boldsymbol{A}_2)$ for every parameter choice θ.

Therefore every continuous invariant approximable by the architecture is determined by the projected representation $\phi(\boldsymbol{A})$. By the definition of $\mathcal{I}_{\mathrm{CE}}$, this gives $\mathcal{I}(\mathcal{F}_{\mathrm{CE}}) \subseteq \mathcal{I}_{\mathrm{CE}}$. □

A.4 Proof of Theorem 3.6

We first record the equivalent indistinguishability interpretation of reference classes in Def. 3.4.

Lemma A.4 (Width-r quotient characterization)**.** *For every $r \geq 1$, a continuous hypergraph invariant f belongs to $\mathcal{H}_r$ if and only if $\boldsymbol{A} \equiv_r \boldsymbol{B} \Longrightarrow f(\boldsymbol{A}) = f(\boldsymbol{B})$.*

Proof. Recall that $\boldsymbol{A} \equiv_r \boldsymbol{B} \iff t(F, \boldsymbol{A}) = t(F, \boldsymbol{B})$ for every pattern F with $W_{\mathrm{gh}}(F) \leq r$. Since each pattern density is continuous, $\equiv_r$ is a closed equivalence relation on the compact space $\mathcal{X}_{k,n}$. The

quotient $\mathcal{Q}_r := \mathcal{X}_{k,n}/\equiv_r$ is therefore compact Hausdorff. Every pattern density with $W_{\mathrm{gh}}(F) \leq r$ descends to a continuous function on $\mathcal{Q}_r$. By definition of $\equiv_r$, these descended functions separate distinct points of $\mathcal{Q}_r$. Their unital algebra consequently contains constants and separates points. The Stone–Weierstrass theorem implies that its uniform closure is $C(\mathcal{Q}_r)$. Pulling these functions back to $\mathcal{X}_{k,n}$ gives exactly the continuous functions that are constant on the $\equiv_r$-equivalence classes. By Definition 3.4, this class is $\mathcal{H}_r$. □

Proof of Theorem 3.6. The proof has two parts: first, we establish the pattern-width filtration; second, we exhibit a width-2 invariant that is lost by clique expansion.

Pattern-width filtration. If $W_{\mathrm{gh}}(F) \leq r$, then also $W_{\mathrm{gh}}(F) \leq r+1$. Hence the generating algebra defining $\mathcal{H}_r$ is contained in the one defining $\mathcal{H}_{r+1}$, and therefore $\mathcal{H}_r \subseteq \mathcal{H}_{r+1}$. Every pattern density is a continuous permutation-invariant function, so $\mathcal{H}_{r+1} \subseteq \mathcal{I}_{\mathrm{all}}$. Every finite pattern has finite generalized hypertree width. Consequently, $\bigcup_{r\geq 1} \mathcal{H}_r$ contains the algebra generated by all finite pattern densities. By Theorem 2.4, the uniform closure of the full pattern-density algebra is $\mathcal{I}_{\mathrm{all}}$. It follows that

$$\overline{\bigcup_{r\geq 1} \mathcal{H}_r}^{\|\cdot\|_\infty} = \mathcal{I}_{\mathrm{all}}.$$

Clique-expansion separation at width 2. Consider the Steiner pair $(\mathcal{H}_1, \mathcal{H}_2)$ from Appendix E.1. Both are Steiner triple systems on 13 vertices. Since every vertex pair occurs in exactly one triple in each system, their pairwise co-occurrence matrices are identical: $\phi(\boldsymbol{A}_1) = \phi(\boldsymbol{A}_2) = J - I$. Hence every invariant in $\mathcal{I}_{\mathrm{CE}}$ assigns the same value to $\mathcal{H}_1$ and $\mathcal{H}_2$.

Let F_{P} denote the Pasch configuration with $E(F_{\mathrm{P}}) = \big\{\{1,2,3\},\{1,4,5\},\{2,4,6\},\{3,5,6\}\big\}$. We first verify that $W_{\mathrm{gh}}(F_{\mathrm{P}}) = 2$. For the upper bound, consider a two-node decomposition tree with bags $B_1 = \{1,2,3,4,5\}, B_2 = \{2,3,4,5,6\}$. The bag B_1 is covered by $\{1,2,3\},\{1,4,5\}$, while B_2 is covered by $\{2,4,6\},\{3,5,6\}$. All four hyperedges are contained in a bag, and the bags containing each vertex form a connected subtree. Thus $W_{\mathrm{gh}}(F_{\mathrm{P}}) \leq 2$.

For the lower bound, recall that generalized hypertree width one is equivalent to hypergraph acyclicity, characterized by the existence of a join tree (Gottlob et al., 2002). In F_{P}, every pair of its four hyperedges intersects in a distinct vertex. Since each such vertex belongs to exactly those two hyperedges, the join-tree running-intersection condition would require every pair of hyperedges to be adjacent. This would require all six edges of K_4 in a tree on four nodes, which is impossible. Hence F_{P} has no join tree, so $W_{\mathrm{gh}}(F_{\mathrm{P}}) > 1$. Combining the two bounds gives $W_{\mathrm{gh}}(F_{\mathrm{P}}) = 2$.

It remains to show that its homomorphism density separates the Steiner pair. Appendix E.1 verifies that $\mathcal{H}_1$ contains 13 Pasch configurations whereas $\mathcal{H}_2$ contains 8. Injective homomorphisms from F_{P} therefore occur with different multiplicities in the two systems.

Non-injective homomorphisms contribute equally. Indeed, vertices belonging to a common hyperedge cannot be identified under a homomorphism into a Steiner triple system. The only possible identifications are therefore among the three pairs of opposite vertices of the Pasch configuration. If one opposite pair is identified, uniqueness of the triple containing each vertex pair in a Steiner triple system forces the other two opposite pairs to be identified as well. The image then collapses onto a single target triple. The number of such collapsed homomorphisms depends only on the number of triples, which is the same for both $\mathcal{H}_1$ and $\mathcal{H}_2$.

Thus, $\hom(F_{\mathrm{P}}, \mathcal{H}_1) \neq \hom(F_{\mathrm{P}}, \mathcal{H}_2)$, and, since both targets have 13 vertices, $t(F_{\mathrm{P}}, \mathcal{H}_1) \neq t(F_{\mathrm{P}}, \mathcal{H}_2)$. Therefore, $t(F_{\mathrm{P}}, \cdot) \in \mathcal{H}_2$. At the same time, this invariant cannot belong to $\mathcal{I}_{\mathrm{CE}}$, because $\mathcal{H}_1$ and $\mathcal{H}_2$ have identical clique expansions. Hence, $t(F_{\mathrm{P}}, \cdot) \in \mathcal{H}_2 \setminus \mathcal{I}_{\mathrm{CE}}$, which proves the claimed projection separation. □

Proof of Proposition 3.7. Suppose that $\boldsymbol{A} \equiv_r \boldsymbol{B} \Longrightarrow f(\boldsymbol{A}) = f(\boldsymbol{B})$ for every invariant f realizable by $\mathcal{F}$. Thus every such f is constant on the $\equiv_r$-equivalence classes. By Lemma A.4, $f \in \mathcal{H}_r$. The same property is preserved under uniform limits, and therefore $\mathcal{I}(\mathcal{F}) \subseteq \mathcal{H}_r$. By Definition 3.5, $\mathcal{F}$ consequently faces a Width Wall at level r. □

A.5 Proof of Theorem 4.1

We first establish the separation property of the exact pattern-alignment family.

Lemma A.5 (Exact alignment separates permutation orbits)**.** *Let $\boldsymbol{A}_1, \boldsymbol{A}_2 \in \mathcal{X}_{k,n}$. If $\boldsymbol{A}_1$ and $\boldsymbol{A}_2$ do not belong to the same $\mathrm{Sym}(n)$-orbit, then there exists a template P in the ambient real tensor space such that $\Theta_P(\boldsymbol{A}_1) \neq \Theta_P(\boldsymbol{A}_2)$.*

Proof. For $\boldsymbol{A} \in \mathcal{X}_{k,n}$, let $\mathcal{O}(\boldsymbol{A}) = \{\Pi \cdot \boldsymbol{A} : \Pi \in \mathrm{Sym}(n)\}$ denote its permutation orbit, and let $\mathcal{C}(\boldsymbol{A}) = \operatorname{conv} \mathcal{O}(\boldsymbol{A})$ be the corresponding orbit polytope.

For every template P, $\Theta_P(\boldsymbol{A}) = \max_{\boldsymbol{X} \in \mathcal{O}(\boldsymbol{A})} \langle P, \boldsymbol{X} \rangle = \max_{\boldsymbol{X} \in \mathcal{C}(\boldsymbol{A})} \langle P, \boldsymbol{X} \rangle$. Thus $\Theta_P(\boldsymbol{A})$ is the support function of the orbit polytope $\mathcal{C}(\boldsymbol{A})$ evaluated at P.

Suppose, toward a contradiction, that $\Theta_P(\boldsymbol{A}_1) = \Theta_P(\boldsymbol{A}_2)$ for every P. Two compact convex sets with identical support functions are equal, so $\mathcal{C}(\boldsymbol{A}_1) = \mathcal{C}(\boldsymbol{A}_2)$.

