LEARNING-AUGMENTED APPROXIMATION ALGORITHMS FOR DENSEST k-SUBGRAPH

Anonymous authors
Paper under double-blind review

ABSTRACT

We study learning-augmented approximation algorithms for the Densest k-Subgraph (DkS) problem under vertex and edge predictions. In the persistent Vertex Prediction Model, each vertex receives a single noisy prediction of its membership in a fixed, unknown optimum $S^\star$; the predictions are independent and each is correct with probability $1/2 + \epsilon$. We give a polynomial-time algorithm that requires no knowledge of ϵ and combines a prediction-guided construction with prediction-free approximation algorithms. For every even $k \geq 2$, it achieves, with high probability over the predictions, an $O_\eta\big(\min\big\{k, 1 + \sqrt{\Delta \log n}/\epsilon, n^{1/4+\eta}\big\}\big)$-approximation, where Δ is the maximum degree and $\eta > 0$ is fixed.

Under the Strongish Planted Clique Hypothesis (SPCH), we show that there is a constant $c > 0$ such that, for every fixed $0 < \epsilon < 1/2$, no randomized polynomial-time algorithm using persistent vertex predictions achieves an $o(k)$-approximation or, separately, an $O(2^{c(\log n)^{2/3}})$-approximation with constant success probability. We also establish approximation guarantees for non-persistent vertex predictions and edge predictions. Experiments on real-world and planted graphs with up to millions of vertices show that moderately informative predictions often improve solution quality over prediction-free baselines while maintaining practical running times.

1 INTRODUCTION

Given an undirected graph $G = (V, E)$ and an integer k, the Densest k-Subgraph (DkS) problem asks for exactly k vertices maximizing the number of induced edges. This cardinality-constrained dense-subgraph problem generalizes k-Clique and is closely connected to community detection in machine learning and network science. Its objective serves as the exhaustive scan statistic for detecting a planted dense community, with applications to social and biological networks, event detection, and cluster monitoring (Arias-Castro & Verzelen, 2014; Hajek et al., 2015). Related planted dense-subgraph models also arise in graph clustering and community recovery (Chen & Xu, 2014). Despite extensive study, a wide gap remains between polynomial-time approximation guarantees and conditional hardness.

The best-known polynomial-time classical approximation algorithm for general DkS, due to Bhaskara, Charikar, Chlamtac, Feige, and Vijayaraghavan (BCCFV) (Bhaskara et al., 2010), achieves an $O_\eta(n^{1/4+\eta})$-approximation for any fixed $\eta > 0$, with running time $n^{O(1/\eta)}$, where $n = |V|$. On the hardness side, the Exponential Time Hypothesis (ETH) rules out polynomial-time $n^{1/(\log\log n)^c}$-approximation for some constant $c > 0$ (Manurangsi, 2017), while randomized Gap-ETH rules out $k^{o(1)}$-approximation in fixed-parameter time (Chalermsook et al., 2017). Under the Strongish Planted Clique Hypothesis (SPCH), Manurangsi et al. (2021) rule out $o(k)$-approximation in $f(k)\mathrm{poly}(n)$ time for every function f. Under a strengthened planted-clique hypothesis, Alon et al. (2011) rule out polynomial-time $2^{c(\log n)^{2/3}}$-approximation for some $c > 0$. These lower bounds concern algorithms given only the graph and k.

Learning-augmented algorithms use predictions to go beyond the worst-case analysis. They have recently attracted increasing attention, partly due to numerous advances in machine learning. Prior work has considered both online problems (Lykouris & Vassilvitskii, 2018; Purohit et al., 2018; Gupta et al., 2022) and offline combinatorial optimization (Cohen-Addad et al., 2024). Two standard goals

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

are *consistency*, outperforming the best-known classical algorithms when predictions are accurate, and *robustness*, retaining guarantees close to or matching the classical bounds when predictions are inaccurate. For DkS, we seek better approximation guarantees from informative predictions while retaining a classical fallback.

We study both persistent and non-persistent settings of the Vertex Prediction Model (Cohen-Addad et al., 2024; Braverman et al., 2024), as well as the Edge Prediction Model (Aamand et al., 2025). In the persistent setting, fix an unknown optimum $S^\star$ before drawing independent binary membership labels, each correct with probability $1/2 + \epsilon$; repeated queries return the same label. Write $m^\star := |E[S^\star]|$ for the optimum edge count. Non-persistent queries return fresh independent labels, while edge predictions supply noisy membership labels for both endpoints of each edge. We ask how much these predictions improve DkS approximation and what computational barriers remain.

Contributions. Our two main results concern persistent vertex predictions. The first gives an approximation algorithm that does not need to know the prediction accuracy.

Theorem 1 (Main approximation guarantee)**.** *Let $\eta > 0$ be fixed and Δ be the maximum degree of G. Under the persistent Vertex Prediction Model described above (formalized in Definition 3), for every even $k \in \{2, \dots, n\}$, every prediction advantage $\epsilon \in (0, 1/2)$, and every $\delta \in (0, 1)$, the algorithm in Section 4 does not need to know ϵ and returns a size-k set S such that, with probability at least $1 - \delta$,*

$$|E[S]| \geq \frac{m^\star}{\min\left\{2(k-1),\, 4 + \frac{2}{\epsilon}\sqrt{\frac{\Delta}{2}\log\frac{2n}{\delta}},\, C_\eta n^{1/4+\eta}\right\}}. \tag{1}$$

Here $C_\eta > 0$ is the constant in the BCCFV guarantee. The algorithm runs in time $n^{O(1/\eta)} + O(|E| + n\log n)$. The same approximation factor holds for degree density relative to $S^\star$.

Taking $\delta = n^{-a}$ for fixed $a > 0$ gives high probability. The prediction bound improves BCCFV's general guarantee for $\Delta = o(\epsilon^2 n^{1/2+2\eta}/\log n)$. For example, fixed ϵ and $\Delta \leq n^{1/3}$ give $O(n^{1/6}\sqrt{\log n})$, compared with $O_\eta(n^{1/4+\eta})$: the bounds differ by a factor of $n^{1/12+\eta}/\sqrt{\log n}$. This also improves the matching bound when $k = \omega(n^{1/6}\sqrt{\log n})$. Omitting BCCFV preserves the first two bounds and reduces the running time to $O(|E| + n\log n)$.

The second establishes hardness even when the prediction accuracy ϵ is known.

Theorem 2 (SPCH hardness for DkS with persistent vertex predictions)**.** *Assume SPCH. There is a constant $c > 0$ such that, for every fixed $\epsilon \in (0, 1/2)$ in the persistent setting of the Vertex Prediction Model described above (formalized in Definition 3), no randomized polynomial-time DkS algorithm, even if given ϵ, with positive input-independent constant success probability achieves an $o(k)$-approximation; separately, no such algorithm achieves an $O(2^{c(\log n)^{2/3}})$-approximation. Probability is over the persistent labels and the algorithm's internal randomness; the success constant may depend on ϵ and the algorithm.*

The first part extends the classical $o(k)$ barrier of Manurangsi et al. (2021) to polynomial-time algorithms with persistent predictions; their prediction-free result allows $f(k)\mathrm{poly}(n)$ time. For the dependence on n, Manurangsi (2017) obtains the larger prediction-free hardness factor $n^{1/(\log\log n)^C}$ under ETH, for some constant $C > 0$. Our second part applies under SPCH even when algorithms receive persistent predictions about an optimum. Its $2^{c(\log n)^{2/3}}$ dependence matches that obtained by Alon et al. (2011) under a strengthened planted-clique hypothesis. In particular, persistent predictions still do not permit any fixed polylogarithmic approximation factor in polynomial time.

Challenges and techniques. *Approximation algorithm.* The main new ingredient in our algorithm is a prediction-guided *score-and-complete* subroutine that drives our $O(1 + \sqrt{\Delta\log n}/\epsilon)$ guarantee. The intuition is to use predictions to find one side of many edges, then use the graph itself to find the other side. A positive label says little about whether a vertex belongs to $S^\star$. But after subtracting the background rate of positive labels, the labels in v's neighborhood estimate $t(v) = |N(v) \cap S^\star|$. Vertices with high scores therefore have many likely connections to the optimum, even if they do not belong to it.

We select only $k/2$ high-scoring vertices, reserving the other $k/2$ positions for their best-connected neighbors. This reservation matters: the high-scoring vertices might have few edges among themselves, as in a bipartite graph. Concentration ensures that score errors cost at most $L =$

$O(\sqrt{\Delta \log n}/\epsilon)$ in $t(v)$ per displaced optimum vertex. The first half thus retains much of the optimum's signal above L; completion turns that signal into induced edges. A matching candidate handles the signal below L, yielding the $O(1 + \sqrt{\Delta \log n}/\epsilon)$ guarantee.

Hardness. First note that a classical graph-only reduction does not immediately yield hardness with predictions: to supply valid labels, the reduction would have to know the fixed optimum $S^\star$ to which those labels are correlated. For the $o(k)$ bound, we simulate persistent labels by independently assigning every vertex a $\mathrm{Bernoulli}(1/2 - \epsilon)$ label. For any fixed $S^\star$, the probability of every label vector under this simulation is at least c_ϵ^{-k} times its probability under genuine predictions, where $c_\epsilon = (1/2+\epsilon)/(1/2-\epsilon)$. Repeating the algorithm $O_\epsilon(c_\epsilon^k)$ times recovers constant success probability. This cost is compatible with the classical $f(k)\mathrm{poly}(n)$-time hardness result, but is too large for our stronger bound in terms of n.

To avoid that exponential loss, we choose the graph-product parameters so that a YES instance contains a clique of size $M = k^3$. Every k-subset of this clique is an optimum, and the assumed algorithm must succeed for each fixed choice. Averaging genuine predictions over a uniformly chosen such optimum makes their distribution close to the easily simulated independent $\mathrm{Bernoulli}(1/2 - \epsilon)$ labels: two random targets overlap in only k^2/M vertices on average, yielding $O_\epsilon(k^{-1/2})$ total-variation distance. Enlarging the YES clique takes the product outside the range of the original soundness lemma, so we prove a stronger NO-case statement. From any overly dense k-set, we extract a sparse witness; dispersion of the sampled base vertices forces that witness to use many distinct base edges, making it unlikely in an Erdős–Rényi input. This bounds every NO-case k-set by kd edges for $d = \Theta(\sqrt{\log m})$, while a YES instance has $\binom{k}{2}$ edges, giving the claimed n-dependent hardness gap.

Further results. We also provide learning augmented algorithms for the non-persistent setting and edge prediction setting. (1) We give a constant-factor approximation for DkS with $O((n/\epsilon^2)\log(4/\delta))$ queries in the non-persistent setting, beating the naive approach that takes $\Omega(n \log n)$ queries for fixed ϵ, δ, by adapting the approach of Braverman et al. (2024) (Theorem 22). (2) We give an $O(\log(n/\delta)/\epsilon^2)$-approximation for DkS in the Edge Prediction Model of Aamand et al. (2025) (Theorem 27). Note that this approximation significantly outperforms those in Theorem 1 for fixed ϵ, δ when $k = \omega(\log n)$ and $\Delta = \omega(\log n)$. (3) Experiments on real and planted graphs with millions of vertices show improvements over classical baselines in solution quality at practical running times (Section 6).

2 Related Work

Approximation and hardness for DkS. Earlier approximation bounds include $O(n^\delta)$ for some $\delta < 1/3$ (Feige et al., 2001) and $O(n^{0.3159})$ (Goldstein & Langberg, 2009); the best-known general bound and the conditional hardness results are summarized in Section 1. Bhaskara et al. (2012) prove polynomial integrality gaps for strong Sherali–Adams and Lasserre, and Jones et al. (2023) give SoS lower bounds in the log-density regime for $k \leq \sqrt{n}$.

Practical DkS algorithms. Our experiments compare against four prediction-free DkS methods. Greedy selects a high-degree seed set and completes it using neighborhood overlap (Feige et al., 2001). Rank-1 PC instantiates a low-rank bilinear optimization framework (Papailiopoulos et al., 2014), while FW and EP-Prox use Frank–Wolfe and proximal-gradient optimization, respectively (Lu et al., 2025; Liu et al., 2026). All four methods receive only G and k.

Learning-augmented dense-subgraph algorithms. Bampis et al. (2024) also study exactly-k DkS on dense graphs with $k = \Omega(n)$, using logarithmically many one-bit predictions rather than persistent labels for every vertex. Bui et al. (2026) study the densest subgraph and densest at-most-k subgraph problems under two prediction models: a predicted vertex set with bounded false-positive and false-negative errors, and a probability vector with bounded global ℓ_p-error relative to a fixed optimum. They give $O(n + m)$-time algorithms with error-dependent approximation guarantees that approach the optimum as the prediction error decreases. Their at-most-k objective permits fewer than k vertices, whereas we require exactly k; moreover, their assumptions bound aggregate prediction error rather than using independent $1/2 + \epsilon$-accurate persistent membership labels.

Vertex Prediction Model. Braverman et al. (2024) study Maximum Independent Set in the same persistent setting of the Vertex Prediction Model and obtain an $\widetilde{O}(\sqrt{\Delta}/\epsilon)$-approximation; they also study the non-persistent setting. Cohen-Addad et al. (2024) consider offline MAX-CUT and Boolean Max-2-CSP with noisy predictions of a fixed optimum assignment, while Dong et al. (2025) study streaming MAX-CUT under a related per-vertex prediction model. Ghoshal et al. (2025) study constraint satisfaction problems with advice about a fixed optimum assignment. The additional difficulty for DkS is its exactly-k induced-edge objective, as discussed above.

Edge Prediction Model. Aamand et al. (2025) introduced the Edge Prediction Model and a high–low-degree framework for NP-hard graph problems. We instantiate this framework for DkS.

3 Problem Setup and Persistent Vertex Predictions

Let $G = (V, E)$ be an undirected simple graph, $n = |V|$, and $k \in \{1, \dots, n\}$. For nonempty $S \subseteq V$, let $E[S]$ be its induced edges and $\mathrm{dens}(S) := |E[S]|/|S|$. DkS asks for exactly k vertices with optimum

$$\mathrm{OPT}(G, k) := \max_{S \subseteq V:\, |S|=k} |E[S]|.$$

Before generating predictions, fix one optimum $S^\star$, unknown to the algorithm, and write $|S^\star| = k$ and $m^\star := |E[S^\star]| = \mathrm{OPT}(G, k)$. Since every feasible set has size k, edge count and degree density have identical approximation factors. We henceforth assume $k \geq 2$, as $k = 1$ is trivial.

Definition 3 (Persistent Vertex Prediction Model)**.** Fix $(G, k, S^\star)$ and an advantage parameter $\epsilon \in (0, 1/2)$. The oracle assigns each $u \in V$ a label $\ell(u) \in \{0, 1\}$ predicting membership in $S^\star$. Conditioned on $(G, k, S^\star)$, these labels are mutually independent and, for every u,

$$\mathbb{P}[\ell(u) = \mathbf{1}\{u \in S^\star\}] = \frac{1}{2} + \epsilon.$$

Each label is sampled once and remains fixed, so repeated queries return the same value.

Consequently,

$$\mathbb{E}\left[\ell(u) - \left(\frac{1}{2} - \epsilon\right)\right] = 2\epsilon\, \mathbf{1}\{u \in S^\star\},$$

which motivates the centered score below. Our main algorithm does not need ϵ; the lower bound holds even when ϵ is supplied.

4 Algorithm and Approximation for Persistent Vertex Predictions

In this section, we prove our main algorithmic result, Theorem 1. We first analyze a hybrid algorithm given ϵ, then remove that requirement using the global label count.

4.1 Hybrid Algorithm

Throughout this section, we assume that k is an even integer with $k \geq 2$. We first assume that ϵ is supplied. For each v, let $t(v) := |N(v) \cap S^\star|$ be its unobserved optimum-neighborhood signal and define the observable score

$$s(v) := \sum_{u \in N(v)} \ell(u) - \left(\frac{1}{2} - \epsilon\right) \deg(v). \tag{2}$$

Then $\mathbb{E}[s(v)] = 2\epsilon t(v)$, so $s(v)/(2\epsilon)$ is an unbiased estimator of $t(v)$.

Order $v_1, \dots, v_n$ by nonincreasing score with fixed tie-breaking, and put $A_h := \{v_1, \dots, v_h\}$. For $x \notin A_h$, let $c_h(x) := |N(x) \cap A_h|$; let B_h be the $k - h$ outside vertices of largest c_h, and set $S_h := A_h \cup B_h$ (with $B_k = \varnothing$).

Although $h = k/2$ suffices for the worst-case guarantee, we adaptively choose the prefix length using

$$Q(h) := |E[A_h]| + \sum_{x \in B_h} c_h(x). \tag{3}$$

Here $|E[S_h]| \geq Q(h)$. Let $\widehat{h}$ be the smallest maximizer of Q, and take the denser of $S_{\widehat{h}}$ and $S_{k/2}$. Retaining $S_{k/2}$ preserves the fixed-prefix guarantee because Q omits edges inside B_h.

Algorithm 1 Adaptive hybrid algorithm with known prediction advantage

Input: $G = (V, E)$, even $k \in \{2, \ldots, |V|\}$, the oracle in Definition 3, its advantage $\epsilon \in (0, 1/2)$, and fixed BCCFV parameter $\eta > 0$

Output: A set $S \subseteq V$ with $|S| = k$

1: Query and store every $\ell(u)$; compute and sort the scores in equation 2.
2: Compute $Q(h)$ implicitly for $1 \leq h \leq k$ by one prefix sweep; set $\widehat{h} \leftarrow \min \arg\max_{1 \leq h \leq k} Q(h)$.
3: Reconstruct $S_{\widehat{h}}$ and $S_{k/2}$; let S^{pred} be the denser one.
4: From a maximal matching M, retain $\min\{|M|, k/2\}$ edges and pad their endpoints to a size-k set S^{match}.
5: Run $\mathsf{BCCFV}_\eta(G, k)$ to obtain a size-k set S^{bccfv}.
6: **return** a set maximizing $|E[T]|$ over $T \in \{S^{\mathrm{pred}}, S^{\mathrm{match}}, S^{\mathrm{bccfv}}\}$.

For the prefix sweep, store outside vertices in buckets indexed by c_h. When v_h enters, remove it and increment each still-outside neighbor; the completion size drops by one and the cutoff moves monotonically. All updates and cutoff moves total $O(|E| + n)$. Adding sorting, matching, and BCCFV gives the time bound below.

