NETBREAKER: BENCHMARKING NETWORK DISMANTLING ACROSS OBJECTIVES, SCALES, AND TOPOLOGIES

Anonymous authors
Paper under double-blind review

ABSTRACT

Network dismantling seeks to fragment a network with as few node removals as possible, which is widely applied in vulnerability analysis, epidemic containment, and infrastructure protection. Although learning-based methods have become increasingly prominent, their progress remains difficult to assess because existing studies differ in method coverage, datasets, evaluation criteria, and experimental protocols. Thus, we introduce NETBREAKER, a comprehensive benchmark designed for the learning era of network dismantling. NETBREAKER organizes 30 methods (18 non-learning and 12 learning-based) under a common taxonomy and evaluates them on 185 networks from eight unified domains, ranging from 14 to 1.69 million nodes. To enable fair comparison, heterogeneous method outputs are converted into node removal sequences and assessed under a unified execution protocol. The evaluation covers four complementary dimensions: target-based effectiveness, trajectory-based effectiveness, structural diagnostics, and applicability & computational efficiency. Extensive experiments show that current learning-based methods do not consistently outperform strong non-learning approaches, while exposing substantial differences in effectiveness, scalability, and efficiency across network conditions and evaluation perspectives. NETBREAKER provides a reproducible basis for measuring progress and developing reliable, scalable, and computationally efficient network dismantling methods.

1 INTRODUCTION

Complex networks underpin many natural and engineered systems, where interconnectivity supports collective function but can also amplify the impact of disruptions. A network may remain largely intact under random failures yet become highly vulnerable when perturbations are strategically targeted, making such structural fragility a central concern in network robustness (Artime et al., 2024). **Network dismantling (ND)** studies this vulnerability by examining how a small targeted set of nodes can be removed to progressively break network connectivity (Braunstein et al., 2016). Beyond characterizing how susceptible a network is to deliberate disruption, dismantling also provides a basis for reasoning about where connectivity should be protected or intentionally interrupted (Fan et al., 2020b; Artime et al., 2024). Efficient network dismantling is intrinsically difficult because identifying the smallest set of removals that achieves a prescribed level of fragmentation is an NP-hard combinatorial optimization problem (Braunstein et al., 2016). The challenge is further compounded in real-world networks, whose heterogeneous and multiscale structures make the effectiveness of a removal strategy strongly dependent on topology (Wandelt et al., 2018; Artime et al., 2024; Adler et al., 2026).

To tackle these challenges, a wide range of methods have been proposed. Early studies relied mainly on structural centrality measures, ranking nodes according to their individual topological importance (Wandelt et al., 2018). As it became clear that effective dismantling depends on the joint effect of multiple removals rather than on isolated node scores, the focus shifted toward optimal-percolation, decycling, message-passing, and graph-partitioning formulations that explicitly account for collective structural effects (Morone & Makse, 2015; Braunstein et al., 2016; Mugisha & Zhou, 2016; Zdeborová et al., 2016; Ren et al., 2019). More recently, machine learning has provided a different way to capture these dependencies. GDM learns graph representations from optimally solved small instances,

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.

while FINDER uses deep reinforcement learning to acquire sequential removal policies that can be transferred to unseen networks (Grassia et al., 2021; Fan et al., 2020b). Subsequent work has further developed supervised, unsupervised, and reinforcement-learning approaches for transferable dismantling (Zhang & Wang, 2022; 2023; Tian et al., 2026; Zhang et al., 2026).

Despite the rapid development of network dismantling methods, the field still lacks a sufficiently unified basis for assessing progress across studies. This inconsistency arises mainly from three aspects. First, network preparation differs substantially across studies. Methods are evaluated on different collections of synthetic and real-world networks, often with different graph construction and preprocessing procedures, making results difficult to compare directly. Second, the dismantling problem itself is not formulated uniformly. Studies may adopt different dismantling targets, stopping conditions, and assumptions about how removals are performed, so nominally similar experiments can correspond to different optimization problems. Third, evaluation protocols remain inconsistent. Different metrics are used to summarize dismantling performance, while computational cost and applicability are not always reported under comparable conditions. Although previous comparative studies have provided valuable assessments of representative methods (Wandelt et al., 2018; Artime et al., 2024) and recent work has expanded evaluation to large collections of real-world networks (Adler et al., 2026), these differences continue to hinder a consistent assessment of methodological progress.

