Learning to Rank: How GNNs Solve Max-Clique and Sparse PCA

Elad Shoham [1], Omri Haber [2], Havana Rika [3], Dan Vilenchik[1]

[1]Ben-Gurion University of the Negev
[2]The Open University of Israel
[3] The Academic College of Tel Aviv-Yaffo
shellad@post.bgu.ac.il, haomri9@post.openu.ac.il, havanari@mta.ac.il, vilenchi@bgu.ac.il

Abstract

Graph neural networks (GNNs) have shown promise on combinatorial problems such as MAX-CLIQUE, yet it remains unclear what algorithmic principles they actually learn. This paper introduces a concept-driven framework for evaluating and interpreting GNNs on such tasks. We begin with a principled benchmark based on synthetic graphs with known difficulty levels—easy, medium, and hard—derived from theoretical thresholds for planted cliques. Using this setup, we show that GNNs reliably learn a simple yet powerful concept: degree-based ranking. This insight motivates a new decoder, Least-Probable Removal (LPR), which significantly outperforms the common top-k strategy, especially on harder and real-world instances. Our analysis pipeline connects latent representations to classical heuristics, improving both interpretability and performance. Finally, we demonstrate cross-domain generalization to sparse PCA, showing that the same GNN architecture and decoding strategy succeed in recovering sparse principal components—revealing a shared underlying principle across domains.

1 Introduction

AI systems now achieve remarkable performance on complex computational tasks, including combinatorial optimization problems such as the MAX-CLIQUE problem which is identifying the largest fully connected subgraph in a graph G. Yet their decision-making processes often remain opaque, motivating a surge of interest in explainable AI.

While popular XAI techniques such as SHAP (Lundberg and Lee 2017), LIME (Ribeiro, Singh, and Guestrin 2016), or Grad-CAM (Selvaraju et al. 2017) focus on feature attribution, they often fall short in domains like combinatorial optimization, where explanations must capture algorithmic principles rather than simply assigning importance to individual features. For example, in the case of MAX-CLIQUE, a meaningful explanation would reveal how properties like vertex degree or connectivity influence the model's predictions of clique membership.

The MAX-CLIQUE problem, a canonical NP-hard problem (Karp 1972), exemplifies this challenge. While NP-hardness implies that some instances are computationally intractable, it does not mean that all instances are equally hard. In fact, many instances can be easy, and without controlling for instance difficulty, it is unclear whether good performance reflects real algorithmic power or simply the ease of the test data.

Despite the empirical success of graph neural networks (GNNs) on graph-structured data (Karalias and Loukas 2020; Min et al. 2022), current evaluations lack this critical control as they typically test on real-world or unstructured datasets without any principled characterization of input hardness. Moreover, standard GNN decoding strategies, such as selecting the top-k vertices by confidence, implicitly assume that output scores align with true clique membership. This assumption often fails on harder instances, where misleading "impostor" vertices (e.g., those with spuriously strong local signals) can degrade performance.

The gap. To summarize, the following gaps remain in the study of GNNs for the MAX-CLIQUE problem:

- The absence of principled benchmarks that control for instance difficulty, making it unclear how meaningful current performance evaluations are.
- A lack of understanding of the algorithmic principles that GNNs learn when trained on MAX-CLIQUE instances. This can hamper real progress—new architectures may appear novel but end up learning the same principles.
- The lack of decoding strategies that exploit these learned representations to improve both performance and interpretability.

Although we focus on MAX-CLIQUE as a case study, these gaps are not unique to this problem. Similar challenges arise throughout the application of GNNs to combinatorial optimization tasks, and the key insights and methods we develop here aim to have broader relevance.

Our contribution. This paper narrows these gaps by contributing both a methodology and new algorithmic insights:

1. **Principled evaluation.** We introduce a benchmark based on synthetic graphs with planted cliques, categorized into easy, medium, and hard regimes according to known theoretical thresholds (Bollobás 1985; Håstad 1999). This allows us to systematically probe GNN performance in difficulty-aware settings, complemented by tests on six real-world datasets.

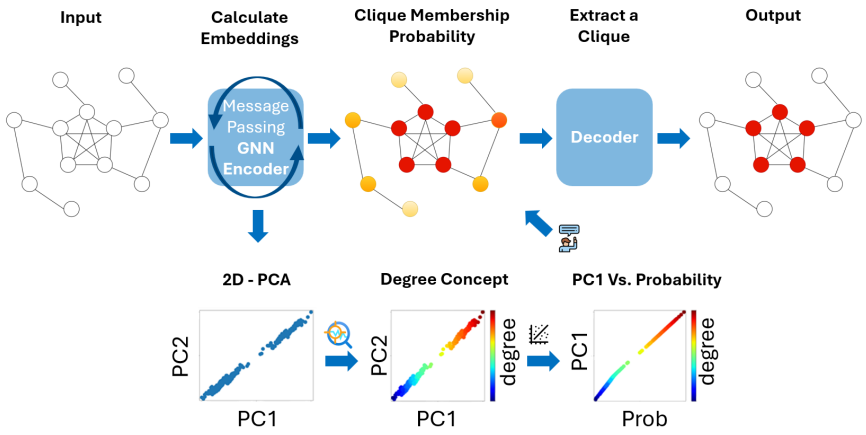


Figure 1: Illustration of our method to excavate concepts and explain how they underpin the GNN's decision-making process. The GNN's flow is given at the top row; the bottom row shows how the learned concepts are discovered in the embedding and drive the clique membership probability output. PC1 refers to the projection of latent space onto the first principal component.