4.2 Approximation Guarantee

For known ϵ, we prove the approximation factor $\min\{2(k-1), 4 + 2L, C_\eta n^{1/4+\eta}\}$ in equation 1. Fix $\delta \in (0, 1)$ and define

$$\tau := \sqrt{\frac{\Delta}{2} \log \frac{2n}{\delta}}, \qquad L := \frac{\tau}{\epsilon}, \qquad T_L := \sum_{v \in S^\star} (t(v) - L)_+, \tag{4}$$

where $(x)_+ := \max\{x, 0\}$; thus T_L is the optimum-neighborhood signal above the effective noise floor L.

We first show that all neighborhood scores concentrate around their expectations.

Lemma 4 (Uniform concentration of neighborhood scores)**.** *With probability at least $1 - \delta$, simultaneously for every $v \in V$, $|s(v) - 2\epsilon t(v)| \leq \tau$.*

This follows from Hoeffding's inequality and a union bound; see Appendix C.1.

Next, prefix selection and completion recover the above-noise signal.

Lemma 5 (Recovery of the above-noise signal)**.** *On the event in Lemma 4, $T_L \leq 4|E[S^{\mathrm{pred}}]|$.*

We analyze $S_{k/2}$, since $|E[S^{\mathrm{pred}}]| \geq |E[S_{k/2}]|$. Let I contain the $k/2$ optimum vertices with largest $t(v)$, and write $S_1 = A_{k/2}$. Pair each $b \in I \setminus S_1$ with a distinct $a \in S_1 \setminus I$. Score ordering and concentration give $t(a) \geq (t(b) - L)_+$, hence $\sum_{v \in S_1} t(v) \geq \sum_{v \in I} (t(v) - L)_+ \geq T_L/2$. The completion $B_{k/2}$ dominates the expected completion score of a uniform $k/2$-subset of $S^\star \setminus S_1$. Including edges inside S_1 gives

$$|E[S_{k/2}]| \geq \frac{1}{2} \sum_{v \in S_1} t(v) \geq T_L/4.$$

Appendix C.1 gives the full pairing and edge-counting argument.

The matching candidate controls the remaining signal and provides a baseline guarantee.

Lemma 6 (Guarantees from a maximal matching)**.** *The maximal-matching candidate satisfies $m^\star \leq 2(k-1)|E[S^{\mathrm{match}}]|$ and $m^\star \leq T_L + 2L|E[S^{\mathrm{match}}]|$.*

Let M be the maximal matching. If $|M| < k/2$, its endpoints form a vertex cover, giving $m^\star \leq T_L + 2L|M| \leq T_L + 2L|E[S^{\text{match}}]|$. If $|M| \geq k/2$, then $|E[S^{\text{match}}]| \geq k/2$ and $2m^\star \leq T_L + kL$, which imply the same residual bound. The baseline factor follows from $t(v) \leq k-1$ in the first case and $m^\star \leq \binom{k}{2}$ in the second. Appendix C.2 gives the complete argument.

Proof of equation 1. On the concentration event, the preceding lemmas give $m^\star \leq T_L + 2L|E[S^{\text{match}}]| \leq (4+2L)|E[S]|$. The matching and BCCFV guarantees give the other two terms in equation 1. The algorithm uses exactly n queries and time $n^{O(1/\eta)} + O(|E| + n\log n)$; the density factor is identical because $|S| = |S^\star| = k$. □

Removing the need to know ϵ. For $k \leq n/4$, query each label once and replace the score in Algorithm 1 by

$$\widehat{s}(v) := \sum_{u \in N(v)} \ell(u) - \frac{\deg(v)}{n-2k}(Z-k), \qquad Z := \sum_{u \in V} \ell(u). \tag{5}$$

Its expectation is $2\epsilon t(v)$, and the same uniform concentration bound holds, so equation 1 remains valid. For $k > n/4$, repeatedly delete a minimum-degree vertex until k remain, then take the densest of this set and the matching and BCCFV candidates. The returned set also satisfies equation 1, without querying predictions. Thus both branches prove the precise bound in Theorem 1; neither ϵ nor δ is an input. Appendix C.4 provides the concentration and degree-based arguments completing the proof. Appendix C.5 extends the known-advantage analysis to fixed feasible targets.

5 Hardness in the Persistent Vertex Prediction Model

In this section, we prove our main lower bound, Theorem 2: the $o(k)$ and, separately, $O(2^{c(\log n)^{2/3}})$ hardness bounds with persistent predictions under SPCH. SPCH rules out an $m^{o(\log m)}$-time distinguisher, with error at most $1/3$, between $\mathcal{G}(m, 1/2)$ and the same distribution with a planted $\lceil m^{\delta_{\text{SPCH}}} \rceil$-clique, for some constant $\delta_{\text{SPCH}} \in (0, 1/2)$ (Manurangsi et al., 2021).

We first remove persistent predictions with two lemmas, then analyze the graph product reduction of Manurangsi et al. (2021).

5.1 Removing Persistent Predictions

The persistent vertex predictions offer a limited advantage: we can replace their labels at a cost in success probability or running time. We use Lemma 7 for the bound in n and Lemma 8 for the $o(k)$ bound.

For the bound in n, we give every vertex an independent $\text{Bernoulli}(1/2-\epsilon)$ label. When the YES graph contains a clique of size k^3, this label distribution approaches genuine predictions averaged over its optimal k-subsets as k grows. Write $p := 1/2+\epsilon$, $q := 1/2-\epsilon$, and $\nu_\epsilon := (p-q)^2/(pq)$.

Lemma 7 (Prediction removal in graphs with a large clique)**.** *Let $\mathcal{A}$ be a randomized polynomial-time DkS algorithm in the persistent setting of the Vertex Prediction Model (Definition 3), with fixed $\epsilon \in (0, 1/2)$. Suppose G contains an M-clique, $M \geq k$, and $\mathcal{A}$ achieves a ρ-approximation with probability at least $\gamma > 0$ for every fixed optimal target $S^\star$. If predictions are replaced by independent $\text{Bernoulli}(q)$ labels, then $\mathcal{A}$ returns a ρ-approximation with probability at least*

$$\gamma - \frac{1}{2}\sqrt{\exp(\nu_\epsilon k^2/M) - 1}.$$

Proof sketch. Let H be a clique of size M. Every $S \subseteq H$ with $|S| = k$ is an optimal target. We simulate predictions by giving every vertex an independent $\text{Bernoulli}(q)$ label; denote their distribution by $\mathcal{D}_{\text{fake}}$. Genuine predictions for S use $\text{Bernoulli}(p)$ labels on S and $\text{Bernoulli}(q)$ labels elsewhere, so the replacement changes the distribution on the k vertices of S.

For the analysis, choose S uniformly among the subsets of H of size k, before generating its genuine labels. Denote the resulting mixture by $\overline{\mathcal{D}}$. The algorithm succeeds with probability at least γ for

every fixed S. Success means returning at least $\binom{k}{2}/\rho$ edges for each target, so averaging over S preserves this guarantee.

As S varies, the k vertices receiving the extra bias vary across H. This increases the expected number of ones in H by $(p-q)k$; with simulated labels, this count has standard deviation $\sqrt{Mpq}$. For fixed ϵ and $M = k^3$, these quantities have orders k and $k^{3/2}$, respectively.

We measure the difference between distributions P, Q by their chi-square divergence $\chi^2(P\|Q) := \mathbb{E}_{X\sim Q}[(P(X)/Q(X)-1)^2]$. Choose two sets $S, T \subseteq H$ independently and uniformly with $|S| = |T| = k$. Their expected overlap is k^2/M. In the second moment of the mixture likelihood ratio, each shared vertex contributes $1+\nu_\epsilon$, and every other vertex contributes 1. Hence

$$1 + \chi^2(\overline{\mathcal{D}}\|\mathcal{D}_{\text{fake}}) = \mathbb{E}_{S,T}(1+\nu_\epsilon)^{|S\cap T|} \le \exp(\nu_\epsilon k^2/M).$$

Any specified j vertices of S all belong to T with probability at most $(k/M)^j$; expanding over shared subsets gives the inequality. Cauchy–Schwarz bounds total variation, and hence the loss in success probability, by $\frac{1}{2}\sqrt{\exp(\nu_\epsilon k^2/M)-1}$. With $M = k^3$, this loss is $O_\epsilon(k^{-1/2})$, giving success probability at least $\gamma/2$ for sufficiently large k. The complete proof is in Appendix D.2. □

For the $o(k)$ bound, the simulation must work for every graph and fixed optimal target. Lemma 8 uses repetition to offset an exponential loss in one run's success probability.

Lemma 8 (Prediction removal in general graphs)**.** *Fix $\epsilon \in (0, 1/2)$ in the persistent setting of the Vertex Prediction Model (Definition 3), and let $c_\epsilon := (1/2+\epsilon)/(1/2-\epsilon)$. If a randomized polynomial-time DkS algorithm $\mathcal{A}$ in this setting returns a ρ-approximation with probability at least $\gamma > 0$ for every fixed $(G, k, S^\star)$, where γ is input-independent, then for every $\theta \in (0,1)$ there exists a prediction-free algorithm $\mathcal{A}'$ with the same approximation factor, success probability at least $1-\theta$, and time $O((1+\gamma^{-1}c_\epsilon^k \ln(1/\theta))\text{poly}(n))$.*

Proof sketch. Fix $S^\star$ for analysis. Let $\mathcal{D}_{\text{true}}$ be the genuine label distribution: independent $\text{Bernoulli}(p)$ labels on $S^\star$ and $\text{Bernoulli}(q)$ labels elsewhere. Let $\mathcal{D}_{\text{fake}}$ assign every vertex an independent $\text{Bernoulli}(q)$ label, stored for all queries in a run. The distributions differ only on the k coordinates in $S^\star$. On each such coordinate, the ratio of fake to genuine probabilities is at least $q/p = c_\epsilon^{-1}$. Independence gives $\mathcal{D}_{\text{fake}}(x) \ge c_\epsilon^{-k}\mathcal{D}_{\text{true}}(x)$ for every label vector x. The algorithm's success probability under fake labels is at least γc_ϵ^{-k}. Define $\mathcal{A}'$ to run $\mathcal{A}$ independently $T = \lceil \gamma^{-1}c_\epsilon^k \ln(1/\theta)\rceil$ times and retain the output with the most edges. The probability that all runs fail is at most $\exp(-\gamma c_\epsilon^{-k}T) \le \theta$. The complete proof is in Appendix D.1. □

5.2 The SPCH Graph Product Reduction

For the bound in n, we follow the reduction of Manurangsi et al. (2021). We choose its parameters to obtain a k^3-clique in the planted case, so that we can apply Lemma 7. Given a graph G on m vertices, we independently sample ℓ vertices uniformly with replacement for each of N product vertices. We join two product vertices if their combined samples induce a clique in G, obtaining $G' \sim \text{RP}_{N,\ell}(G)$.

Their completeness lemma gives a K-clique in G' with probability at least 0.9 if G has a $\lceil m^\delta\rceil$-clique, $K \ge 20$, and $N \ge 10Km^{(1-\delta)\ell}$. Their soundness lemma gives, for $G \sim \mathcal{G}(m, 1/2)$, at most $10^7 \log m/(\delta^2\ell)$ edges per vertex in every set of at most K product vertices with probability at least 0.9, provided $N \le 1000Km^{(1-\delta)\ell}$, $\ell \ge K \ge 20$, and $K\ell \le m^{0.99\delta}$ (Manurangsi et al., 2021, Lemmas 4–5).

Fix a rational $0 < \bar\delta < \delta_{\text{SPCH}}$, let $L_m = \lceil \log m\rceil$ and $s = \lceil 16/\bar\delta\rceil$, with logarithms to base two. Choose

$$\begin{aligned} \ell &= s\lceil\sqrt{L_m}\rceil, & k &= 2\left\lfloor m^{\bar\delta/8}/2\right\rfloor, \\ N &= \left\lceil 100k^3 m^{(1-\bar\delta)\ell}\right\rceil, & d &= \lceil 1024L_m/(\bar\delta^2\ell)\rceil. \end{aligned} \tag{6}$$

Completeness. In the planted case, each product vertex samples wholly inside the planted clique with probability at least $m^{-(1-\bar\delta)\ell}$. The expected number is at least $100k^3$, so the completeness argument above gives a k^3-clique in G' with probability at least 0.9.

Soundness. The original lemma assumes $\ell \geq K$, whereas here $k \gg \ell$. We extend its dispersion argument by counting sparse witnesses. Set $a = \bar{\delta}/2$ and $\xi = a^2/64$. With high probability, every $r \leq k$ product vertices cover at least $ar\ell/4$ base vertices. If a set of at most k vertices has more than d edges per vertex, deleting vertices of degree below d leaves a nonempty subgraph on $r \leq k$ vertices with minimum degree at least d. Choose d neighbors per vertex to form a witness of at most rd product edges. Dispersion forces at least $\xi rd\ell^2$ distinct base edges, so a fixed witness occurs under $\mathcal{G}(m, 1/2)$ with probability at most $2^{-\xi rd\ell^2}$. There are at most $N^r r^{rd}$ witnesses, and our parameters give $\sum_{r=1}^{k} N^r r^{rd} 2^{-\xi rd\ell^2} \leq 1/(N-1)$. Including dispersion failures, every set of k vertices has at most kd edges with probability at least 0.9 (see Appendix D.3 for the full argument).

Gap and time. These choices give $\log N = \Theta((\log m)^{3/2})$ and $d = \Theta(\sqrt{\log m})$. For a constant $c > 0$ depending only on $\bar{\delta}$, the ratio of the YES and NO bounds is

$$\frac{\binom{k}{2}}{kd} = \frac{k-1}{2d} \geq 2^{2c(\log N)^{2/3}}. \tag{7}$$

We construct G' by checking all pairs of product vertices in $O(m^2 + N^2\ell^2)$ time. Including a polynomial-time algorithm on G', the total time is $N^{O(1)} = m^{O(\sqrt{\log m})} = m^{o(\log m)}$.

Proof sketch of Theorem 2. For the $o(k)$ claim, Lemma 8 with $\theta = 1/3$ preserves the factor and gives success at least $2/3$ without predictions in $O(c_\epsilon^k \mathrm{poly}(n))$ time, contradicting Manurangsi et al. (2021, Theorem 2).

For the bound in n, suppose $\rho(N) \leq C2^{c(\log N)^{2/3}}$ for a constant $C > 0$. Give each run on G' fresh independent $\mathrm{Bernoulli}(q)$ labels. In the planted case, the k^3-clique and Lemma 7 give success at least $\gamma/2$ per run for sufficiently large m, with input-independent $\gamma > 0$. By equation 7, success returns a k-set with more than kd edges. Accept if any of $T = \lceil (2/\gamma) \ln 20 \rceil$ independent runs does so. The YES acceptance probability is at least $0.9 \cdot 0.95 > 2/3$, and the NO probability is at most $0.1 < 1/3$. This violates SPCH in the time above (Appendix D.4). □

6 Experiments

We evaluate LAA from Algorithm 1 with the known generation parameter ϵ in equation 2. The implementation retains the adaptive and fixed prefix candidates and the matching candidate, and omits the BCCFV fallback.

6.1 Experimental Setup

Graphs and targets. Table 1 lists three SNAP graphs (Leskovec et al., 2007), two OGB graphs (Hu et al., 2020), and `cit-Patents` (Leskovec et al., 2005). Gurobi (Gurobi Optimization, LLC, 2026) certifies the optima for nine small instances; the nine targets on large graphs are feasible references frozen before label generation. We also generate five graphs per scale for four families with planted cliques in $G(n, 0.5)$, and five LFR graphs with one million vertices each (Lancichinetti et al., 2008) for each $\mu \in \{0.5, 0.6, 0.7\}$ using NetworKit (Staudt et al., 2016). The targets are the planted cliques and selected LFR communities. See Appendix F.1 for graph generation and reference construction.

Methods and measurements. For each fixed target and $\epsilon \in \{0, 0.01, \ldots, 0.50\}$, we sample ten persistent label vectors by independently labeling each vertex correctly with probability $1/2 + \epsilon$. LAA and its two ablations share these vectors. *Label Only* uniformly subsamples or pads the positive-label set to obtain exactly k vertices; *Score Only* selects the k vertices with the largest scores in equation 2. We compare them with Greedy (Feige et al., 2001), Rank1PC (Papailiopoulos et al., 2014), FW (Lu et al., 2025), and EP-Prox (Liu et al., 2026), each run once per graph. We average $\mathrm{dens}(S) = |E[S]|/k$ over labels and, for synthetic families, over the means from five graphs. The grid endpoints are diagnostic; Appendix F.1 gives implementation details, handling of odd k, and timing protocol.

Table 1: Preprocessed real-world instances. *Offline ref.* is a frozen feasible target without an optimality certificate.

Dataset	n	m	Values of k	Target
`ca-AstroPh`	18,772	198,050	200, 500, 1000	Exact OPT
`ca-CondMat`	23,133	93,439	150, 300, 600	Exact OPT
`ca-HepTh`	9,875	25,973	81, 161, 322	Exact OPT
`ogbl-collab`	235,868	967,632	500, 1000, 2000	Offline ref.
`ogbn-products`	2,449,029	61,859,012	4000, 8000, 15000	Offline ref.
`cit-Patents`	3,774,768	16,518,947	2000, 4000, 8000	Offline ref.

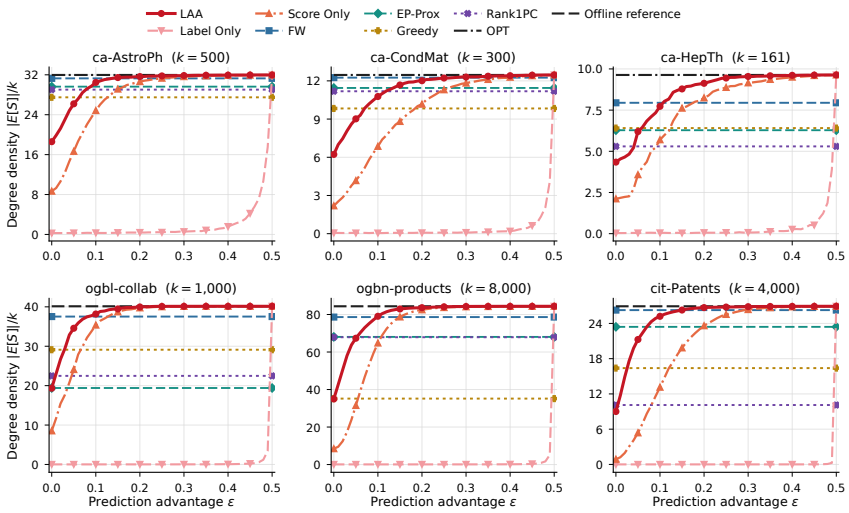

Figure 1: Degree density on six real graphs at $k \approx \sqrt{m}$. Targets are certified optima (top) or frozen references (bottom). Complete results with standard-error ribbons appear in Appendix F.2.