To provide a common basis for comparison, we introduce NETBREAKER, a systematic benchmark for network dismantling. NETBREAKER standardizes network preparation and problem instantiation across methods, while a method-independent evaluation procedure assesses the resulting removal sequences under consistent criteria. The benchmark covers 30 dismantling methods and variants across 185 real-world networks. By separating method execution from evaluation, NETBREAKER preserves the original behavior of heterogeneous algorithms while enabling their performance, computational cost, and execution coverage to be examined under the same experimental setting.

Our contributions can be summarized as follows:

- **Comprehensive Benchmark for the Learning Era of Network Dismantling.** NetBreaker enables a fair and direct comparison among 18 non-learning and 12 learning-based network dismantling methods and variants by unifying their experimental settings across 185 networks with diverse domains, scales, and topological characteristics. By bringing traditional algorithms and recently developed learning-based approaches into the same benchmark, NetBreaker provides a comprehensive view of the methodological progress in network dismantling.
- **Multi-faceted Evaluation and Analysis.** We systematically evaluate existing network dismantling methods from multiple complementary dimensions, including dismantling effectiveness, computational efficiency, completion coverage, scalability, and performance across different network characteristics. Our analysis reveals the respective strengths, limitations, and applicable conditions of different families of methods, providing practical insights to guide future research.
- **A Unified and Resource-Aware Benchmarking Codebase.** We develop a unified and extensible codebase that accommodates the heterogeneous execution procedures of existing network dismantling methods. In addition to solution quality, the codebase is able to record computational costs and method completion under predefined runtime, memory, and scale constraints. It provides a reproducible foundation for evaluating existing methods and integrating future network dismantling approaches.

2 PROBLEM AND POSITIONING

Task. We study node dismantling on a simple, undirected, and unweighted graph $G = (V, E)$ with $N = |V|$ nodes. Removing a set $S \subseteq V$ yields the residual graph $G[V \setminus S]$. Given a fragmentation threshold $\theta \in (0, 1)$, the objective is to find the smallest removal set whose deletion reduces the largest connected component to at most a θ fraction of the original network:

$$S^* = \arg\min_{S \subseteq V} |S| \quad \text{s.t.} \quad \frac{|\mathrm{LCC}(G[V \setminus S])|}{N} \leq \theta.$$

Why a new benchmark. Existing evaluations of network dismantling methods broadly fall into two groups. Comparative studies evaluate multiple methods under shared experimental settings.

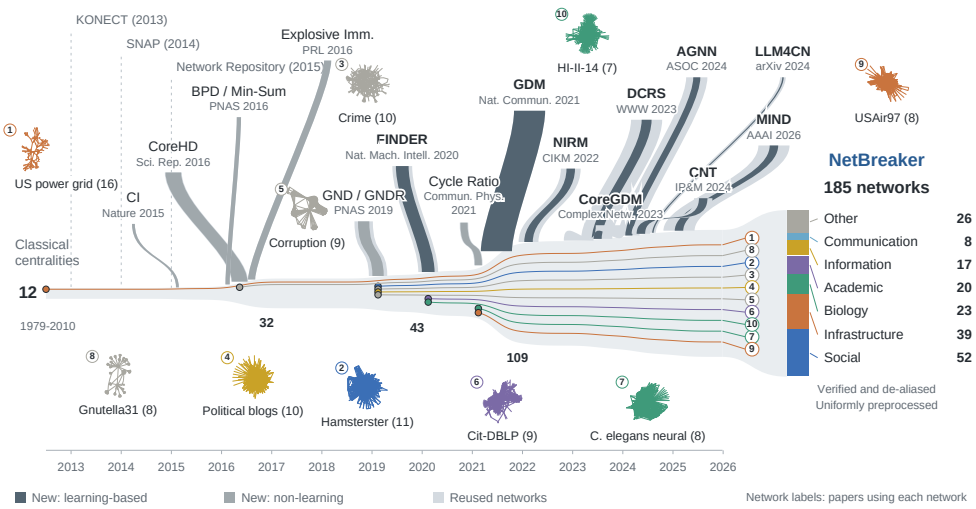


Figure 1: Evolution and important network datasets in network dismantling.