2. **XAI-driven analysis.** We analyze the latent spaces of trained GNNs and find that they consistently learn a degree-based ranking of vertices. This provides a conceptual bridge between learned representations and classical heuristics. Figure 1 illustrates our XAI pipeline.
3. **Decoder innovation.** Motivated by this insight, we replace the naive top-k decoder with a Least-Probable Removal (LPR) strategy: iteratively removing low-likelihood vertices until a clique remains. This leads to substantial performance improvements, particularly on medium-difficulty and real-world instances.
4. **Cross-domain generalization.** We show that the same GNN and decoder architecture generalizes seamlessly to single-spike SPARSE-PCA instances (Johnstone 2001), where a clique is replaced with a strong correlation structure in the covariance matrix. This transfer succeeds because the GNN implicitly learns a universal principle, row-sum ranking, that extends beyond graph structure. We further verify the reverse: a GNN trained on SPARSE-PCA instances can successfully solve MAX-CLIQUE.
5. **New algorithmic result.** Covariance Thresholding (CT) stands as the state-of-the-art for solving medium-SNR single-spike sparse PCA instances, both theoretically and empirically (Bickel and Levina 2008; Deshpande and Montanari 2016) . For years, it remained an open question whether a non-spectral, combinatorial algorithm could match its performance. We answer this affirmatively and show that the MAX-CLIQUE GNN with LPR achieves comparable results, offering the first combinatorial alternative, no hyperparameters, whose operation is interpretable through our concept learning framework.

The remainder of the paper is organized as follows. Section 2 expands our methodology, where we formally define the Max-Clique problem, and the top-k and LPR decoding. Section 3 covers related work, Section 4 details the GNN architectures, and Section 5 describes the datasets. Section 6 presents our experiments and Section 7 concludes with a discussion.

2 Methodology

Our methodology has two main components: a principled benchmarking framework for MAX-CLIQUE, and a concept learning pipeline for interpreting GNNs. Here we expand on the benchmarking framework, as it may be new to readers less acquainted with theoretical aspects of planted clique in random graphs. The concept learning pipeline, illustrated in Figure 1, is described and elaborated in Sections 6 and 7.

Throughout this paper, we sometimes use the terms "Maximum Clique" and "finding a clique of size k" interchangeably. These two problems are equivalent, given an algorithm for one, the other can be solved by iterating over possible values of k from 1 to n, the number of vertices in the graph. In this work, we focus on the version where the target clique size k is given as input, but for brevity we occasionally refer to the problem simply as "Max-Clique."

The Maximum-clique problem is easily expressed in mathematical form. Given an $n \times n$ adjacency matrix A of a graph G, and a vector $x \in \{0,1\}^n$ with exactly k ones, it is readily verified that the bi-linear form $x^T Ax$ equals twice the number of edges in the subgraph indexed by x. Thus, to find if the graph has a clique of size k one can solve

$$\omega_k(G) = \max_{\substack{x \in \{0,1\}^n \\ \|x\|_0 = k}} x^T Ax, \tag{1}$$

where $\|x\|_0$ is the number of non-zero elements in x, and check if $\omega_k(G) = 2\binom{k}{2}$. To find the largest clique, one simply iterates this for $k = 1, \ldots n$.

Replacing the adjacency matrix A in Eq. (1) with a covariance matrix Σ, and changing the constraint $x \in \{0,1\}^n$ to $\|x\|^2 = 1$, gives the SPARSE-PCA problem.

What makes solving the quadratic optimization in Eq. (1) computationally hard is the non-convex constraints on x (both the binary constraint and the sparsity).

2.1 Two Decoding Heuristics: Top-k and LPR

A key challenge in solving MAX-CLIQUE is converting vertex scores into valid solutions. Two general rounding heuristics, *Top-k* and *Least-Probable Removal (LPR)*, are commonly used. Both start from a vertex scoring (e.g., degree or GNN output) and define how to round it into a discrete solution, e.g., clique membership.

Top-k heuristics proceed by ranking vertices according to their scores, and then selecting vertices in order of decreasing score. A simple variant selects the top-k vertices as a candidate clique without verification; a more refined variant incrementally adds vertices if they form a clique with all previously selected ones, stopping when k clique vertices are found or the list is exhausted. In the MAX-CLIQUE literature, both variants have been analyzed (e.g., (Feige and Ron 2010)), and the Top-k approach is known as *diagonal thresholding* in SPARSE-PCA (Johnstone and Lu 2009).

In contrast, *LPR* heuristics iteratively remove the vertex with the lowest score, recompute scores after each removal, and repeat until a clique remains. This strategy has also been studied theoretically for MAX-CLIQUE (Feige and Ron 2010; Dekel, Gurel-Gurevich, and Peres 2014) and embodies a fundamentally different search principle—rather than building up a solution from high-scoring vertices, it prunes low-scoring vertices until feasibility is achieved.

In terms of computational complexity, *Top-k* heuristics run in $O(n^2)$ time while *LPR* heuristics require $O(n^3)$ since scores must be recomputed after each removal, and up to n removals may be needed. While this added computational cost might seem significant, we will show that it pays off: LPR consistently outperforms Top-k on harder instances, where naive ranking heuristics fail to account for subtle structural cues that mislead Top-k decoders. In practice, the runtime averaged 25 seconds, peaking at 90 seconds for the largest graph (47 K nodes), staying within practical limits.

2.2 Max-Clique in Random Graphs

The Erdős-Rényi model $G(n, p)$ generates graphs by connecting each pair of n vertices independently with probability p. For $G(n, 1/2)$, the largest clique has size roughly $2\log_2 n$ with high probability, and finding it is believed to be computationally hard (Feige and Grinberg 2020).

