Linear GCNs Need Better Bias, Not More Expressive Power

Celia Rubio-Madrigal [1] **Rebekka Burkholz** [1]

Abstract

Do modern GNNs succeed because of architectural expressive power or because of implicit bias? We show that simple linear graph convolutions with sum aggregation and MLP classifiers can match the generalization performance of state-of-the-art graph learning methods. Theoretically, sum aggregation after a linear feature transformation can preserve neighborhood multiset information, yielding Weisfeiler–Lehman-level expressive power. Empirically, however, this potential is realized only through distillation from more complex GNN teachers. When trained from scratch, linear GCNs often fail to generalize. This reveals a gap between what simple models can express and what standard training selects. Our findings suggest that progress in graph learning may depend less on increasing expressiveness and more on controlling implicit bias, trainability, and model selection. Our experimental design provides a framework to distinguish these concepts.

1. Introduction

Graph Neural Networks (GNNs) are a standard tool for learning on graph structured data, with applications spanning social networks (Fan et al., 2019), knowledge graphs (Schlichtkrull et al., 2018), biology (Weighill et al., 2021; Gotovos et al., 2021; Burkholz et al., 2021; Hossain et al., 2024a;b), and physical systems (Sanchez-Gonzalez et al., 2020). At the heart of message-passing GNNs is a neighborhood aggregation step (Gori et al., 2005; Scarselli et al., 2009; Gilmer et al., 2017b; Bronstein et al., 2021). Many modern architectures boost performance by making aggregation (and the associated updates) increasingly expressive and layer-specific, often at the cost of more parameters, hyperparameters, and training complexity (You et al., 2020).

A striking counterpoint was recently established by Luo et al. (2024; 2025): When equipped with standard training components—normalization, dropout, residual connections, and broader hyperparameter tuning—classic models such as GCNs (Kipf & Welling, 2017) can match or rival substantially more complex GNNs (including Transformer-style variants (Wu et al., 2023; Deng et al., 2024b; Kong et al., 2023; Wu et al., 2022; Chen et al., 2023; Rampášek et al., 2022; Shirzad et al., 2023)) across many node and graph classification benchmarks. These results raise a basic question that remains theoretically underexplained: *Are strong results from well-trained GCNs a consequence of hidden expressive power already present in simple aggregation schemes?*

A follow-up line of work by Rubio-Madrigal & Burkholz (2026) provides a partial theoretical answer: Aggregation need not be learned *in theory*. Their fixed aggregation features (FAFs) recursively compute layerwise aggregation statistics to form a tabular representation that can be classified by a sufficiently powerful MLP. Empirically, however, a paradox appears. The maximally expressive fixed multiset functions work badly in practice, while the generally successful choices (e.g., sum/mean-based) are not maximally expressive, and exhibit a substantial performance gap on datasets that require depth and residual connections—notably, Minesweeper and Roman-Empire. Thus, the gap between in-principle expressive power, trainability, and generalization is not fully closed.

In this work, we prove that linear graph convolutions with sum aggregation can always preserve neighborhood multiset information at the same level as WL-equivalent architectures such as GIN (Xu et al., 2019). Concretely, fix any task-relevant equivalence relation on node representations—that is, whatever distinctions one wishes the aggregation process to preserve—and call its equivalence classes clusters. If the representations are sufficiently separable with respect to these clusters, then a single linear transformation can map embeddings to (approximately) orthogonal codes, after which sum aggregation acts as a counting mechanism that preserves multiset statistics. From this viewpoint, the "amount of aggregation learning" a dataset requires is tied to how easily the model can maintain (or induce) separable representations as neighborhoods are repeatedly mixed.

[1]CISPA Helmholtz Center for Information Security. Correspondence to: Celia Rubio-Madrigal <celia.rubio-madrigal@cispa.de>.

The Workshop on Graph Foundation Models at the 43rd International Conference on Machine Learning (ICML 2026), Seoul, South Korea. Copyright 2026 by the author(s).

Our theory establishes the existence of generalizing solutions for linear GNNs. To recover such solutions in practice, we use knowledge distillation (KD) from SOTA teachers. Unlike Zhang et al. (2022a), who distill into less expressive graph-agnostic MLPs, we distill into linear students that remain simpler than the teacher, but are expressive enough for any task, as we show. We use the same KL-divergence regularization between teacher and student logits, steering training toward generalizing solutions that exist but are not naturally reached.

On the medium-sized node classification benchmarks of Luo et al. (2024), our students match, and in some cases improve upon, their SOTA teachers. We obtain similar results on 6 of the the more recent GraphLand node prediction benchmarks (Bazhenov et al., 2025), including 3 classification and 3 regression tasks, and on a 16-hop long-range dataset from City-Networks (Liang et al., 2026). These results suggest that linear GCNs are expressive enough for these tasks, but that they are biased toward solutions that generalize worse than those found by nonlinear models. The limitation is therefore implicit bias, rather than expressive power or range.

These findings motivate a shift in graph architecture design: from increasing expressive power toward improving and controlling implicit bias. Yet what constitutes a good bias for graph data remains poorly understood, and we currently lack principled tools for distinguishing well-generalizing models from poorly generalizing ones beyond intuition and empirical evaluation. Moreover, current SOTA models cannot be viewed as definitive, as further improvements remain possible under this framework.

Our contributions are as follows:

1. **Theory:** We prove that sum-aggregation linear GCN layers can preserve neighborhood multiset counts under separability and sufficient-width assumptions, clarifying an underappreciated expressive power mechanism of linear graph convolution.

2. **Diagnostic:** We use KD as a model-selection intervention to test whether high-performing predictors exist inside linear GNN student classes.

3. **Experiments:** Across transductive node-prediction benchmarks (Luo et al., 2024; Bazhenov et al., 2025; Liang et al., 2026), distilled linear GCNs match or surpass strong GNN teachers, while training them from scratch often do not. These results suggest that the main limitation of linear students is not expressive power, but their implicit bias during training.

2. Related Work

A recurring question in graph learning is how to attribute the gains of GNNs. Much of the literature explains these gains through the discriminative power of message passing, the ability of aggregation to preserve and transmit graph signal, or dataset-level heuristics that predict when graph information is useful. These perspectives have motivated increasingly complex architectures that modify aggregation, propagation, or graph conditioning to address apparent limitations of standard message passing.

Expressive power Expressive power in message-passing GNNs is most commonly understood as discriminative power: which graph-level inputs, or which neighborhood multisets, an architecture can map to distinct representations. This view is formalized by comparisons with the Weisfeiler–Lehman hierarchy (Gilmer et al., 2017a; Xu et al., 2019). A related but distinct line of work studies whether this expressive power can be effectively realized through signal propagation. Repeated aggregation may make node representations indistinguishable, leading to over-smoothing (Li et al., 2019; Oono & Suzuki, 2020; Zhou et al., 2021), or may compress exponentially many distant signals through bottlenecks, leading to over-squashing (Topping et al., 2022; Jamadandi et al., 2024; Rubio-Madrigal et al., 2025). Many proposed remedies therefore focus on increasing architectural expressive power with the assumption that the model will realize those gains.

Benchmark-driven heuristics The relationship between graph learning benchmarks and GNN model generalization has been studied from heuristic-based and empirical perspectives. Li et al. (2023) use regression analysis to identify data properties that are predictive of model performance, while You et al. (2020) explore a large number of model-task combinations to transfer architectural insights across tasks. For node classification in particular, performance is often explained through graph-level heuristics such as homophily/heterophily, motivating architectures explicitly designed for such regimes (Zhu et al., 2020; 2021; Chien et al., 2021; Maurya et al., 2022; Li et al., 2022). Beyond homophily, Platonov et al. (2023a) propose label informativeness as a more direct measure of how much information a neighbor's label provides about a node's label, while Luan et al. (2024) argue that GNNs help when aggregation makes intra-class node embeddings closer than inter-class ones. These heuristic explanations have also driven increasingly complex aggregation mechanisms, often designed to address specific perceived failures, and sometimes introducing additional optimization machinery or hyperparameters (Finkelshtein et al., 2024; Errica et al., 2025). However, recent benchmark studies show that carefully trained classical GNNs can match much more complex architectures

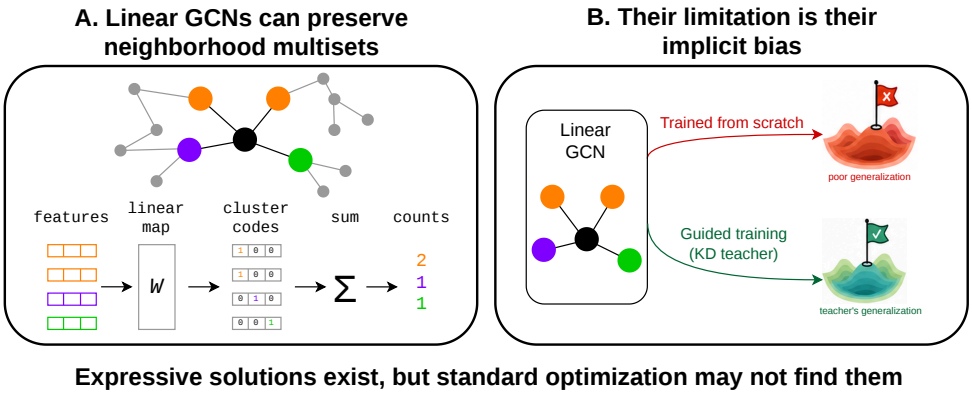


Figure 1. Our main contributions are a demonstration that Linear GCNs can preserve WL-relevant neighborhood multiset information, and that Knowledge Distillation enables the discovery of a better generalizing solution.

on many tasks (Luo et al., 2024; 2025). We therefore view many such architectural choices as introducing useful but indirect biases, whose effects are not necessarily controllable. While they may improve generalization, they do not by themselves explain whether the gains arise from additional expressive power or from a more favorable inductive bias; nor do they clarify how such biases might be steered.