Every distinct point of $\mathcal{O}(\boldsymbol{A})$ is an exposed point of $\mathcal{C}(\boldsymbol{A})$. Indeed, for $\boldsymbol{X} \in \mathcal{O}(\boldsymbol{A})$, all elements of the orbit have the same Euclidean norm because permutations act orthogonally. Hence, for any distinct $\boldsymbol{Y} \in \mathcal{O}(\boldsymbol{A})$, $\langle \boldsymbol{X}, \boldsymbol{Y} \rangle < \|\boldsymbol{X}\|_2^2 = \langle \boldsymbol{X}, \boldsymbol{X} \rangle$, where the strict inequality follows from Cauchy–Schwarz. Thus the linear functional $\boldsymbol{Z} \mapsto \langle \boldsymbol{X}, \boldsymbol{Z} \rangle$ uniquely exposes $\boldsymbol{X}$.

Consequently, the exposed points of $\mathcal{C}(\boldsymbol{A}_i)$ are precisely the distinct elements of $\mathcal{O}(\boldsymbol{A}_i)$. Equality of the two orbit polytopes therefore implies $\mathcal{O}(\boldsymbol{A}_1) = \mathcal{O}(\boldsymbol{A}_2)$, contradicting the assumption that $\boldsymbol{A}_1$ and $\boldsymbol{A}_2$ lie in different permutation orbits. Hence some template P must separate them. □

Proof of Theorem 4.1. We prove the three statements in order.

Part 1: exact alignment. For a fixed template P, $\Theta_P(\boldsymbol{A}) = \max_{\Pi \in \mathrm{Sym}(n)} \langle P, \Pi \cdot \boldsymbol{A} \rangle$ is the maximum of finitely many linear functions of $\boldsymbol{A}$ and is therefore continuous. Let $\Gamma \in \mathrm{Sym}(n)$. Then

$$\begin{aligned}\Theta_P(\Gamma \cdot \boldsymbol{A}) &= \max_{\Pi \in \mathrm{Sym}(n)} \langle P, \Pi \cdot (\Gamma \cdot \boldsymbol{A}) \rangle \\ &= \max_{\Pi \in \mathrm{Sym}(n)} \langle P, (\Pi\Gamma) \cdot \boldsymbol{A} \rangle \\ &= \Theta_P(\boldsymbol{A}),\end{aligned}$$

because $\Pi \mapsto \Pi\Gamma$ is a bijection of $\mathrm{Sym}(n)$. Thus Θ_P is a continuous permutation invariant. Lemma A.5 shows that the family $\{\Theta_P\}_P$ separates distinct permutation orbits.

Part 2: global Birkhoff relaxation. Recall $\Theta_P^{\mathcal{B}}(\boldsymbol{A}) = \max_{D \in \mathcal{B}_n} \langle P, D^{\otimes k} \cdot \boldsymbol{A} \rangle$. Since every permutation matrix belongs to $\mathcal{B}_n$, the feasible set of the exact alignment problem is contained in that of the relaxed problem. Hence, $\Theta_P(\boldsymbol{A}) \leq \Theta_P^{\mathcal{B}}(\boldsymbol{A})$. The map $(D, \boldsymbol{A}) \longmapsto \langle P, D^{\otimes k} \cdot \boldsymbol{A} \rangle$ is continuous, and $\mathcal{B}_n$ is compact. It follows that $\Theta_P^{\mathcal{B}}$ is continuous in $\boldsymbol{A}$.

For permutation invariance, let $\Gamma \in \mathrm{Sym}(n)$. Then

$$\begin{aligned}\Theta_P^{\mathcal{B}}(\Gamma \cdot \boldsymbol{A}) &= \max_{D \in \mathcal{B}_n} \langle P, D^{\otimes k} \cdot (\Gamma \cdot \boldsymbol{A}) \rangle \\ &= \max_{D \in \mathcal{B}_n} \langle P, (D\Gamma)^{\otimes k} \cdot \boldsymbol{A} \rangle.\end{aligned}$$

Right multiplication by a permutation matrix is a bijection of $\mathcal{B}_n$. Therefore, $\Theta_P^{\mathcal{B}}(\Gamma \cdot \boldsymbol{A}) = \Theta_P^{\mathcal{B}}(\boldsymbol{A})$. Thus the globally optimized Birkhoff relaxation is a continuous permutation invariant.

Part 3: finite Sinkhorn implementation. Let $S_\psi(\Gamma \cdot \boldsymbol{A}) = S_\psi(\boldsymbol{A})\Gamma^\top$ for every $\Gamma \in \mathrm{Sym}(n)$, as assumed in Eqn. (6). Write $D_T(\boldsymbol{A}) = \mathrm{Sinkhorn}_T\left(S_\psi(\boldsymbol{A})/\tau\right)$. For $\tau > 0$, the entrywise exponential used to initialize the Sinkhorn iteration has strictly positive entries. Both row normalization and column normalization commute with a permutation of the columns. Consequently, by induction over the finite sequence of Sinkhorn normalizations, $D_T(\Gamma \cdot \boldsymbol{A}) = D_T(\boldsymbol{A})\Gamma^\top$.

Using the compatibility of the tensor action with matrix multiplication, we obtain

$$\begin{aligned}D_T(\Gamma \cdot \boldsymbol{A})^{\otimes k} \cdot (\Gamma \cdot \boldsymbol{A}) &= \left(D_T(\boldsymbol{A})\Gamma^\top\right)^{\otimes k} \cdot (\Gamma \cdot \boldsymbol{A}) \\ &= D_T(\boldsymbol{A})^{\otimes k} \cdot \boldsymbol{A}.\end{aligned}$$

Therefore

$$\begin{aligned}\widetilde{\Theta}_{P,\psi,T}(\Gamma \cdot \boldsymbol{A}) &= \langle P, D_T(\Gamma \cdot \boldsymbol{A})^{\otimes k} \cdot (\Gamma \cdot \boldsymbol{A})\rangle \\ &= \langle P, D_T(\boldsymbol{A})^{\otimes k} \cdot \boldsymbol{A}\rangle \\ &= \widetilde{\Theta}_{P,\psi,T}(\boldsymbol{A}).\end{aligned}$$

Hence the finite-Sinkhorn feature is permutation invariant.

If S_ψ is continuous, then $\boldsymbol{A} \mapsto D_T(\boldsymbol{A})$ is continuous because a finite Sinkhorn iteration is a finite composition of continuous maps on strictly positive matrices. Eqn. (4) then shows that $\widetilde{\Theta}_{P,\psi,T}$ is continuous. Consequently, if the local branch has a permutation-invariant readout and each score map S_{ψ_j} satisfies Eqn. (6), then the full architecture in Eqn. (5) is a continuous permutation-invariant function. □

Remark A.6 (Separation versus finite template sets). Lemma A.5 concerns the family of exact alignment scores obtained by allowing the template P to range over the full ambient tensor space. It does not imply that an arbitrary fixed collection of J learned templates separates every pair of non-isomorphic hypergraphs. Likewise, neither the global Birkhoff relaxation nor a finite-Sinkhorn implementation automatically inherits the orbit-separation property of the exact alignment family. Their separating power depends on the relaxation and on the learned templates.

B Additional Extensions

This appendix collects two extensions of the invariant framework developed in Section 2.2. We first give a finite-sample concentration bound for estimating individual pattern densities from random vertex maps. We then extend the fixed-n density result to non-uniform hypergraphs using mixed-arity patterns evaluated under a common vertex map.

B.1 Concentration of Pattern-Density Estimators

For a fixed pattern F, its homomorphism density can be estimated by sampling random vertex maps.

Proposition B.1 (Pattern-density concentration)**.** *Let F be a k-uniform pattern with $|V(F)| = v$, and let $\boldsymbol{A} \in \mathcal{X}_{k,n}$. Draw N independent uniformly random maps $\phi_i : V(F) \to [n], i = 1, \dots, N$, and define*

$$\widehat{t}(F, \boldsymbol{A}) = \frac{1}{N} \sum_{i=1}^{N} \prod_{e \in E(F)} A_{\phi_i(e)}.$$

Then, $\mathbb{E}[\widehat{t}(F, \boldsymbol{A})] = t(F, \boldsymbol{A})$, and, for every $\epsilon > 0$, $\Pr\big[\big|\widehat{t}(F, \boldsymbol{A}) - t(F, \boldsymbol{A})\big| \geq \epsilon\big] \leq 2\exp(-2N\epsilon^2)$. Consequently, additive accuracy ϵ with confidence at least $1 - \delta$ requires $N = O\Big(\frac{\log(1/\delta)}{\epsilon^2}\Big)$ independent samples.

Proof. For each sampled map ϕ_i, define

$$X_i = \prod_{e \in E(F)} A_{\phi_i(e)}.$$

Since $\boldsymbol{A} \in \mathcal{X}_{k,n}$, we have $X_i \in [0, 1]$. Moreover,

$$\mathbb{E}[X_i] = \frac{1}{n^v} \sum_{\phi : V(F) \to [n]} \prod_{e \in E(F)} A_{\phi(e)} = t(F, \boldsymbol{A}).$$

Thus $\widehat{t}(F, \boldsymbol{A})$ is unbiased. The concentration bound follows directly from Hoeffding's inequality applied to the independent bounded random variables $X_1, \dots, X_N$. □

The sample count above is independent of the target size n for additive estimation of a fixed pattern density. This does not imply that all patterns have the same computational cost: evaluating a single sample requires processing the hyperedges of F. Moreover, relative estimation becomes more demanding for rare patterns. For example, in the unweighted setting, when $t(F, H)$ is small, achieving a fixed relative accuracy generally requires a number of samples that grows with $1/t(F, H)$.