The *planted clique* problem adds a clique of size k into $G(n, 1/2)$. The difficulty of recovering this clique depends on k— a larger k makes recovery easier because clique vertices receive a noticeable degree "boost". Specifically:

- The **Easy** regime: When $k = \Omega(\sqrt{n}\log n)$, simple heuristics like Top-k degree ranking recover the clique (Kucera 1995).
- The **Medium** regime: When $k = \Omega(\sqrt{n})$, recovery requires more sophisticated methods such as spectral algorithms (AKS (Alon, Krivelevich, and Sudakov 1998)) or LPR based on degree (Feige and Grinberg 2020).
- The **Hard** regime: When $k = O(\sqrt{n})$, the planted clique hardness conjecture asserts that recovery is computationally infeasible (Manurangsi et al. 2021; Hajek 2015; Ma and Wu 2015; Berthet and Rigollet 2013a; Hazan and Krauthgamer 2011).

2.3 Defining Easy, Medium, and Hard Instances

While these asymptotic results define the general landscape, they provide no direct guidance for fixed, finite graphs (e.g., $n = 500$). To label our benchmarks, we used the algorithmic insights to empirically calibrate the regime thresholds for k.

Concretely, for a fixed n (we used $n = 500, 1000$), we identified the minimal k where the Top-k heuristic recovered the planted clique 90% of the time, and we mark it as k_1. We then further reduced k until LPR's success rate drops below 90% and mark it as k_2. Lastly, we set $k_3 = 2\log_2 n$, the size of the largest clique in a random graph. The **Easy** regime is $[k_1, n]$, **Medium** is $[k_2 + 1, k_1 - 1]$ and **Hard** $[k_3, k_2]$.

2.4 Single Spike SPARSE-PCA

In addition to graphs, we evaluate GNN generalization on synthetic single-spike SPARSE-PCA instances, which were introduced in (Johnstone 2001) and have been widely used in the literature to study high-dimensional data.

In this setting, there are two key dimensions: p, the number of features, and n, the number of samples. The most interesting regime is high-dimensional (Johnstone 2001; Johnstone and Lu 2009), where $p = \Omega(n)$. For simplicity and without loss of generality, we set $p = n$ throughout to avoid carrying unnecessary notation.

The single-spike instances are generated as follows. First, we construct a population covariance matrix of the form

$$\Sigma = I + \beta vv^\top, \tag{2}$$

where v is a sparse "spike" vector of support size k, and $\beta > 0$ controls the signal strength. In our canonical construction, v has exactly k nonzero entries, each equal to $\pm 1/\sqrt{k}$, ensuring unit norm.

Next, using this covariance, we sample n independent p-dimensional multivariate normal observations to form a data matrix $X \in \mathcal{R}^{n\times p}$, and construct the empirical covariance matrix $S = X^\top X/n$. The task is then to recover the support of v, i.e., identify the k variables most correlated with the spike component, the solution to Eq. (1).

MAX-CLIQUE and SPARSE-PCA are structurally similar—recovering the support of v in sparse PCA parallels identifying a planted clique in a graph. However, the monotonicity of complexity is reversed, larger k makes sparse PCA instances harder, since the spike signal is spread over more entries (while v remains unit-norm), causing Gaussian noise to dominate and mask the signal. Also, a similar taxonomy of difficulty regimes—easy, medium, and hard—has also been established for SPARSE-PCA (Berthet and Rigollet 2013b; Krauthgamer, Nadler, and Vilenchik 2015; Johnstone and Lu 2009), allowing us to repeat the framework posed in Sections 2.2 and 2.3.

3 Related Work

Our work lies at the intersection of two areas: neural networks for combinatorial optimization and explainable AI (XAI), a largely unexplored space with few exceptions such as (Shoham et al. 2026).

Combinatorial Optimization. Many graph problems, including Max-Clique, are NP-hard and typically solved with

exact algorithms (Rossi, Gleich, and Gebremedhin 2015) or heuristics like Top-k and LPR. Recently, GNNs have emerged as data-driven heuristics for problems such as TSP, Max-Cut, Max-Independent Set, and Max-Clique (Garmendia et al. 2024; Karalias and Loukas 2020; Min et al. 2022; Yau et al. 2023). Some approaches leverage specialized loss functions inspired by convex relaxations (Yau et al. 2023), while others combine imitation learning with classic heuristics (Liu, Lodi, and Tanneau 2021). In contrast, the GNNs we study are trained with a "natural" loss directly derived from the clique objective, enabling us to examine which algorithmic concepts they learn spontaneously.

Explainable AI. XAI methods like LIME, SHAP (Ribeiro, Singh, and Guestrin 2016; Lundberg and Lee 2017), GradCAM (Selvaraju et al. 2017), and RISE (Petsiuk, Das, and Saenko 2018) focus on feature importance or saliency, but these approaches are less suited to combinatorial tasks where algorithmic principles matter. Other work probes learned representations for human-understandable concepts (Kim et al. 2018; McGrath et al. 2022; Qiu et al. 2024), though these often require auxiliary classifiers that may themselves introduce confounds. More relevant to our approach is (Shoham et al. 2026), which interprets intermediate embeddings by aligning them with known heuristics. We adopt and extend this concept-driven perspective, focusing on aligning latent spaces with interpretable algorithmic principles.

4 GNN Models for Max-Clique

We evaluate two recent GNN architectures for solving MAX-CLIQUE: **ENN** (Karalias and Loukas 2020) and **SNN** (Min et al. 2022). Both follow the standard pipeline of GNN-based combinatorial solvers. They process a graph using message passing, and decode the resulting vertex scores into a clique using a fixed strategy. While the results were comparable across the two models, we focus on SNN for brevity, as it is more recent (2022 vs. 2020), includes additional architectural components (notably, scattering layers), and slightly outperformed ENN in the tests we ran. Parameter count and size analysis of each model appears in the Supplementary.