Linear GNNs SGC (Wu et al., 2019) is a simple architecture in which nonlinearities are removed from a GCN (Kipf & Welling, 2017). This is based on their hypothesis that "the nonlinearity between GCN layers is not critical - but the majority of the benefit arises from the local averaging". However, they do not explore or prove the extent of this claim, and they do not consider the necessary power of the predictor at use. Subsequent work has explored the simplification of GNNs through precomputing multi-hop aggregation, which at inference is theoretically equivalent to an explicit collapse of SGC's weight matrices into the predictor. This approach is mainly considered for efficiency purposes, but not for expressive power. SIGN (Frasca et al., 2020) precompute and concatenate features via a variety of linear diffusion operators. They discuss that exploring more expressive or wider local operators is a worthwhile study. GAMLP (Zhang et al., 2022b) propagate both features and labels at multiple depths, and learns node-specific mixtures of hops using recursive attention. HOGA (Deng et al., 2024a) uses instead a gated self-attention module over the precomputed hops. Meanwhile, Rubio-Madrigal & Burkholz (2026) study what is needed for the precomputed multi-hop features—here called FAFs—to be as performant as SOTA GCNs. By just using a well-tuned MLP as predictor, they can match 12 out of 14 benchmarks from Luo et al. (2024). Moreover, they prove that a specific kind of fixed aggregation can be maximally expressive—but does not work well in practice. We instead consider a more general model where we take the (non-collapsed, learnable) linear layers of SGC with an MLP predictor, and we optionally concatenate the layer-wise representations in the spirit of the precomputed features. Our setup thus takes all the elements that make these prior works successful, and explains the extent of their well performance—and also shows some of their limitations.

Implicit bias In overparameterized learning, implicit bias refers to the tendency of a model parameterization, initialization, and optimization procedure to select particular predictors among many solutions with vanishing or comparable training loss. Classical results show that, in separable classification, gradient descent on exponential-tailed losses can make the predictor direction converge to max-margin-type solutions, and related work studies analogous function-space biases in wide neural networks (Soudry et al., 2018; Chizat & Bach, 2020) and parameter level optimization (Saxe et al., 2014; Gunasekar et al., 2018; Pesme et al., 2021; Tarzanagh et al., 2023; Gadhikar et al., 2025; Jacobs et al., 2026a;b). Controlling such biases through explicit regularization (Kobayashi et al., 2024; Jacobs & Burkholz, 2025; Jacobs et al., 2025) is an effective lever to boost generalization and select solutions with favorable properties like sparsity or inference efficiency. In this work, we actively seek to inherit the implicit bias of other model classes while respecting simplicity constraints like linearity through knowledge distillation. In graph learning, the implicit bias perspective is especially relevant because aggregation not only determines what information is available, but also shapes which uses of the input graph are favored during training (Mustafa et al., 2023; Mustafa & Burkholz, 2024b). For instance, GNNs can be biased toward relying on graph structure even when ignoring it would generalize better (Bechler-Speicher et al., 2024; Mustafa & Burkholz, 2024a). In a related direction, training as an MLP and propagating only at inference time can yield strong generalization in some settings (Yang et al., 2023).

Knowledge distillation Knowledge distillation (KD) provides a way to impose a favorable, teacher-induced bias on a student model, making the resulting preference on the solution space no longer purely implicit. Zhang et al. (2022a) show that graph-less MLP students can inherit much of the performance of stronger GNN teachers in settings that have high mutual information between node features and labels. In contrast, we distill into linear (graph-aware) students which, by our theory, can already contain suitable generalizing solutions regardless of the setting. Thus, KD is not used to add expressive power in the model class, but to steer optimization toward solutions that standard supervised training does not naturally select or find. While KD can provide more signal than standard training through the use of soft labels, our paper focuses on establishing that the solution exist within the class, and not how it is reached. Moreover, we note that current graph learning practices do not necessarily control the solution that is reached either.

3. Theory: How Powerful Are Linear GCNs?

The simplicity of linear GCNs has many advantages, including high efficiency and low memory demands, which makes them highly scalable. Concretely, the model admits several equivalent algebraic representations at inference time. Graph propagations and feature transformations can collapse into a single linear operator—with the adjacency matrix absorbed within–enabling parallelized GPU computation. Alternatively, repeatedly applying a shared linear operator—interpretable as a matrix-root factorization of the collapsed form—can reduce memory requirements and yield a sparser operator.

Empirically, they define a surprisingly strong baseline (Wu et al., 2019), in particular if oversmoothing is addressed (Wang et al., 2021). This leads to the hypothesis that more complex variants with nonlinear activation functions are not required in practice.

Our first theoretical contribution is to show that linear GCNs are highly expressive, in particular, if they rely on sum aggregations rather than mean aggregations (generally proposed in the literature). We further argue that residual connections or concatenations of node and neighborhood features contribute significantly to the competitive performance.

Notation The definition of the aggregation process is:

$$\mathbf{X}^{(0)} = \mathbf{X};\ \mathbf{X}_v^{(l+1)} = \sum_{j=1}^{N} a_{vj}\mathbf{X}_j^{(l)}\mathbf{M}^{(l)} + \mathbf{X}_v^{(l)}\mathbf{R}^{(l)} \quad (1)$$

where $\mathbf{A} = (a_{ij})$ is the adjacency matrix of a given graph $\mathcal{G} = (V, E)$ consisting of $N = |V|$ nodes with features $\mathbf{X} \in \mathbb{R}^{N\times p}$. $\mathbf{M}^{(l)}, \mathbf{R}^{(l)} \in \mathbb{R}^{p_l \times p_{l+1}}$ are linear maps for the aggregated neighbor contribution and the self contribution. We use $\|\cdot\|_2$ for the Euclidean norm on vectors and the induced spectral norm on matrices.

Throughout this section, the word cluster should be understood operationally rather than as a data assumption. A cluster denotes a type of node representations whose distinction from other types is to be preserved under aggregation; this may correspond to labels, structural roles, learned latent states, or other signals which need not be known beforehand.

Preservation of multisets We prove that a single layer is sufficient to preserve information about multisets that define node neighborhoods. Implicitly, our choice of $\mathbf{M}^{(l)}$ assumes that neighborhood aggregation only needs to extract discrete information by counting how many neighbors are of a specific node type. For instance, node representations could be clustered and the sum aggregation calculates how many neighbors belong to each of the available clusters. The higher the hidden feature dimension, the more clusters a linear GCN can potentially consider. Note that nodes can also have continuous features and are not limited to discrete ones. Sum aggregations can also make use of continuous information, e.g. how distant neighbor embeddings are from cluster centers, as we show later. For precise guarantees, however, we first focus on the information that is provably preserved.