6.2 Results

We call the first evaluated ϵ at which LAA's mean density exceeds the strongest baseline's mean under the same graph aggregation the *crossover threshold*. This is a comparison of means on the evaluated grid; we do not infer statistical significance.

Real graphs. LAA surpasses the strongest baseline on all 18 configurations on real graphs (Figure 1 and table 2). The crossover threshold ranges from 0.11 to 0.35 for certified optima and from 0.07 to 0.17 for the frozen references. For large graphs, a higher value than the reference establishes a denser feasible size-k set.

Planted graphs. Across the four clique families, the crossover thresholds are 0.27–0.29, and LAA curves averaged over graphs reach the target objective at 0.47–0.48 (Appendix F.3). For the nine LFR configurations these ranges are 0.11–0.21 and 0.27–0.48, respectively. Attaining the clique objective shows that the output is a size-k clique, which may differ from the planted clique. Attaining an LFR community's edge count gives no optimality certificate or guarantee of recovering its vertices.

Ablations and runtime. Score Only usually exceeds Label Only; LAA usually exceeds both at low and intermediate ϵ. LAA is faster than all four baselines on the 13 planted configurations and faster than FW, EP-Prox, and Rank1PC on the nine large real instances. Greedy is faster on all 18 real instances. These comparisons exclude graph loading and common preprocessing; Appendices F.1 and F.4 give the timing protocol and complete runtime results.

AI USE STATEMENT

We used generative AI to assist with related work discovery, manuscript drafting, language editing, and formatting. For the theoretical results, the authors independently developed the approximation algorithm, derived its approximation guarantees, and developed the proof ruling out an $o(k)$-approximation. AI tools were used to help check these proofs for correctness. For the hardness bound in terms of the number of vertices n, we followed the reduction framework of Manurangsi et al. (2021) and used AI to assist with parameter selection and the derivation of specific proof steps. The authors reviewed the AI-assisted material and verified the final arguments. All experimental code was generated using AI tools and subsequently checked and validated by the authors. The authors also verified the experimental results and reviewed the final manuscript, and take responsibility for all its contents, including AI-assisted text, proofs, and code.

REPRODUCIBILITY STATEMENT

Complete proofs are provided in Appendices C to E. Experimental protocols and implementation details are described in Appendix F. The experimental code is available at `https://anonymous.4open.science/r/ICLR2027-LAA-DkS-code-91642`.

REFERENCES

Anders Aamand, Justin Y. Chen, Siddharth Gollapudi, Sandeep Silwal, and Hao Wu. Improved approximations for hard graph problems using predictions. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 73–101. PMLR, 13–19 Jul 2025. URL `https://proceedings.mlr.press/v267/aamand25c.html`.

Noga Alon, Sanjeev Arora, Rajsekar Manokaran, Dana Moshkovitz, and Omri Weinstein. Inapproximability of densest κ-subgraph from average case hardness. Unpublished manuscript, 2011. URL `https://www.tau.ac.il/~nogaa/PDFS/dks8.pdf`.

Ery Arias-Castro and Nicolas Verzelen. Community detection in dense random networks. *The Annals of Statistics*, 42(3):940–969, 2014. doi: 10.1214/14-AOS1208.

Evripidis Bampis, Bruno Escoffier, and Michalis Xefteris. Parsimonious learning-augmented approximations for dense instances of $\mathcal{NP}$-hard problems. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 2700–2714. PMLR, 21–27 Jul 2024. URL `https://proceedings.mlr.press/v235/bampis24a.html`.

Aditya Bhaskara, Moses Charikar, Eden Chlamtac, Uriel Feige, and Aravindan Vijayaraghavan. Detecting high log-densities: an $O(n^{1/4})$ approximation for densest k-subgraph. In *Proceedings of the Forty-Second ACM Symposium on Theory of Computing*, STOC '10, pp. 201–210, New York, NY, USA, 2010. Association for Computing Machinery. ISBN 9781450300506. doi: 10.1145/1806689.1806719. URL `https://doi.org/10.1145/1806689.1806719`.

Aditya Bhaskara, Moses Charikar, Venkatesan Guruswami, Aravindan Vijayaraghavan, and Yuan Zhou. Polynomial integrality gaps for strong SDP relaxations of densest k-subgraph. In Yuval Rabani (ed.), *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 388–405. SIAM, 2012. doi: 10.1137/1.9781611973099.34.

Vladimir Braverman, Prathamesh Dharangutte, Vihan Shah, and Chen Wang. Learning-Augmented Maximum Independent Set. In Amit Kumar and Noga Ron-Zewi (eds.), *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2024)*, volume 317 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pp. 24:1–24:18, Dagstuhl, Germany, 2024. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. ISBN 978-3-95977-348-5. doi: 10.4230/LIPIcs.APPROX/RANDOM.2024.24. URL `https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.APPROX/RANDOM.2024.24`.

Thai Bui, Luan Nguyen, and Hoa T. Vu. Fast and simple densest subgraph with predictions, 2026. URL https://arxiv.org/abs/2505.12600.

Parinya Chalermsook, Marek Cygan, Guy Kortsarz, Bundit Laekhanukit, Pasin Manurangsi, Danupon Nanongkai, and Luca Trevisan. From Gap-ETH to FPT-inapproximability: Clique, dominating set, and more. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, pp. 743–754, 2017. doi: 10.1109/FOCS.2017.74.

Yudong Chen and Jiaming Xu. Statistical-computational phase transitions in planted models: The high-dimensional setting. In *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pp. 244–252. PMLR, 2014.

Vincent Cohen-Addad, Tommaso d'Orsi, Anupam Gupta, Euiwoong Lee, and Debmalya Panigrahi. Learning-augmented approximation algorithms for maximum cut and related problems. In *Advances in Neural Information Processing Systems*, volume 37, pp. 25961–25980. Curran Associates, Inc., 2024. doi: 10.52202/079017-0816. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/2db08b94565c0d582cc53de6cee5fd47-Paper-Conference.pdf.

Yinhao Dong, Pan Peng, and Ali Vakilian. Learning-Augmented Streaming Algorithms for Approximating MAX-CUT. In Raghu Meka (ed.), *16th Innovations in Theoretical Computer Science Conference (ITCS 2025)*, volume 325 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pp. 44:1–44:24, Dagstuhl, Germany, 2025. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. ISBN 978-3-95977-361-4. doi: 10.4230/LIPIcs.ITCS.2025.44. URL https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.ITCS.2025.44.

Uriel Feige, Guy Kortsarz, and David Peleg. The dense k-subgraph problem. *Algorithmica*, 29(3): 410–421, 2001. doi: 10.1007/s004530010050.

Suprovat Ghoshal, Konstantin Makarychev, and Yury Makarychev. Constraint satisfaction problems with advice. In Yossi Azar and Debmalya Panigrahi (eds.), *Proceedings of the 2025 Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2025, New Orleans, LA, USA, January 12-15, 2025*, pp. 1202–1221. SIAM, 2025. doi: 10.1137/1.9781611978322.36. URL https://doi.org/10.1137/1.9781611978322.36.

Doron Goldstein and Michael Langberg. The dense k subgraph problem. *CoRR*, abs/0912.5327, 2009. doi: 10.48550/arXiv.0912.5327. URL https://arxiv.org/abs/0912.5327v2.

Anupam Gupta, Debmalya Panigrahi, Bernardo Subercaseaux, and Kevin Sun. Augmenting online algorithms with ε-accurate predictions. In *Advances in Neural Information Processing Systems*, volume 35, pp. 2115–2127. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/0ea048312aa812b2711fe765a9e9ef05-Abstract-Conference.html.

Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual, 2026. URL https://www.gurobi.com.

Bruce Hajek, Yihong Wu, and Jiaming Xu. Computational lower bounds for community detection on random graphs. In Peter Grünwald, Elad Hazan, and Satyen Kale (eds.), *Proceedings of The 28th Conference on Learning Theory*, volume 40 of *Proceedings of Machine Learning Research*, pp. 899–928, Paris, France, 03–06 Jul 2015. PMLR. URL https://proceedings.mlr.press/v40/Hajek15.html.

Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 22118–22133. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/fb60d411a5c5b72b2e7d3527cfc84fd0-Paper.pdf.

Chris Jones, Aaron Potechin, Goutham Rajendran, and Jeff Xu. Sum-of-squares lower bounds for densest k-subgraph. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, pp. 84–95. Association for Computing Machinery, 2023. doi: 10.1145/3564246.3585221.

Andrea Lancichinetti, Santo Fortunato, and Filippo Radicchi. Benchmark graphs for testing community detection algorithms. *Physical Review E*, 78(4):046110, 2008. doi: 10.1103/PhysRevE.78.046110.

Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. Graphs over time: Densification laws, shrinking diameters and possible explanations. In *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*, KDD '05, pp. 177–187, New York, NY, USA, 2005. Association for Computing Machinery. ISBN 159593135X. doi: 10.1145/1081870.1081893. URL https://doi.org/10.1145/1081870.1081893.

Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. Graph evolution: Densification and shrinking diameters. *ACM Trans. Knowl. Discov. Data*, 1(1):2–es, March 2007. ISSN 1556-4681. doi: 10.1145/1217299.1217301. URL https://doi.org/10.1145/1217299.1217301.

Ya Liu, Junbin Liu, Wing-Kin Ma, and Aritra Konar. A scalable and exact relaxation for densest k-subgraph via error bounds. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 15360–15368, 2026. doi: 10.1609/aaai.v40i18.38562.

Qiheng Lu, Nicholas D. Sidiropoulos, and Aritra Konar. Densest k-subgraph mining via a provably tight relaxation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 12291–12299, 2025. doi: 10.1609/aaai.v39i12.33339.

Thodoris Lykouris and Sergei Vassilvitskii. Competitive caching with machine learned advice. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 3296–3305. PMLR, 2018. URL https://proceedings.mlr.press/v80/lykouris18a.html.

Pasin Manurangsi. Almost-polynomial ratio ETH-hardness of approximating densest k-subgraph. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2017, pp. 954–961, New York, NY, USA, 2017. Association for Computing Machinery. ISBN 9781450345286. doi: 10.1145/3055399.3055412. URL https://doi.org/10.1145/3055399.3055412.

Pasin Manurangsi, Aviad Rubinstein, and Tselil Schramm. The Strongish Planted Clique Hypothesis and Its Consequences. In James R. Lee (ed.), *12th Innovations in Theoretical Computer Science Conference (ITCS 2021)*, volume 185 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pp. 10:1–10:21, Dagstuhl, Germany, 2021. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. ISBN 978-3-95977-177-1. doi: 10.4230/LIPIcs.ITCS.2021.10. URL https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.ITCS.2021.10.

Dimitris Papailiopoulos, Ioannis Mitliagkas, Alexandros Dimakis, and Constantine Caramanis. Finding dense subgraphs via low-rank bilinear optimization. In Eric P. Xing and Tony Jebara (eds.), *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pp. 1890–1898. PMLR, 2014. URL https://proceedings.mlr.press/v32/papailiopoulos14.html.

Manish Purohit, Zoya Svitkina, and Ravi Kumar. Improving online algorithms via ML predictions. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper/2018/hash/73a427badebe0e32caa2e1fc7530b7f3-Abstract.html.

Christian L. Staudt, Aleksejs Sazonovs, and Henning Meyerhenke. NetworKit: A tool suite for large-scale complex network analysis. *Network Science*, 4(4):508–530, 2016. doi: 10.1017/nws.2016.20.

A Overview of the Appendix

The appendix is organized as follows:

- Appendix B: Technical preliminaries, including the BCCFV guarantee and a size-reduction lemma.
- Appendix C: Complete proofs for persistent vertex predictions, including unknown prediction advantage and fixed feasible targets.
- Appendix D: Complete hardness proofs, including prediction removal, optimal-target mixtures, and the graph-product reduction.
- Appendix E: Definitions, algorithms, and guarantees for non-persistent vertex predictions and edge predictions.
- Appendix F: Experimental protocols, implementation details, complete solution-quality results, and runtime comparisons.

B Deferred Technical Preliminaries

We first recall the prediction-free approximation guarantee used by the BCCFV candidate.

Theorem 9 (BCCFV approximation Bhaskara et al. (2010))**.** *For every fixed constant $\eta > 0$, there is an algorithm BCCFV_η that, on every n-vertex DkS instance (G, k), runs in time $n^{O(1/\eta)}$ and returns a set S^{bccfv} of size k satisfying*

$$|E[S^{\mathrm{bccfv}}]| \geq \frac{1}{C_\eta n^{1/4+\eta}} \max_{T \subseteq V:\, |T|=k} |E[T]|,$$

where $C_\eta > 0$ depends only on η. If an implementation returns at most k vertices, arbitrary padding to size k cannot decrease the induced edge count.

The following lemma enforces the exact-cardinality constraint while retaining a constant fraction of the induced edges.

Lemma 10 (Reducing at most $2k$ vertices to exactly k)**.** *Let $k \geq 2$ and $W \subseteq V$ with $|W| \leq 2k$. A size-k set $\widehat{S} \subseteq V$ satisfying*

$$|E[\widehat{S}]| \geq \frac{1}{6}|E[W]|$$

can be found in polynomial time.

Proof. If $|W| < k$, pad W arbitrarily to size k. Now suppose $k \leq m := |W| \leq 2k$ and choose a uniformly random k-subset $\widehat{S} \subseteq W$. Each edge of $E[W]$ is retained with probability

$$\frac{\binom{m-2}{k-2}}{\binom{m}{k}} = \frac{k(k-1)}{m(m-1)} \geq \frac{k(k-1)}{2k(2k-1)} = \frac{k-1}{2(2k-1)} \geq \frac{1}{6}.$$

Therefore $\mathbb{E}|E[\widehat{S}]| \geq |E[W]|/6$. The method of conditional expectation derandomizes the choice in polynomial time. □

C Complete Proofs for the Persistent Vertex-Prediction Result

C.1 Prediction-Guided Recovery Above the Noise Floor

Recall that

$$t(v) = |N(v) \cap S^\star|$$

and

$$s(v) = \sum_{u \in N(v)} \left(\ell(u) - \left(\frac{1}{2} - \epsilon \right) \right).$$

For disjoint sets $X, Y \subseteq V$, let $E[X, Y]$ denote the set of edges with one endpoint in X and the other in Y.

The first lemma identifies the signal captured by the centered neighborhood score.

Lemma 11 (Unbiased neighborhood scores)**.** *For every $v \in V$,*

$$\mathbb{E}[s(v)] = 2\epsilon t(v).$$

Proof. For each $u \in N(v)$,

$$\mathbb{E}[\ell(u)] = \begin{cases} \frac{1}{2} + \epsilon, & u \in S^\star, \\ \frac{1}{2} - \epsilon, & u \notin S^\star. \end{cases}$$

After subtracting $1/2 - \epsilon$, a neighbor in $S^\star$ contributes 2ϵ in expectation, while a neighbor outside $S^\star$ contributes zero. Summing over $N(v)$ proves the claim. □

Remark 12 (Why calibration is required). The centered score relies on the exact identities

$$\mathbb{P}[\ell(u) = 1 \mid u \in S^\star] = \frac{1}{2} + \epsilon, \qquad \mathbb{P}[\ell(u) = 1 \mid u \notin S^\star] = \frac{1}{2} - \epsilon.$$

If the model merely lower-bounded the correctness probability, the expected centered contribution of a nonmember would not necessarily be zero. The positive result is therefore proved under the calibrated persistent model of Definition 3.

Proof of Lemma 4. Fix a non-isolated vertex v. The labels of the vertices in $N(v)$ are mutually independent, and the centered summands defining $s(v)$ lie in intervals of length one. By Lemma 11 and Hoeffding's inequality,

$$\begin{aligned} \mathbb{P}\left[|s(v) - 2\epsilon t(v)| \geq \tau\right] &\leq 2\exp\left(-\frac{2\tau^2}{\deg(v)}\right) \\ &\leq 2\exp\left(-\frac{2\tau^2}{\Delta}\right) \\ &= 2\exp\left(-\log\frac{2n}{\delta}\right) = \frac{\delta}{n}. \end{aligned}$$

For an isolated vertex, $s(v) = t(v) = 0$, so the desired inequality holds deterministically. A union bound over all n vertices proves that, with probability at least $1 - \delta$,

$$|s(v) - 2\epsilon t(v)| \leq \tau$$

simultaneously for every $v \in V$. □

The next lemma translates uniform score concentration into a lower bound on the signal retained by the top-score vertices.

Lemma 13 (Truncated-signal replacement)**.** *Let $I \subseteq S^\star$ consist of the $k/2$ vertices with the largest values of $t(v)$, and let*

$$S_1 := A_{k/2}$$

consist of the $k/2$ vertices with the largest scores $s(v)$. On the event

$$|s(v) - 2\epsilon t(v)| \leq \tau \qquad \textit{for every } v \in V,$$

we have

$$\sum_{v \in S_1} t(v) \geq \sum_{v \in I} (t(v) - L)_+ \geq \frac{1}{2} T_L.$$

Proof. Define

$$X := S_1 \setminus I, \qquad Y := I \setminus S_1.$$

Since $|S_1| = |I| = k/2$, the sets X and Y have the same cardinality. Pair every $b \in Y$ with a distinct vertex $a \in X$. Because S_1 consists of the $k/2$ largest-score vertices,

$$s(a) \geq s(b).$$

On the concentration event,

$$2\epsilon t(a) + \tau \geq s(a) \geq s(b) \geq 2\epsilon t(b) - \tau.$$

Therefore,

$$t(a) \geq t(b) - \frac{\tau}{\epsilon} = t(b) - L.$$

Since $t(a) \geq 0$, this implies

$$t(a) \geq (t(b) - L)_+.$$

Summing over all pairs and using

$$t(v) \geq (t(v) - L)_+$$

for every $v \in S_1 \cap I$, we obtain

$$\sum_{v \in S_1} t(v) \geq \sum_{v \in I} (t(v) - L)_+.$$

The function $x \mapsto (x - L)_+$ is nondecreasing. Hence I also contains the largest $k/2$ values among the k nonnegative quantities

$$\{(t(v) - L)_+ : v \in S^\star\}.$$

The sum of the largest half of k nonnegative numbers is at least half their total sum. Consequently,

$$\sum_{v \in I} (t(v) - L)_+ \geq \frac{1}{2} \sum_{v \in S^\star} (t(v) - L)_+ = \frac{1}{2} T_L.$$

□

We next show that the completion step converts this retained signal into induced edges.