Wandelt et al. (Wandelt et al., 2018) compare 13 predominantly non-learning methods on 20 networks, assessing dismantling performance through the robustness measure R over the LCC trajectory together with execution time. Artime et al. (Artime et al., 2024) compare 13 methods on 65 networks using a 10% LCC target and the corresponding trajectory AUC. Although these studies provide valuable common test basis, the performance evaluation is conducted by a limited set of perspectives and coverlittle learning-based dismantling methods.

Method-centric studies constitute the second group. To substantiate the effectiveness of a proposed method, these studies often employ broader experimental designs tailored to that method. GDM (Grassia et al., 2021), for example, considers multiple evaluation perspectives to validate its dismantling performance, whereas LGD-NA (Adler et al., 2026) strengthens the empirical effectiveness for its proposed method by comparing 17 methods across 1,475 networks. These experiments provide informative evidence for the corresponding methods, but their baselines, network collections, and experimental protocols are selected around different research questions. Their results therefore cannot be compared directly across studies and do not constitute a shared benchmark spanning non-learning and learning-based methods.

Besides, the datasets used in both groups of studies are organized under inconsistent domain classifications, and the complete network collections required to reproduce the reported comparisons are not always available. These inconsistencies further affect the direct comparison across studies.

NETBREAKER is designed to provide a unified benchmark for the learning era of network dismantling. It evaluates 18 non-learning methods together with 12 recent learning-based methods under the same experimental protocol. Drawing from the networks used in prior studies, the benchmark selects 185 datasets covering a broad range of network scales and application domains, and reorganizes their inconsistent domain labels into a unified domain taxonomy. It further establishes an explicit taxonomy of dismantling methods, provides the complete benchmark dataset collection, and evaluates methodological progress through four complementary dimensions. By integrating these perspectives within a comprehensive evaluation framework, NETBREAKER enables fair and direct comparison across methodological generations.

3 THE NETBREAKER BENCHMARK

NETBREAKER follows a three-stage procedure (Figure 2). The first stage constructs a common network corpus that includes networks of different scales and application domains, and all methods operate on the same canonically prepared network instances. Second, each method retains its method-specific training, optimization, inference, and post-processing procedure, while its output is converted into a common node removal sequence. The third stage evaluates their outputs through a common representation and four complementary evaluation dimensions under a unified execution protocol.

Table 1: Comparison with related network dismantling study. Domains report dataset-level coverage under the unified domain taxonomy. Evaluation dimensions count target-based evaluation, trajectory-based evaluation, structural diagnostics, and applicability and computational efficiency. For dataset availability, ✓ denotes that the exact experimental graphs are provided, Partial that only a subset or acquisition instructions are provided, and ✗ that neither is provided.

Study	Benchmark	Methods	Datasets	Domains	Max. N	Eval. dims.	Dataset availability	Method taxonomy
Wandelt et al. (Wandelt et al., 2018)	✓	13	20	6/8	8.1×10^4	2	Partial	✓
Artime et al. (Artime et al., 2024)	✓	13	65	8/8	1.4×10^6	2	✓	✓
FINDER (Fan et al., 2020b)	✗	11	9	4/8	1.6×10^6	3	✓	✓
GDM (Grassia et al., 2021)	✗	15	57	8/8	1.8×10^6	4	Partial	✗
MIND (Tian et al., 2026)	✗	9	48	7/8	1.5×10^6	2	✓	✓
LGD-NA (Adler et al., 2026)	✗	17	1,475	8/8	2.3×10^4	3	Partial	✓
NETBREAKER (this work)	✓	30	185	8/8	1.7×10^6	4	✓	✓