Theorem 3.1. *Assume that the node features in Eq. (1) are of the form $\mathbf{X}^{(l)} = \mathbf{TC}$, where $\mathbf{C} \in \mathbb{R}^{k\times p_l}$ defines a cluster center matrix, where row i equals center c_i, and each row of $\mathbf{T} \in \mathbb{R}^{n\times k}$ indicates the cluster membership of the respective node. Furthermore, assume $\operatorname{rank}(\mathbf{C}) = k > 0$ and $p_{l+1} \geq k$. Then there exists a linear sum-aggregation layer $\mathbf{M}^{(l)}$ that preserves the full information about neighborhood multisets and node cluster identities. In particular, $\mathbf{M}^{(l)} = [\mathbf{C}^+, \mathbf{0}] \in \mathbb{R}^{p_l\times p_{l+1}}$, where $\mathbf{C}^+$ denotes the Moore–Penrose inverse of $\mathbf{C}$.*

Proof. To preserve information about multisets even after sum aggregation, we would like to map the node features $\mathbf{X}^{(l)}$ to one-hot encoded features $\mathbf{T}^{(l)}$ so that every node of a specific type i with features x_i is mapped to a vector e_i that is the i-th standard basis vector of $\mathbb{R}^{p_{l+1}}$. However, since we only need to represent $k \leq p_{l+1}$ clusters, it suffices to define $\mathbf{T}^{(l)} = [\mathbf{T}, \mathbf{0}]$. It follows immediately from the construction that $\mathbf{M}^{(l)} = [\mathbf{C}^+, \mathbf{0}]$ maps $\mathbf{X}^{(l)}$ to $\mathbf{T}^{(l)}$, since $\mathbf{X}^{(l)}\mathbf{M}^{(l)} = \mathbf{TC}[\mathbf{C}^+, \mathbf{0}] = \mathbf{T}[\mathbf{I}, \mathbf{0}] = \mathbf{T}^{(l)}$.

Last but not least, the skip connection $\mathbf{X}_v^{(l)}$ needs to be represented in the same coordinate system than the neighbor representations. Thus, we can instantiate the residual term $\mathbf{R}^{(l)} := (1+\epsilon)\mathbf{M}^{(l)} = (1+\epsilon)[\mathbf{C}^+, \mathbf{0}]$, i.e., $\mathbf{X}_v^{(l+1)} = \sum_{j=1}^{n} a_{vj}\mathbf{X}_j^{(l)}\mathbf{M}^{(l)} + (1+\epsilon)\mathbf{X}_v^{(l)}\mathbf{M}^{(l)}$.

Since $\mathbf{X}^{(l)} = \mathbf{TC}$, we have, for $i \leq k$, $X_{vi}^{(l+1)} =$

$\sum_{j=1}^{n} a_{vj}T_{ji} + (1+\epsilon)T_{vi}$. For unweighted sum aggregation, this becomes $X_{vi}^{(l+1)} = n_i(v) + (1+\epsilon)T_{vi}$, where $n_i(v) \in \mathbb{N}$ is the number of neighbors of v belonging to cluster i. Since $T_{vi} \in \{0,1\}$ and ϵ is irrational, this value uniquely determines both $n_i(v)$ and T_{vi}. Hence the layer preserves the neighborhood multiset counts together with the node's own cluster identity. Since $\mathbf{X}_v^{(l)} = \mathbf{T}_v\mathbf{C}$, preserving $\mathbf{T}_v$ also preserves the original node feature $\mathbf{X}_v^{(l)}$.

More generally, if we want to map the features $\mathbf{X}^{(l)}$ to target representations $\mathbf{T}^{(l)}$, we have to solve the following system of linear equations: $\mathbf{T}^{(l)} = \mathbf{X}^{(l)}\mathbf{M}^{(l)}$. If a solution exists, it is given by $\mathbf{M}^{(l)} = (\mathbf{X}^{(l)})^{+}\mathbf{T}^{(l)}$, where defines $(\mathbf{X}^{(l)})^{+}$ the Moore-Penrose inverse of the feature matrix. $\mathbf{X}^{(l)}\mathbf{M}^{(l)}$ will generally minimize the L_2-distance to the target $\mathbf{T}^{(l)}$, which can also preserve most of the relevant information if the target is chosen wisely.

To derive the conditions of exact equality, we utilize the singular value decomposition of the features $\mathbf{X}^{(l)} = \mathbf{U}\mathbf{D}\mathbf{V}^{\top}$, where the diagonal matrix is of the form $\mathbf{D} = [\tilde{\mathbf{D}}, \mathbf{0}]^{\top}$, where $\tilde{\mathbf{D}}$ is a k-dimensional diagonal matrix. Accordingly, we have $\mathbf{U}^{\top}\mathbf{T}^{(l)} = [\tilde{\mathbf{D}}\mathbf{M}^{(l)}\mathbf{V}^{\top}, \mathbf{0}]^{\top}$. While $\mathbf{M}^{(l)}$ has sufficient degrees of freedom to solve the k upper equations, whether the lower set $\mathbf{P}\mathbf{U}^{\top}\mathbf{T}^{(l)}$ can be fulfilled is determined by the target alignment $\mathbf{P}\mathbf{U}^{\top}\mathbf{T}^{(l)} = \mathbf{0}$, where $\mathbf{P}$ denotes a projection to the last $n-k$ rows. □

If the hidden feature dimension p_{l+1} is large enough and the relevant nodes are separable (so that the cluster centers defined by the rows of $\mathbf{C}$ are linearly independent), we can map nodes to any target node embedding, including one-hot encodings of node cluster memberships. These are special, as sum aggregations of neighbors with one-hot encoded features perform simple counting of cluster members (Xu et al., 2019; Rubio-Madrigal & Burkholz, 2026). In contrast, mean aggregation, as implemented by Wu et al. (2019); Wang et al. (2021), would lose information about exact numbers of neighbors in favor of fractions—but usually still works well in practice.

To judge the significance of the above result, we have to ask how restrictive it is to describe the information to be preserved through such clusters. In other words, for a given task, can the relevant distinctions be represented by sufficiently separable classes, and does preserving their neighborhood counts transport enough information? First and foremost, in principle it could cover information about all nodes if we make the feature dimension large enough, for instance, $p_{l+1} = N$ so that every node has their own cluster, effectively corresponding to a node positional encoding. This is likely not very meaningful for solving a task, as learning seeks to compress information by assigning the right nodes to the same cluster. Furthermore, note that the same construction of the linear layer in the proof would also work even if the nodes in one cluster do not have identical but only similar node features.

Theorem 3.2. *Assume that the node features in Eq. (1) satisfy $\|\mathbf{X}_v^{(l)} - \mathbf{T}_v\mathbf{C}\|_2 \leq \epsilon'$ for all nodes $v \in V$, where $\mathbf{C}$ and $\mathbf{T}$ are defined as in Thm. 3.1. Assume further that $0 < \epsilon' < \frac{\sigma_{\min}(\mathbf{C})}{\sqrt{2}}$, where $\sigma_{\min}(\mathbf{C})$ denotes the smallest nonzero singular value of $\mathbf{C}$. Then the linear layer from Thm. 3.1 approximately preserves neighborhood multiset information.*

Proof. Since $\mathbf{C}$ has full row rank, $\|\mathbf{C}^{+}\|_2 = \frac{1}{\sigma_{\min}(\mathbf{C})}$. Let $\mathbf{M}^{(l)} = [\mathbf{C}^{+}, \mathbf{0}]$. If node v belongs to cluster i, then $\mathbf{T}_v = e_i$. Hence $\|\mathbf{X}_v^{(l)}\mathbf{M}^{(l)} - \mathbf{T}_v^{(l)}\|_2 \leq \|\mathbf{X}_v^{(l)} - \mathbf{T}_v\mathbf{C}\|_2\|\mathbf{C}^{+}\|_2 \leq \frac{\epsilon'}{\sigma_{\min}(\mathbf{C})}$. Define $\delta := \frac{\epsilon'}{\sigma_{\min}(\mathbf{C})} < 1/\sqrt{2}$. For any other cluster $j \neq i$, $\|\mathbf{X}_v^{(l)}\mathbf{M}^{(l)} - e_j\|_2 \geq \|e_i - e_j\|_2 - \|\mathbf{X}_v^{(l)}\mathbf{M}^{(l)} - e_i\|_2 \geq \sqrt{2} - \delta > \delta$. Thus $\mathbf{X}_v^{(l)}\mathbf{M}^{(l)}$ remains closer to the correct cluster code e_i than to any other cluster code.

Moreover, the aggregation error for a node u is bounded by $\left\|\sum_{v=1}^{N} a_{uv}\mathbf{X}_v^{(l)}\mathbf{M}^{(l)} - \sum_{v=1}^{N} a_{uv}\mathbf{T}_v^{(l)}\right\|_2 \leq \sum_{v=1}^{N} |a_{uv}| \frac{\epsilon'}{\sigma_{\min}(\mathbf{C})}$. Therefore, the aggregated neighborhood counts are preserved up to an error controlled by the feature perturbation, the conditioning of $\mathbf{C}$, and the weighted degree of the node. □

A complementary spectral perspective is given by Kanatsoulis & Ribeiro (2024), who argue that useful graph structure can appear through spectral representations of well-behaved graphs. This provides one natural mechanism by which separable node clusters may arise. Our result abstracts away from the origin of these clusters: whether they come from spectral structure, learned features, or other positional information, once the node representations are sufficiently separable, a linear transformation followed by sum aggregation can preserve the corresponding neighborhood multiset counts.