B.2 Non-Uniform Hypergraphs and Mixed-Arity Densities

We extend Theorem 2.4 to non-uniform hypergraphs. The key distinction is that different arity components are acted on by the *same* vertex permutation. Accordingly, independent invariant summaries of each arity need not retain the relative alignment between relations of different orders. The appropriate query objects are therefore mixed-arity patterns whose relations share a common vertex set.

Fix a maximum arity $K \geq 2$ and recall the compact domain $\mathcal{X}_{\leq K,n} = \prod_{j=2}^{K} [0,1]^{\binom{[n]}{j}}$, with the diagonal action of $\mathrm{Sym}(n)$. A *mixed-arity pattern* is a tuple $F = \big(V(F), E_2(F), \ldots, E_K(F)\big)$, where $E_j(F)$ is a collection of j-vertex hyperedges on the common vertex set $V(F)$. As in Section 2.2, finite hyperedge multiplicities are permitted when evaluating densities on the weighted domain.

For $\boldsymbol{A} = (A^{(2)}, \ldots, A^{(K)}) \in \mathcal{X}_{\leq K,n}$, define the mixed-arity homomorphism density by

$$t(F, \boldsymbol{A}) = \frac{1}{n^{|V(F)|}} \sum_{\phi: V(F) \to [n]} \prod_{j=2}^{K} \prod_{e \in E_j(F)} A^{(j)}_{\phi(e)}. \tag{8}$$

The same map ϕ is used simultaneously for all arities. Thus, unlike a collection of independently evaluated per-order statistics, $t(F, \boldsymbol{A})$ can record how relations of different sizes intersect on common vertices. This is the natural finite relational-structure notion of homomorphism counting (Lovász, 1967; 2012).

Corollary B.2 (Mixed-arity pattern-density universality)**.** *Fix $n \geq 1$ and $K \geq 2$. Let $C\big(\mathcal{X}_{\leq K,n}/\mathrm{Sym}(n)\big)$ denote the continuous functions on $\mathcal{X}_{\leq K,n}$ that are invariant under the diagonal action of $\mathrm{Sym}(n)$. Then the unital algebra generated by mixed-arity pattern densities $t(F, \cdot)$ is uniformly dense in $C\big(\mathcal{X}_{\leq K,n}/\mathrm{Sym}(n)\big)$. Equivalently, for every continuous invariant f and every $\epsilon > 0$, there exist mixed-arity patterns $F_1, \ldots, F_m$ and a polynomial $q : \mathbb{R}^m \to \mathbb{R}$ such that*

$$\sup_{\boldsymbol{A} \in \mathcal{X}_{\leq K,n}} \big| f(\boldsymbol{A}) - q\big(t(F_1, \boldsymbol{A}), \ldots, t(F_m, \boldsymbol{A})\big) \big| < \epsilon.$$

Proof. The proof follows the orbit-sum argument of Appendix A.2. The coordinate space underlying $\mathcal{X}_{\leq K,n}$ is $V_{\leq K,n} = \bigoplus_{j=2}^{K} \mathbb{R}^{\binom{[n]}{j}}$, and $\mathrm{Sym}(n)$ acts diagonally by simultaneously permuting the vertex labels in every arity component.

A coordinate monomial on $V_{\leq K,n}$ has the form $m(\boldsymbol{A}) = \prod_{j=2}^{K} \prod_{e \in \binom{[n]}{j}} \big(A^{(j)}_e\big)^{a_{j,e}}$, where only finitely many exponents $a_{j,e}$ are nonzero. Such a monomial determines a finite mixed-arity pattern on a common labeled vertex set: an edge e of arity j occurs with multiplicity $a_{j,e}$.

Averaging m over $\mathrm{Sym}(n)$ produces an invariant orbit sum. As in the proof of Lemma A.2, this orbit sum is, up to a constant depending only on the labeling and automorphisms of the pattern, a weighted injective homomorphism count of the corresponding mixed-arity pattern. Möbius inversion over partitions of the common vertex set expresses each such injective count as a finite linear combination of mixed-arity homomorphism counts. Identifications of vertices belonging to the same hyperedge give zero because repeated-index coordinates are excluded, while other identifications may create parallel hyperedges and are handled by the allowed edge multiplicities.

Hence every $\mathrm{Sym}(n)$-invariant polynomial on $V_{\leq K,n}$ lies in the linear span of mixed-arity homomorphism counts. Invariant polynomials separate distinct $\mathrm{Sym}(n)$-orbits by the same finite-group averaging argument used in Lemma A.3. Since $\mathcal{X}_{\leq K,n}$ is compact and $\mathrm{Sym}(n)$ is finite, the orbit space $\mathcal{X}_{\leq K,n}/\mathrm{Sym}(n)$ is compact Hausdorff. The mixed-arity density algebra contains constants, is closed under addition and multiplication, and separates points of this orbit space. The Stone–Weierstrass theorem therefore gives the claimed uniform density. □

Why per-order invariants are insufficient. The use of a common vertex map in (8) is essential. Suppose two non-uniform hypergraphs have isomorphic j-uniform components for every j, but the permutations realizing these isomorphisms cannot be chosen consistently across arities. Every invariant computed independently within each arity then agrees on the two hypergraphs, and taking products of these independently invariant quantities cannot recover the lost relative alignment.

For example, on four vertices consider

$$H_1: \quad E_2 = \{\{1,2\}\}, \qquad E_3 = \{\{1,2,3\}\},$$
$$H_2: \quad E_2 = \{\{1,2\}\}, \qquad E_3 = \{\{1,3,4\}\}.$$

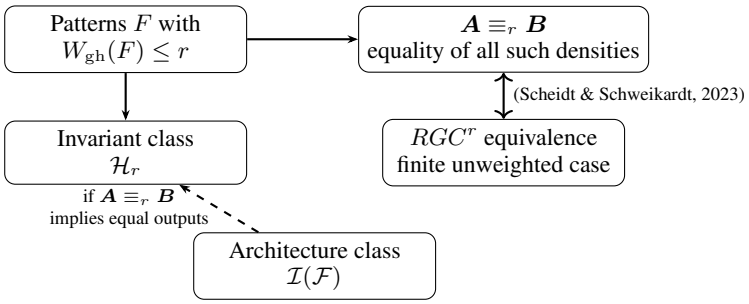
The 2-uniform components are isomorphic, as are the 3-uniform components, but the full mixed-arity structures are not: in H_1 the 2-edge and 3-edge intersect in two vertices, whereas in H_2 they intersect in one. A mixed-arity pattern containing one 2-edge and one 3-edge sharing two vertices therefore records information that no collection of independently invariant per-order summaries can recover.

Products of per-order densities correspond only to mixed-arity patterns whose different-arity components have disjoint vertex sets. They are consequently a proper subset of the mixed-arity pattern algebra needed to capture the diagonal $\mathrm{Sym}(n)$-action.

C Architecture Classification of Modern HGNNs

This appendix situates representative HGNN architectures relative to the information restrictions developed in Section 3. The classification is deliberately architectural rather than a direct assignment of generalized-hypertree-width levels. An architecture is placed behind a Width Wall at level r only when an independent indistinguishability or refinement argument establishes the upper bound required by Proposition 3.7.

Pattern width, logic, and architecture. For finite unweighted hypergraphs, (Scheidt & Schweikardt, 2023) establish an exact correspondence between homomorphism indistinguishability over patterns of generalized hypertree width at most r and equivalence in a two-sorted guarded counting logic with r hyperedge variables. This correspondence concerns structural indistinguishability; an HGNN belongs to the corresponding function class only after an architecture-specific upper bound has been proved.



The dashed arrow is the architecture-dependent step. Neither message-passing depth nor hidden dimension alone determines its position in the width filtration.

Equivariant higher-order networks. Higher-order invariant and equivariant networks process tensor-valued features and can exceed the distinguishing power of first-order message passing (Maron et al., 2019a; Keriven & Peyré, 2019). When sufficiently high tensor orders are allowed, such constructions admit universality results for permutation-invariant or equivariant functions. A fixed tensor order, however, should not automatically be identified with a particular generalized-hypertree-width level. Their increasing $O(n^q)$ storage and computation at tensor order q motivate lower-cost structural summaries such as the pattern features studied in this paper.

Relation to generalized Weisfeiler–Leman refinements. Zhang et al. (Zhang et al., 2025) introduce a higher-dimensional generalized Weisfeiler–Leman hierarchy for hypergraphs and show that increasing its dimension yields a strict increase in distinguishing power. Their k-HNN construction provides a neural realization associated with that refinement hierarchy. We treat this as a complementary expressivity scale. Without an additional transfer theorem, the k parameter of k-GWL is not identified here with the generalized-hypertree-width index r of $\mathcal{H}_r$.