Lemma 14 (Completion converts signal into induced edges)**.** *Let*

$$S_2 := B_{k/2}$$

consist of the $k/2$ vertices outside $S_1 = A_{k/2}$ with the largest completion scores

$$c(v) := |N(v) \cap S_1|.$$

Then

$$|E[S_1 \cup S_2]| \geq \frac{1}{2} \sum_{v \in S_1} t(v).$$

Proof. Define

$$A := S_1 \setminus S^\star, \qquad B := S^\star \setminus S_1, \qquad C := S_1 \cap S^\star.$$

Since $|S_1| = k/2$, we have

$$\frac{k}{2} \leq |B| \leq k.$$

Because S_2 consists of the $k/2$ vertices with the largest completion scores, its total completion score is at least the expected score of a uniformly random $k/2$-vertex subset of B. Therefore,

$$\begin{aligned} |E[S_1, S_2]| = \sum_{v \in S_2} c(v) &\geq \frac{k/2}{|B|} \sum_{v \in B} c(v) \\ &= \frac{k/2}{|B|} |E[S_1, B]| \\ &\geq \frac{1}{2} \left(|E[A, B]| + |E[B, C]| \right). \end{aligned}$$

On the other hand, expanding the optimum-neighborhood signal over A and C gives

$$\sum_{v \in S_1} t(v) = |E[A, B]| + |E[A, C]| + |E[B, C]| + 2|E[C]|.$$

Moreover,

$$|E[A,C]| + 2|E[C]| \le 2|E[A]| + 2|E[A,C]| + 2|E[C]| \\ = 2|E[S_1]|.$$

Combining these inequalities yields

$$\sum_{v \in S_1} t(v) \le 2|E[S_1, S_2]| + 2|E[S_1]| \\ \le 2|E[S_1 \cup S_2]|.$$

Rearranging proves the lemma. □

Proof of Lemma 5. Assume the event in Lemma 4. By Lemmas 13 and 14,

$$|E[S_{k/2}]| = |E[S_1 \cup S_2]| \ge \frac{1}{2} \sum_{v \in S_1} t(v) \ge \frac{1}{4} T_L.$$

Since S^{pred} is the better of $S_{\widehat{h}}$ and $S_{k/2}$,

$$|E[S^{\mathrm{pred}}]| \ge |E[S_{k/2}]|.$$

Consequently,

$$T_L \le 4|E[S^{\mathrm{pred}}]|.$$

□

C.2 Residual Control via a Maximal Matching

Proof of Lemma 6. Let M be the maximal matching used by the algorithm and define

$$r := \min\{|M|, k/2\}.$$

The r selected matching edges are contained in $G[S^{\mathrm{match}}]$. Hence

$$|E[S^{\mathrm{match}}]| \ge r.$$

We first prove the prediction-independent guarantee. Suppose that $|M| < k/2$, and let U be the set of all endpoints of M. Since M is maximal, U is a vertex cover of G: otherwise, an edge with both endpoints in $V \setminus U$ could be added to M. It follows that $U \cap S^\star$ is a vertex cover of $G[S^\star]$. Therefore,

$$\begin{aligned} m^\star &\le \sum_{u \in U \cap S^\star} t(u) \\ &\le (k-1)|U \cap S^\star| \\ &\le 2(k-1)|M| \\ &\le 2(k-1)|E[S^{\mathrm{match}}]|. \end{aligned}$$

If instead $|M| \ge k/2$, then

$$|E[S^{\mathrm{match}}]| \ge \frac{k}{2}$$

and

$$m^\star \le \binom{k}{2} \le (k-1)|E[S^{\mathrm{match}}]| \le 2(k-1)|E[S^{\mathrm{match}}]|.$$

Thus, in both cases,

$$m^\star \le 2(k-1)|E[S^{\mathrm{match}}]|.$$

We next prove the residual guarantee. Suppose first that $|M| < k/2$, and let U again denote the set of all endpoints of M. As shown above, $U \cap S^\star$ is a vertex cover of $G[S^\star]$, and hence

$$m^\star \le \sum_{u \in U \cap S^\star} t(u).$$

Using

$$t(u) \le (t(u) - L)_+ + L,$$

we obtain

$$\begin{aligned} m^\star &\le \sum_{u \in U \cap S^\star} (t(u) - L)_+ + L|U \cap S^\star| \\ &\le T_L + 2L|M| \\ &\le T_L + 2L|E[S^{\mathrm{match}}]|. \end{aligned}$$

Finally, suppose that $|M| \ge k/2$. By the handshake lemma in $G[S^\star]$,

$$2m^\star = \sum_{v \in S^\star} t(v).$$

Again using $t(v) \le (t(v) - L)_+ + L$, we obtain

$$2m^\star \le T_L + kL.$$

Since $|E[S^{\mathrm{match}}]| \ge k/2$,

$$\begin{aligned} m^\star &\le \frac{1}{2} T_L + \frac{kL}{2} \\ &\le \frac{1}{2} T_L + L|E[S^{\mathrm{match}}]| \\ &\le T_L + 2L|E[S^{\mathrm{match}}]|. \end{aligned}$$

This proves both claimed guarantees. □

C.3 Combining the Three Candidates

The remaining candidate supplies the classical prediction-free guarantee.

Lemma 15 (Prediction-free fallback guarantee)**.** *The BCCFV candidate satisfies*

$$|E[S^{\mathrm{bccfv}}]| \ge \frac{m^\star}{C_\eta n^{1/4+\eta}}.$$

Proof. By Theorem 9,

$$|E[S^{\mathrm{bccfv}}]| \ge \frac{1}{C_\eta n^{1/4+\eta}} \max_{T \subseteq V : |T| = k} |E[T]|.$$

The maximum on the right-hand side is $m^\star$, which proves the claim. □

Analysis of the known-advantage algorithm. We prove the factor in equation 1 for Algorithm 1, which receives ϵ.

For each $h \in \{\widehat{h}, k/2\}$, the sets A_h and B_h are disjoint and have cardinalities h and $k-h$, respectively. Therefore, each set $S_h = A_h \cup B_h$ contains exactly k vertices, and so does S^{pred}. The matching candidate contains the endpoints of $r \le k/2$ disjoint matching edges and is padded to size k when necessary. The BCCFV candidate also contains exactly k vertices.

The first guarantee in Lemma 6 and the final comparison among the candidates deterministically give

$$m^\star \le 2(k-1)|E[S^{\mathrm{match}}]| \le 2(k-1)|E[S]|.$$

By Lemma 4, with probability at least $1 - \delta$, the concentration event holds simultaneously for all vertices. Condition on this event. Then Lemma 5 gives

$$T_L \le 4|E[S^{\mathrm{pred}}]|.$$

Combining this inequality with the second guarantee in Lemma 6, we obtain

$$\begin{aligned} m^\star &\le T_L + 2L|E[S^{\mathrm{match}}]| \\ &\le 4|E[S^{\mathrm{pred}}]| + 2L|E[S^{\mathrm{match}}]| \\ &\le (4 + 2L)|E[S]|. \end{aligned}$$

Substituting

$$L = \frac{1}{\epsilon}\sqrt{\frac{\Delta}{2}\log\frac{2n}{\delta}}$$

gives

$$|E[S]| \geq \frac{m^\star}{4 + \frac{2}{\epsilon}\sqrt{\frac{\Delta}{2}\log\frac{2n}{\delta}}}.$$

Independently, Lemma 15 and the final comparison among the candidates imply

$$|E[S]| \geq \frac{m^\star}{C_\eta n^{1/4+\eta}}.$$

Taking the strongest of the three lower bounds yields

$$|E[S]| \geq \frac{m^\star}{\min\left\{2(k-1),\, 4 + \frac{2}{\epsilon}\sqrt{\frac{\Delta}{2}\log\frac{2n}{\delta}},\, C_\eta n^{1/4+\eta}\right\}}.$$

Because both S and $S^\star$ contain exactly k vertices, dividing their induced-edge counts by k gives the same approximation factor for density.

Finally, the algorithm makes exactly n oracle queries. Computing the neighborhood scores takes $O(|E| + n)$ time, and sorting the vertices by score takes $O(n \log n)$ time. During the prefix sweep, inserting a vertex only increments the completion scores of its remaining neighbors, so every edge causes at most one update. The cutoff separating the $k - h$ largest completion scores moves monotonically through the frequency buckets, and therefore all values $Q(h)$ are computed in $O(|E| + n)$ total time after sorting. Constructing the two selected completion sets and computing a maximal matching require another $O(|E| + n)$ time. Together with the BCCFV call, the total running time is

$$n^{O(1/\eta)} + O(|E| + n \log n).$$

□

C.4 Unknown Prediction Advantage

We retain the label distribution of Definition 3, but do not supply ϵ to the algorithm. In particular, the target $S^\star$ is fixed before its independent persistent labels are drawn. The input consists of G, an even $k \in \{2, \ldots, n\}$, fixed $\eta > 0$, and access to these labels.

Guarantee. We complete the proof of Theorem 1 by removing the known-advantage assumption. For every $\epsilon \in (0, 1/2)$ and $\delta \in (0, 1)$, the construction below preserves the factor in equation 1 over the full range of even k, with probability at least $1 - \delta$. It uses at most n persistent-label queries and runs in time $n^{O(1/\eta)} + O(|E| + n \log n)$. Neither ϵ nor δ is required as an input.

Construction. As in equation 5, if $k \leq n/4$, query each label once and define

$$Z := \sum_{u \in V} \ell(u), \qquad \widehat{s}(v) := \sum_{u \in N(v)} \ell(u) - \frac{\deg(v)}{n - 2k}(Z - k).$$

Here Z is the total number of positive labels. Replace $s(v)$ by $\widehat{s}(v)$ in Algorithm 1, keeping the same fixed tie-breaking, prefix completion, and prediction-free candidates. The scores require only G, k, and the observed labels. Multiplying them by the positive integer $n - 2k$ gives integer scores with the same ordering.

If $k > n/4$, start with all vertices and repeatedly delete a vertex of minimum current degree until exactly k remain. Return the densest of this set and the matching and BCCFV candidates from Algorithm 1. This branch uses no predictions.

Completion of the proof of Theorem 1. The case $k \le n/4$. Write $q := 1/2 - \epsilon$. Since the fixed target has exactly k vertices,

$$\mathbb{E}Z = nq + 2\epsilon k, \qquad \mathbb{E}[Z - k] = (n - 2k)q, \qquad \mathbb{E}[\widehat{s}(v)] = 2\epsilon t(v).$$

Thus the global correction removes the same expected background contribution as the original score, without knowing ϵ.

Fix a vertex v, and write $d := \deg(v)$ and $\beta := d/(n - 2k)$. The centered score is

$$\widehat{s}(v) - 2\epsilon t(v) = (1 - \beta) \sum_{u \in N(v)} (\ell(u) - \mathbb{E}\ell(u)) - \beta \sum_{u \notin N(v)} (\ell(u) - \mathbb{E}\ell(u)).$$

These are independent centered labels with deterministic coefficients. The squared interval lengths of their weighted contributions sum to

$$d(1 - \beta)^2 + (n - d)\beta^2 = d + \frac{d^2(4k - n)}{(n - 2k)^2} \le d,$$

where the last inequality uses $k \le n/4$. With τ as in equation 4, Hoeffding's inequality therefore gives, for $d > 0$,

$$\mathbb{P}[|\widehat{s}(v) - 2\epsilon t(v)| > \tau] \le 2\exp(-2\tau^2/d) \le \delta/n.$$

Isolated vertices have zero score error. A union bound yields the same uniform error bound as Lemma 4, now for $\widehat{s}$. This calculation uses independence of the original labels; it does not assume that the global correction and the neighborhood sum are independent.

Set $L = \tau/\epsilon$ for analysis only. The replacement and completion argument in Appendix C.1 uses only score ordering and this uniform error bound. It therefore applies with $\widehat{s}$ in place of s. The matching and BCCFV arguments are unchanged, so the candidate analysis in Appendix C.3 gives exactly the factor in equation 1.

The case $k > n/4$. If $\Delta = 0$, then $m^\star = 0$ and the guarantee is trivial. If $k = 2$, the matching guarantee has factor $2(k - 1) = 2$, which is at most $4 + 2L$. We therefore assume $\Delta \ge 1$ and $k \ge 4$. Let e_r be the number of edges remaining when the deletion process has r vertices. A minimum-degree vertex has degree at most $2e_r/r$, so $e_{r-1} \ge (r - 2)e_r/r$. Multiplying these bounds gives

$$e_k \ge \frac{k(k-1)}{n(n-1)}|E| \ge \frac{k(k-1)}{n(n-1)}m^\star.$$

Since $n \le 4k - 1$ and $k \ge 4$,

$$\frac{n(n-1)}{k(k-1)} \le \frac{(4k-1)(4k-2)}{k(k-1)} = 16 + \frac{4k+2}{k(k-1)} \le \frac{35}{2}.$$

The matching candidate also has approximation factor at most 2Δ. Indeed, if the maximal matching M has fewer than $k/2$ edges, its endpoint set U is a vertex cover, all its endpoints are retained, and

$$m^\star \le \sum_{u \in U \cap S^\star} t(u) \le 2\Delta|M| \le 2\Delta|E[S^{\mathrm{match}}]|.$$

If $|M| \ge k/2$, then $|E[S^{\mathrm{match}}]| \ge k/2$, whereas $m^\star \le k\Delta/2$; this even gives factor Δ. The denser of the deletion and matching candidates consequently has factor at most $\min\{35/2, 2\Delta\}$.

We now compare this factor with $4 + 2L$. Since $\epsilon < 1/2$ and $\delta < 1$,

$$L = \frac{1}{\epsilon}\sqrt{\frac{\Delta}{2}\log\frac{2n}{\delta}} > \sqrt{2\Delta\log(2n)}.$$

For $\Delta \le 2$, we have $2\Delta \le 4$. For $3 \le \Delta \le 7$, using $n \ge 4$ gives

$$\frac{(\Delta - 2)^2}{2\Delta} \le \frac{25}{14} < \log 8 \le \log(2n),$$

and hence $2\Delta \le 4 + 2L$. For $\Delta \ge 8$, the graph is simple, so $n \ge 9$ and

$$4 + 2L > 4 + 2\sqrt{16 \log 18} > \frac{35}{2}.$$

Thus the returned set satisfies $m^\star \le (4 + 2L)|E[S]|$ in every case. Together with the unchanged matching and BCCFV guarantees, this proves the exact factor in equation 1, and hence Theorem 1, for all even k.

Finally, the first branch queries each vertex once; the second makes no queries. The global count costs $O(n)$ time, and the first branch otherwise retains the implementation of Algorithm 1. Degree buckets implement minimum-degree deletion in $O(|E| + n)$ total time. Including matching and BCCFV gives the stated running time. □

C.5 Guarantees Relative to a Fixed Feasible Target

The target need not be optimal for the persistent upper-bound argument. Fix an even $k \in \{2, \dots, n\}$ and a set $T \subseteq V$ with $|T| = k$, before generating any labels. Each vertex receives one independent persistent label satisfying

$$\mathbb{P}[\ell(u) = \mathbf{1}\{u \in T\}] = \frac{1}{2} + \epsilon, \qquad \epsilon \in (0, 1/2).$$

The known-advantage algorithm receives ϵ and these labels, but does not know T. The label distribution differs from Definition 3 only in allowing the prediction target to be nonoptimal.

Guarantee. For every $\delta \in (0, 1)$ and fixed $\eta > 0$, Algorithm 1 returns a size-k set S such that, with probability at least $1 - \delta$ over the labels,

$$|E[S]| \ge \max\left\{ \frac{\mathrm{OPT}(G,k)}{2(k-1)}, \quad \frac{|E[T]|}{4 + 2L}, \quad \frac{\mathrm{OPT}(G,k)}{C_\eta n^{1/4+\eta}} \right\}, \qquad L := \frac{1}{\epsilon}\sqrt{\frac{\Delta}{2} \log \frac{2n}{\delta}}.$$

The algorithm makes n queries and runs in time $n^{O(1/\eta)} + O(|E| + n \log n)$, as established in Appendix C.3. If BCCFV is omitted, the first two bounds still hold, with n queries and $O(|E| + n \log n)$ time.

Proof. Write $t_T(v) := |N(v) \cap T|$ for the number of neighbors of v in the target, and let $R_T := \sum_{v \in T} (t_T(v) - L)_+$. The score expectation and concentration arguments in Appendix C.1 give $\mathbb{E}[s(v)] = 2\epsilon t_T(v)$ and, with probability at least $1 - \delta$, $|s(v) - 2\epsilon t_T(v)| \le \epsilon L$ for every vertex. On this event, the replacement and completion arguments use only $|T| = k$, not optimality, and yield $R_T \le 4|E[S^{\mathrm{pred}}]|$.

Let M be the maximal matching. If $|M| < k/2$, its endpoint set U is a vertex cover, so

$$|E[T]| \le \sum_{u \in U \cap T} t_T(u) \le R_T + 2L|M| \le R_T + 2L|E[S^{\mathrm{match}}]|.$$

If $|M| \ge k/2$, then $|E[S^{\mathrm{match}}]| \ge k/2$ and $2|E[T]| = \sum_{v \in T} t_T(v) \le R_T + kL$, which implies the same bound. Since the algorithm keeps the densest candidate,

$$|E[T]| \le R_T + 2L|E[S^{\mathrm{match}}]| \le (4 + 2L)|E[S]|.$$

The prediction-free matching and BCCFV guarantees continue to hold against $\mathrm{OPT}(G, k)$, proving the other two bounds. The algorithm and its implementation are unchanged. □

If $|E[T]| \ge a\,\mathrm{OPT}(G, k)$ for some $a \in (0, 1]$, the prediction term gives approximation factor $(4 + 2L)/a$ against $\mathrm{OPT}(G, k)$. Without a target-quality certificate, that term compares only with $|E[T]|$. In particular, this result applies to the fixed reference sets and LFR communities in Section 6 when k is even, under the stated independent label-generation model; it does not certify those targets as optimal.

D Proofs for Hardness in the Persistent Vertex Prediction Model

This section proves two prediction-removal lemmas, analyzes the randomized graph product at the required parameter values, and completes both parts of Theorem 2.

D.1 Prediction Removal in General Graphs

Fix a DkS instance (G, k), with $G = (V, E)$ and $n := |V|$. In this subsection and the next, $X = (X_u)_{u \in V}$ is the random prediction vector, and $x = (x_u)_{u \in V} \in \{0,1\}^V$ is one possible value. Thus x has n binary coordinates, with x_u recording the prediction for vertex u. For any label distribution $\mathcal{D}$, $\mathcal{D}(x)$ denotes the probability of generating the entire vector x.

Fix an optimum $S^\star$ only for the analysis; the reduction does not know it. For the fixed prediction advantage $\epsilon \in (0, 1/2)$, let

$$p := \frac{1}{2} + \epsilon, \qquad q := \frac{1}{2} - \epsilon, \qquad c_\epsilon := \frac{p}{q} > 1.$$

Under the genuine distribution $\mathcal{D}_{\text{true}}$, the coordinates of X are independent, with X_u distributed as $\text{Bernoulli}(p)$ on $S^\star$ and $\text{Bernoulli}(q)$ elsewhere. Under the simulated distribution $\mathcal{D}_{\text{fake}}$, every coordinate is independently $\text{Bernoulli}(q)$. Each run samples its vector once and answers every query to u with the stored bit X_u, ensuring persistence even for adaptive or repeated queries.