Scattering-based GNN (SNN). SNN uses a graph diffusion module to extract multiscale structural features. Specifically, the diffusion operator $P := \frac{1}{2}(I_n + AD^{-1})$ mixes self-information with normalized neighbor information. From this, the model constructs scattering wavelets for each scale r as $\psi_r := P^{2^{r-1}} - P^{2^r}$, which isolate frequency bands that capture local-to-global graph structure. An attention mechanism selects the most informative bands, and the aggregated signal is passed through an MLP to produce vertex scores, normalized to $[0, 1]$.

The model and is trained in an unsupervised manner with loss

$$L(p) = -p^\top A p + \beta p^\top \bar{A} p,$$

where A and $\bar{A}$ are the adjacency matrices of the graph and its complement respectively, p is the vector of clique membership probabilities. The first term is derived directly from Eq. (1), promoting dense subgraphs (cliques), while the second penalizes mass assigned to non-edges.

To isolate learned representations, we omitted handcrafted features (e.g., eccentricity, degree) used in the original paper as part of the input, alongise the graph G. We also evaluated ENN and found that all major trends replicate; for space reasons, we report results for SNN only.

Dataset	# Graphs	n	k	Dist
Easy-1k	1000	1000	[62, 1000]	Random
Medium-1k	1000	1000	[36, 61]	Random
Hard-1k	1000	1000	[20, 35]	Random
Easy-0.5k	1000	500	[71, 500]	Random
Medium-0.5k	1000	500	[26, 70]	Random
Hard-0.5k	1000	500	[15, 25]	Random
Youtube	16386	7.8±42.2	[2, 12]	Real
IMDB	1000	19.8±10	[4, 30]	Real
Orkut	5000	215±320	[3, 50]	Real
COLLAB	5000	74.5±62	[4, 239]	Real
Twitter	973	131±64	[2, 70]	Real
Facebook	10	416±329	[7, 69]	Real

Table 1: Description of datasets used in our exploration. The average size of the graph is n ± std. k is the range of clique sizes. Dist stands for distribution

5 Datasets

Our dataset collection includes both synthetic and real-world graphs, selected to span different levels of algorithmic difficulty (Table 1). The synthetic datasets comprise three categories—Easy, Medium, Hard—generated as described in Section 2.2 using the planted clique model $G(n, 1/2, k)$.

Each category consists of 1000 instances generated by first picking a k uniformly at random from the relevant range, then choosing k vertices uniformly at random, and connecting all edges between them (the clique). Then, include each remaining possible edge with probability 1/2.

We varied the number of vertices n from 100 to 2000 and observed similar qualitative performance of the GNNs across sizes; for brevity, we report results for $n = 500, 1000$.

To complement the synthetic data, we include six real-world graph datasets: Twitter (McAuley and Leskovec 2012a), COLLAB (Morris et al. 2020), IMDB (Morris et al. 2020; Yanardag and Vishwanathan 2015), Facebook (McAuley and Leskovec 2012b; Leskovec and Krevl 2014), YouTube, and Orkut (Yang and Leskovec 2012; Leskovec and Krevl 2014). These datasets introduce additional structural challenges, such as disconnected components and skewed degree distributions, providing a broader test of the GNNs' generalization capabilities.

Sparse PCA data. In addition, we generated single-spike SPARSE-PCA instances according to the model described in Eq. (2), where a rank-one perturbation $\beta vv^\top$ is added to a standard Gaussian covariance matrix, with v a sparse unit vector whose non-zero entries are $\pm 1/\sqrt{k}$ (sign chosen uniformly at random). In line with prior work on SPARSE-

PCA (Amini and Wainwright 2008; Deshpande and Montanari 2016; Johnstone 2001; Berthet and Rigollet 2013b), we set the number of features equal to the number of samples, $p = n$, and fix $\beta = 1$, placing us in the high-dimensional, weak-signal regime. As in the MAX-CLIQUE case, the SPARSE-PCA instances in that parameter regime were categorized into easy, medium, and hard regimes (in the aforecited papers). Empirically, for $n = p = 1000$, we found the k interval that spans the easy to hard regime to be 1 to 50. We tested the GNN across this entire range. For brevity, we omit a full table summarizing these instances.

To the best of our knowledge, there are no real-world SPARSE-PCA benchmarks with labeled ground-truth support, and so our evaluation is limited to synthetic data.

6 Experimental Setting

Our experimental setting follow the structure of our main contributions: a principled evaluation of GNN performance on synthetic and real-world graphs; an XAI analysis revealing that the models learn a degree-based ranking; performance improvements by replacing Top-k with LPR; and cross-domain generalization to sparse PCA, reflecting a universal row-sum ranking principle.

Experiments were conducted on an Nvidia RTX 3080 GPU with 64GB RAM and an AMD Ryzen 9 5650X CPU, using PyTorch. The code can be found here https://github.com/HavanaLab/clique_aaai_2026. We trained two GNN models, the Scattering-based model (SNN) described in Section 4, and the Erdős Goes Neural model (ENN) (Karalias and Loukas 2020). Each was trained for 100 epochs with batch size 32. At test time, we applied the two decoding strategies, Top-k and LPR.

Evaluation metrics. Performance is measured by the approximation ratio which is the size of the clique found divided by the true maximum clique size. For synthetic datasets, ground truth is known by construction; for real-world datasets, we computed the optimum using an exact solver (Rossi, Gleich, and Gebremedhin 2015).