The linear layer above might even transport more valuable information than only the cluster membership. The neighborhood aggregation also takes the distance of the node feature to the cluster center into account. The ability to identify hidden clustering has also been linked to good out-of-distribution robustness (Reddy et al., 2026a;b) and might thus also lead to well generalizing and robust solutions.

Expressive power of non-linear GCNs The ability of a linear layer to perform clustering and also collect information about the cluster membership of neighbors is the requirement to achieve expressive power at the level of a Weisfeller-Lehmann tests (Xu et al., 2019). The more complex GIN architecture, consisting of nonlinear MLP transformations before each sum aggregation layer, was originally

proposed for this purpose, questioning the expressiveness of regular GCNs. It is a trivial corollary or our previous results that GCNs with ReLU activation functions achieve at least the same expressive power as linear GCNs.

4. Methodology

The results above should be interpreted as existence results of generalizing solutions within the model class of linear graph networks. They show that linear convolutions can preserve the multiset information needed by expressive message-passing architectures. However, this does not imply that empirical risk minimization will naturally find the corresponding predictor. In practice, a failure of a linear GCN may therefore have two explanations: either the representation lacks task-relevant information, or the training procedure selects a poorly generalizing solution among many expressible ones.

We treat knowledge distillation as a controlled way to test whether generalizing solutions already exist inside linear student classes. The architectural changes may be useful for finding or biasing toward a solution, but not for representing it at inference time.

Knowledge distillation Following Zhang et al. (2022a), we generate soft targets z_v for each node v with a teacher GNN that is the best classic message-passing model in the respective dataset. We train a student model with true labels y_z and with z_v, in the following manner:

$$\mathcal{L} = \sum_{v \in \mathcal{V}^{\text{train}}} \mathcal{L}_{\text{label}}(\hat{y}_v, y_v) + \lambda \sum_{v \in \mathcal{V}} \mathcal{L}_{\text{teacher}}(\hat{y}_v, z_v) \quad (2)$$

where λ is a hyperparameter and we generally use $\lambda = 0.3$; $\mathcal{L}_{\text{label}}$ is RMSE for regression tasks and CE for classification; $\mathcal{L}_{\text{teacher}}$ is the KL-divergence.

Datasets We evaluate on the 14 medium-sized node classification datasets from (Luo et al., 2024). We also consider 6 of the recently proposed node benchmark GraphLand (Bazhenov et al., 2025) under RL split: 3 node classification and 3 node regression tasks. Finally, we use the Paris dataset from the long-range benchmark City-Networks (Liang et al., 2026), which was proposed as a 16-hop long-range benchmark.

Teachers For each benchmark, we re-implement one of the best-performing reported methods and, per seed or split, use the model with the best validation performance as the teacher. In Luo et al. (2024), all classic GNNs perform competitively, so we use GCN (Kipf & Welling, 2017). Note that these match the Graph Transformers and heterophily-aware models. For GraphLand (Bazhenov et al., 2025), where GAT and GT tend to perform equally best, we use GAT (Veličković et al., 2018). As hyperparameters are not directly reported, we re-run their hyperparameter tuning loop for our own architectural setup. For City-Networks (Liang et al., 2026), where GraphSAGE performs slightly better than other classical GNNs, we use GraphSAGE (Hamilton et al., 2017). This gives us a set of strong benchmark-specific teachers drawn from different classical GNN architectures. In general, we construct them following the architectures and environment from Luo et al. (2024). More details of the implementation can be found in Appendix A.

Student model Given a GCN as described above, we build a Linear GCN (LGCN) with the same architecture but no ReLUs before the prediction head. While sum aggregation is theoretically more expressive than mean aggregation, in practice we do not observe a significant performance gap, so we report the more commonly used mean—note there are normalization layers in between. Unless otherwise specified, students inherit the teacher's training hyperparameters, with occasional adjustments to width and number of epochs; all hyperparameters are shown in Tables 4, 5. We also compare to an MLP student, coined GLNN by Zhang et al. (2022a). We use the MLP hyperparameters reported by Rubio-Madrigal & Burkholz (2026) when available.

We furthermore add an architectural change to the LGCN: we take the layer-wise node representations and concatenate them into the prediction head (-C), instead of using only the last layer. This avoids having to deal with over-smoothing and follows a similar pattern than other simplified architectures (Frasca et al., 2020; Zhang et al., 2022b; Deng et al., 2024a; Rubio-Madrigal & Burkholz, 2026). Empirically, this seems to aid in training, but student models without it that are given enough training time perform equally well.

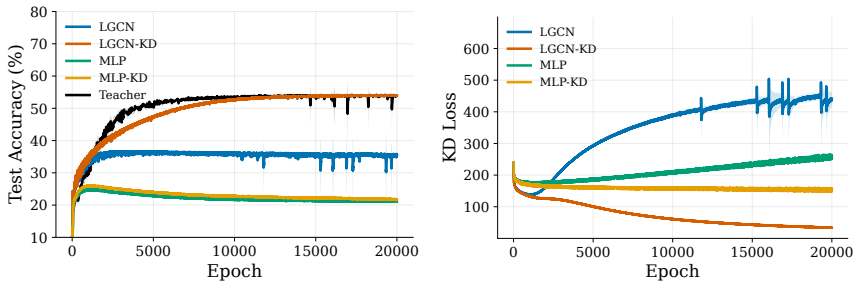
5. Experiments

We present test performance in Table 1 for benchmarks from Luo et al. (2024), Table 2 for Bazhenov et al. (2025), and Table 3 for Liang et al. (2026). We **bolden** the best results, as well as the all results within error (calculated with RSS over the corresponding number of runs per dataset).

For all datasets, there exists a linear GCN student model with layer-wise concatenated embeddings (LGCN-C-KD) such that its test performance is equal or greater than the respective SOTA teacher.

We further report the following observations:

- Concatenating layer-wise embeddings into the prediction head (-C) is usually better performing, especially for the longer-range datasets; this seems to aid in fitting the data faster rather than being due to information loss.



(a) Test accuracy *(b)* Knowledge distillation loss

Figure 2. Averaged performance measurements for LGCN and MLP with and without knowledge distillation (KD) for the Paris dataset. We report test accuracy and the KD loss ($\mathcal{L}_{\text{teacher}}$), which is only enforced on the KD models and merely tracked on the non-distilled ones.

Table 1. Test performance comparison between a GCN teacher from (Luo et al., 2024) and linear GCN (LGCN), with and without layer-wise concatenation into the (nonlinear) prediction head (-C), with and without knowledge distillation (KD).

Dataset:	A-Computer	A-Photo	A-Ratings	Chameleon	Citeseer	Coauthor-CS	Coauthor-Physics
GCN (teacher)	92.76 ± 0.10	95.45 ± 0.23	**54.12 ± 0.55**	**48.25 ± 2.22**	**69.84 ± 1.13**	95.24 ± 0.06	97.18 ± 0.04
MLP	88.01 ± 0.15	93.90 ± 0.04	48.90 ± 0.50	41.71 ± 1.83	55.12 ± 0.70	93.78 ± 0.12	95.89 ± 0.05
LGCN	92.07 ± 0.00	93.66 ± 0.58	49.84 ± 0.67	**47.36 ± 2.58**	**70.20 ± 0.14**	95.13 ± 0.06	97.13 ± 0.01
LGCN-C	92.36 ± 0.16	95.27 ± 0.19	51.51 ± 0.96	46.62 ± 2.87	**69.40 ± 0.37**	95.31 ± 0.03	97.11 ± 0.03
MLP-KD	89.07 ± 0.21	**95.90 ± 0.04**	52.20 ± 0.88	**50.57 ± 2.29**	**70.04 ± 2.63**	**95.64 ± 0.04**	**97.29 ± 0.07**
LGCN-KD	92.65 ± 0.16	95.64 ± 0.15	53.15 ± 0.38	**47.97 ± 2.69**	**70.80 ± 1.45**	**95.55 ± 0.15**	**97.26 ± 0.03**
LGCN-C-KD	**93.07 ± 0.09**	95.73 ± 0.08	**54.49 ± 0.59**	**49.70 ± 2.09**	**70.64 ± 1.81**	95.53 ± 0.07	**97.26 ± 0.04**