C.1 Extended Classification Table

Table 2 summarizes how representative HGNNs obtain structural information. The final column records only information restrictions justified by the architecture itself; it does not assign a generalized-hypertree-width level unless such a bound has been independently established.

C.2 Conditions for Architecture–Width Transfer

The following conditions delimit the structure-only architecture class to which a refinement or indistinguishability bound may be applied. They do *not*, by themselves, imply a formula relating message-passing depth to generalized hypertree width.

- **(C1) Permutation equivariance.** Node- and hyperedge-level computations commute with relabelings of the vertex set, and invariant outputs use permutation-invariant readouts.
- **(C2) Locality when a local-message-passing bound is invoked.** If an expressivity upper bound relies on local propagation, each update must use only the neighborhood specified by that bound. Global attention, global pooling inside intermediate layers, or other all-pairs operations require separate analysis. Locality alone does not determine a GHW level.
- **(C3) Symmetric aggregation.** Aggregation over unordered incident nodes or hyperedges must itself be permutation invariant or equivariant, as appropriate. Examples include sum, mean, DeepSets-style aggregation, and permutation-equivariant attention.
- **(C4) No external symmetry-breaking information.** A structure-only width claim excludes node identifiers, random identifiers, positional encodings, or other auxiliary features that distinguish vertices beyond information supplied by the hypergraph itself.
- **(C5) The full preprocessing pipeline is part of the architecture.** Any reduction, spectral preprocessing, sampling rule, positional encoding, or learned preprocessing step must satisfy the same claimed indistinguishability bound. A bound for the downstream message-passing network does not apply if preprocessing has already introduced information outside that bound.

Under these conditions, Proposition 3.7 can be applied once one has separately proved that $\boldsymbol{A} \equiv_r \boldsymbol{B} \implies f(\boldsymbol{A}) = f(\boldsymbol{B})$ for every $f \in \mathcal{F}$. Only then may one conclude $\mathcal{I}(\mathcal{F}) \subseteq \mathcal{H}_r$. Neither the number of message-passing layers nor the hidden dimension alone supplies this implication.

Remark C.1 (Positional encodings and the Width Wall). Positional encodings, random node features, node identifiers, and related augmentations change the information presented to the architecture. They can therefore increase distinguishing power without changing the underlying message-passing operator. Such models are not contradicted by a Width-Wall result proved for structure-only inputs; rather, they fall outside the corresponding indistinguishability assumption.

The purpose of the structure-only analysis is to isolate which invariants are recoverable from the hypergraph representation itself. When external or globally computed features are supplied, the relevant expressivity analysis must include the information contained in those features as part of the architecture.

D Additional Experimental Details

This appendix provides the dataset statistics, training protocol, computational verification of the Steiner witness, and additional real-data results supporting Section 5. The real-data analysis in the revised manuscript uses four benchmarks: Cora, Citeseer, PubMed, and Cora-CA. For these datasets, the benchmark provides input node features that are independent of the target labels.

D.1 Dataset Statistics

Dataset statistics showing the number of nodes, hyperedges, features, and classes for the benchmarks used in Sec. 5 is shown in Table 3.

Table 2: Structural classification of representative HGNN architectures. A Width-Wall level is not assigned unless supported by a separate indistinguishability or refinement bound.

Model	Operator	Structural view	Key consideration
Graph reduction			
HGNN (Feng et al., 2019)	Incidence-derived graph propagation	Clique projection	Prop. matrix $D_v^{-1/2} H W D_e^{-1} H^\top D_v^{-1/2}$ depends on pairwise co-incidence information. When ϕ retains this weighted pairwise projection, the model factors through ϕ and Proposition 3.2 applies.
HyperGCN (Yadati et al., 2019)	Mediator graph	Graph reduction	Hyperedges replaced by mediator edges before graph convolution. Should not automatically be identified with $\mathcal{I}_{\mathrm{CE}}$; full preprocessing pipeline must be included in analysis.
Native incidence message passing			
HNHN (Dong et al., 2020)	Nonlinear n↔e	Native incidence	Degree-weighted node–hyperedge message passing preserves incidence representation. No GHW level asserted without architecture-specific indistinguishability bound.
UniGNN (Huang & Yang, 2021)	Unified MP	Native incidence	GCN/GAT/GIN-style operators are applied through the incidence structure. Locality alone does not determine a generalized-hypertree-width level.
AllDeepSets (Chien et al., 2022)	DeepSets/edge	Native set aggregation	Uses symmetric set functions within hyperedges and across incident hyperedges. Its precise Width-Wall level, if any, requires a separate refinement analysis.
AllSetTransformer (Chien et al., 2022)	Attention/edge	Native set aggregation	Attention is permutation equivariant over hyperedge multisets, but this fact alone does not determine the GHW of the patterns represented by the architecture.
Diffusion / operator learning			
ED-HNN (Wang et al., 2023)	Equivariant diffusion	Native hypergraph operator	Uses permutation-equivariant hypergraph diffusion. Local/ iterative operator structure does not imply particular GHW bound.
Higher-order / topological			
k-GWL HNNs (Zhang et al., 2025)	k-dim. GWL	Higher-dimensional refinement	Strict hierarchy in k-GWL dimension. We do not identify its level k with $\mathcal{H}_k$ without an explicit transfer theorem.
SheafHNN (Duta et al., 2023)	Sheaf Laplacian	Additional sheaf structure	Input contains structure beyond the bare hypergraph domain considered in Sections 2–3.
Preprocessing-heavy / implicit			
TF-HNN (Tang et al., 2025)	Training-free MP	Preprocessed propagation	Expressivity depends on complete preprocessing and propagation pipeline, not only on number of learned layers.
ZEN (Bae et al., 2025)	Closed-form linear	Precomputed propagation	Closed-form propagation operator must be analyzed as part of architecture; propagation depth alone does not determine GHW level.
IHGNN (Li et al., 2025)	Implicit equilibrium	Implicit incidence operator	No bounded-width conclusion follows from nominal network depth alone.
InvNet (ours)	Pattern alignment	Global invariant alignment	Exact alignment scores separate permutation orbits. Globally optimized Birkhoff score is also invariant, while finite Sinkhorn implementation invariant under score-equivariance condition of Thm. 4.1.

Table 3: Statistics for node-classification benchmarks used in Section 5.

Dataset	Nodes	Hyperedges	Features	Classes	Avg. HE size
Cora	2,708	1,579	1,433	7	3.03
Citeseer	3,312	1,079	3,703	6	3.20
PubMed	19,717	7,963	500	3	4.35
Cora-CA	2,708	1,072	1,433	7	4.28

D.2 Steiner Pair Verification

The synthetic structural witness used in our separation experiments is the pair of non-isomorphic $\mathrm{STS}(13)$ hypergraphs constructed in Appendix E.1. Appendix E.1 gives the explicit systems and establishes their theoretical properties; here we report computational consistency checks on the instances used in the experiments.

For each system, we compute the pairwise co-occurrence matrix $C_{ij} = \big|\{e \in \mathcal{E} : i \in e,\ j \in e\}\big|$. For both $\mathcal{H}_1$ and $\mathcal{H}_2$, we obtain $C_{ij} = 1$ for every $i \neq j$. Thus the two experimental instances have identical clique expansions, $\phi(\boldsymbol{A}_1) = \phi(\boldsymbol{A}_2) = J - I$.

We also independently enumerate Pasch configurations and recover the counts used in Appendix E.1: $\mathcal{H}_1$ contains 13 Pasch configurations whereas $\mathcal{H}_2$ contains 8. These computational checks verify that the instances used experimentally match the theoretical Steiner witness. The width-2 separation itself is established analytically in Section 3 and Appendix A.

D.3 Training Details

Data splits. For each retained real-data benchmark, we use random $50/25/25$ train/validation/test node splits. We generate 10 deterministic splits using seeds $0, \ldots, 9$ and report mean test accuracy and standard deviation across the resulting runs.

Optimization. All models are trained with the Adam optimizer (Kingma & Ba, 2015). For node classification, we use learning rate 10^{-3}, weight decay 0, gradient clipping 1.0, label smoothing 0.1, a cosine-annealing learning-rate schedule, and at most 200 epochs with early stopping of patience 50. Normalization parameters and bias terms are excluded from weight decay whenever weight decay is used. The baselines use hidden dimension 64 and dropout 0.2, following the AllSet-style benchmark configuration. The primary DensNet-D runs use hidden dimension 128 and dropout 0.5. Consequently, the broad comparison in Table 1 is not a matched-capacity ablation.

To isolate the contribution of the density branch, we therefore perform a separate controlled comparison in Section D.4. There, the AllDeepSets backbone and DensNet-D use the same hidden dimension, dropout, optimization settings, and training schedule; the density branch is the architectural difference.

InvNet-specific settings. For the small synthetic alignment experiments, the number of templates is selected from $J \in \{4, 8, 16\}$, the number of finite Sinkhorn iterations from $T_{\mathrm{iter}} \in \{5, 10, 20\}$, and the temperature is annealed from 1.0 to 0.01 during training. These finite-Sinkhorn computations correspond to the implementation described in Appendix F.

Compute. All experiments are run on NVIDIA H100 GPUs. Node classification uses 8 GPUs in parallel with per-job isolation (no DDP); full sweep completes in < 12 hours.