For the XAI part, where we analyze what the GNNs learn, we used two methods: (i) *Visualization*, projecting vertex embeddings onto two dimensions via PCA; PC1 typically explains over 95% of variance, suggesting a dominant low-dimensional structure. (ii) *Quantification*, computing Spearman rank correlations between degree ranking and both PC1 projections and softmax outputs, to measure alignment between learned representations, predictions, and degree. We track these correlations layer-by-layer to understand how concept learning evolves during message passing.

For SPARSE-PCA, we compare the GNN to the state-of-the-art Covariance Thresholding method (Deshpande and Montanari 2016; Bickel and Levina 2008). Given a threshold t, it zeros out all off-diagonal covariance entries with absolute value below t, then computes the top eigenvector and selects the top k entries by absolute value.

7 Results

We now turn to discuss the results that were obtained according to the experimental setting just described. For space reasons, we report results for the scattering-based GNN (SNN) here; results for the Erdős Goes Neural network (ENN) are similar and included in the Supplementary material. Also all the raw results, which the tables below summarize, appear in the Supplementary material.

7.1 Performance Results

We evaluated the GNN performance on both synthetic planted-clique datasets and real-world graphs.

Insight 1. Our first key observation is that performance is remarkably insensitive to the choice of training data. The type of training graphs (random vs. structured), their difficulty (easy, medium, hard), and the size of planted cliques all have minimal influence on test outcomes. This suggests that the concepts the GNNs learn are highly transferable and largely independent of clique-specific features. Detailed results for all train-test combinations are reported in Supplementary Table 5. Given this robustness, Table 2 reports performance for models trained on the Medium-1k and Medium-0.5k planted-clique dataset, which achieved the best, but only marginally better, results overall.

Insight 2. A key result from Table 2 is the validation of our principled dataset design. The separation into Easy, Medium, and Hard regimes was defined independently of the GNNs, based solely on classical theoretical insights. The results show that GNN performance tracks this categorization precisely. Top-k decoding works well on Easy instances but collapses on Medium and Hard; LPR succeeds on both Easy and Medium but fails on Hard, as expected. This demonstrates that our framework provides a differentiated and meaningful way to evaluate any future GNN architecture, enabling targeted improvements.

Insight 3. Turning to real-world datasets, we observe a dramatic improvement when switching from Top-k to LPR decoding across all datasets except YouTube, confirming our second key contribution: aligning decoding strategies with the concepts learned by the model improves performance substantially even outside synthetic benchmarks. The complete comparison against classical algorithm baselines is provided in the Supplementary.

7.2 Concept Evaluation

We now evaluate how the GNNs have learned the underlying concept guiding their predictions.

Visual analysis. We begin by visualizing the learned embeddings using PCA. Figure 2 shows 2D projections of SNN embeddings across layers, with vertices color-coded by degree. A clear alignment between degree and the horizontal axis (PC1, explaining over 95% of the variance) emerges after just one layer, though in some cases deeper layers refine this structure further. We retain all four layers for consistency with the original SNN architecture (Min et al. 2022). The vertical axis (PC2), accounting for less than 5% of the variance, introduces dataset-specific noise but does not obscure the dominant degree signal. Similar visualizations for ENN and for real-world graphs (e.g., Twitter) appear in the Supplementary material.

Test	Top-k	LPR
Easy-1k, Easy-0.5k	0.99 ± 0.01	0.99 ± 0.01
Medium-1k, Medium-0.5k	0.55 ± 0.19	0.99 ± 0.06
Hard-1k, Hard-0.5k	0.55 ± 0.23	0.55 ± 0.23
Average Synthetic	0.70	0.84
Youtube	0.99 ± 0.04	0.99 ± 0.05
IMDB	0.9 ± 0.2	1 ± 0.15
Orkut	0.78 ± 0.2	0.9 ± 0.17
COLLAB	0.77 ± 0.28	1 ± 0.26
Twitter	0.76 ± 0.21	0.91 ± 0.23
Facebook	0.71 ± 0.25	0.99 ± 0.02
Average Real-world	0.81	0.96

Table 2: Approximation ratio (mean $\pm$ std) of top-k and LPR decoders for the SNN trained on Medium-1k. Synthetic results for $n = 1k, 0.5k$ are grouped as they were the same.

We observe the same structure in SPARSE-PCA instances (Figure 2 bottom part), where replacing degree with the row-sum of the design matrix yields a similar alignment with the principal embedding axis.

Quantitative analysis. We next quantify how degree ranking is encoded in the embeddings and outputs.

Insight 1. For synthetic datasets, we observe, in Table 3, a perfect Spearman correlation between degree and PC1 across all layers and datasets. However this perfect alignment does not translate into perfect performance—in the Medium regime, degree ranking does not naively predict clique membership, and in the Hard regime, the planted clique may not even exhibit a detectable degree signal under the Planted Clique Hardness Conjecture.

Insight 2.1 For real-world datasets, we find strong but imperfect correlations between degree and PC1 (bottom part of Table 3), with variation across datasets. Notably, the two datasets with the weakest correlations—Twitter and Facebook—are precisely those where Top-k decoding performs worst, highlighting its sensitivity to degree alignment and its inferiority to LPR. However, correlation alone does not explain differences in LPR performance across datasets; we explore this in Insight 3.

Insight 2.2 In some cases, we observe a slight drop in degree-PC1 correlation from Layer 1 to Layer 4. While this degradation is often minor and at times even within the standard deviation, it highlights a potential inefficiency: had the GNN been designed with concept learning in mind, it may have been possible to omit deeper layers—reducing computation and potentially improving performance.

Insight 3. Table 4 shows the fraction of clique vertices in the lowest PC1 quartile (Q1). Such vertices are problematic for both decoders: Top-k overlooks them, and LPR risks falsely removing them.