Dataset:	Cora	Minesweeper	Pubmed	Questions	Roman-Empire	Squirrel	WikiCS
GCN (teacher)	81.68 ± 0.61	**97.07 ± 0.33**	79.68 ± 0.33	**79.09 ± 0.40**	**90.56 ± 0.28**	**44.34 ± 1.11**	**81.44 ± 0.19**
MLP	62.64 ± 0.62	52.19 ± 0.60	71.44 ± 0.65	72.12 ± 0.91	66.38 ± 0.16	41.20 ± 1.31	74.86 ± 0.19
LGCN	81.00 ± 0.37	90.50 ± 0.86	80.32 ± 0.88	78.20 ± 0.44	81.34 ± 0.33	**43.24 ± 1.61**	81.17 ± 0.15
LGCN-C	80.64 ± 0.43	90.43 ± 1.16	79.44 ± 0.38	78.08 ± 0.58	83.44 ± 0.59	39.44 ± 1.11	81.05 ± 0.33
MLP-KD	74.80 ± 1.12	52.20 ± 0.60	**81.96 ± 0.64**	73.62 ± 1.24	66.60 ± 0.13	41.30 ± 1.12	75.94 ± 0.26
LGCN-KD	**82.76 ± 0.57**	**96.78 ± 0.19**	80.92 ± 0.27	**78.65 ± 0.62**	82.54 ± 0.32	**44.04 ± 1.32**	**81.73 ± 0.24**
LGCN-C-KD	**82.84 ± 0.22**	**97.00 ± 0.27**	80.76 ± 0.62	**78.80 ± 0.88**	**90.20 ± 0.30**	**43.21 ± 1.72**	**81.94 ± 0.46**

- The student models can even be strictly better than the teachers, as the teachers' best validation epoch is not usually an interpolating one. Meanwhile, the student models can train towards zero loss while being regularized to the better-generalizing solution, avoiding overfitting at the same time.
- The GCN and GAT teachers train for 2,500 epochs, while we train some students for 10,000 or 20,000. With a more tuned trainability, fewer epochs or narrower widths could be possible, as the combined loss can be reduced by increasing the speed of training.
- Long-range datasets such as Minesweeper and Paris require more training time and width. (We set it to 20,000 and 1024, respectively.) Distilling these architectures necessitates a long and stable time in the interpolation region, allowing the KD regularization to move the model to a well-generalizing solution. In Fig. 2, we can see the averaged test accuracy curve across epochs, as well as the KD loss, which could benefit from more epochs.

6. Conclusions

We revisit a basic question in graph learning: whether the success of modern GNNs should be attributed primarily to increased architectural expressive power or to the inductive biases imposed by architecture and training. Our results suggest that expressive power alone is not the main bottleneck. We show theoretically that linear graph convolutions with sum aggregation can preserve any neighborhood multiset information whenever node representations are sufficiently separable, thereby matching the multiset-preservation role

Table 2. Test performance comparison between a GAT teacher from GraphLand (Bazhenov et al., 2025) and linear GCN (LGCN), with and without layer-wise concatenation into the (nonlinear) prediction head (-C), with and without knowledge distillation (KD).

Dataset:	artnet-exp	artnet-views	city-reviews	city-roads-L	city-roads-M	tolokers-2
GAT (teacher)	**45.47 ± 0.33**	53.26 ± 0.68	77.25 ± 0.16	52.66 ± 0.49	**58.68 ± 0.53**	55.78 ± 0.52
MLP	35.41 ± 0.34	22.76 ± 0.68	65.86 ± 0.24	45.38 ± 0.55	51.72 ± 1.80	38.19 ± 2.19
LGCN	42.95 ± 0.33	51.88 ± 0.39	77.36 ± 0.09	50.40 ± 0.29	56.99 ± 0.33	51.51 ± 0.54
LGCN-C	42.75 ± 0.31	53.50 ± 0.28	77.47 ± 0.06	50.24 ± 0.32	57.32 ± 0.28	53.74 ± 0.81
MLP-KD	35.98 ± 0.61	41.32 ± 0.71	73.56 ± 0.17	47.38 ± 0.15	53.48 ± 0.44	40.65 ± 0.39
LGCN-KD	45.10 ± 0.29	**56.83 ± 0.45**	**77.79 ± 0.06**	**52.98 ± 0.24**	**58.76 ± 0.47**	54.52 ± 1.18
LGCN-C-KD	**45.60 ± 0.33**	**56.81 ± 0.44**	**77.67 ± 0.22**	**53.30 ± 0.40**	**59.19 ± 0.35**	**56.55 ± 0.38**

Table 3. Test performance comparison between a GAT teacher on Paris from City-Networks (Liang et al., 2026) and linear GCN (LGCN), with and without knowledge distillation (KD).

Method	SAGE (teacher)	MLP	LGCN	MLP-KD	LGCN-KD
Paris	**54.27 ± 0.17**	24.71 ± 0.09	36.80 ± 0.14	25.97 ± 0.05	53.97 ± 0.20

usually associated with WL-expressive architectures. Our theory proves a prominent empirical hypothesis found in literature for the simplification of GNNs (Wu et al., 2019; Frasca et al., 2020; Wang et al., 2021) and downscales the urgency for increasingly more expressive architectures (Xu et al., 2019). It also theoretically provides an explanation for the recent empirical success of well-tuned or augmented classical GNNs in exchange of more complex architectures (Luo et al., 2024; 2025).

Empirically, we use knowledge distillation as a diagnostic for separating representational capacity from training bias in transductive settings. Distilled linear GCNs matched or exceeded strong GNN teachers across standard node classification benchmarks (Luo et al., 2024), recent industrial GraphLand tasks (Bazhenov et al., 2025), and a long-range City-Networks benchmark (Liang et al., 2026). Importantly, this does not hold equally for graph-agnostic MLP students (Zhang et al., 2022a). In contrast, the same linear architectures trained from scratch often generalize substantially worse. This suggests that the relevant solutions exist inside the linear model class, but are not reliably selected by ordinary supervised training.

These findings alter the interpretation of numerous architectural improvements in graph learning. More complex architectures may be beneficial not because additional expressive power is strictly required at inference time, but because they impose useful biases that make good solutions easier to find. However, it is common in the literature to accept the benefits of an architectural improvement based on the existence of better solutions. From our perspective, progress in graph learning should not only address how to expand the space of representable functions, but also how to control model selection within an already expressive domain.

Our work also leaves several open directions. First, knowledge distillation identifies the existence of good linear solutions, but it does not by itself explain how to find them without a teacher. Second, the long training times and increased width required on harder long-range datasets suggest that optimization remains a central challenge. Finally, a more principled understanding of graph-specific implicit bias is needed: what biases are induced by different aggregation schemes, and when do they improve generalization? We hope that distinguishing expressive power from implicit bias provides a useful framework for answering these questions and for designing more efficient and more reliable graph learning methods.

Acknowledgments and Disclosure of Funding

We thank Tom Jacobs and Nimrah Mustafa for helpful discussions and input on the related work. We also gratefully acknowledge funding from the European Research Council (ERC) under the Horizon Europe Framework Programme (HORIZON) for proposal number 101116395 SPARSE-ML.

Impact Statement

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

LLM Statement

To improve fluency of the text sentence level, editing has been done using large language models.

References

Bazhenov, G., Platonov, O., and Prokhorenkova, L. Graphland: Evaluating graph machine learning models on diverse industrial data. In *The Fourth Learning on Graphs Conference*, 2025. URL https://openreview.net/forum?id=DDpToPGYOu.

Bechler-Speicher, M., Amos, I., Gilad-Bachrach, R., and Globerson, A. Graph neural networks use graphs when they shouldn't. In *International Conference on Machine Learning (ICML)*, 2024. URL https://arxiv.org/abs/2309.04332.

Bronstein, M. M., Bruna, J., Cohen, T., and Velickovic, P. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *CoRR*, abs/2104.13478, 2021. URL https://arxiv.org/abs/2104.13478.

Burkholz, R., Quackenbush, J., and Bojar, D. Using graph convolutional neural networks to learn a representation for glycans. *Cell Reports*, 35(11), 2021.

Chen, J., Gao, K., Li, G., and He, K. NAGphormer: A tokenized graph transformer for node classification in large graphs. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=8KYeilT3Ow.

Chien, E., Peng, J., Li, P., and Milenkovic, O. Adaptive universal generalized PageRank graph neural network. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=n6jl7fLxrP.

Chizat, L. and Bach, F. Implicit bias of gradient descent for wide two-layer neural networks trained with the logistic loss. In Abernethy, J. and Agarwal, S. (eds.), *Proceedings of Thirty Third Conference on Learning Theory*, volume 125 of *Proceedings of Machine Learning Research*, pp. 1305–1338. PMLR, 09–12 Jul 2020. URL https://proceedings.mlr.press/v125/chizat20a.html.