D.4 Density Ablation

To isolate the effect of the density branch, Table 4 compares DensNet-D with an AllDeepSets backbone under an identical training configuration: hidden dimension 128, dropout 0.5, label smoothing 0.1, the same optimizer and cosine schedule, and the same train/validation/test splits.

The density-aware variant improves mean test accuracy on all four benchmarks, with differences ranging from 0.6 to 1.8 percentage points. These results provide evidence that explicitly estimated

Table 4: Matched density ablation on the four benchmarks. AllDeepSets and DENSNET-D use the same training configuration; the *only architectural difference* is the density branch. Values are test accuracy (%), reported as mean ± standard deviation over 10 splits.

Dataset	AllDeepSets	DENSNET-D	Difference
Cora	76.1 ± 1.7	77.5 ± 1.2	$+1.4$
Citeseer	70.8 ± 1.7	72.6 ± 0.5	$+1.8$
PubMed	87.1 ± 0.6	87.7 ± 0.5	$+0.6$
Cora-CA	80.4 ± 1.2	81.2 ± 1.0	$+0.8$

local pattern features can contain predictive information beyond that used by the matched AllDeepSets backbone.

D.5 NODE CLASSIFICATION RESULTS WITH ERROR BARS

Table 5 reports the full real-data results used in Section 5. The ED-HNN and SheafHyperGNN entries are previously published values obtained using their original architecture-specific training recipes; the remaining rows are run under our ANCS baseline protocol.

Table 5: Node classification accuracy (%) on the four real-data benchmarks. ‡ denotes published results obtained with the corresponding architecture-specific training protocol. All other rows are run under our ANCS baseline configuration.

Method	Cora	Citeseer	PubMed	Cora-CA
MLP	74.1 ± 1.0	72.3 ± 0.3	87.1 ± 0.3	74.6 ± 1.0
HGNN	69.9 ± 6.7	68.1 ± 0.7	71.2 ± 9.7	76.8 ± 1.0
HyperGCN	71.0 ± 5.9	68.1 ± 1.7	78.3 ± 5.4	74.3 ± 0.6
HNHN	72.8 ± 5.1	68.3 ± 0.7	70.0 ± 11.5	76.6 ± 0.8
UniGNN	77.3 ± 0.7	72.4 ± 0.7	87.6 ± 0.5	82.9 ± 0.8
AllDeepSets	76.2 ± 1.5	70.9 ± 1.1	87.2 ± 0.4	80.6 ± 1.1
ED-HNN‡	80.3 ± 1.2	73.7 ± 1.5	89.0 ± 0.6	84.0 ± 1.0
SheafHyperGNN‡	81.3 ± 1.1	74.7 ± 1.3	87.7 ± 0.5	85.5 ± 0.9
DENSNET-D	77.5 ± 1.2	72.6 ± 0.5	87.7 ± 0.5	81.2 ± 1.0

E SEPARATION CONSTRUCTION

This appendix gives the explicit hypergraph pair used to illustrate two distinct information losses discussed in the main text. The construction witnesses the failure of clique expansion in Theorem 3.6 and provides a concrete example for the pattern-alignment mechanism of Section 4. The same pair is also used in the corresponding synthetic separation experiment.

E.1 THE STEINER PAIR

We construct two non-isomorphic 3-uniform hypergraphs on $n = 13$ vertices with *identical* clique expansions using Steiner triple systems.

A *Steiner triple system* $\mathrm{STS}(v)$ is a collection of 3-element subsets of $[v]$ such that every 2-element subset is contained in exactly one triple. An $\mathrm{STS}(v)$ has

$$\frac{\binom{v}{2}}{\binom{3}{2}} = \frac{v(v-1)}{6}$$

triples, and its clique expansion is K_v. The number of non-isomorphic $\mathrm{STS}(v)$ systems is 1 for $v \in \{7, 9\}$ and 2 for $v = 13$ (Colbourn & Dinitz, 2007). Thus $v = 13$ is the smallest order admitting a non-isomorphic STS pair.

Explicit construction. Let $\mathcal{H}_1$ be the *cyclic* $\mathrm{STS}(13)$, constructed from the base blocks $\{0,1,4\}$ and $\{0,2,7\}$ under the action of $\mathbb{Z}_{13}$:

$$\mathcal{H}_1 = \big\{\{i, i{+}1, i{+}4\} : i \in \mathbb{Z}_{13}\big\} \cup \big\{\{i, i{+}2, i{+}7\} : i \in \mathbb{Z}_{13}\big\},$$

where all additions are modulo 13. This gives 26 triples.

Let $\mathcal{H}_2$ be obtained from $\mathcal{H}_1$ by a *Pasch trade*: replace

$$\{3,4,7\},\ \{3,5,10\},\ \{4,10,12\},\ \{5,7,12\}$$

with

$$\{3,4,10\},\ \{3,5,7\},\ \{4,7,12\},\ \{5,10,12\}.$$

The trade preserves the pairwise incidences on these six vertices and therefore preserves the Steiner property.

Identical clique expansions. Every pair of vertices occurs in exactly one triple in both systems. Consequently, their pairwise co-occurrence matrices agree and $\phi(\boldsymbol{A}_1) = \phi(\boldsymbol{A}_2) = J - I$, the adjacency matrix of K_{13}. Hence every invariant in $\mathcal{I}_{\mathrm{CE}}$ assigns the same value to $\mathcal{H}_1$ and $\mathcal{H}_2$.

Non-isomorphism and Pasch separation. The two systems have different numbers of Pasch configurations: $\mathcal{H}_1$ contains 13, whereas $\mathcal{H}_2$ contains 8. Since the number of Pasch configurations is an isomorphism invariant, $\mathcal{H}_1$ and $\mathcal{H}_2$ are non-isomorphic.

Let F_{P} denote the Pasch pattern $E(F_{\mathrm{P}}) = \big\{\{1,2,3\},\{1,4,5\},\{2,4,6\},\{3,5,6\}\big\}$. As shown in Appendix A.4, $W_{\mathrm{gh}}(F_{\mathrm{P}}) = 2$. Moreover, $t(F_{\mathrm{P}}, \mathcal{H}_1) \neq t(F_{\mathrm{P}}, \mathcal{H}_2)$. Thus the Steiner pair is indistinguishable after clique expansion but is separated by a width-2 hypergraph pattern density. This gives the explicit witness used in Theorem 3.6.

Exact alignment separation. The same pair also illustrates the distinction between pairwise projection and global pattern alignment. Take the template $P = \boldsymbol{A}_1$. Then, $\Theta_{\boldsymbol{A}_1}(\boldsymbol{A}_1) = 156$, since $\mathcal{H}_1$ contains 26 triples and each contributes 3! ordered tensor entries. If $\Theta_{\boldsymbol{A}_1}(\boldsymbol{A}_2) = 156$, some permutation would align every hyperedge of $\mathcal{H}_2$ with a hyperedge of $\mathcal{H}_1$, yielding an isomorphism between the two systems. Since no such isomorphism exists, $\Theta_{\boldsymbol{A}_1}(\boldsymbol{A}_2) < \Theta_{\boldsymbol{A}_1}(\boldsymbol{A}_1)$. Thus exact pattern alignment separates a pair that clique expansion cannot.

F InvNet Architecture Details

This appendix expands on the InvNet architecture introduced in Section 4. We distinguish the globally optimized Birkhoff relaxation from the finite Sinkhorn map used in the implementation and state the condition under which the latter preserves permutation invariance.

F.1 Birkhoff Relaxation and Finite Sinkhorn Alignment

Recall the exact pattern-alignment score $\Theta_P(\boldsymbol{A}) = \max_{\Pi \in \mathrm{Sym}(n)} \langle P, \Pi \cdot \boldsymbol{A} \rangle$. Replacing permutation matrices by the Birkhoff polytope gives the global relaxation

$$\Theta_P^{\mathcal{B}}(\boldsymbol{A}) = \max_{D \in \mathcal{B}_n} \langle P, D^{\otimes k} \cdot \boldsymbol{A} \rangle, \tag{9}$$

where $\mathcal{B}_n$ is the set of $n \times n$ doubly stochastic matrices. Since permutation matrices belong to $\mathcal{B}_n$, $\Theta_P(\boldsymbol{A}) \leq \Theta_P^{\mathcal{B}}(\boldsymbol{A})$. As shown in Appendix A.5, $\Theta_P^{\mathcal{B}}$ is a continuous permutation invariant.

The finite implementation used by InvNet is different from solving Eqn. (9) globally. For each template P_j, a learned score map produces $S_{\psi_j}(P_j, \boldsymbol{A}) \in \mathbb{R}^{n \times n}$. Starting from the positive matrix obtained by exponentiating $S_{\psi_j}(P_j, \boldsymbol{A})/\tau$, we apply T_{iter} alternating row and column normalizations: $D_{j,T}(\boldsymbol{A}) = \mathrm{Sinkhorn}_{T_{\mathrm{iter}}}\big(S_{\psi_j}(P_j, \boldsymbol{A})/\tau\big)$. The implemented alignment feature is then

$$\widetilde{\Theta}_{P_j, \psi_j, T}(\boldsymbol{A}) = \langle P_j, D_{j,T}(\boldsymbol{A})^{\otimes k} \cdot \boldsymbol{A} \rangle. \tag{10}$$

Thus finite Sinkhorn iterations define a learned alignment map; they are not assumed to solve the global optimization problem in Eqn. (9).