In real-world graphs, Orkut, COLLAB, and Facebook have the most clique vertices in Q1, mirroring Top-k's worst performance and its sensitivity to naive degree signals. Twitter also struggles, despite fewer Q1 vertices, likely due to clique members scattered in Q2–Q3.

Dataset	L1 vs Deg	L4 vs Deg	L4 vs Probs
Easy	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
Medium	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
Hard	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
YouTube	0.91 ± 0.08	0.92 ± 0.09	0.98 ± 0.05
IMDB	0.93 ± 0.09	0.92 ± 0.12	0.98 ± 0.07
Orkut	0.94 ± 0.06	0.92 ± 0.08	0.99 ± 0.01
COLLAB	0.94 ± 0.06	0.92 ± 0.08	0.98 ± 0.04
Twitter	0.91 ± 0.08	0.88 ± 0.09	0.99 ± 0.01
Facebook	0.88 ± 0.05	0.86 ± 0.06	1 ± 0

Table 3: Spearman correlation between vertex degree and PC1 projection of SNN layers (L2,L3 omitted due to similarity). Also shown: correlation between final softmax outputs (Prob) and PC1 projection of L4 embedding.

LPR does well even with Q1 clique vertices as removing one vertex at a time minimizes false eliminations. Table 4 shows only the first step; as LPR proceeds, non-clique vertices tend to fall in rank while clique vertices regain rank. For lack of space, we omit this dynamic analysis.

For synthetic graphs, trend aligns with theory: no clique vertices in Q1 for Easy, a few for Medium, more for Hard.

Insight 4. Finally, we confirm that the latent-space learned degree structure propagates through to the model outputs: PC1 scores are highly correlated with predicted clique membership probabilities, as seen in the last column of Table 3, establishing a coherent explainability link from embeddings to decisions.

7.3 Sparse-PCA Results

Figure 3 presents the approximation ratio achieved by SNN trained on MAX-CLIQUE and tested on single-spike SPARSE-PCA (left) and MAX-CLIQUE (right) instances.

Insight 1. The trends are symmetric once we account for reversed hardness: in SPARSE-PCA, larger k makes instances harder, while in MAX-CLIQUE, it makes them easier. In both domains, the top-k decoder performs well in the

Dataset	Clique in Q1	Non-clique in Q1
Easy	0 ± 0	0.33 ± 0.02
Medium	0.03 ± 0.03	0.33 ± 0.02
Hard	0.07 ± 0.03	0.33 ± 0.02
Twitter	0.1 ± 0.09	0.26 ± 0.04
Youtube	0.07 ± 0.15	0.1 ± 0.14
IMDB	0.09 ± 0.11	0.15 ± 0.12
Orkut	0.24 ± 0.11	0.24 ± 0.05
COLLAB	0.18 ± 0.11	0.19 ± 0.09
Facebook	0.26 ± 0.09	0.24 ± 0.01

Table 4: Proportion of clique (and non-clique) vertices in the lowest quartile ($Q1$) of PC1 scores at layer 4, measuring how often true clique members fall among the lowest-ranked nodes and similarly non-clique.

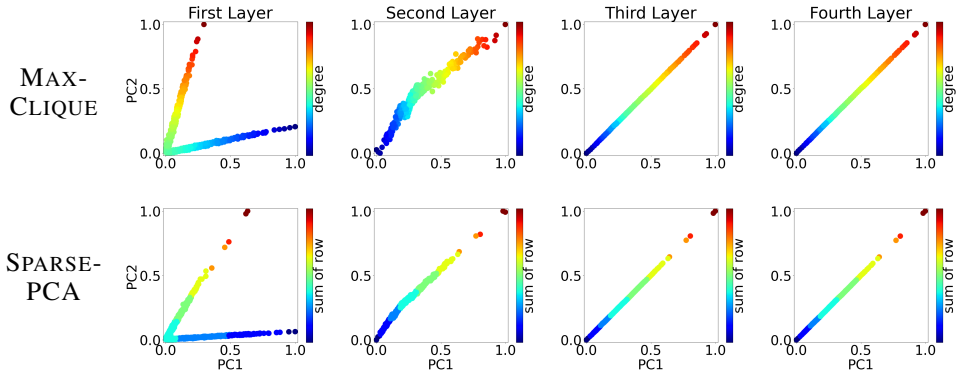
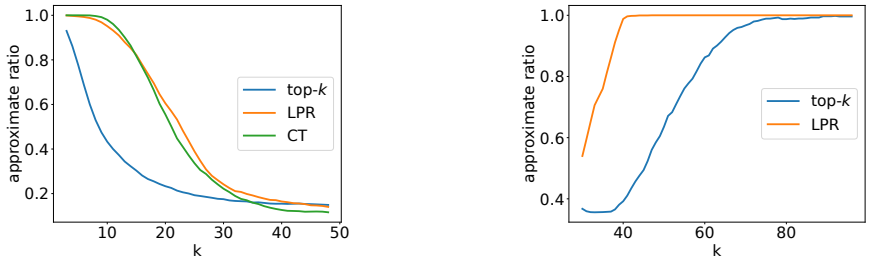


Figure 2: 2D PCA projection of SNN vertex embeddings across four layers, shown for a Medium-1k synthetic instance. Colors indicate true degree (MAX-CLIQUE) or row sum (SPARSE-PCA). PC1 captures over 95% of the variance and aligns with degree after just one layer. We include four layers for consistency with the original architecture of the SNN (Min et al. 2022), though comparable performance was achieved with a single layer.