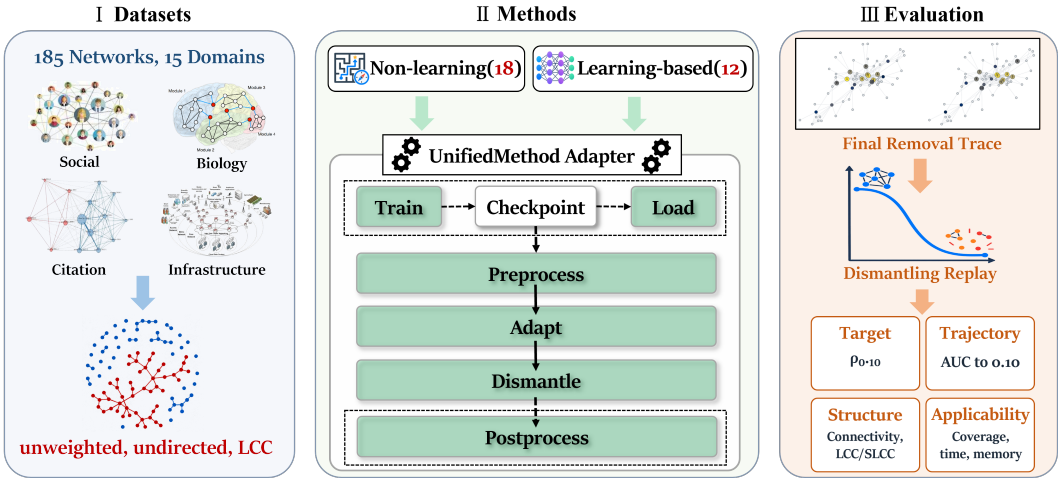


Figure 2: NETBREAKER at a glance. Our benchmark mainly consists of Datasets, Methods, and Evaluation modules. Within the Datasets module, we collect 185 network datasets and convert each network into a unified verison. As for Method module, we design a standard procedure to execute different methods. The output of all the methods are then set to be node removal sequence, which is served as the basis for the comprehensive evaluation.

3.1 NETWORKS

Collection and preparation. We construct the corpus through a literature-guided collection process, tracing networks evaluated in network dismantling and related critical-node identification studies. The registry covers classical centrality baselines and subsequent non-learning and learning-based methods, including studies published in venues such as Nature, PNAS, Nature Machine Intelligence, Nature Communications, CIKM, and The Web Conference, alongside relevant preprints. We link reported networks to their source studies and retrieve the data from KONECT (Kunegis, 2013), SNAP (Leskovec & Krevl, 2014), Network Repository (Rossi & Ahmed, 2015), LINQS (Sen et al., 2008), and author-released resources. Figure 1 illustrates the introduction and reuse of networks across representative studies. After reconciling dataset identities and provenance across different sources, we retain 185 networks. Each network is represented in the simple, undirected, and unweighted setting defined in Section 2, providing identical structural inputs to all evaluated methods. Detailed identity checks and conversion rules are reported in Appendix A.

Coverage and characteristics. The processed networks range from 14 to approximately 1.69 million nodes, comprising 52 small ($N < 10^3$), 99 medium ($10^3 \leq N < 5\times10^4$), and 34 large ($N \geq 5\times10^4$) networks. They span social, infrastructure, biological, academic, information, communication, and other application settings, organized through a common domain taxonomy documented in Appendix A. Average degree ranges from approximately 2.1 to 685, average clustering from nearly zero to 0.94, and degree assortativity from -0.88 to 0.89. This coverage supports comparisons across network scales, domains, and structural characteristics; all statistics refer to the processed graphs.

3.2 METHODS

NETBREAKER evaluates 30 methods, comprising 18 non-learning and 12 learning-based methods. Static, adaptive, and reinsertion variants are retained when they correspond to distinct dismantling procedures and produce different removal sequences. The complete method list, taxonomy, implementation sources, and configurations are reported in Appendix B.

Non-learning methods. The non-learning methods include a random-based baseline, classical centrality heuristics, approximate-theory methods, decycling algorithms, and graph-partitioning or optimization-based approaches. They range from widely used strategies such as Degree, Betweenness, PageRank, and k-core to methods developed specifically for optimal percolation and network dismantling, including Collective Influence, BPD, Min-Sum, CoreHD, GND, Explosive Immunization, and LGD-NA CND (Morone & Makse, 2015; Mugisha & Zhou, 2016; Braunstein et al., 2016; Zdeborová et al., 2016; Ren et al., 2019; Clusella et al., 2016; Adler et al., 2026).