For the formal permutation-invariance guarantee of Theorem 4.1, the score map must satisfy

$$S_\psi(P, \Pi \cdot \boldsymbol{A}) = S_\psi(P, \boldsymbol{A})\Pi^\top \qquad \text{for every } \Pi \in \mathrm{Sym}(n). \tag{11}$$

Under Eqn. (11), finite Sinkhorn normalization preserves the same transformation law: $D_T(\Pi \cdot \boldsymbol{A}) = D_T(\boldsymbol{A})\Pi^\top$, and consequently $\widetilde{\Theta}_{P,\psi,T}$ is permutation invariant.

A sufficient way to enforce Eqn. (11) is to form the score matrix from shared vertex-level descriptors. For example, let $U_\eta(\Pi \cdot \boldsymbol{A}) = \Pi U_\eta(\boldsymbol{A})$ be a permutation-equivariant vertex encoder for the input and let $Q_\eta(P)$ contain descriptors of the template vertices. A shared pairwise scoring function $[S_\psi(P, \boldsymbol{A})]_{iv} = s_\psi([Q_\eta(P)]_i, [U_\eta(\boldsymbol{A})]_v)$ then satisfies (11). More generally, any score construction satisfying this transformation law is admissible.

F.2 Full Architecture

Algorithm 1 InvNet: Invariant-Theoretic Hypergraph Neural Network

Require: Hypergraph tensor $\boldsymbol{A} \in \mathcal{X}_{k,n}$, node features $X \in \mathbb{R}^{n \times d_0}$
Require: Learnable templates $\{P_j\}_{j=1}^J \subset \mathbf{S}^{k,n}$, offsets $\{\alpha_j\}_{j=1}^J$
Require: Message-passing layers, score maps $\{S_{\psi_j}\}_{j=1}^J$, temperature τ, iterations T_{iter}

// Local branch: hyperedge message passing
$\boldsymbol{h}^{(0)} \leftarrow X$
for $\ell = 1, \dots, L$ **do**
  $\boldsymbol{m}_e^{(\ell)} \leftarrow \phi_e\big(\{\boldsymbol{h}_v^{(\ell-1)} : v \in e\}\big)$ ▷ Hyperedge aggregation
  $\boldsymbol{h}_v^{(\ell)} \leftarrow \phi_v\big(\boldsymbol{h}_v^{(\ell-1)}, \{\boldsymbol{m}_e^{(\ell)} : v \in e\}\big)$ ▷ Node update
end for
$\boldsymbol{z}_{\mathrm{local}} \leftarrow \mathrm{ReadOut}(\boldsymbol{h}^{(L)})$

// Alignment branch: finite Sinkhorn features
for $j = 1, \dots, J$ **do**
  $S_j \leftarrow S_{\psi_j}(P_j, \boldsymbol{A})$ ▷ Alignment score matrix
  $D_j \leftarrow \mathrm{Sinkhorn}_{T_{\mathrm{iter}}}(S_j/\tau)$
  $\widetilde{\theta}_j \leftarrow \langle P_j, D_j^{\otimes k} \cdot \boldsymbol{A}\rangle - \alpha_j$
end for
$\boldsymbol{z}_{\mathrm{inv}} \leftarrow [\widetilde{\theta}_1, \dots, \widetilde{\theta}_J]$

// Combine and predict
return $\mathrm{MLP}_{\mathrm{out}}(\boldsymbol{z}_{\mathrm{local}} \| \boldsymbol{z}_{\mathrm{inv}})$

For inputs with node features, vertex relabeling acts jointly as $(\boldsymbol{A}, X) \longmapsto (\Pi \cdot \boldsymbol{A}, \Pi X)$. The usual shared aggregate–combine operations make the local branch equivariant under this joint action, and an invariant pooling operation makes $\boldsymbol{z}_{\mathrm{local}}$ invariant. The corresponding guarantee for the alignment branch additionally requires the score-equivariance condition in Eqn. (11).

Computational complexity. For each template, T_{iter} Sinkhorn normalizations of an $n \times n$ score matrix require $O(T_{\mathrm{iter}} n^2)$ operations. Thus the normalization component of the alignment branch costs $O(JT_{\mathrm{iter}} n^2)$. The total cost also includes construction of the score matrices and evaluation of $\langle P_j, D_j^{\otimes k} \cdot \boldsymbol{A}\rangle$; these costs depend on the template and sparse-tensor parameterization used in the implementation. We therefore report them separately rather than identifying the entire alignment branch with the $O(n^2)$ Sinkhorn normalization cost.

Template parameterization. For variable-size or input-conditioned variants, both template generation and alignment-score construction must respect the appropriate permutation action for the invariance guarantee of Theorem 4.1 to apply. Such parameterizations should therefore be distinguished from the fixed-template construction analyzed in the theorem.

Remark F.1 (Ideal and implemented alignment). The family of exact alignment scores separates permutation orbits by Lemma A.5. This does not imply that a finite learned template set, the global

Birkhoff relaxation, or a finite-Sinkhorn implementation has the same separation power. Decreasing τ sharpens the finite Sinkhorn map but does not, by itself, eliminate a possible relaxation or optimization gap. Accordingly, the implemented alignment branch is used as a tractable approximation whose empirical separating behavior is evaluated directly.

G DensNet-D Architecture Details

This appendix provides the architecture specification of DensNet-D, the density-aware node classifier evaluated in Table 1 of the main text. DensNet-D augments an AllDeepSets backbone (Chien et al., 2022) with per-node structural features obtained by Monte Carlo estimation of local pattern densities. The construction is motivated by the pattern-density framework of Sections 2–3, but is not claimed to realize a particular level $\mathcal{H}_r$ of the generalized hypertree-width filtration.

G.1 Architecture

Let $\mathcal{H} = (V, \mathcal{E})$ be a hypergraph with node features $X \in \mathbb{R}^{n\times d}$. For each node v, DensNet-D computes a backbone representation and a density-aware structural representation, which are fused before classification.

1. AllDeepSets backbone. We use a two-layer incidence-based message-passing backbone with permutation-invariant aggregation. The initial node representation is

$$h_v^{(0)} = W_{\text{in}} x_v. \tag{12}$$

At layer ℓ, node representations are first aggregated within each hyperedge,

$$m_e^{(\ell)} = \text{MLP}_e^{(\ell)} \left(\sum_{u\in e} h_u^{(\ell-1)} \right), \tag{13}$$

and the resulting hyperedge representations are then aggregated at each incident node,

$$a_v^{(\ell)} = \sum_{e\ni v} m_e^{(\ell)}. \tag{14}$$

The node update is

$$h_v^{(\ell)} = \text{LN}\left(h_v^{(\ell-1)} + \text{MLP}_v^{(\ell)}\big(a_v^{(\ell)}\big) \right), \tag{15}$$

where the MLPs use ReLU activations and dropout and LN denotes LayerNorm. All four real-data benchmarks use this same aggregation form.

2. Local density branch. For each node v, define its incident-star subhypergraph

$$\mathcal{H}_v = (V_v, \mathcal{E}_v), \qquad \mathcal{E}_v = \{e \in \mathcal{E} : v \in e\}, \qquad V_v = \bigcup_{e\in\mathcal{E}_v} e.$$

Given a library of patterns $\mathcal{P} = \{F_1, \ldots, F_P\}$, the density branch forms

$$\rho_v = \left[\widehat{t}(F_1, \mathcal{H}_v), \ldots, \widehat{t}(F_P, \mathcal{H}_v)\right], \tag{16}$$

where each $\widehat{t}(F_p, \mathcal{H}_v)$ is a Monte Carlo estimate of the corresponding local pattern density. We use 200 Monte Carlo samples per node, selected from the validation grid $\{100, 200, 500\}$; complete settings are reported in Table 8.

The pattern library is selected by the DatasetProfiler using coarse structural statistics such as the edge-size distribution and mean node degree. This profiling rule is an implementation heuristic rather than an assignment of the dataset to a level of the width filtration.

The estimated density vector is projected to the hidden dimension via

$$z_v = \text{LN}\big(W_\rho \rho_v + b_\rho\big). \tag{17}$$

The Monte Carlo estimates are treated as fixed structural inputs: gradients are not propagated through the sampling procedure.

3. Per-node gate. A learned gate controls the contribution of the density representation:

$$g_v = \sigma\left(w^\top h_v^{(L)} + b_g\right), \tag{18}$$

where σ is the sigmoid function. The gate weight w is initialized to zero, so its initial value is determined by b_g.

For the four benchmarks, the initialization is chosen using the same descriptive mean-edge-size rule used by the implementation:

- $\bar{k} < 4$ (Cora, Citeseer): $b_g = -5.0$;
- $4 \le \bar{k} < 7$ (PubMed, Cora-CA): $b_g = -2.0$.

These bins are used only as implementation heuristics and should not be interpreted as generalized-hypertree-width levels.