Deng, C., Yue, Z., Yu, C., Sarar, G., Carey, R., Jain, R., and Zhang, Z. Less is more: Hop-wise graph attention for scalable and generalizable learning on circuits. In *Proceedings of the 61st ACM/IEEE Design Automation Conference*, DAC '24, New York, NY, USA, 2024a. Association for Computing Machinery. ISBN 9798400706011. doi: 10.1145/3649329.3657386. URL https://doi.org/10.1145/3649329.3657386.

Deng, C., Yue, Z., and Zhang, Z. Polynormer: Polynomial-expressive graph transformer in linear time. In *The Twelfth International Conference on Learning Representations*, 2024b. URL https://openreview.net/forum?id=hmv1LpNfXa.

Errica, F., Christiansen, H., Zaverkin, V., Maruyama, T., Niepert, M., and Alesiani, F. Adaptive message passing: A general framework to mitigate oversmoothing, oversquashing, and underreaching. In *International Conference on Machine Learning (ICML)*, 2025. URL https://arxiv.org/abs/2312.16560.

Fan, W., Ma, Y., Li, Q., He, Y., Zhao, E., Tang, J., and Yin, D. Graph neural networks for social recommendation. In *The World Wide Web Conference*, WWW '19, pp. 417–426, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450366748. doi: 10.1145/3308558.3313488. URL https://doi.org/10.1145/3308558.3313488.

Finkelshtein, B., Huang, X., Bronstein, M., and İsmail İlkan Ceylan. Cooperative graph neural networks. In *International Conference on Machine Learning (ICML)*, 2024. URL https://arxiv.org/abs/2310.01267.

Frasca, F., Rossi, E., Eynard, D., Chamberlain, B., Bronstein, M., and Monti, F. Sign: Scalable inception graph neural networks. In *ICML 2020 Workshop on Graph Representation Learning and Beyond*, 2020.

Gadhikar, A., Jacobs, T., Zhou, C., and Burkholz, R. Sign-in to the lottery: Reparameterizing sparse training. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=iwKT7MEZZw.

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., and Dahl, G. E. Neural message passing for quantum chemistry, 2017a. URL https://arxiv.org/abs/1704.01212.

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., and Dahl, G. E. Neural message passing for quantum chemistry. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, pp. 1263–1272. JMLR.org, 2017b.

Gori, M., Monfardini, G., and Scarselli, F. A new model for learning in graph domains. In *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, volume 2, pp. 729–734 vol. 2, 2005. doi: 10.1109/IJCNN.2005.1555942.

Gotovos, A., Burkholz, R., Quackenbush, J., and Jegelka, S. Scaling up continuous-time markov chains helps resolve underspecification. *Advances in Neural Information Processing Systems*, 34:14580–14592, 2021.

Gunasekar, S., Lee, J., Soudry, D., and Srebro, N. Characterizing implicit bias in terms of optimization geometry. In Dy, J. and Krause, A. (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1832–1841. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/gunasekar18a.html.

Hamilton, W., Ying, Z., and Leskovec, J. Inductive representation learning on large graphs. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/5dd9db5e033da9c6fb5ba83c7a7ebea9-Paper.pdf.

Hossain, I., Fanfani, V., Fischer, J., Quackenbush, J., and Burkholz, R. Biologically informed neuralodes for genome-wide regulatory dynamics. *Genome Biology*, 25(1):1–29, 2024a.

Hossain, I., Fischer, J., Burkholz, R., and Quackenbush, J. Pruning neural network models for gene regulatory dynamics using data and domain knowledge. In *Neural Information Processing Systems*, 2024b.

Jacobs, T. and Burkholz, R. Mask in the mirror: Implicit sparsification. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=U47ymTS3ut.

Jacobs, T., Zhou, C., and Burkholz, R. Mirror, mirror of the flow: How does regularization shape implicit bias? In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=MLiR9LS5PW.

Jacobs, T., Gadhikar, A., Rubio-Madrigal, C., and Burkholz, R. Hyperbolic aware minimization: Implicit bias for sparsity. In *The Fourteenth International Conference on Learning Representations*, 2026a. URL https://openreview.net/forum?id=XKB5Hu0ACY.

Jacobs, T., Zhou, C., and Burkholz, R. Never saddle for reparameterized steepest descent as mirror flow. In *The Fourteenth International Conference on Learning Representations*, 2026b. URL https://openreview.net/forum?id=YgudIlQ9nC.

Jamadandi, A., Rubio-Madrigal, C., and Burkholz, R. Spectral graph pruning against over-squashing and over-smoothing. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=EMkrwJY2de.

Kanatsoulis, C. and Ribeiro, A. Counting graph substructures with graph neural networks. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=qaJxPhkYtD.

Kipf, T. N. and Welling, M. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=SJU4ayYgl.

Kobayashi, S., Akram, Y., and Oswald, J. V. Weight decay induces low-rank attention layers. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=oDeqjIM9Sk.

Kong, K., Chen, J., Kirchenbauer, J., Ni, R., Bruss, C. B., and Goldstein, T. GOAT: A global transformer on large-scale graphs. In Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., and Scarlett, J. (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 17375–17390. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/kong23a.html.

Li, G., Müller, M., Thabet, A., and Ghanem, B. Deepgcns: Can gcns go as deep as cnns? In *The IEEE International Conference on Computer Vision (ICCV)*, 2019.

Li, T. W., Mei, Q., and Ma, J. A metadata-driven approach to understand graph neural networks. In *Advances in Neural Information Processing Systems*, 2023. URL https://arxiv.org/abs/2310.19263.

Li, X., Zhu, R., Cheng, Y., Shan, C., Luo, S., Li, D., and Qian, W. Finding global homophily in graph neural networks when meeting heterophily. In Chaudhuri, K., Jegelka, S., Song, L., Szepesvari, C., Niu, G., and Sabato, S. (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 13242–13256. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/li22ad.html.

Liang, H., de Ocáriz Borde, H. S., Sripathmanathan, B., Bronstein, M. M., and Dong, X. Towards quantifying long-range interactions in graph machine learning: a large graph dataset and a measurement. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=fylMiUmg39.

Luan, S., Hua, C., Xu, M., Lu, Q., Zhu, J., Chang, X.-W., Fu, J., Leskovec, J., and Precup, D. When do graph neural

networks help with node classification? investigating the impact of homophily principle on node distinguishability. 2024. URL https://arxiv.org/abs/2304.14274.

Luo, Y., Shi, L., and Wu, X.-M. Classic GNNs are strong baselines: Reassessing GNNs for node classification. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024. URL https://openreview.net/forum?id=xkljKdGe4E.

Luo, Y., Shi, L., and Wu, X.-M. Can classic GNNs be strong baselines for graph-level tasks? simple architectures meet excellence. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=ZH7YgIZ3DF.

Maurya, S. K., Liu, X., and Murata, T. Simplifying approach to node classification in graph neural networks. *Journal of Computational Science*, 62:101695, 2022. ISSN 1877-7503. doi: https://doi.org/10.1016/j.jocs.2022.101695. URL https://www.sciencedirect.com/science/article/pii/S1877750322000990.

McCallum, A. K., Nigam, K., Rennie, J., and Seymore, K. Automating the construction of internet portals with machine learning. *Information Retrieval*, 3(2):127–163, 2000. doi: 10.1023/A:1009953814988. URL https://doi.org/10.1023/A:1009953814988.

Mernyei, P. and Cangea, C. Wiki-cs: A wikipedia-based benchmark for graph neural networks. In *ICML Workshop: Graph Representation Learning and Beyond*, 2020. URL https://arxiv.org/abs/2007.02901.

Mustafa, N. and Burkholz, R. Dynamic rescaling for training GNNs. In *Neural Information Processing Systems (NeurIPS)*, 2024a.

Mustafa, N. and Burkholz, R. Gate: How to keep out intrusive neighbors. In *International Conference on Machine Learning (ICML)*, 2024b. URL https://arxiv.org/abs/2406.00418.

Mustafa, N., Bojchevski, A., and Burkholz, R. Are GATs out of balance? In *Neural Information Processing Systems (NeurIPS)*, 2023.

Namata, G., London, B., Getoor, L., and Huang, B. Query-driven active surveying for collective classification. In *Proceedings of the Workshop on Mining and Learning with Graphs*, 2012.

Oono, K. and Suzuki, T. Graph neural networks exponentially lose expressive power for node classification. In *International Conference on Learning Representations*, 2020.

Pesme, S., Pillaud-Vivien, L., and Flammarion, N. Implicit bias of SGD for diagonal linear networks: a provable benefit of stochasticity. In Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=vvi7KqHQiA.

Platonov, O., Kuznedelev, D., Babenko, A., and Prokhorenkova, L. Characterizing graph datasets for node classification: Homophily-heterophily dichotomy and beyond, 2023a. URL https://arxiv.org/abs/2209.06177.