Complete proof of Lemma 8. The two distributions agree outside $S^\star$. On each coordinate in $S^\star$, the fake-to-true likelihood ratio is either q/p or p/q. Independence therefore gives, for every prediction vector x,

$$\mathcal{D}_{\text{fake}}(x) \geq c_\epsilon^{-k} \mathcal{D}_{\text{true}}(x).$$

Multiply this pointwise bound by the conditional probability that $\mathcal{A}$ returns a feasible ρ-approximation given x, and sum over all $x \in \{0,1\}^V$. Including the algorithm's internal randomness, we obtain

$$\mathbb{P}_{X \sim \mathcal{D}_{\text{fake}}}[\mathcal{A} \text{ succeeds}] \geq c_\epsilon^{-k} \mathbb{P}_{X \sim \mathcal{D}_{\text{true}}}[\mathcal{A} \text{ succeeds}] \geq \gamma c_\epsilon^{-k}.$$

Define $\mathcal{A}'$ as follows. Independently sample $T := \lceil \gamma^{-1} c_\epsilon^k \ln(1/\theta) \rceil$ fake label vectors, run $\mathcal{A}$ with fresh internal randomness on each, and validate every output, replacing an infeasible output by an arbitrary size-k set. Return the feasible output inducing the most edges. All runs fail with probability at most

$$(1 - \gamma c_\epsilon^{-k})^T \leq \exp(-\gamma c_\epsilon^{-k} T) \leq \theta.$$

Any successful run supplies the required approximation, and the total time is $O((1 + \gamma^{-1} c_\epsilon^k \ln(1/\theta))\text{poly}(n))$. □

D.2 Prediction Removal in Graphs with a Large Clique

The purpose of the calculation is to control the loss in success probability when genuine predictions are replaced by simulated labels. After averaging over optimal targets, it suffices to bound a total variation distance. We first relate this distance to χ^2-divergence, then bound the divergence through the overlap of two auxiliary targets.

Complete proof of Lemma 7. Averaging over optimal targets. Fix an M-clique $H \subseteq V$, with $M \geq k$. Every k-subset $S \subseteq H$ is an optimal target, since it induces $\binom{k}{2}$ edges. Success means returning a size-k set with at least $\binom{k}{2}/\rho$ edges, the same condition for every target. By hypothesis, $\mathcal{A}$ succeeds with probability at least γ for each fixed S.

Let $\mathcal{D}_S$ be the genuine distribution of the random prediction vector $X = (X_u)_{u \in V}$ for this target: its coordinates are independent, with X_u distributed as $\text{Bernoulli}(p)$ on S and $\text{Bernoulli}(q)$ elsewhere. A realization $x = (x_u)_{u \in V} \in \{0,1\}^V$ has n binary coordinates, and $\mathcal{D}_S(x)$ is the probability of this entire vector.

For analysis, first choose S uniformly from $\binom{H}{k}$, the collection of all k-subsets of H, and then draw X from $\mathcal{D}_S$. Thus the target is fixed before its independent labels are generated. The resulting *mixture*

distribution satisfies

$$\overline{\mathcal{D}}(x) := \frac{1}{\binom{M}{k}} \sum_{S \in \binom{H}{k}} \mathcal{D}_S(x),$$

$$\mathbb{P}_{X \sim \overline{\mathcal{D}}}[\mathcal{A} \text{ succeeds}] = \mathbb{E}_S \mathbb{P}_{X \sim \mathcal{D}_S}[\mathcal{A} \text{ succeeds}] \geq \gamma.$$

All success probabilities also include the algorithm's independent internal randomness. This averaging preserves the guarantee for legal fixed targets; the mixture is only a tool for the analysis.

Reducing the success loss to total variation. The reduction samples $\mathcal{D}_{\text{fake}}$, giving every vertex an independent $\text{Bernoulli}(q)$ label and storing it for all later queries. We compare this distribution with $\overline{\mathcal{D}}$; no closeness to an individual $\mathcal{D}_S$ is required.

The total variation distance

$$\text{TV}(\overline{\mathcal{D}}, \mathcal{D}_{\text{fake}}) := \frac{1}{2} \sum_{x \in \{0,1\}^V} |\overline{\mathcal{D}}(x) - \mathcal{D}_{\text{fake}}(x)|$$

bounds the change in the probability of any event. Adding the algorithm's internal randomness preserves this bound, since its random coins have the same independent distribution in both experiments. Applying the bound to the common success event gives

$$\mathbb{P}_{X \sim \mathcal{D}_{\text{fake}}}[\mathcal{A} \text{ succeeds}] \geq \gamma - \text{TV}(\overline{\mathcal{D}}, \mathcal{D}_{\text{fake}}).$$

Thus our remaining task is to bound this total variation distance.

Passing from total variation to χ^2-divergence. For each fixed vector x, the quantity $\overline{\mathcal{D}}(x)/\mathcal{D}_{\text{fake}}(x) - 1$ is the relative difference between its two probabilities. Averaging the square of this difference under the fake distribution defines

$$\chi^2(\overline{\mathcal{D}} \| \mathcal{D}_{\text{fake}}) := \sum_{x \in \{0,1\}^V} \mathcal{D}_{\text{fake}}(x) \left(\frac{\overline{\mathcal{D}}(x)}{\mathcal{D}_{\text{fake}}(x)} - 1 \right)^2.$$

The sum weights each full prediction vector by its probability under $\mathcal{D}_{\text{fake}}$; all denominators are positive because $0 < q < 1$. Factoring $\mathcal{D}_{\text{fake}}(x)$ out of the absolute difference and applying Cauchy–Schwarz gives

$$\begin{aligned} \text{TV}(\overline{\mathcal{D}}, \mathcal{D}_{\text{fake}}) &= \frac{1}{2} \mathbb{E}_{X \sim \mathcal{D}_{\text{fake}}} \left| \frac{\overline{\mathcal{D}}(X)}{\mathcal{D}_{\text{fake}}(X)} - 1 \right| \\ &\leq \frac{1}{2} \sqrt{\chi^2(\overline{\mathcal{D}} \| \mathcal{D}_{\text{fake}})}. \end{aligned}$$

This is why a mean squared difference suffices to control the probability loss. With $\nu_\epsilon := (p - q)^2/(pq)$, it remains to prove

$$\chi^2(\overline{\mathcal{D}} \| \mathcal{D}_{\text{fake}}) \leq \exp(\nu_\epsilon k^2/M) - 1.$$

Expressing the second moment through target overlap. The square in χ^2 is useful because it turns the mixture into an average over pairs of targets. First, the mixture's likelihood ratio has mean one under the fake distribution:

$$\mathbb{E}_{X \sim \mathcal{D}_{\text{fake}}} \frac{\overline{\mathcal{D}}(X)}{\mathcal{D}_{\text{fake}}(X)} = \sum_{x \in \{0,1\}^V} \overline{\mathcal{D}}(x) = 1.$$

The last equality holds because $\overline{\mathcal{D}}$ is a probability distribution. Expanding the square in the definition of χ^2 therefore shows that $1 + \chi^2$ is the second moment of this ratio.

For a fixed target S, define its likelihood ratio $\Lambda_S(x)$ by comparing the probability of the same complete vector x under $\mathcal{D}_S$ and $\mathcal{D}_{\text{fake}}$. Coordinates outside S cancel. For $u \in S$, $x_u = 1$ contributes p/q, whereas $x_u = 0$ contributes q/p. Independence, followed by averaging over targets, gives

$$\Lambda_S(x) := \frac{\mathcal{D}_S(x)}{\mathcal{D}_{\text{fake}}(x)} = \prod_{u \in S} \left[\left(\frac{p}{q} \right)^{x_u} \left(\frac{q}{p} \right)^{1 - x_u} \right],$$

$$\frac{\overline{\mathcal{D}}(x)}{\mathcal{D}_{\text{fake}}(x)} = \frac{1}{\binom{M}{k}} \sum_{S \in \binom{H}{k}} \Lambda_S(x) = \mathbb{E}_S \Lambda_S(x).$$

For a fixed x, this target average is a deterministic value that still depends on x. Its square averages products over all ordered pairs of targets. Introduce independent uniform $S, T \in \binom{H}{k}$, independently of $X \sim \mathcal{D}_{\text{fake}}$. Then

$$\begin{aligned} 1 + \chi^2(\overline{\mathcal{D}} \| \mathcal{D}_{\text{fake}}) &= \mathbb{E}_{X \sim \mathcal{D}_{\text{fake}}} \big(\mathbb{E}_S \Lambda_S(X) \big)^2 \\ &= \mathbb{E}_{S,T} \mathbb{E}_{X \sim \mathcal{D}_{\text{fake}}} [\Lambda_S(X) \Lambda_T(X)]. \end{aligned}$$

The inner expectation fixes S, T and averages the label vector X; the outer expectation averages the target pair. These targets are auxiliary indices in the calculation, and the reduction selects neither of them.

Under $\mathcal{D}_{\text{fake}}$, each coordinate is one with probability q and zero with probability $p = 1 - q$. The corresponding factor in a single likelihood ratio has first and second moments

$$q \frac{p}{q} + p \frac{q}{p} = 1, \qquad q \left(\frac{p}{q} \right)^2 + p \left(\frac{q}{p} \right)^2 = 1 + \nu_\epsilon.$$

In $\Lambda_S(X) \Lambda_T(X)$, a vertex belonging to only one target contributes a factor with mean one. A shared vertex uses the same random bit X_u in both ratios, so its factor is squared and has mean $1 + \nu_\epsilon$. Distinct coordinates remain independent. Evaluating the inner expectation for fixed S, T, then averaging over the target pair, gives

$$\begin{aligned} \mathbb{E}_{X \sim \mathcal{D}_{\text{fake}}} [\Lambda_S(X) \Lambda_T(X)] &= (1 + \nu_\epsilon)^{|S \cap T|}, \\ 1 + \chi^2(\overline{\mathcal{D}} \| \mathcal{D}_{\text{fake}}) &= \mathbb{E}_{S,T} (1 + \nu_\epsilon)^{|S \cap T|}. \end{aligned}$$

We have reduced the divergence bound to controlling the overlap of two random optimal targets.

Bounding the overlap. Condition on the first auxiliary target S, keeping T uniform. This fixes an index only for the calculation below. The resulting bound will hold for every S, so it can then be averaged over S. Let $Z := |S \cap T|$, a random variable counting shared vertices. Each of the k vertices in S belongs to T with probability k/M, hence $\mathbb{E}_T Z = k^2/M$.

To bound the exponential moment, expand it before taking the expectation:

$$\mathbb{E}_T (1 + \nu_\epsilon)^Z = \sum_{j=0}^{k} \nu_\epsilon^j \mathbb{E}_T \binom{Z}{j},$$

where $\binom{Z}{j} = 0$ for $j > Z$. This expansion is useful because $\binom{Z}{j}$ counts the j-subsets of S contained entirely in T, so its expectation can be obtained by counting. For any fixed such subset U, the remaining $k - j$ vertices of T must come from $H \setminus U$, giving

$$\mathbb{P}_T[U \subseteq T] = \frac{\binom{M-j}{k-j}}{\binom{M}{k}} = \prod_{i=0}^{j-1} \frac{k-i}{M-i}.$$

There are $\binom{k}{j}$ choices of U. By linearity of expectation,

$$\mathbb{E}_T \binom{Z}{j} = \binom{k}{j} \prod_{i=0}^{j-1} \frac{k-i}{M-i} \leq \binom{k}{j} \left(\frac{k}{M} \right)^j,$$

where the empty product is one and each ratio $(k-i)/(M-i) \leq k/M$. Substituting this bound into the expansion yields

$$\begin{aligned} \mathbb{E}_T (1 + \nu_\epsilon)^Z &\leq \sum_{j=0}^{k} \binom{k}{j} (\nu_\epsilon k / M)^j \\ &= (1 + \nu_\epsilon k / M)^k \leq \exp(\nu_\epsilon k^2 / M). \end{aligned}$$

The subset count justifies this bound despite dependence between the vertex-inclusion events. Averaging over S now gives the required estimate

$$\chi^2(\overline{\mathcal{D}} \| \mathcal{D}_{\text{fake}}) \leq \exp(\nu_\epsilon k^2 / M) - 1.$$

Completing the probability transfer. Substituting this estimate into the Cauchy–Schwarz bound, and then into the success-probability inequality established above, yields

$$\mathbb{P}_{X\sim\mathcal{D}_{\text{fake}}}[\mathcal{A}\text{ succeeds}] \geq \gamma - \frac{1}{2}\sqrt{\exp(\nu_\epsilon k^2/M) - 1}. \qquad \square$$

For $M = k^3$ and fixed ϵ, the loss is $O_\epsilon(k^{-1/2})$, hence at most $\gamma/2$ for sufficiently large k. The reduction only samples and stores a fake label vector and runs $\mathcal{A}$; it needs neither the clique H nor any target.

D.3 A Randomized Graph Product with a Large Clique Gap

Throughout the remainder of this section, logarithms are to base two unless written as $\ln$. We use the randomized graph product of Manurangsi et al. (2021): m is the base-graph size, N is the product-graph size, ℓ is the product length, and k is the target cardinality. Thus ℓ here is not a prediction label. Let $L_m := \lceil \log m \rceil$, and define

$$\operatorname{den}(H) := \frac{|E(H)|}{|V(H)|}, \qquad \operatorname{den}_{\leq k}(H) := \max_{\substack{\emptyset \neq W \subseteq V(H) \\ |W| \leq k}} \operatorname{den}(H[W]).$$

To sample $G' \sim \mathrm{RP}_{N,\ell}(G)$, independently sample ℓ base vertices uniformly with replacement for each $i \in [N]$, and write S_i for their support. The product vertices are the distinct indices, even when their supports coincide. Indices $i \neq j$ are adjacent exactly when $S_i \cup S_j$ induces a clique in G.

The following proposition states the completeness and soundness gap needed for the hardness reduction.

Proposition 16 (Large-clique graph-product gap)**.** *Fix a rational $\bar{\delta} \in (0, \delta_{\mathrm{SPCH}})$. There is a constant $c > 0$ such that, for all sufficiently large m, parameters N, ℓ, k, d, with k even, can be chosen so that*

$$\ell = \Theta_{\bar{\delta}}(\sqrt{\log m}), \qquad k = \Theta(m^{\bar{\delta}/8}), \qquad \log N = \Theta_{\bar{\delta}}((\log m)^{3/2}), \qquad d = \Theta_{\bar{\delta}}(\sqrt{\log m}),$$

and the following hold:

- *(i) If G contains a $\lceil m^{\bar{\delta}} \rceil$-clique, then $G' \sim \mathrm{RP}_{N,\ell}(G)$ contains a k^3-clique with probability at least 0.9.*
- *(ii) If $G \sim \mathcal{G}(m, 1/2)$ and $G' \sim \mathrm{RP}_{N,\ell}(G)$, then $\operatorname{den}_{\leq k}(G') \leq d$ with probability at least 0.9.*

Moreover, $\binom{k}{2}/(kd) \geq 2^{2c(\log N)^{2/3}}$, and the product can be constructed in $m^{O_{\bar{\delta}}(\sqrt{\log m})}$ time.

The graph-product construction is unchanged. We enlarge the planted product clique by choosing N, and extend soundness to larger k through sparse witnesses in which each vertex specifies only d neighbors. The next two lemmas justify this parameter range.

Lemma 17 (Dispersion of sampled product vertices)**.** *Let $a \in (0, 1)$, and let $m \geq 2$, N, ℓ, k be positive integers such that $N \leq m^{(1-a)\ell}$ and $k\ell \leq m^{a/2}$. With failure probability at most $m^{-a\ell/4}/(1 - m^{-a\ell/4})$, the sampled supports satisfy*

$$\left|\bigcup_{i\in J} S_i\right| \geq \frac{a}{4}|J|\ell \quad \textit{for every nonempty } J \subseteq [N] \textit{ with } |J| \leq k. \tag{8}$$

Proof. Fix J of size $r \leq k$, and put $t := \lceil ar\ell/4 \rceil - 1 \leq ar\ell/4$. Failure of Equation (8) means that the $r\ell$ independent samples are supported on at most t base vertices. For $t = 0$, this is impossible. Otherwise its probability is at most

$$\binom{m}{t}(t/m)^{r\ell} \leq m^t (t/m)^{r\ell} \leq \left(\frac{t}{m^{1-a/4}}\right)^{r\ell}.$$

There are at most N^r choices for J. Since $N \leq m^{(1-a)\ell}$ and $t \leq k\ell \leq m^{a/2}$, the probability that any size-r set fails is at most

$$\left(\frac{t}{m^{3a/4}}\right)^{r\ell} \leq m^{-ar\ell/4}.$$

Summing over $r \geq 1$ gives the stated bound. The event depends only on the sampled supports, independently of the base-graph edges. □

We now use dispersion to bound the probability of a dense product subgraph through sparse witnesses.

Lemma 18 (Soundness from sparse witnesses)**.** *Under the hypotheses of Lemma 17, suppose $N \geq 2$. Set $\xi := a^2/64$ and $d := \lceil 4L_m/(\xi\ell) \rceil$, and assume*

$$\log k \leq \frac{\xi}{2}\ell^2, \qquad ad\ell \geq 8.$$

For $G \sim \mathcal{G}(m, 1/2)$ and $G' \sim \mathrm{RP}_{N,\ell}(G)$,

$$\mathbb{P}[\mathrm{den}_{\leq k}(G') > d] \leq \frac{m^{-a\ell/4}}{1 - m^{-a\ell/4}} + \frac{1}{N-1}.$$

Proof. Fix sampled supports satisfying Equation (8). If a subgraph on at most k vertices has density greater than d, iteratively deleting vertices of degree less than d leaves a nonempty subgraph with minimum degree at least d. Otherwise counting edges when their first endpoint is deleted would give fewer than d edges per original vertex, a contradiction.

Let $J \subseteq [N]$ be the remaining index set, with $|J| = r \leq k$. For each vertex, choose d of its neighbors, and let F consist of all chosen undirected edges. Then F has minimum degree at least d and at most rd edges. For fixed J, there are at most

$$\binom{r-1}{d}^r \leq r^{rd} = 2^{rd \log r}$$

such choices. When $r \leq d$, no such witness exists.