4. Concatenation fusion. The gated density embedding is concatenated with the backbone representation,

$$r_v = h_v^{(L)} \,\|\, g_v z_v, \tag{19}$$

and the class logits are

$$\hat{y}_v = \mathrm{MLP}_{\mathrm{out}}(r_v), \tag{20}$$

where $\mathrm{MLP}_{\mathrm{out}} : \mathbb{R}^{2d_h} \to \mathbb{R}^C$ is a two-layer classifier. Thus the backbone and density representation occupy separate coordinates before the final prediction layer.

5. Staged unfreezing. The density-side trainable parameters are initially frozen and are unfrozen later in training. For the benchmarks, the unfreezing epoch is determined by the same descriptive profiler rule:

- Cora and Citeseer: epoch 50;
- PubMed and Cora-CA: epoch 30.

During the frozen phase, the backbone continues to train while the density-side trainable parameters remain fixed. The staged schedule is an optimization heuristic intended to reduce early interference between the two branches; it has no role in the expressivity results of Sections 3–4.

G.2 Training Recipe

All four datasets use the same core optimization settings: hidden dimension 128, learning rate 10^{-3}, dropout 0.5, two message-passing layers, gradient clipping at 1.0, label smoothing 0.1, a cosine learning-rate schedule, and at most 200 epochs with early-stopping patience 50.

The dataset-dependent choices in DensNet-D are the pattern library selected by the DatasetProfiler, the gate initialization, and the staged-unfreezing epoch. These quantities are chosen from dataset-level structural statistics and are treated as engineering choices rather than theoretically determined width parameters. The complete configuration is reported in Table 9.

G.3 Relationship to the Pattern-Density Framework

DensNet-D uses the same basic structural primitive as the theoretical framework: counts, or Monte Carlo estimates of counts, of small hypergraph patterns. There are, however, two important distinctions.

First, the theoretical classes $\mathcal{H}_r$ are generated by global homomorphism densities of all patterns satisfying a generalized hypertree-width constraint. DensNet-D instead uses a finite, selected library of *local* star-neighborhood density features. Consequently, its density branch should not be identified with $\mathcal{H}_r$ for any specified r without an additional realization argument.

Second, the pattern-density theory concerns structural invariant information, whereas DensNet-D combines these structural coordinates with ordinary node features and an incidence-based neural backbone. The empirical ablation in Appendix D.4 therefore tests whether the selected density coordinates provide useful information beyond the matched AllDeepSets backbone; it does not constitute an empirical measurement of the backbone's Width Wall.

H EXAMPLES OF HYPERGRAPH INVARIANTS

This appendix illustrates how representative hypergraph invariants fit within the framework of Sections 2–3. The classifications below are deliberately information-theoretic. Membership in a class such as $\mathcal{H}_r$ does not imply that every neural architecture informally associated with width r computes the corresponding invariant; an architecture-specific upper or realization result is required for such a conclusion. Likewise, when an invariant admits a representation using patterns of generalized hypertree width at most r, we obtain a certified inclusion in $\mathcal{H}_r$. We do not claim that r is the minimum possible width unless a matching separation result is available. Let $\boldsymbol{A} \in \mathcal{X}_{k,n}$ denote a weighted k-uniform hypergraph.

Pattern-density invariants. For every finite k-uniform pattern F, $\boldsymbol{A} \longmapsto t(F, \boldsymbol{A})$ is a continuous permutation-invariant function. By definition, $t(F, \cdot) \in \mathcal{H}_{W_{\mathrm{gh}}(F)}$. Polynomial combinations of densities whose patterns have width at most r belong to the algebra generating $\mathcal{H}_r$.

The simplest example is the number of hyperedges. Let E_k denote the pattern consisting of one k-edge. For an unweighted simple hypergraph,

$$|\mathcal{E}| = \frac{n^k}{k!}\, t(E_k, \boldsymbol{A}),$$

and $W_{\mathrm{gh}}(E_k) = 1$. Hence edge count is a width-1 pattern invariant.

A more informative example is the Pasch pattern P_{Pasch} used in Section 3. Since $W_{\mathrm{gh}}(P_{\mathrm{Pasch}}) = 2$, its homomorphism density satisfies $t(P_{\mathrm{Pasch}}, \cdot) \in \mathcal{H}_2$. On the Steiner pair of Appendix E.1, this invariant distinguishes two hypergraphs having identical clique expansions. Thus it provides a concrete example of structural information available beyond the clique projection.

Clique-projection invariants. Let $\phi : \mathcal{X}_{k,n} \to \mathcal{X}_{2,n}$ be the clique projection used in Section 3. Any continuous graph invariant g gives a hypergraph invariant $f(\boldsymbol{A}) = g(\phi(\boldsymbol{A}))$ belonging to the reference class $\mathcal{I}_{\mathrm{CE}}$.

Examples include continuous spectral statistics of the projected graph, such as $\lambda_{\max}(\phi(\boldsymbol{A}))$ or continuous functions of its eigenvalues. Such quantities depend only on the clique projection and therefore assign the same value to any two hypergraphs with identical $\phi(\boldsymbol{A})$. This statement concerns information retained by the projection; it does not imply that an arbitrary graph-message-passing implementation computes every invariant in $\mathcal{I}_{\mathrm{CE}}$.

Exact alignment invariants. For a real template P in the ambient tensor space, the exact alignment score $\Theta_P(\boldsymbol{A}) = \max_{\Pi \in \mathrm{Sym}(n)} \langle P, \Pi \cdot \boldsymbol{A} \rangle$ is a continuous permutation invariant. As established in Section 4, the full family $\{\Theta_P : P\}$ separates distinct permutation orbits.

Alignment scores therefore provide a second construction of hypergraph invariants, conceptually different from homomorphism densities. For example, choosing P to be the tensor of a fixed template hypergraph yields the maximum weighted overlap between that template and a relabeling of the input. We do not assign such a score to a particular $\mathcal{H}_r$ without an additional width analysis.

General continuous invariants. The density theorem of Section 2 applies beyond quantities that have an obvious small-pattern interpretation. For example, on the weighted fixed-n domain define

$$\mathrm{MaxCut}(\boldsymbol{A}) = \max_{S \subseteq [n]} \sum_{e \in \binom{[n]}{k}} A_e\, \mathbf{1}\left\{e \cap S \neq \varnothing,\ e \cap S^c \neq \varnothing\right\}.$$

This is a maximum over finitely many linear functions of $\boldsymbol{A}$ and is therefore a continuous permutation invariant. Consequently, $\mathrm{MaxCut} \in \mathcal{I}_{\mathrm{all}}$, and the density theorem guarantees approximation by finite combinations of pattern densities on the fixed-n compact domain. The theorem alone does not identify the smallest width r for which $\mathrm{MaxCut} \in \mathcal{H}_r$.

Table 6: Representative hypergraph invariants and their certified relationship to our expressivity framework. The table reports only inclusions established directly by the definitions or results of this paper; it does not assign neural architectures to minimum width levels.

Invariant	**Certified class**	**Reason**
Hyperedge count	$\mathcal{H}_1$	Single-edge density
$t(F,\cdot)$	$\mathcal{H}_{W_{\mathrm{gh}}(F)}$	Definition of the width filtration
Pasch density	$\mathcal{H}_2$	$W_{\mathrm{gh}}(P_{\mathrm{Pasch}}) = 2$
Continuous invariant of $\phi(\boldsymbol{A})$	$\mathcal{I}_{\mathrm{CE}}$	Factors through clique projection
CE spectral statistic	$\mathcal{I}_{\mathrm{CE}}$	Function of $\phi(\boldsymbol{A})$
Exact alignment score Θ_P	$\mathcal{I}_{\mathrm{all}}$	Continuous permutation invariant
Weighted hypergraph MAXCUT	$\mathcal{I}_{\mathrm{all}}$	Continuous invariant on fixed n

I Related Work

We situate our contributions within five threads of related work. Our perspective connects homomorphism-based structural invariants with architectural information loss in hypergraph neural networks.

Homomorphism counting and logical expressivity. Lovász (1967) showed that graph homomorphism counts determine finite graphs up to isomorphism; Lovász (2012) developed this viewpoint further in graph limit theory. Dell et al. (2018) connected graph homomorphism counts to Weisfeiler–Leman expressivity. For hypergraphs, Scheidt & Schweikardt (2023) characterized indistinguishability by homomorphism counts from patterns of bounded generalized hypertree width using a two-sorted guarded counting logic with bounded hyperedge variables. Scheidt (2024) developed related connections for hypertree depth. Our work uses these results as a structural reference for organizing continuous hypergraph invariants by pattern width, while keeping architecture-specific realizability as a separate question.

GNN expressivity and WL hierarchies. Xu et al. (2019) and Morris et al. (2019) established fundamental connections between message-passing GNNs and Weisfeiler–Leman refinement. Higher-order invariant and equivariant graph architectures provide increasingly expressive alternatives beyond standard message passing (Maron et al., 2019a;b). For hypergraphs, Böker (2019) studied color refinement, while Zhang et al. (2025) introduced k-dimensional generalized WL and proved a strict hierarchy in the refinement dimension, together with corresponding higher-order HNN constructions. This hierarchy is complementary to our generalized-hypertree-width filtration: both organize structural distinguishability, but we do not identify k-GWL level k with $\mathcal{H}_k$ without an explicit transfer theorem. Choi et al. (2026) further extended WL-style refinement to categorical structures.