Platonov, O., Kuznedelev, D., Diskin, M., Babenko, A., and Prokhorenkova, L. A critical look at evaluation of gnns under heterophily: Are we really making progress? In *The Eleventh International Conference on Learning Representations*, 2023b.

Rampášek, L., Galkin, M., Dwivedi, V. P., Luu, A. T., Wolf, G., and Beaini, D. Recipe for a general, powerful, scalable graph transformer. In Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., and Oh, A. (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 14501–14515. Curran Associates, Inc., 2022.

Reddy, A. G., Rubio-Madrigal, C., Burkholz, R., and Muandet, K. When shift happens - confounding is to blame. *International Conference on Learning Representations (ICLR)*, 2026a.

Reddy, A. G., Verma, R., Rubio-Madrigal, C., Muandet, K., and Burkholz, R. Boosting for predictive sufficiency. In *International Conference on Learning Representations (ICLR)*, 2026b.

Rozemberczki, B., Allen, C., and Sarkar, R. Multi-Scale attributed node embedding. *Journal of Complex Networks*, 9(2), 05 2021. ISSN 2051-1329. doi: 10.1093/comnet/cnab014. URL https://doi.org/10.1093/comnet/cnab014.

Rubio-Madrigal, C. and Burkholz, R. Fixed aggregation features can rival GNNs. In *Forty-third International Conference on Machine Learning*, 2026.

Rubio-Madrigal, C., Jamadandi, A., and Burkholz, R. GNNs getting ComFy: Community and feature similarity guided rewiring. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=g6v09VxgFw.

Sanchez-Gonzalez, A., Godwin, J., Pfaff, T., Ying, R., Leskovec, J., and Battaglia, P. W. Learning to simulate complex physics with graph networks. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org, 2020.

Saxe, A. M., McClelland, J. L., and Ganguli, S. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks, 2014. URL https://arxiv.org/abs/1312.6120.

Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., and Monfardini, G. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2009. doi: 10.1109/TNN.2008.2005605.

Schlichtkrull, M., Kipf, T. N., Bloem, P., van den Berg, R., Titov, I., and Welling, M. Modeling relational data with graph convolutional networks. In Gangemi, A., Navigli, R., Vidal, M.-E., Hitzler, P., Troncy, R., Hollink, L., Tordai, A., and Alam, M. (eds.), *The Semantic Web*, pp. 593–607, Cham, 2018. Springer International Publishing. ISBN 978-3-319-93417-4.

Sen, P., Namata, G., Bilgic, M., Getoor, L., Galligher, B., and Eliassi-Rad, T. Collective classification in network data. *AI Magazine*, 29 (3):93, Sep. 2008. doi: 10.1609/aimag.v29i3.2157. URL https://ojs.aaai.org/index.php/aimagazine/article/view/2157.

Shchur, O., Mumme, M., Bojchevski, A., and Günnemann, S. Pitfalls of graph neural network evaluation. *Relational Representation Learning Workshop, NeurIPS*, 2018.

Shirzad, H., Velingker, A., Venkatachalam, B., Sutherland, D. J., and Sinop, A. K. Exphormer: Sparse transformers for graphs. In Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., and Scarlett, J. (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 31613–31632. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/shirzad23a.html.

Soudry, D., Hoffer, E., and Srebro, N. The implicit bias of gradient descent on separable data. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=r1q7n9gAb.

Tarzanagh, D. A., Li, Y., Thrampoulidis, C., and Oymak, S. Transformers as support vector machines. In *NeurIPS 2023 Workshop on Mathematics of Modern Machine Learning*, 2023. URL https://openreview.net/forum?id=gLwzzmh79K.

Topping, J., Giovanni, F. D., Chamberlain, B. P., Dong, X., and Bronstein, M. M. Understanding over-squashing and bottlenecks on graphs via curvature. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=7UmjRGzp-A.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., and Bengio, Y. Graph attention networks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rJXMpikCZ.

Wang, Y., Wang, Y., Yang, J., and Lin, Z. Dissecting the diffusion process in linear graph convolutional networks. In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 5758–5769. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/2d95666e2649fcfc6e3af75e09f5adb9-Paper.pdf.

Weighill, D., Guebila, M. B., Lopes-Ramos, C., Glass, K., Quackenbush, J., Platig, J., and Burkholz, R. Gene regulatory network inference as relaxed graph matching. *AAAI Conference on Artificial Intelligence*, 2021.

Wu, F., Souza, A., Zhang, T., Fifty, C., Yu, T., and Weinberger, K. Simplifying graph convolutional networks. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 6861–6871. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/wu19e.html.

Wu, Q., Zhao, W., Li, Z., Wipf, D. P., and Yan, J. Nodeformer: A scalable graph structure learning transformer for node classification. In Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., and Oh, A. (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 27387–27401. Curran Associates, Inc., 2022.

Wu, Q., Zhao, W., Yang, C., Zhang, H., Nie, F., Jiang, H., Bian, Y., and Yan, J. SGFormer: Simplifying and empowering transformers for large-graph representations. In Oh, A., Naumann, T., Globerson, A., Saenko, K., Hardt, M., and Levine, S. (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 64753–64773. Curran Associates, Inc., 2023.

Xu, K., Hu, W., Leskovec, J., and Jegelka, S. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=ryGs6iA5Km.

Yang, C., Wu, Q., Wang, J., and Yan, J. Graph neural networks are inherently good generalizers: Insights by bridging GNNs and MLPs. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=dqnNW2omZL6.

You, J., Ying, R., and Leskovec, J. Design space for graph neural networks. In *Advances in Neural Information Processing Systems*, 2020. URL https://arxiv.org/abs/2011.08843.

Zhang, S., Liu, Y., Sun, Y., and Shah, N. Graph-less neural networks: Teaching old MLPs new tricks via distillation. In *International Conference on Learning Representations*, 2022a. URL https://openreview.net/forum?id=4p6_5HBWPCw.

Zhang, W., Yin, Z., Sheng, Z., Li, Y., Ouyang, W., Li, X., Tao, Y., Yang, Z., and Cui, B. Graph attention multi-layer perceptron. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '22, pp. 4560–4570, New York, NY, USA, 2022b. Association for Computing Machinery. ISBN 9781450393850. doi: 10.1145/3534678.3539121. URL https://doi.org/10.1145/3534678.3539121.

Zhou, K., Huang, X., Zha, D., Chen, R., Li, L., Choi, S.-H., and Hu, X. Dirichlet energy constrained learning for deep graph neural networks. *Advances in neural information processing systems*, 2021.

Zhu, J., Yan, Y., Zhao, L., Heimann, M., Akoglu, L., and Koutra, D. Beyond homophily in graph neural networks: Current limitations and effective designs. *Advances in Neural Information Processing Systems*, 33, 2020.

Zhu, J., Rossi, R. A., Rao, A., Mai, T., Lipka, N., Ahmed, N. K., and Koutra, D. Graph neural networks with heterophily. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(12): 11168–11176, May 2021. doi: 10.1609/aaai.v35i12.17332. URL https://ojs.aaai.org/index.php/AAAI/article/view/17332.

A. Experimental details

This appendix provides the experimental details needed to reproduce our empirical results. Teacher/student families and the distillation objective are described in the main text; here we specify the protocol used to obtain the reported numbers.

A.1. Experimental scope

The experimental suite contains three groups of transductive node-prediction benchmarks:

Table 4. Architecture hyperparameters for (Luo et al., 2024).