(a) SPARSE-PCA: $n = p = 1000$

(b) MAX-CLIQUE: $n = 1000$

Figure 3: Performance of SNN when trained on MAX-CLIQUE Medium-1k and tested on Single-Spike SPARSE-PCA (left) and random MAX-CLIQUE (right) instances as k varies. Larger k values make MAX-CLIQUE easier but SPARSE-PCA harder. The Covariance Thresholding algorithm (CT) was parameterized (independently for each instance) with a threshold t via grid search over 0.1–8 in steps of 0.1, reporting the best result. SNN+LPR is as good as the state-of-the-art CT.

easy regime but quickly deteriorates as difficulty increases. In contrast, LPR maintains strong performance throughout and achieves state-of-the-art accuracy in the medium-difficulty regime, on par with the spectral baseline Covariance Thresholding (Deshpande and Montanari 2016).

Insight 2. We observed the same performance patterns when SNN was trained on SPARSE-PCA instances rather than MAX-CLIQUE, underscoring that a universal concept, row-sum ranking, was learned and successfully transferred across relevant domains. Again, details omitted for brevity.

8 Discussion

We present a framework showing that GNNs trained on MAX-CLIQUE consistently learn vertex degree ranking, across datasets, architectures, and even domains, SPARSE-PCA. Remarkably, this concept alone is sufficient to drive competitive performance on both synthetic and real-world benchmarks. These findings support Cappart et al. (2023), which says that GNNs often resemble polynomial-time heuristics rather than solvers for NP-hard problems.

Notably, both ENN and the more recent SNN converge to the same internal concept—*degree ranking*—despite SNN's added scattering layers. This convergence suggests that, had concept learning been analyzed earlier, research effort might have shifted from refining ENN to pursuing fundamentally new architectural directions beyond degree-based heuristics. Looking ahead, our framework motivates the principle of *algorithmic alignment*: designing learning objectives and architectures that remain grounded in the reasoning patterns of the algorithms they aim to emulate. For example, one could steer concept learning toward richer notions such as *core number* or *local connectivity motifs* instead of defaulting to degree. Likewise, integrating domain-specific decoders like LPR directly into training could reinforce solver-aligned reasoning and improve efficiency—potentially reducing runtime from $O(n^3)$ to $O(n^2)$. Future work may further employ imitation learning (Liu, Lodi, and Tanneau 2021) to guide networks toward explicit emulation of algorithmic decision logic.

References

Alon, N.; Krivelevich, M.; and Sudakov, B. 1998. Finding a Large Hidden Clique in a Random Graph. In *ACM-SIAM*

Symposium on Discrete Algorithms (SODA), volume 13, 594–598.

Amini, A. A.; and Wainwright, M. J. 2008. High-dimensional analysis of semidefinite relaxations for sparse principal components. In *2008 IEEE international symposium on information theory*, 2454–2458. IEEE.

Berthet, Q.; and Rigollet, P. 2013a. Complexity theoretic lower bounds for sparse principal component detection. *Journal of Machine Learning Research*, 30: 1046–1066.

Berthet, Q.; and Rigollet, P. 2013b. Optimal detection of sparse principal components in high dimension. *The Annals of Statistics*, 41(4): 1780–1815.

Bickel, P. J.; and Levina, E. 2008. Covariance regularization by thresholding.

Bollobás, B. 1985. *Random Graphs*. Academic Press, Inc.

Cappart, Q.; Chételat, D.; Khalil, E. B.; Lodi, A.; Morris, C.; and Velickovic, P. 2023. Combinatorial Optimization and Reasoning with Graph Neural Networks. *Journal of Machine Learning Research (JMLR)*, 24: 130:1–130:61.

Dekel, Y.; Gurel-Gurevich, O.; and Peres, Y. 2014. Finding hidden cliques in linear time with high probability. *Combinatorics, Probability and Computing*, 23(1): 29–49.

Deshpande, Y.; and Montanari, A. 2016. Sparse PCA via covariance thresholding. *Journal of Machine Learning Research*, 17(141): 1–41.

Feige, U.; and Grinberg, V. 2020. How to Hide a Clique? In *International Colloquium on Automata, Languages and Programming (ICALP)*, 44:1–44:13.

Feige, U.; and Ron, D. 2010. Finding hidden cliques in linear time. *Discrete Mathematics & Theoretical Computer Science*, DMTCS Proceedings vol. AM,...

Garmendia, A. I.; Cappart, Q.; Ceberio, J.; and Mendiburu, A. 2024. MARCO: A Memory-Augmented Reinforcement Framework for Combinatorial Optimization. In *Proceedings of the Thirty-ThirdInternational Joint Conference on Artificial Intelligence*, 6931–6939. International Joint Conferences on Artificial Intelligence Organization.

Hajek, W. Y. X. J., Bruce. 2015. Computational lower bounds for community detection on sparse random graphs. *Journal of Machine Learning Research*, 16(1): 3261–3298.

Håstad, J. 1999. Clique is hard to approximate within $n^{1-\epsilon}$. *Acta Mathematica*, 182(1): 105–142.

Hazan, E.; and Krauthgamer, R. 2011. How hard is it to approximate the best Nash equilibrium? *SIAM Journal on Computing*, 40(1): 79–91.

Johnstone, I. M. 2001. On the distribution of the largest eigenvalue in principal components analysis. *The Annals of statistics*, 29(2): 295–327.

Johnstone, I. M.; and Lu, A. Y. 2009. On consistency and sparsity for principal components analysis in high dimensions. *Journal of the American Statistical Association*, 104(486): 682–693.

Karalias, N.; and Loukas, A. 2020. Erdos Goes Neural: an Unsupervised Learning Framework for Combinatorial Optimization on Graphs. In *Conference on Neural Information Processing Systems (NeurIPS)*.