For a fixed witness F, define

$$V_G(F) := \bigcup_{i \in J} S_i, \qquad E_G(F) := \bigcup_{\{i,j\} \in E(F)} \big\{\{u, v\} : u \in S_i,\ v \in S_j,\ u \neq v\big\}.$$

Dispersion gives $|V_G(F)| \geq ar\ell/4$. For any $u \in V_G(F)$, choose an index $i \in J$ with $u \in S_i$. Applying Equation (8) to the neighbor-index set of i in F shows that their supports cover at least $ad\ell/4$ base vertices. This application is valid because the neighbor-index set has size at most $r - 1 \leq k$. All but possibly u are adjacent to u in $(V_G(F), E_G(F))$. As $ad\ell \geq 8$, its degree is at least $ad\ell/4 - 1 \geq ad\ell/8$, and hence

$$|E_G(F)| \geq \frac{1}{2}\frac{ar\ell}{4}\frac{ad\ell}{8} = \xi r d \ell^2.$$

If $F \subseteq G'$, all these distinct base edges must be present. They are independent under $\mathcal{G}(m, 1/2)$, after the supports have been fixed, so

$$\mathbb{P}_G[F \subseteq G'] \leq 2^{-\xi r d \ell^2}.$$

The witness need not be induced; ignoring further clique constraints only increases this probability bound. A union bound over its choices gives

$$\mathbb{P}_G[J \text{ admits a witness}] \leq 2^{rd \log r - \xi r d \ell^2} \leq 2^{-(\xi/2) r d \ell^2},$$

where $\log r \leq \log k \leq \xi\ell^2/2$. Finally, $N \leq m^{(1-a)\ell} \leq 2^{L_m \ell}$ and the choice of d imply

$$\begin{aligned}\mathbb{P}_G[\mathrm{den}_{\leq k}(G') > d \mid (S_i)_{i \in [N]}] &\leq \sum_{r=1}^{k} N^r 2^{-(\xi/2) r d \ell^2} \\ &\leq \sum_{r=1}^{k} N^r 2^{-2 r L_m \ell} \leq \sum_{r=1}^{\infty} N^{-r} = \frac{1}{N-1}.\end{aligned}$$

This bound holds for every realization satisfying dispersion. Adding its failure probability from Lemma 17 proves the claim. □

Complete proof of Proposition 16. Fix $\bar{\delta}$ as in the proposition and set $s := \lceil 16/\bar{\delta} \rceil$. For all sufficiently large m, take

$$\begin{aligned} \ell &:= s\lceil \sqrt{L_m} \rceil, & k &:= 2\left\lfloor \frac{m^{\bar{\delta}/8}}{2} \right\rfloor, \\ N &:= \left\lceil 100k^3 m^{(1-\bar{\delta})\ell} \right\rceil, & d &:= \left\lceil \frac{1024 L_m}{\bar{\delta}^2 \ell} \right\rceil. \end{aligned} \tag{9}$$

These choices give the four asymptotic relations in the statement, with k even and $2 \le k \le N$.

For completeness, fix a $\lceil m^{\bar{\delta}} \rceil$-clique in G. Each sampled product vertex falls entirely in it with probability at least $m^{-(1-\bar{\delta})\ell}$, independently of the other samples. The number Z of such vertices is therefore binomial, with $\mathbb{E}Z \ge 100k^3$. Chernoff's inequality gives $\mathbb{P}[Z < k^3] \le \exp(-\Omega(k^3)) = o(1)$. Their indices form a clique in G', proving part (i).

For soundness, use $a := \bar{\delta}/2$, so that $\xi = a^2/64 = \bar{\delta}^2/256$. We check all hypotheses of Lemmas 17 and 18. Since $\log k \le (\bar{\delta}/8)\log m$ and $\ell \to \infty$,

$$\log N \le (1-\bar{\delta})\ell \log m + 3\log k + \log 200 \le (1-\bar{\delta}/2)\ell \log m$$

for all sufficiently large m. Thus $N \le m^{(1-a)\ell}$. Also,

$$k\ell = O_{\bar{\delta}}(m^{\bar{\delta}/8}\sqrt{\log m}) \le m^{\bar{\delta}/4} = m^{a/2}.$$

The sparse-witness condition follows from

$$\log k \le \frac{\bar{\delta}}{8} L_m \le \frac{1}{2} L_m \le \frac{\xi}{2} s^2 L_m \le \frac{\xi}{2}\ell^2,$$

because $s \ge 16/\bar{\delta}$. The chosen d is exactly $\lceil 4L_m/(\xi\ell) \rceil$, and $ad\ell \ge 4aL_m/\xi \ge 8$ eventually. The failure bound in Lemma 18 tends to zero, proving part (ii).

In the planted case, $\mathrm{OPT}(G', k) = \binom{k}{2}$ on the completeness event. In the null case, every size-k set induces at most kd edges on the soundness event. Their gap satisfies

$$g(m) := \frac{\binom{k}{2}}{kd} = \frac{k-1}{2d} = \Omega_{\bar{\delta}}\left(\frac{m^{\bar{\delta}/8}}{\sqrt{\log m}} \right).$$

For all sufficiently large m,

$$\log g(m) \ge \frac{\bar{\delta}}{16} L_m, \qquad \log N \le \ell L_m \le 2s L_m^{3/2}.$$

Consequently the constant

$$c := \frac{\bar{\delta}}{32(2s)^{2/3}} > 0$$

satisfies $g(m) \ge 2^{2c(\log N)^{2/3}}$. Finally, constructing the product by checking all pairs of sampled supports takes $O(m^2 + N^2\ell^2)$ time, which is $2^{O_{\bar{\delta}}(L_m^{3/2})} = m^{O_{\bar{\delta}}(\sqrt{\log m})}$. All parameters are determined by m and fixed rational constants; the integer powers and roots required for the roundings can be computed within this time bound. □

D.4 Completing the Hardness Proof

Complete proof of Theorem 2. Fix $\epsilon \in (0, 1/2)$, and suppose that a randomized polynomial-time algorithm $\mathcal{A}$ under Definition 3 succeeds with probability at least $\gamma_\epsilon > 0$ for every fixed $(G, k, S^\star)$, where γ_ϵ is independent of the input.

For an $o(k)$-approximation, apply Lemma 8 with $\theta = 1/3$. The resulting prediction-free algorithm preserves the factor, succeeds with probability at least $2/3$, and runs in $O(c_\epsilon^k)\mathrm{poly}(n)$ time. This contradicts Manurangsi et al. (2021, Theorem 2), which rules out an $o(k)$-approximation in $f(k)\mathrm{poly}(n)$ time for every function f, even under perfect completeness.

For the bound in n, fix a rational $\bar{\delta} \in (0, \delta_{\mathrm{SPCH}})$ and take c from Proposition 16, and suppose that

$$\rho(N) \le C\, 2^{c(\log N)^{2/3}}$$

for some constant $C > 0$ and all sufficiently large N. Given an m-vertex SPCH instance G, sample $G' \sim \mathrm{RP}_{N,\ell}(G)$ using Equation (9). Generate $T := \lceil (2/\gamma_\epsilon) \ln 20 \rceil$ independent fake label vectors, each with independent $\mathrm{Bernoulli}(1/2 - \epsilon)$ coordinates, and run $\mathcal{A}$ on (G', k) with fresh internal randomness each time. Store each run's labels to ensure persistence. Validate outputs and accept exactly when a feasible output induces more than kd edges.

In the planted case, G contains a $\lceil m^{\bar{\delta}} \rceil$-clique. With probability at least 0.9, the product contains a k^3-clique. Condition on any such product graph. By Lemma 7, each fake-label run returns a $\rho(N)$-approximation with probability at least

$$\gamma_\epsilon - \frac{1}{2}\sqrt{\exp(\nu_\epsilon/k) - 1} \ge \gamma_\epsilon/2$$

for all sufficiently large m. The probability that all T runs fail is at most $\exp(-\gamma_\epsilon T/2) \le 0.05$. Moreover,

$$\frac{\binom{k}{2}}{kd} \ge 2^{2c(\log N)^{2/3}} > C\, 2^{c(\log N)^{2/3}} \ge \rho(N)$$

eventually, so any successful run exceeds the acceptance threshold. The completeness probability is at least $0.9 \cdot 0.95 > 2/3$.

In the null case, with probability at least 0.9, every size-k output has at most kd edges, so the reduction rejects regardless of the labels and internal randomness. Its soundness error is at most $0.1 < 1/3$, without invoking any approximation guarantee for fake labels.

Since T is constant, construction, simulation, and output validation take total time

$$N^{O(1)}\mathrm{poly}(m, \ell) = 2^{O_{\bar{\delta}}(L_m^{3/2})} = m^{O_{\bar{\delta}}(\sqrt{\log m})} = m^{o(\log m)},$$

contradicting SPCH. Truncate $\mathcal{A}$ at its polynomial time bound on every simulated transcript if necessary. The constant c depends only on $\bar{\delta}$; the minimum instance size may depend on ϵ, γ_ϵ, and C. □

D.5 Additional Consequences

The hardness bound also yields a formulation in terms of the maximum degree.

Corollary 19 (Maximum-degree formulation)**.** *Assume SPCH, and let Δ be the maximum degree. There is a constant $c > 0$ such that, for every fixed $\epsilon \in (0, 1/2)$, no randomized polynomial-time algorithm in the persistent setting of the Vertex Prediction Model, with positive input-independent constant success probability, achieves an $O(2^{c(\log(\Delta+2))^{2/3}})$-approximation.*

Complete proof of Corollary 19. Take the constant c from Theorem 2. Since $\Delta + 2 \le n + 1$ and $(\log(n+1))^{2/3} - (\log n)^{2/3} = o(1)$, such an algorithm would also achieve an $O(2^{c(\log n)^{2/3}})$-approximation, contradicting that theorem. □

Remark 20 (Larger at-least-accuracy classes). If a guarantee is required uniformly over persistent oracles satisfying

$$\mathbb{P}[\ell(u) = 1 \mid u \in S^\star] \ge \frac{1}{2} + \epsilon, \qquad \mathbb{P}[\ell(u) = 0 \mid u \notin S^\star] \ge \frac{1}{2} + \epsilon.$$

it must hold for the calibrated independent oracle of Definition 3. Thus Theorem 2 also rules out these uniform guarantees. Individual oracles can be easier; perfect labels recover $S^\star$ exactly.

E Extensions Beyond Persistent Vertex Predictions

The main text focuses on the persistent setting of the Vertex Prediction Model. This appendix gives complete statements, algorithms, and proofs for two extensions: repeated fresh queries in the non-persistent setting of the Vertex Prediction Model and endpoint-specific predictions in the Edge Prediction Model. Throughout this section, $\log$ denotes the natural logarithm.

Both extensions use the exact-cardinality size-reduction lemma stated and proved in Lemma 10.

E.1 The Vertex Prediction Model in the Non-Persistent Setting

Following the non-persistent oracle model of Braverman et al. (2024), we use the following DkS specialization.

Definition 21 (Vertex Prediction Model in the Non-Persistent Setting)**.** Fix $(G, k, S^\star)$ and an advantage parameter $\epsilon \in (0, 1/2)$ known to the algorithm. On its j-th query to a vertex $u \in V$, the oracle returns a label $\ell_j(u) \in \{0, 1\}$ predicting membership in $S^\star$. Conditioned on $(G, k, S^\star)$, the family $\{\ell_j(u) : u \in V,\ j \geq 1\}$ is mutually independent and, for every $u \in V$ and $j \geq 1$,

$$\mathbb{P}[\ell_j(u) = \mathbf{1}\{u \in S^\star\}] = \frac{1}{2} + \epsilon.$$

Thus, repeated queries to the same vertex return fresh independent samples.

Its one-query marginal agrees with Definition 3; only the behavior under repeated queries differs. We adapt the multi-round elimination method of Braverman et al. (2024) to the induced-edge objective of DkS.

Fix a target failure probability $\delta \in (0, 1)$, and set

$$L := \log \frac{4}{\delta}, \qquad R := \left\lceil 1 + \log \frac{n}{k} \right\rceil.$$

For each round $r \in \{1, \ldots, R\}$, define

$$q_r := \left\lceil \frac{4(r + L)}{\epsilon^2} \right\rceil, \qquad B := \left\lceil 50 \frac{n}{\epsilon^2} L \right\rceil.$$

The algorithm maintains nested surviving sets $V = V_0 \supseteq V_1 \supseteq \cdots$. In round r, it queries each $v \in V_{r-1}$ independently q_r times, lets $A_r(v)$ be the number of positive answers, and retains v if a majority is positive. Whenever at most $2k$ vertices remain, it uses Lemma 10 to form a size-k candidate and retains the densest candidate seen so far. The budget check makes the query bound deterministic.

Algorithm 2 Multi-round filtering with fresh vertex predictions

Input: $G = (V, E)$, an integer k with $2 \leq k \leq |V|$, $\epsilon \in (0, 1/2)$, $\delta \in (0, 1)$, and access to the oracle in Definition 21
Output: A set $S \subseteq V$ with $|S| = k$

Set $V_0 \leftarrow V$, $Q \leftarrow 0$, and let S^{best} be any size-k subset of V.
for $r = 1, \ldots, R$ **do**
  if $Q + |V_{r-1}| q_r > B$ **then**
    break
  end if
  Query each $v \in V_{r-1}$ independently q_r times; let $A_r(v)$ be the number of positive answers.
  Set $V_r \leftarrow \{v \in V_{r-1} : A_r(v) \geq \lceil q_r/2 \rceil\}$.
  Set $Q \leftarrow Q + |V_{r-1}| q_r$.
  if $|V_r| \leq 2k$ **then**
    Apply Lemma 10 to V_r to obtain a size-k set S_r.
    if $|E[S_r]| > |E[S^{\mathrm{best}}]|$ **then**
      Set $S^{\mathrm{best}} \leftarrow S_r$.
    end if
  end if
end for
return S^{best}.

We now state the approximation and query guarantees of Algorithm 2.

Theorem 22 (Constant-factor approximation with fresh vertex predictions)**.** *For every integer k with $2 \leq k \leq n$, $\epsilon \in (0, 1/2)$, and $\delta \in (0, 1)$, Algorithm 2 always returns a set $S \subseteq V$ of exactly k vertices. With probability at least $1 - \delta$ over the fresh oracle responses,*

$$|E[S]| \geq \frac{1}{12} m^\star.$$

The algorithm uses at most

$$B := \left\lceil \frac{50n}{\epsilon^2} \log \frac{4}{\delta} \right\rceil$$

oracle queries and runs in time polynomial in n, $1/\epsilon$, *and* $\log(4/\delta)$. *For constant* δ, *the query complexity is* $O(n/\epsilon^2)$.

To separate the probabilistic filtering argument from the deterministic budget check, let $(\overline{V}_r)_{r=0}^{R}$ denote the *ideal filtering process*. It uses the same oracle responses and filtering rule as Algorithm 2, but ignores the budget check and therefore always completes all R rounds. Let

$$\overline{Q}_R := \sum_{r=1}^{R} |\overline{V}_{r-1}| q_r$$

denote its total query cost.

Lemma 23 (Filtering accuracy and query complexity)**.** *The ideal filtering process satisfies the following properties:*

(i) for every $u \in S^\star$,

$$\mathbb{P}[u \notin \overline{V}_R] \le \frac{\delta}{300};$$

(ii)

$$\mathbb{P}\left[|\overline{V}_R \setminus S^\star| > \frac{k}{100}\right] \le \frac{\delta}{4};$$

(iii)

$$\mathbb{P}\left[\overline{Q}_R > 50\frac{n}{\epsilon^2}L\right] \le \frac{\delta}{4}.$$

Moreover, on the event $\overline{Q}_R \le 50nL/\epsilon^2$, *the budget check in Algorithm 2 never fires, and its sets* $V_0, \dots, V_R$ *coincide with* $\overline{V}_0, \dots, \overline{V}_R$.

Proof. For a fixed vertex present at the beginning of round r, Hoeffding's inequality bounds its filtering error by

$$\beta_r := \exp(-2\epsilon^2 q_r) \le \exp(-8(r+L)) \le \delta^8 e^{-8r}.$$

Consider first a vertex $u \in S^\star$. A union bound over the rounds gives

$$\mathbb{P}[u \notin \overline{V}_R] \le \sum_{r=1}^{R} \beta_r \le \frac{\delta^8}{e^8 - 1} \le \frac{\delta}{300},$$

where the last inequality follows from $\delta \le 1$. This proves part (i).

Now consider a vertex $v \notin S^\star$. Conditioned on $v \in \overline{V}_{r-1}$, the probability that v survives round r is at most β_r. In particular,

$$\begin{aligned}\mathbb{P}[v \in \overline{V}_R] &\le \beta_R \\ &\le \delta^8 e^{-8} \left(\frac{k}{n}\right)^8 \\ &\le \delta^8 e^{-8} \frac{k}{n},\end{aligned}$$

where we used $R \ge 1 + \log(n/k)$ and $k \le n$. Therefore,

$$\mathbb{E}\big[|\overline{V}_R \setminus S^\star|\big] \le \delta^8 e^{-8} k.$$

Markov's inequality yields

$$\mathbb{P}\left[|\overline{V}_R \setminus S^\star| > \frac{k}{100}\right] \le 100\delta^8 e^{-8} \le \frac{\delta}{4},$$

which proves part (ii).

It remains to bound the query cost. Since $L \geq \log 4 > 1$ and $\epsilon < 1/2$,

$$q_r \leq \frac{5}{\epsilon^2}(r + L).$$

Writing $x = n/k \geq 1$, the definition of R implies

$$kR^2 \leq 4n, \qquad kR \leq 2n.$$

Consequently, the total query contribution of vertices in $S^\star$ is deterministically bounded by

$$\begin{aligned}\sum_{r=1}^{R} |\overline{V}_{r-1} \cap S^\star| q_r &\leq k \sum_{r=1}^{R} q_r \\ &\leq \frac{5k}{\epsilon^2}\left(\frac{R(R+1)}{2} + RL\right) \\ &\leq 25\frac{n}{\epsilon^2}L.\end{aligned}$$

The first-round contribution of vertices outside $S^\star$ is at most

$$nq_1 \leq 10\frac{n}{\epsilon^2}L.$$

For their contribution in the remaining rounds, define

$$Z := \sum_{r=2}^{R} |\overline{V}_{r-1} \setminus S^\star| q_r.$$

For every $r \geq 2$ and every $v \notin S^\star$,

$$\mathbb{P}[v \in \overline{V}_{r-1}] \leq \delta^8 e^{-8(r-1)}.$$

Writing $q := e^{-8} < 10^{-3}$, linearity of expectation and the bound $q_r \leq 5(r + L)/\epsilon^2$ give

$$\begin{aligned}\mathbb{E}[Z] &\leq \frac{5n\delta^8}{\epsilon^2} \sum_{r=2}^{\infty} (r + L) q^{r-1} \\ &= \frac{5n\delta^8}{\epsilon^2}\left(\frac{q}{(1-q)^2} + (1 + L)\frac{q}{1-q}\right) \\ &\leq \delta^8 \frac{n}{\epsilon^2} L.\end{aligned}$$

Indeed, $(1 - q)^{-1} < 2$ and $L = \log(4/\delta) > 1$, so five times the parenthesized term is at most $(3 + L)/100 \leq L$. Markov's inequality therefore gives

$$\mathbb{P}\Big[Z > 15\frac{n}{\epsilon^2}L\Big] \leq \frac{\delta^8}{15} \leq \frac{\delta}{4}.$$

Adding the three contributions proves part (iii).