Hypergraph neural networks. The HGNN landscape has evolved from graph-reduction approaches (Feng et al., 2019; Yadati et al., 2019) to incidence-based hyperedge processing (Dong et al., 2020; Huang & Yang, 2021; Chien et al., 2022; Arya et al., 2020) and permutation-equivariant hypergraph architectures (Kim et al., 2022; Wang et al., 2023). Zaheer et al. (2017) established the Deep Sets framework underlying many symmetric aggregation mechanisms, while Duta et al. (2023) incorporated additional sheaf-theoretic structure. Our framework differs in focusing on which invariant information is retained by a representation before making an architecture-specific expressivity claim.

Topological and geometric deep learning. Bodnar et al. (2021) extended message passing to simplicial complexes, and Hajij et al. (2022) developed a broader topological deep-learning perspective. Bronstein et al. (2021) emphasized symmetry and group actions as organizing principles for geometric deep learning. Our formulation adopts this viewpoint for the action of the symmetric group on hypergraph tensors and studies invariant structural coordinates under vertex relabeling.

Tensor methods. Classical tensor methods provide powerful tools for multilinear data analysis (Kolda & Bader, 2009), but are not generally formulated around invariance to hypergraph vertex

relabeling. Our exact pattern-alignment scores instead combine tensor representations with permutation optimization, providing an invariant template-based complement to homomorphism-density features.

J Dataset Statistics and Preprocessing

Table 7 summarizes the structural characteristics of the four real-data benchmarks used in Table 1. The reported statistics characterize the size and edge-cardinality distribution of each dataset and provide the dataset-level quantities used by the DensNet-D implementation.

Data provenance. Cora, Citeseer, and PubMed are co-citation hypergraphs from the AllSet benchmark (Chien et al., 2022), originally constructed from the datasets used by Yadati et al. (2019). Each hyperedge groups papers co-cited by a common paper. Cora-CA is a co-authorship hypergraph from the same benchmark, where each hyperedge groups the co-authors of one publication. All four datasets use the benchmark-provided node features. Data-loading scripts and the deterministic splits used in our experiments are included in the supplementary code.

Data splits. All experiments use $50/25/25$ train/validation/test random splits. We generate 10 deterministic splits using seeds $0, \ldots, 9$ and report mean $\pm$ standard deviation across the corresponding runs.

Table 7: Structural statistics of the four benchmarks. n: nodes, m: hyperedges, d: input-feature dimension, C: number of classes, $\bar{k}$: mean hyperedge cardinality, k_{95}: 95th-percentile hyperedge cardinality, $k_{\max}$: maximum hyperedge cardinality, and $\bar{\deg}$: mean node degree. These quantities are descriptive dataset statistics and are not proxies for generalized hypertree width.

Dataset	n	m	d	C	$\bar{k}$	k_{95}	$k_{\max}$	$\bar{\deg}$
Cora	2,708	1,579	1,433	7	3.03	5	5	1.8
Citeseer	3,312	1,079	3,703	6	3.20	7	26	1.0
PubMed	19,717	7,963	500	3	4.35	12	171	1.8
Cora-CA	2,708	1,072	1,433	7	4.28	13	43	1.7

DatasetProfiler inputs. For the experiments, the profiler uses mean hyperedge cardinality to select the gate initialization and staged-unfreezing schedule described in Appendix K. Specifically,

$$\bar{k} < 4 \quad \Longrightarrow \quad b = -5.0, \text{ unfreeze epoch} = 50,$$

whereas

$$4 \leq \bar{k} < 7 \quad \Longrightarrow \quad b = -2.0, \text{ unfreeze epoch} = 30.$$

These thresholds are fixed implementation heuristics. They do not correspond to levels of the generalized-hypertree-width filtration.

The remaining statistics in Table 7, including k_{95}, $k_{\max}$, and $\bar{\deg}$, are reported to characterize the datasets and to make the preprocessing pipeline reproducible. In particular, the substantial difference between mean and maximum hyperedge cardinality on PubMed and Cora-CA illustrates why mean hyperedge size alone should not be interpreted as a measure of structural complexity.

K Hyperparameter Configuration and Reproducibility

This appendix records the training and model-specific settings used for the results in Table 1.

Shared training protocol. All models use the Adam optimizer with learning rate 10^{-3}, weight decay 0, gradient clipping at 1.0, label smoothing 0.1, and a cosine-annealing learning-rate schedule. Training runs for at most 200 epochs with early-stopping patience 50. For each dataset, we use the same 10 deterministic train/validation/test splits described in Appendix D.

Baseline models. The baselines in Table 1 are MLP, HGNN, HyperGCN, HNHN, UniGNN, and AllDeepSets. They use hidden dimension 64 and dropout 0.2. The hypergraph models use two message-passing layers. These settings follow the AllSet-style benchmark configuration (Chien et al., 2022).

DENSNET-D. DENSNET-D uses hidden dimension 128, dropout 0.5, two AllDeepSets backbone layers, and 200 Monte Carlo samples per node for local density estimation. Its per-node gate is $g_v = \sigma\left(\mathbf{w}^\top \mathbf{h}_v + b\right)$, as described in Appendix G.

For the four benchmarks, the implementation uses two descriptive mean-edge-size settings:

$$\begin{aligned} \bar{k} < 4 \quad &: \quad b = -5.0, \qquad \text{unfreeze at epoch } 50, \\ 4 \le \bar{k} < 7 \quad &: \quad b = -2.0, \qquad \text{unfreeze at epoch } 30. \end{aligned}$$

The gate weight $\mathbf{w}$ is initialized to zero, so these choices give initial gate values $\sigma(-5) \approx 0.007$ and $\sigma(-2) \approx 0.119$, respectively.

These thresholds are engineering heuristics based on coarse dataset statistics. They are not generalized-hypertree-width estimates and do not assign the datasets or the model to levels of the Width-Wall filtration. No special density-feature normalization is used on the four benchmarks.

Table 8 summarizes the model-level settings, and Table 9 gives the complete per-dataset DENSNET-D configuration.

Table 8: Model-specific hyperparameters for the experiments. All models use Adam with lr$= 10^{-3}$, weight decay$= 0$, gradient clipping$= 1.0$, label smoothing$= 0.1$, and cosine learning-rate annealing. "MC" denotes the number of Monte Carlo samples used for local density estimation.

Model	Hidden	Dropout	Layers	Gate init	MC
MLP	64	0.2	2	—	—
HGNN	64	0.2	2	—	—
HyperGCN	64	0.2	2	—	—
HNHN	64	0.2	2	—	—
UniGNN	64	0.2	2	—	—
AllDeepSets	64	0.2	2	—	—
DENSNET-D	128	0.5	2	dataset-dependent	200

Table 9: DENSNET-D configuration on the four benchmarks. The mean-edge-size bins are implementation heuristics only and do not represent Width-Wall levels. Gate initialization b determines the initial contribution $\sigma(b)$; "unfreeze epoch" denotes when the density-side trainable parameters begin updating.

Dataset	$\bar{k}$	Gate init b	$\sigma(b)$	Unfreeze epoch	MC
Cora	3.03	-5.0	0.007	50	200
Citeseer	3.20	-5.0	0.007	50	200
PubMed	4.35	-2.0	0.119	30	200
Cora-CA	4.28	-2.0	0.119	30	200

Hyperparameter selection. For baselines, we use the common benchmark configuration above rather than tuning each architecture separately. For DENSNET-D, we considered hidden dimensions $\{64, 128, 256\}$, dropout values $\{0.2, 0.5\}$, and Monte Carlo sample counts $\{100, 200, 500\}$. The gate initialization and staged-unfreezing schedule follow the fixed mean-edge-size rules stated above. These rules are deterministic once the dataset statistics are known, but they are implementation heuristics rather than theoretically prescribed hyperparameters.

Software and hardware. Model training and baseline sweeps are run on $8\times$ NVIDIA H100 GPUs using PyTorch 2.6.0, CUDA 12.4, and Python 3.10 (conda `hginv` environment). The DENSNET-D

density features are precomputed once and then treated as fixed structural inputs during neural-network training. Precomputation is performed on an Apple M4 Max with 128 GB unified memory. It requires less than one minute for Cora, Citeseer, and Cora-CA and approximately two hours for PubMed ($n = 19{,}717$). Configuration files, random seeds, and launch commands for the reported experiments are included in the supplementary code.

L COMPUTE BUDGET

The total computational cost of our experimental evaluations is $\approx$200 H100 GPU-hours, of which approximately 40% ($\approx$80 GPU-hours) produced the final reported results and 60% ($\approx$120 GPU-hours) was spent on development iterations, debugging, and hyperparameter exploration. The dominant cost is invariant-based models (INVNET), which account for all five timeout events (3-hour cap per job); density-based models (DENSNET-D) and all baselines complete in under 5 minutes per seed on every dataset.

At approximately 700W per H100, the total energy cost is $\approx$140 kWh, corresponding to $\approx$56 kg CO_2e at the US average grid intensity of 0.4 kg/kWh. This is modest relative to the scale of the evaluation (approximately 860 individual training runs across the full sweep alone) and reflects the efficiency of the 8-GPU parallel scheduling: most jobs complete in under 30 minutes, with only InvNet-based models requiring hour-scale compute.