Learning-based methods. The learning-based methods cover supervised learning, self- or unsupervised learning, reinforcement learning, and generative program search. Direct dismantling methods include GDM, NIRM, Stable-AGDM, DCRS, FINDER, and MIND, together with their applicable variants (Grassia et al., 2021; Zhang & Wang, 2022; 2023; Fan et al., 2020b; Tian et al., 2026; Zhang et al., 2026). We additionally include learning-based critical-node methods as explicitly identified proxy baselines because they produce node rankings that can be evaluated through the same removal-sequence interface.

3.3 EVALUATION

Unified Output Representation. Dismantling methods produce heterogeneous outputs, including node scores, removal sets, and sequential decisions. Comparing them requires a common representation from which network fragmentation can be measured consistently. Since each intervention removes a node, we use the node removal sequence as this interface, preserving each method's solution procedure and post-processing. Scores induce a descending removal order, while set-valued solutions are ordered through the corresponding method's post-processing.

Given a removal sequence $\pi = (v_1, v_2, \ldots)$, the residual network and its normalized largest connected component after k removals are

$$G_k = G[V \setminus \{v_1, \ldots, v_k\}], \qquad L(k) = \frac{|\mathrm{LCC}(G_k)|}{N}. \tag{1}$$

A shared evaluator reconstructs these states on the same processed network for every method.

Multi-faceted Evaluation. Our evaluation is designed to assess both the quality of a dismantling solution and the resources required to obtain it. Quality concerns not only how many nodes must be removed to reach the fragmentation target, but also how much connectivity remains during sequential removal. These two summaries can still obscure how connected components split, motivating additional structural diagnostics. Practical use further requires knowing whether a method completes within the execution constraints and at what computational cost. These considerations lead to the following four evaluation dimensions.

Target-based evaluation. To measure the intervention required to reach the prescribed fragmentation level, we use the removal fraction at the first step satisfying the target. With $s = \lceil 0.10N \rceil$,

$$K = \min\{k : |\mathrm{LCC}(G_k)| \le s\}, \qquad \rho_{0.10} = \frac{K}{N}. \tag{2}$$

A smaller $\rho_{0.10}$ indicates that fewer removals are required.

Trajectory-based evaluation. To account for connectivity retained during dismantling, we measure the area under the normalized LCC trajectory up to the same target:

$$\mathrm{AUC}_{0.10} = \frac{1}{N}\,\mathrm{Simpson}\big(L(1), \ldots, L(K);\, \Delta k = 1\big). \tag{3}$$

A smaller value indicates lower accumulated connectivity before the target is reached.

Structural diagnostics. We examine pairwise connectivity and the evolution of the largest and second-largest connected components to characterize how fragmentation occurs. Diagnostics include mean pairwise connectivity up to the target, the largest single-step LCC drop, and the peak size of the second-largest component, together with the removal fractions at which the latter two occur. They describe structural changes along the trajectory rather than define an additional overall ranking.

Applicability and computational efficiency. We report completion coverage, defined as the fraction of networks on which a method produces a valid result under the benchmark constraints, and retain unsuccessful outcomes in cross-method comparisons. Computational cost is measured by observed wall-clock time and peak memory consumption, separating offline training from computation on the target network. The released experiment records preserve execution limits, non-completion categories, aggregation rules, and cost boundaries for every run.

3.4 RESEARCH QUESTIONS

We design NETBREAKER to systematically evaluate existing network dismantling methods from multiple perspectives. In particular, we aim to investigate the following research questions.

RQ1: How effective are learning-based methods for network dismantling?
Motivation and Experiment Design. Learning-based methods have increasingly been introduced into network dismantling, yet they are commonly evaluated on different network collections and against different sets of baselines. It therefore remains unclear whether these methods provide consistent improvements over strong non-learning approaches under a unified experimental setting. We compare learning-based and non-learning methods across the full benchmark, examining their dismantling performance and completion coverage under the same preprocessing, execution, and evaluation protocol.

RQ2: How does the performance of network dismantling methods vary across network scales and domains?
Motivation and Experiment Design. Real-world networks differ substantially in scale and application domain, and a method that performs well on one group of networks may not retain its advantage on another. We therefore evaluate all methods across networks of different scales and across the application domains represented in the benchmark. By comparing their performance rankings and completion coverage within each group, we examine whether the relative strengths of different methods remain stable across heterogeneous network data.