		L	H	LR	epochs	M	WD	D	ln	bn	res	PL	R	R-C
a-computer	GCN	3	512	0.001	2500	2	5e-05	0.5	1	0	0	0	144.7	
	MLP	0	256	0.005	5000	2	5e-05	0.7	0	1	0	0	20	
	LGCN	3	512	0.001	5000	2	5e-05	0.5	1	0	0	0	266.3	263
a-photo	GCN	6	256	0.001	2500	2	5e-05	0.5	1	0	1	0	96.33	
	MLP	0	256	0.005	5000	2	0.0005	0.5	1	0	0	0	16	
	LGCN	6	256	0.001	5000	2	5e-05	0.5	1	0	1	0	167.7	175.7
a-ratings	GCN	4	512	0.001	2500	2	0	0.5	0	1	1	0	159.7	
	MLP	0	256	0.001	5000	2	0	0.2	0	1	0	0	21	
	LGCN	4	512	0.001	5000	2	0	0.5	0	1	1	0	309	338.7
chameleon	GCN	5	512	0.005	2500	2	0.001	0.2	0	0	0	0	31.5	
	MLP	0	512	0.001	5000	5	0.01	0.3	0	1	0	0	20	
	LGCN	5	512	0.005	5000	2	0.001	0.2	0	0	0	0	46	48.6
citeseer	GCN	2	512	0.001	2500	2	0.01	0.5	0	0	0	0	23.4	
	MLP	0	512	0.01	5000	5	0.01	0.3	0	0	0	0	27	
	LGCN	2	512	0.001	5000	2	0.01	0.5	0	0	0	0	36	38
c-cs	GCN	2	512	0.001	2500	2	0.0005	0.3	1	0	1	0	198	
	MLP	0	64	0.005	5000	2	0.01	0.2	0	1	0	0	51.33	
	LGCN	2	512	0.001	5000	2	0.0005	0.3	1	0	1	0	348	335
c-physics	GCN	2	64	0.001	2500	2	0.0005	0.3	1	0	1	0	75.67	
	MLP	0	512	0.005	5000	2	0.001	0	0	1	0	0	365	
	LGCN	2	64	0.001	5000	2	0.0005	0.3	1	0	1	0	136	137
cora	GCN	3	512	0.001	2500	2	0.0005	0.7	0	0	0	0	23.8	
	MLP	0	512	0.01	5000	3	0.01	0.7	1	0	0	0	18	
	LGCN	3	512	0.001	5000	2	0.0005	0.7	0	0	0	0	35	39
minesweeper	GCN	12	64	0.01	2500	2	0	0.2	0	1	1	0	71.67	
	MLP	0	512	0.01	5000	3	0	0.2	0	1	0	0	42.67	
	LGCN	12	1024	0.01	20000	5	0	0.2	0	1	1	0	4434	5583
pubmed	GCN	2	256	0.005	2500	2	0.0005	0.7	0	0	0	0	24.4	
	MLP	0	64	0.01	5000	2	0	0.7	1	0	0	0	14	
	LGCN	2	512	0.005	5000	2	0.0005	0.7	0	0	0	0	46.2	49
questions	GCN	10	512	3e-05	2500	2	0	0.3	0	0	1	1	993	
	MLP	0	512	0.005	5000	3	0.01	0.2	0	1	0	0	134.3	
	LGCN	10	512	3e-05	5000	2	0	0.3	0	0	1	1	1356	1507
roman-empire	GCN	9	512	0.001	2500	2	0	0.5	0	1	1	1	579	
	MLP	0	256	0.01	5000	3	0	0.7	0	1	0	0	28	
	LGCN	9	512	0.001	5000	2	0	0.5	0	1	1	1	491.3	637
squirrel	GCN	4	256	0.01	2500	2	0.0005	0.7	0	1	1	0	31.1	
	MLP	0	512	0.01	5000	5	0.01	0.7	0	1	0	0	24	
	LGCN	4	256	0.01	5000	2	0.0005	0.7	0	1	1	0	62	63
wikics	GCN	3	256	0.001	2500	2	0	0.5	1	0	0	0	63.33	
	MLP	0	64	0.01	5000	2	0.001	0.7	0	1	0	0	15	
	LGCN	3	512	0.001	5000	2	0	0.5	1	0	0	0	116	128

1. **Medium-scale node classification benchmarks.** We use the 14 medium-size datasets considered by Luo et al. (2024): Cora (McCallum et al., 2000), Citeseer (Sen et al., 2008), Pubmed (Namata et al., 2012), Amazon Computer, Amazon Photo, Coauthor CS, Coauthor Physics (Shchur et al., 2018), WikiCS (Mernyei & Cangea, 2020), Squirrel, Chameleon (Rozemberczki et al., 2021), Roman Empire, Amazon Ratings, Minesweeper, and Questions (Platonov et al., 2023b).

2. **GraphLand node-prediction benchmarks.** We use six of the datasets from GraphLand (Bazhenov et al., 2025) under the recommended RL split: artnet-exp, city-reviews, tolokers-2 (classification), artnet-views, city-roads-M, and city-roads-L (regression).

3. **Long-range node-prediction benchmark.** We use the Paris dataset from City-Networks (Liang et al., 2026), which is designed to evaluate long-range graph interactions.

A.2. Datasets, splits, and metrics

We use the preprocessing provided by the corresponding benchmark loaders. Node features are used as released by the benchmark; for GraphLand, there are some transformations given as hyperparameters (see Table 6). We use the environment from Luo et al. (2024); concretely, the version of Pytorch Geometric is 2.6.1.

We use the official benchmark-provided splits or number of runs recommended. For datasets with multiple benchmark splits, we evaluate one run per split and report the mean and standard deviation over all runs. For datasets with a fixed split, we repeat training over multiple random seeds and report the mean and standard deviation over seeds. Concretely, GraphLand benchmarks run 10 times and City-Networks run 5 times; the other datasets differ.

For classification datasets, we report test accuracy in percent, except for Minesweeper and Questions (ROCAUC) and GraphLand (AP). For GraphLand regression datasets, we report R^2. Model selection is performed using the validation metric associated with the dataset. We report the best epoch for a fixed number of epochs.

A.3. Training and hyperparameters

All models are trained with Adam (Luo et al., 2024) or AdamW (Bazhenov et al., 2025; Liang et al., 2026) with full-batch training. We run our experiments on an 80GB NVIDIA A100 GPU from an internal cluster.

Tables 4,5 report the hyperparameters and average runtimes. L is number of local layers or hops. H is hidden dimension. M is number of MLP layers as prediction head. LR is learning rate, and WD is weight decay. D is dropout. PL is pre-linear layer as encoder. R is average runtime per seed in seconds, and R-C is runtime of the concatenated version. We take the values of the non KD versions; the KD versions run at approximately the same speed and with the same parameters.

For benchmarks in Luo et al. (2024); Liang et al. (2026) the best hyperparameters are provided in their papers. For Bazhenov et al. (2025) they are not; instead, there are loops of hyperparameters specifically chosen per dataset and model, which we have re-run to find our best-performing teacher. Our baselines match those reported in the papers. For every seed or split, the teacher checkpoint with the best validation performance is selected.

Table 5. Architecture hyperparameters for (Bazhenov et al., 2025; Liang et al., 2026).

		L	H	LR	epochs	M	WD	D	ln	bn	res	PL	R	R-C
artnet-exp	GAT	3	512	3e-05	2500	2	0	0.2	1	0	1	0	439.1	
	MLP	0	512	3e-05	10000	5	0	0.2	0	1	0	0	404.4	
	LGCN	3	512	3e-05	10000	2	0	0.2	1	0	1	0	1224	1374
artnet-views	GAT	3	512	0.003	2500	2	0	0.2	1	0	1	0	374.9	
	MLP	0	512	0.003	20000	5	0	0.2	0	1	0	0	649.9	
	LGCN	3	512	0.003	20000	2	0	0.2	1	0	1	0	2286	2589
city-reviews	GAT	3	512	3e-05	2500	2	0	0.1	1	0	1	0	1442	
	MLP	0	512	3e-05	10000	5	0	0.1	0	1	0	0	1092	
	LGCN	3	512	3e-05	10000	2	0	0.1	1	0	1	0	4323	4837
city-roads-L	GAT	3	512	0.001	2500	2	0	0.2	1	0	1	0	605	
	MLP	0	512	0.001	10000	5	0	0.2	0	1	0	0	829.2	
	LGCN	3	512	0.001	10000	2	0	0.2	1	0	1	0	1967	2387
city-roads-M	GAT	3	512	0.003	2500	2	0	0.2	1	0	1	0	275.9	
	MLP	0	512	0.003	20000	5	0	0.2	0	1	0	0	730.6	
	LGCN	3	512	0.003	20000	2	0	0.2	1	0	1	0	1779	2009
tolokers-2	GAT	3	512	0.0003	2500	2	0	0.2	1	0	1	0	398.1	
	MLP	0	512	0.0003	10000	5	0	0.2	0	1	0	0	145	
	LGCN	3	512	0.0003	10000	2	0	0.2	1	0	1	0	1237	1273
paris	SAGE	16	128	0.0005	20000	2	0.00001	0.2	0	1	1	1	3171	
	MLP	0	128	0.0005	20000	2	0.00001	0.2	0	1	1	1	138	
	LGCN	16	1024	0.0005	20000	2	0.00001	0.2	0	1	1	1	2962	3522

Table 6. Feature transformations chosen for (Bazhenov et al., 2025).

		numerical_features_transform	fraction_features_transform
artnet-exp	GAT/MLP	quantile-transform-normal	quantile-transform-normal
	LGCN	standard-scaler	none
artnet-views		quantile-transform-normal	quantile-transform-normal
city-reviews		standard-scaler	none
city-roads-L		standard-scaler	none
city-roads-M		standard-scaler	none
tolokers-2		standard-scaler	none