Finally, if $\overline{Q}_R \leq B$, then every prefix of the ideal process also uses at most B queries. Hence, the budget check in Algorithm 2 never fires. Coupling the actual and ideal processes using the same oracle responses then gives

$$V_r = \overline{V}_r \qquad \text{for every } r = 0, \ldots, R.$$

□

The filtering guarantee controls individual vertex losses; the next lemma converts it into preservation of optimum edges.

Lemma 24 (Preservation of optimum edges)**.** *With probability at least* $1 - \delta/75$*,*

$$|E[\overline{V}_R \cap S^\star]| \geq \frac{1}{2}m^\star.$$

Proof. The statement is immediate when $m^\star = 0$. Suppose that $m^\star > 0$, and let X be the number of edges in $E[S^\star]$ having at least one endpoint outside $\overline{V}_R$.

For every $e = \{u, v\} \in E[S^\star]$, Lemma 23(i) and a union bound give

$$\begin{aligned}\mathbb{P}[e \text{ is lost}] &\le \mathbb{P}[u \notin \overline{V}_R] + \mathbb{P}[v \notin \overline{V}_R] \\ &\le \frac{\delta}{150}.\end{aligned}$$

Therefore,

$$\mathbb{E}[X] \le \frac{\delta}{150} m^\star.$$

By Markov's inequality,

$$\mathbb{P}\left[X > \frac{1}{2} m^\star\right] \le \frac{\delta}{75}.$$

Whenever $X \le m^\star/2$, at least $m^\star/2$ optimum edges have both endpoints in $\overline{V}_R \cap S^\star$, proving the claim. □

Complete proof of Theorem 22. The budget check in Algorithm 2 ensures deterministically that the total number of oracle queries is at most

$$B = O\left(\frac{n}{\epsilon^2} \log \frac{4}{\delta}\right).$$

All remaining operations run in polynomial time.

By Lemma 23(ii)–(iii) and Lemma 24, the total failure probability is at most

$$\frac{\delta}{4} + \frac{\delta}{4} + \frac{\delta}{75} = \frac{77}{150}\delta < \delta.$$

Thus, with probability at least $1 - \delta$, the actual algorithm completes all R rounds,

$$|V_R| \le |S^\star| + |V_R \setminus S^\star| \le k + \frac{k}{100} \le 2k,$$

and

$$|E[V_R]| \ge |E[V_R \cap S^\star]| \ge \frac{1}{2} m^\star.$$

Therefore, in round R, Algorithm 2 applies Lemma 10 to form a size-k candidate S_R that preserves at least one sixth of these edges. Thus,

$$|E[S_R]| \ge \frac{1}{6}|E[V_R]| \ge \frac{1}{12} m^\star.$$

Since the algorithm returns the densest candidate encountered,

$$|E[S^{\text{best}}]| \ge |E[S_R]| \ge \frac{1}{12} m^\star.$$

□

For comparison, uniform per-vertex majority voting exactly recovers $S^\star$, but requires an additional $\log n$ factor in the query complexity.

Proposition 25 (Exact recovery by per-vertex majority vote)**.** *Under Definition 21, querying every vertex*

$$q := \left\lceil \frac{1}{2\epsilon^2} \log \frac{n}{\delta} \right\rceil$$

times and classifying each vertex by majority vote recovers $S^\star$ with probability at least $1 - \delta$. The total number of queries is

$$O\left(\frac{n}{\epsilon^2} \log \frac{n}{\delta}\right).$$

Proof. For every vertex, Hoeffding's inequality bounds the probability of an incorrect majority decision, treating a tie as an error when necessary, by

$$\exp(-2\epsilon^2 q) \le \frac{\delta}{n}.$$

A union bound over all vertices shows that every membership bit is recovered correctly with probability at least $1 - \delta$. On this event, the positively classified vertices are exactly $S^\star$. □

E.2 The Edge Prediction Model

We next instantiate the Edge Prediction Model introduced by Aamand et al. (2025) for DkS. In this model, every edge supplies predictions for the optimal-solution assignments of its two incident vertices. Aamand et al. also developed a general high–low-degree framework for using these predictions: predictions are aggregated to classify high-degree vertices, whereas the remaining low-degree subgraph is handled by a prediction-free algorithm. We follow this framework and specialize its two components to the induced-edge objective and exact-cardinality constraint of DkS.

Definition 26 (Edge Prediction Model)**.** Fix $(G, k, S^\star)$ and an advantage parameter $\epsilon \in (0, 1/2)$ known to the algorithm. Every edge $e = \{u, v\} \in E$ provides two prediction bits $(b_u(e), b_v(e)) \in \{0, 1\}^2$. For each $w \in e$, the bit $b_w(e)$ predicts whether $w \in S^\star$, and

$$\mathbb{P}[b_w(e) = \mathbf{1}\{w \in S^\star\}] = \frac{1}{2} + \epsilon.$$

Conditioned on $(G, k, S^\star)$, all prediction bits are mutually independent. Thus, the incident prediction bits $\{b_v(e) : e \ni v\}$ provide $\deg(v)$ independent observations of the same membership bit.

For $\delta \in (0, 1)$, define the degree threshold

$$D_{\text{edge}} := \left\lceil \frac{1}{2\epsilon^2} \log \frac{n}{\delta} \right\rceil.$$

Define the corresponding high- and low-degree vertex sets by

$$V_{\text{high}} := \{v \in V : \deg(v) \geq D_{\text{edge}}\}, \qquad V_{\text{low}} := V \setminus V_{\text{high}}.$$

The algorithm forms two candidates. The high-degree candidate augments the majority-positive vertices in V_{high} with vertices having the most neighbors among them. The low-degree candidate is constructed directly from a maximal matching in $G[V_{\text{low}}]$. Both candidates contain exactly k vertices, and the algorithm returns the denser one.

Algorithm 3 DkS in the Edge Prediction Model

Input: $G = (V, E)$, an integer k with $2 \leq k \leq |V|$, $\epsilon \in (0, 1/2)$, $\delta \in (0, 1)$, and the edge predictions in Definition 26

Output: A set $\widehat{S} \subseteq V$ with $|\widehat{S}| = k$

Set $H \leftarrow \{v \in V_{\text{high}} : \sum_{e \ni v} b_v(e) > \deg(v)/2\}$.
Let S_1 be any subset of H of size $\min\{|H|, k\}$.
Let $S_2 \subseteq V \setminus S_1$ consist of the $k - |S_1|$ vertices with the largest values of $|N(v) \cap S_1|$, breaking ties arbitrarily.
Set $S^{\text{high}} \leftarrow S_1 \cup S_2$.
Compute a maximal matching M in $G[V_{\text{low}}]$.
if $|M| \geq \lfloor k/2 \rfloor$ **then**
  Let U be the set of endpoints of any $\lfloor k/2 \rfloor$ edges in M.
else
  Let U be the set of all endpoints of edges in M.
end if
Pad U with arbitrary vertices of $V \setminus U$ to obtain a size-k set S^{low}.
return A set maximizing $|E[\hat{S}]|$ over $\hat{S} \in \{S^{\text{high}}, S^{\text{low}}\}$.

The following theorem gives the approximation guarantee for Algorithm 3.

Theorem 27 (Approximation in the Edge Prediction Model)**.** *For every integer k with $2 \leq k \leq n$, $\epsilon \in (0, 1/2)$, and $\delta \in (0, 1)$, under Definition 26, Algorithm 3 always returns a set $\widehat{S} \subseteq V$ of exactly k vertices. With probability at least $1 - \delta$ over the edge predictions, its output satisfies*

$$|E[\widehat{S}]| \geq \frac{1}{2D_{\text{edge}} + 1} m^\star.$$

Consequently, its approximation factor is

$$2D_{\mathrm{edge}} + 1 = O\left(\frac{1}{\epsilon^2}\log\frac{n}{\delta}\right).$$

The algorithm runs in polynomial time.

The analysis first establishes the majority-vote classification on V_{high}, then analyzes the maximal-matching construction on V_{low}, and finally combines the two exact-cardinality candidates.

Lemma 28 (Simultaneous classification of high-degree vertices)**.** *With probability at least $1-\delta$, the set H computed by Algorithm 3 satisfies*

$$H = S^\star \cap V_{\mathrm{high}}.$$

Proof. For every $v \in V_{\mathrm{high}}$, let $Z(v) := \sum_{e \ni v} b_v(e)$. If $v \in S^\star$, then

$$\mathbb{E}[Z(v)] = \left(\frac{1}{2} + \epsilon\right)\deg(v),$$

and v can be misclassified only if $Z(v) \le \deg(v)/2$. If $v \notin S^\star$, then

$$\mathbb{E}[Z(v)] = \left(\frac{1}{2} - \epsilon\right)\deg(v),$$

and v can be misclassified only if $Z(v) > \deg(v)/2$.

In either case, Hoeffding's inequality gives

$$\mathbb{P}[v \text{ is misclassified}] \le \exp\bigl(-2\epsilon^2\deg(v)\bigr) \le \exp\bigl(-2\epsilon^2 D_{\mathrm{edge}}\bigr) \le \frac{\delta}{n}.$$

A union bound over at most n high-degree vertices proves the claim. □

We next bound the contribution retained by the maximal matching on the low-degree subgraph.

Lemma 29 (Low-degree matching guarantee)**.** *The set S^{low} constructed by Algorithm 3 has exactly k vertices and satisfies*

$$|E[S^{\mathrm{low}}]| \ge \frac{1}{2D_{\mathrm{edge}}} \max_{\substack{T \subseteq V_{\mathrm{low}} \\ |T| \le k}} |E[T]|.$$

Proof. Let $G' := G[V_{\mathrm{low}}]$, whose maximum degree is strictly less than D_{edge}, and let M be the maximal matching computed by Algorithm 3. First suppose that

$$|M| \ge \left\lfloor \frac{k}{2} \right\rfloor.$$

Then U is the set of endpoints of $\lfloor k/2 \rfloor$ edges in M, so $|U| \le k$ and

$$|E_{G'}[U]| \ge \left\lfloor \frac{k}{2} \right\rfloor \ge \frac{k}{4}$$

edges. On the other hand, every set T of at most k vertices satisfies

$$|E_{G'}[T]| \le \frac{1}{2}\sum_{v \in T} \deg_{G'}(v) < \frac{D_{\mathrm{edge}}k}{2}.$$

This gives the claimed factor.

Now suppose that

$$|M| < \left\lfloor \frac{k}{2} \right\rfloor.$$

Now U is the set of all endpoints of M. If $M = \varnothing$, maximality implies that G' has no edges, so the conclusion is immediate. Hence, assume $M \neq \varnothing$. Then

$$|U| = 2|M| < 2\left\lfloor \frac{k}{2} \right\rfloor \leq k.$$

Because M is maximal, U is a vertex cover of G'. Hence, for every $T \subseteq V(G')$ with $|T| \leq k$,

$$|E_{G'}[T]| \leq \sum_{v \in U} \deg_{G'}(v) < 2D_{\text{edge}}|M|.$$

Moreover, $G'[U]$ contains all edges of M, and therefore

$$|E_{G'}[U]| \geq |M|.$$

The last two inequalities give the desired approximation factor. In both cases, Algorithm 3 pads U to exactly k vertices. Padding cannot remove induced edges, so the same guarantee holds for S^{low}. □

Finally, we combine the high- and low-degree candidates to account for all optimum edges.

Lemma 30 (Combined high- and low-degree candidates)**.** *With probability at least $1 - \delta$, the output of Algorithm 3 satisfies*

$$|E[\widehat{S}]| \geq \frac{1}{2D_{\text{edge}} + 1} m^\star.$$

Proof. Condition on the event in Lemma 28, and write

$$S^\star_{\text{high}} := S^\star \cap V_{\text{high}}, \qquad S^\star_{\text{low}} := S^\star \cap V_{\text{low}}.$$

Then

$$H = S^\star_{\text{high}}.$$

Since $|H| \leq k$, the construction in Algorithm 3 gives

$$S_1 = S^\star_{\text{high}}, \qquad |S_2| = k - |S_1| = |S^\star_{\text{low}}|.$$

Moreover, $S^\star_{\text{low}} \subseteq V \setminus S_1$. Because S_2 contains the vertices with the largest numbers of neighbors in S_1,

$$\begin{aligned} |E[S_1, S_2]| &= \sum_{v \in S_2} |N(v) \cap S_1| \\ &\geq \sum_{v \in S^\star_{\text{low}}} |N(v) \cap S_1| \\ &= |E[S^\star_{\text{high}}, S^\star_{\text{low}}]|. \end{aligned}$$

It follows that

$$|E[S^{\text{high}}]| \geq X, \qquad X := |E[S^\star_{\text{high}}]| + |E[S^\star_{\text{high}}, S^\star_{\text{low}}]|.$$

Since $S^\star_{\text{low}}$ is a feasible at-most-k set in $G[V_{\text{low}}]$, Lemma 29 gives

$$|E[S^{\text{low}}]| \geq \frac{Y}{2D_{\text{edge}}}, \qquad Y := |E[S^\star_{\text{low}}]|.$$

The three edge classes counted by X and Y partition $E[S^\star]$, so

$$m^\star = X + Y.$$

Let $Z := |E[\widehat{S}]|$. Since the algorithm returns the denser candidate,

$$X \leq Z, \qquad Y \leq 2D_{\text{edge}} Z.$$

Therefore,

$$m^\star = X + Y \leq (2D_{\text{edge}} + 1)Z,$$

which proves the claim. □

Complete proof of Theorem 27. By construction, $|S^{\text{high}}| = |S_1| + |S_2| = k$, and Lemma 29 gives $|S^{\text{low}}| = k$. Hence, the algorithm always returns exactly k vertices. Aggregating the edge predictions, selecting the highest-scoring vertices, computing a maximal matching, padding the matching endpoints, and comparing the candidates all take polynomial time.

The approximation guarantee and its success probability follow from Lemma 30. Finally,

$$2D_{\text{edge}} + 1 = O\left(\frac{1}{\epsilon^2} \log \frac{n}{\delta}\right),$$

which proves the theorem. □

F Additional Experimental Details and Results

This section documents preprocessing, implementation settings, uncertainty computation, timing scope, and results for the configurations reported in Section 6.

F.1 Protocol and Implementation Details

Evaluated LAA and ablations. The experimental implementation retains the adaptive and fixed-prefix candidates and the maximal-matching safeguard of Algorithm 1. It returns the size-k candidate with the most induced edges. For odd experimental k, the fixed-prefix candidate uses $h = \lfloor k/2 \rfloor$, while the matching candidate retains $\lfloor k/2 \rfloor$ edges and pads their endpoints to size k. The guarantee in Theorem 1 applies to even k. *Label Only* uniformly adjusts the positive-label set to size k; *Score Only* selects the k highest-scoring vertices in equation 2. The grid endpoints $\epsilon = 0$ and $\epsilon = 0.5$ serve as diagnostics; the guarantee assumes $\epsilon \in (0, 0.5)$.

Graph preprocessing. Every input is converted to an unweighted simple undirected graph. We merge reciprocal directed arcs where applicable, discard self-loops and duplicate edges, remove isolated vertices, and relabel the remaining vertices as $1, \ldots, n$. The values in Table 1 are measured after this common preprocessing. We do not restrict the graphs to their largest connected components.

Certified and offline references. For each of the nine reported small real-world instances, Gurobi Optimizer 13.0.2 terminates with status Optimal and zero final MIP gap. Binary variables x_v select vertices and continuous variables y_{uv} select induced edges in the exact formulation

$$\max \sum_{\{u,v\} \in E} y_{uv} \quad \text{s.t.} \quad \sum_v x_v = k, \quad y_{uv} \le x_u, x_v, \quad x_v \in \{0,1\},\ 0 \le y_{uv} \le 1.$$

For the three larger real-world graphs, exact optimization is infeasible. We therefore run a prediction-free portfolio containing degree- and core-based seeds, localized constructive seeds, restricted one-swap local improvement, and perturb-and-repair. We use 40 independent shards and a five-hour wall-clock budget for each (G, k), retain the best feasible size-k set, and freeze that set before generating any labels. This offline procedure does not call LAA or any of the four evaluated empirical baselines. Consequently, an offline reference is a feasible lower bound on OPT, not an optimality certificate.

Planted graph generation. For each planted-clique size, we generate five independent $G(n, 0.5)$ graphs and then add every missing edge inside the selected k-vertex set. The planted set is therefore a certified optimum, although attaining its objective does not identify which maximum clique was returned.

For LFR, we use $n = 10^6$, average degree 200, maximum degree 500, degree exponent -2.8, community-size exponent -1.5, and community sizes in $[8{,}000, 22{,}000]$. For each graph seed we generate paired instances with $\mu \in \{0.5, 0.6, 0.7\}$. Before changing μ, the generator fixes the degree and community-size sequences, so matching seeds give a controlled comparison across mixing levels. We retain a graph only if it contains a community within 1.5% of each nominal target $10{,}000$, $15{,}000$, and $20{,}000$. Within each window, we select the community with the largest normalized internal edge density and set k to its realized size. Across the five seeds, the mean realized values are $9{,}984.4$, $15{,}004.4$, and $19{,}947.2$, respectively. Degree density is normalized by the graph-specific realized k before averaging.

Baseline implementations. All methods return exactly k vertices, with ties resolved consistently by the implementation. Greedy uses the high-degree seed-and-completion procedure of Feige et al. (2001). FW uses $\lambda = 1$, at most 200 Frank–Wolfe iterations, uniform initialization $x_0 = (k/n)\mathbf{1}$, and top-k rounding. EP-Prox uses at most 100 proximal-gradient iterations, $\lambda_0 = 10^{-10}$, multiplier 20, and uniform initialization; its final iterate is rounded to the k largest coordinates. Diagnostic logging is disabled for runtime measurement because otherwise it periodically performs additional full-graph edge-counting scans that do not affect the iterate. Rank1PC computes the leading rank-one component using `eigsh`, forms the two supports induced by its two signs, and reports the support inducing more edges in the original graph. The numerical configurations are deterministic, so each prediction-free baseline is run once per graph.