Karp, R. M. 1972. Reducibility among combinatorial problems. In *Complexity of computer computations*, 85–103. Springer.

Kim, B.; Wattenberg, M.; Gilmer, J.; Cai, C. J.; Wexler, J.; Viégas, F. B.; and Sayres, R. 2018. Interpretability Beyond Feature Attribution: Quantitative Testing with Concept Activation Vectors (TCAV). In *International Conference on Machine Learning (ICML)*, 2673–2682.

Krauthgamer, R.; Nadler, B.; and Vilenchik, D. 2015. Do SemiDefinite Relaxations Solve Sparse PCA up to the Information Limit? *Annals of Statistics*, 4(3): 1300–1322.

Kucera, L. 1995. Expected Complexity of Graph Partitioning Problems. *Discrete Applied Mathematics*, 57(2-3): 193–212.

Leskovec, J.; and Krevl, A. 2014. SNAP Datasets: Stanford Large Network Dataset Collection. http://snap.stanford.edu/data. Accessed: November 12, 2025.

Liu, D.; Lodi, A.; and Tanneau, M. 2021. Learning chordal extensions. *J. Glob. Optim.*, 81(1): 3–22.

Lundberg, S. M.; and Lee, S.-I. 2017. A Unified Approach to Interpreting Model Predictions. In *Conference on Neural Information Processing Systems (NeurIPS)*, 4765–4774.

Ma, Z.; and Wu, Y. 2015. Computational barriers in minimax submatrix detection. *Annals of Statistics*, 43(3): 1089–1116.

Manurangsi, P.; Raghavendra, P.; Schramm, T.; and Steurer, D. 2021. Strongish Planted Clique Hypothesis and its consequences for hardness of approximation. In *Proceedings of the Annual ACM Symposium on Theory of Computing (STOC)*, 1247–1260.

McAuley, J. J.; and Leskovec, J. 2012a. Learning to Discover Social Circles in Ego Networks. In *Conference on Neural Information Processing Systems (NeurIPS)*, 548–556.

McAuley, J. J.; and Leskovec, J. 2012b. Learning to Discover Social Circles in Ego Networks. In Bartlett, P. L.; Pereira, F. C. N.; Burges, C. J. C.; Bottou, L.; and Weinberger, K. Q., eds., *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States*, 548–556.

McGrath, T.; Kapishnikov, A.; Tomašev, N.; Pearce, A.; Wattenberg, M.; Hassabis, D.; Kim, B.; Paquet, U.; and Kramnik, V. 2022. Acquisition of chess knowledge in AlphaZero. *Proceedings of the National Academy of Sciences*, 119(47): e2206625119.

Min, Y.; Wenkel, F.; Perlmutter, M.; and Wolf, G. 2022. Can Hybrid Geometric Scattering Networks Help Solve the Maximum Clique Problem? In *Conference on Neural Information Processing Systems (NeurIPS)*.

Morris, C.; Kriege, N. M.; Bause, F.; Kersting, K.; Mutzel, P.; and Neumann, M. 2020. TUDataset: A collection of benchmark datasets for learning with graphs. In *ICML 2020 Workshop on Graph Representation Learning and Beyond (GRL+ 2020)*.

Petsiuk, V.; Das, A.; and Saenko, K. 2018. RISE: Randomized Input Sampling for Explanation of Black-box Models. In *British Machine Vision Conference (BMVC)*, 151.

Qiu, Y.; Liu, W.; Wang, J.; and Li, R. 2024. PAGE: Parametric Generative Explainer for Graph Neural Network. In Endriss, U.; Melo, F. S.; Bach, K.; Diz, A. J. B.; Alonso-Moral, J. M.; Barro, S.; and Heintz, F., eds., *ECAI 2024 - 27th European Conference on Artificial Intelligence, 19-24 October 2024, Santiago de Compostela, Spain - Including 13th Conference on Prestigious Applications of Intelligent Systems (PAIS 2024)*, volume 392 of *Frontiers in Artificial Intelligence and Applications*, 858–865. IOS Press.

Ribeiro, M. T.; Singh, S.; and Guestrin, C. 2016. "Why Should I Trust You?" - Explaining the Predictions of Any Classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1135–1144. ACM.

Rossi, R. A.; Gleich, D. F.; and Gebremedhin, A. H. 2015. Parallel Maximum Clique Algorithms with Applications to Network Analysis. *SIAM Journal on Scientific Computing*, 37(5).

Selvaraju, R. R.; Cogswell, M.; Das, A.; Vedantam, R.; Parikh, D.; and Batra, D. 2017. Grad-CAM: Visual Explanations From Deep Networks via Gradient-Based Localization. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.

Shoham, E.; Cohen, H.; Wattad, K.; Rika, H.; and Vilenchik, D. 2026. Concept learning for algorithmic reasoning: Insights from SAT-solving GNNs. *Inf. Sci.*, 726: 122754.

Yanardag, P.; and Vishwanathan, S. 2015. Deep Graph Kernels. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '15, 1365–1374. New York, NY, USA: Association for Computing Machinery. ISBN 9781450336642.

Yang, J.; and Leskovec, J. 2012. Defining and Evaluating Network Communities Based on Ground-Truth. In Zaki, M. J.; Siebes, A.; Yu, J. X.; Goethals, B.; Webb, G. I.; and Wu, X., eds., *12th IEEE International Conference on Data Mining, ICDM 2012, Brussels, Belgium, December 10-13, 2012*, 745–754. IEEE Computer Society.

Yau, M.; Lu, E.; Karalias, N.; Xu, J.; and Jegelka, S. 2023. Are Graph Neural Networks Optimal Approximation Algorithms? *CoRR*, abs/2310.00526.