Timing and uncertainty. All runtime values exclude graph loading and common preprocessing but include method-specific initialization, optimization, final top-k rounding, and the final induced-edge evaluation. For LAA, we average the reported solver time over its ten runs at each ϵ, then average those 51 values. For the planted-clique and LFR families, we subsequently average over five graph instances. Runtime comparisons therefore concern the implemented solvers, not end-to-end data loading. All experiments use an Intel Xeon Platinum 8562Y CPU at $2.80\,\mathrm{GHz}$ with $768\,\mathrm{GB}$ RAM. Prediction-free configurations are timed once per graph, so we do not estimate repeated-system timing variability for these deterministic runs.

Main-text curves omit ribbons for readability. Below, ribbons show ± 1 standard error across ten oracle realizations for a real graph and across five graph-level means for a planted family, after averaging ten oracle realizations within each graph. Prediction-free horizontal lines are unshaded family means.

F.2 Complete Real-World Results

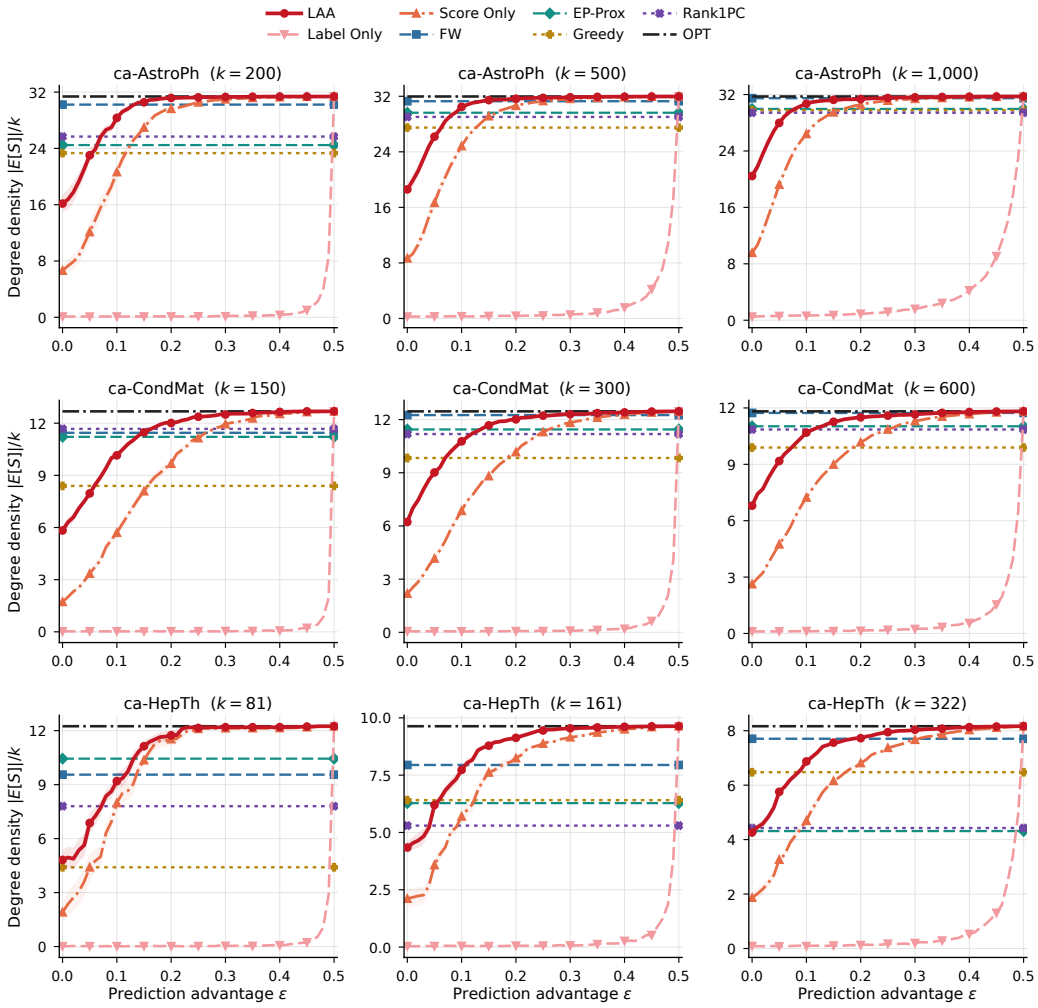

Figure 2: Complete degree-density results for the three reported graphs with certified optima. Columns correspond to the three values of k in Table 1. Ribbons show ± 1 standard error over ten persistent-label realizations.

Table 2 lists descriptive crossover thresholds for all 18 reported real-world configurations. They compare means on the fixed grid $\{0, 0.01, \ldots, 0.50\}$ and are not significance tests or estimates of a continuous transition.

Table 2: Crossover thresholds on reported real-world instances. Entries in each of the last two columns correspond in order.

Dataset	Values of k	Crossover ϵ
`ca-AstroPh`	$200, 500, 1000$	$0.14, 0.14, 0.24$
`ca-CondMat`	$150, 300, 600$	$0.17, 0.28, 0.35$
`ca-HepTh`	$81, 161, 322$	$0.14, 0.11, 0.19$
`ogbl-collab`	$500, 1000, 2000$	$0.08, 0.09, 0.09$
`ogbn-products`	$4000, 8000, 15000$	$0.07, 0.10, 0.10$
`cit-Patents`	$2000, 4000, 8000$	$0.11, 0.15, 0.17$

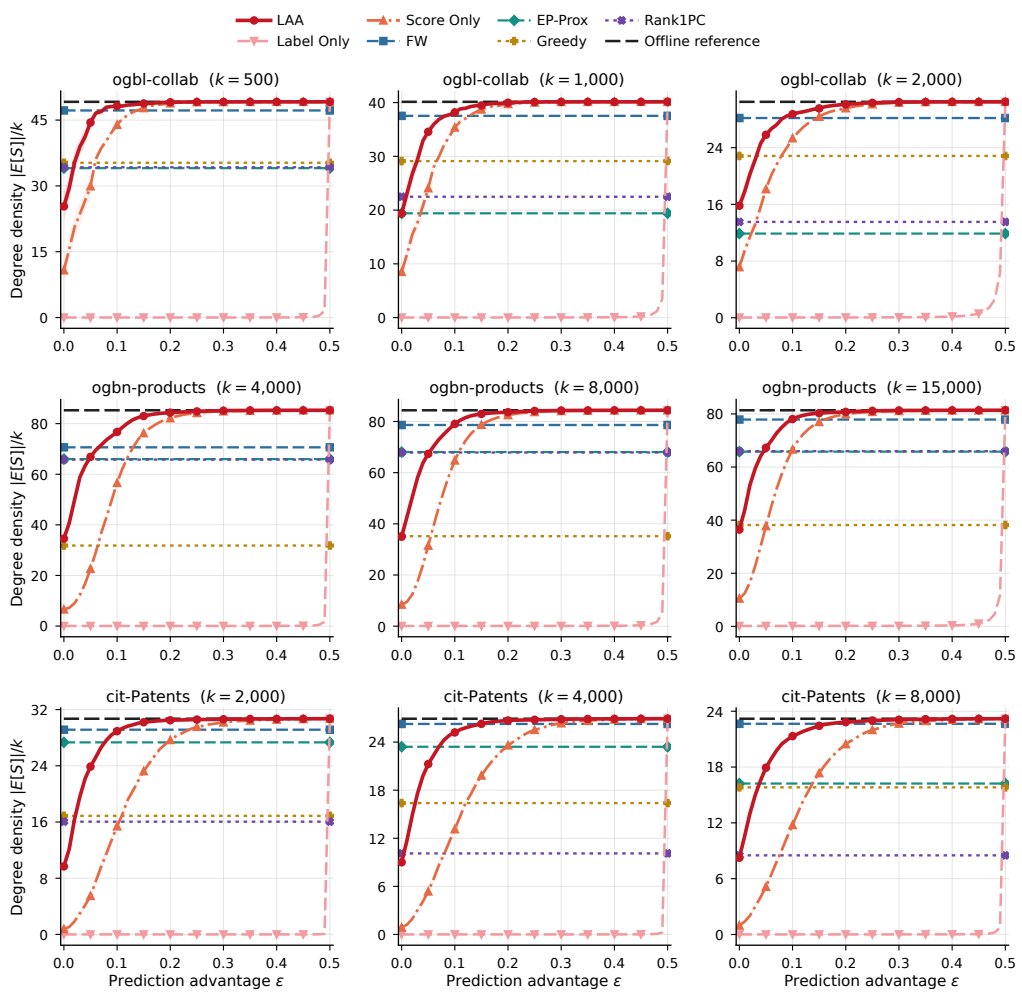

Figure 3: Complete degree-density results for the three reported larger real-world graphs. The black line is the frozen offline reference, not a certified optimum. Ribbons show ± 1 standard error over ten persistent-label realizations.

Reporting scope. The real-world conclusions are restricted to the six graphs and 18 configurations in Table 1; we do not claim exhaustive benchmark coverage.

F.3 Complete Planted-Instance Results

Table 3 summarizes both crossover and target-objective attainment. For LFR, attainment means equality with the selected planted community's edge count; it does not imply exact vertex recovery or global optimality.

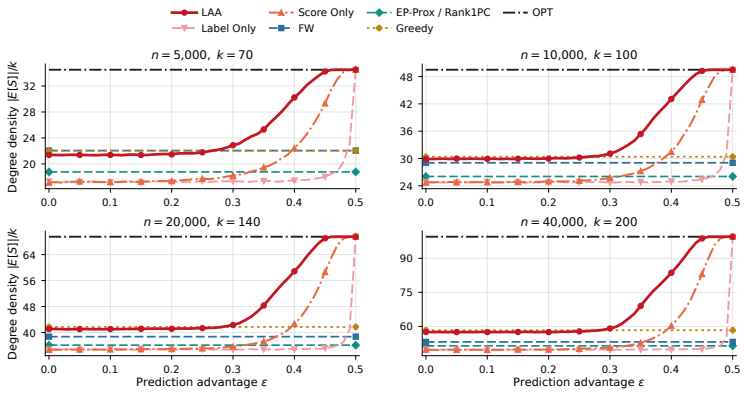

Figure 4: Complete planted-clique results with ± 1 standard-error ribbons across five graph-level means. EP-Prox and Rank1PC coincide in all four panels.

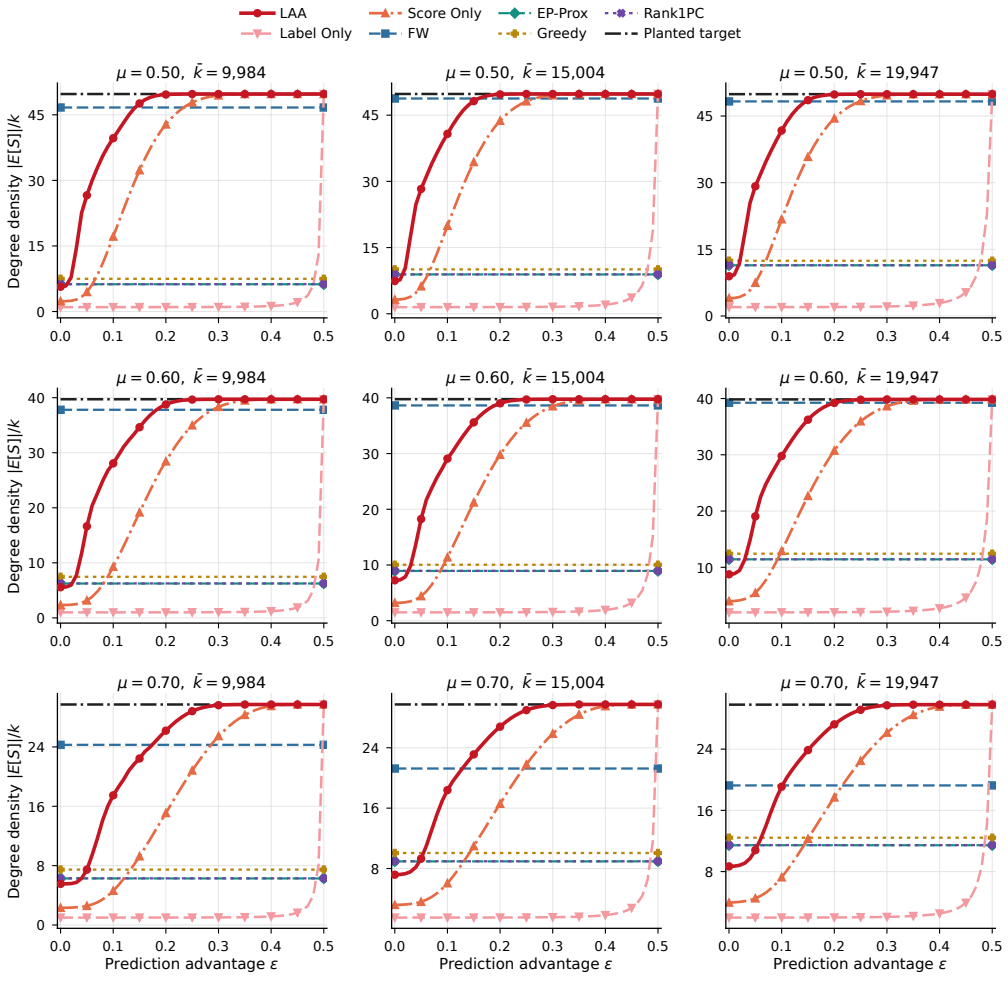

Figure 5: Complete LFR results for three mixing levels and three target-size windows. Each graph uses its realized k before averaging; ribbons show ± 1 standard error across five graph-level means.

Table 3: Grid-level thresholds on planted families. Entries are ordered by increasing n for planted cliques and by target size $10{,}000, 15{,}000, 20{,}000$ for LFR.

Family	Configurations	Crossover ϵ	Target ϵ
Planted clique	$n = 5{,}000, 10{,}000, 20{,}000, 40{,}000$	$0.27, 0.27, 0.29, 0.29$	$0.47, 0.47, 0.48, 0.48$
LFR, $\mu = 0.5$	targets $10{,}000, 15{,}000, 20{,}000$	$0.15, 0.16, 0.15$	$0.27, 0.28, 0.28$
LFR, $\mu = 0.6$	targets $10{,}000, 15{,}000, 20{,}000$	$0.19, 0.19, 0.21$	$0.36, 0.36, 0.36$
LFR, $\mu = 0.7$	targets $10{,}000, 15{,}000, 20{,}000$	$0.18, 0.13, 0.11$	$0.38, 0.46, 0.48$

F.4 Complete Runtime Results

The complete runtime plots below use the preloaded-graph timing convention specified in Appendix F.1. Across all 13 planted configurations, Greedy's solver time is 1.01–1.71 times LAA's, while FW, EP-Prox, and Rank1PC take 14.5–205.7 times as long. On the nine configurations from `ogbl-collab`, `ogbn-products`, and `cit-Patents`, FW, EP-Prox, and Rank1PC take 24.8–128.2, 9.82–44.1, and 3.19–22.0 times LAA's solver time, respectively. Greedy is faster than LAA on all 18 configurations on real graphs.

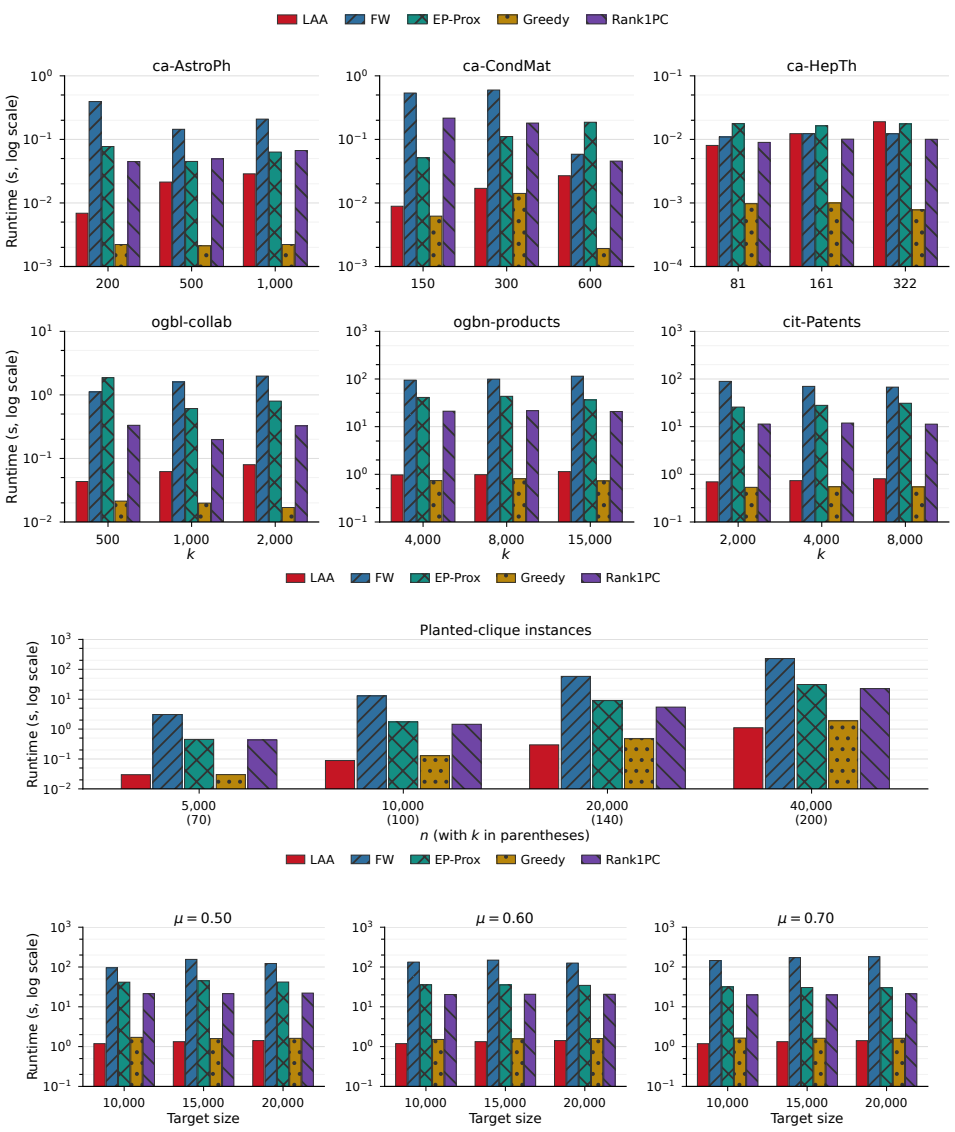

Figure 6: Complete preloaded-runtime results with logarithmic vertical axes: every reported real-world graph and value of k (top), the four planted-clique families (middle), and all LFR configurations (bottom). Each synthetic value averages five graph instances.