RQ3: How do different evaluation metrics affect the assessment of network dismantling methods?
Motivation and Experiment Design. Network dismantling performance can be evaluated from different perspectives. Target-based evaluation measures the number of removals required to reach a prescribed fragmentation level, trajectory-based evaluation measures how effectively connectivity is reduced throughout the dismantling process, and structural diagnostics characterize when and how components split. Since these metrics reward different properties of a removal sequence, they may produce different conclusions about method performance. We compare method rankings and rank changes across all three perspectives to determine how strongly the metric choice affects the resulting assessment.

RQ4: What are the time and space costs of network dismantling methods?
Motivation and Experiment Design. Dismantling quality alone does not determine whether a method can be applied to large real-world networks. Different methods incur substantially different computational costs during training, graph-specific adaptation or search, inference, and post-processing. We therefore compare the observed wall-clock time and peak memory consumption of all methods, while retaining timeout, out-of-memory, and pre-declared skip outcomes as part of the analysis. This evaluation characterizes the trade-off between dismantling effectiveness, computational efficiency, and practical applicability.

Table 2: Failure-aware aggregate comparison across 185 benchmark networks. Values are medians over successful dataset-level evaluations. Superscripts **1–3** mark the three best failure-aware average ranks for each metric. The upward arrows on max ΔLCC and max SLCC indicate larger transition magnitudes, not standalone dismantling superiority.

		Target-based	Trajectory-based		Transition diagnostics				
ID	Method	$\rho_{0.10}\downarrow$	LCC-AUC↓	PairConn↓	max ΔLCC↑	$\rho(\Delta)\downarrow$	max SLCC↑	$\rho(\text{SLCC})\downarrow$	Coverage
1	Random	0.773	0.397	0.349	0.017	0.411	0.004	0.471	100.0%
2	Static Degree	0.200	0.108	0.377	0.047	0.124	0.032	0.142	100.0%
3	Adaptive Degree	0.161	0.089	0.417	0.053	0.116	0.042	0.135	100.0%
4	Static Betweenness	0.234	0.121	**0.316**[1]	0.053	0.148	0.040	0.169	87.0%
5	Adaptive Betweenness	0.103	0.054	0.387	0.193	0.054	0.178	0.058	67.0%
6	PageRank	0.175	0.091	**0.361**[2]	0.047	0.119	0.031	0.133	100.0%
7	k-core / k-shell	0.272	0.148	0.377	0.047	0.189	0.035	0.211	100.0%
8	Cycle Ratio	0.241	0.128	0.347	0.052	0.140	0.037	0.186	83.8%
9	Collective Influence	0.137	**0.079**[2]	0.360	0.052	**0.080**[3]	0.044	0.098	95.1%
10	BPD	**0.107**[1]	**0.077**[3]	0.559	**0.173**[3]	**0.095**[2]	**0.129**[1]	**0.092**[2]	98.4%
11	Min-Sum	**0.109**[3]	0.083	0.582	0.183	0.100	**0.096**[3]	0.101	96.8%
12	CoreHD	**0.107**[2]	0.082	0.583	**0.179**[2]	0.096	**0.101**[2]	**0.093**[3]	100.0%
13	GND	0.147	0.071	0.380	0.134	0.071	0.118	0.087	93.0%
14	GNDR	0.117	0.101	0.736	**0.330**[1]	0.112	0.095	0.105	93.0%
15	Explosive Immunization	0.119	**0.065**[1]	0.379	0.091	**0.075**[1]	0.080	**0.083**[1]	99.5%
16	Ensemble-GND	0.109	0.064	0.409	0.218	0.066	0.201	0.069	63.8%
17	Ensemble-GNDR	0.099	0.082	0.732	0.331	0.093	0.096	0.092	63.8%
18	LGD-NA CND	0.168	0.093	0.396	0.080	0.113	0.056	0.118	78.9%
19	GDM	0.161	0.095	0.402	0.058	0.121	0.028	0.133	91.9%
20	GDM+R	0.122	0.082	0.485	0.148	0.108	0.095	0.108	90.3%
21	CoreGDM	0.120	0.081	0.508	0.155	0.103	0.098	0.104	91.4%
22	NIRM	0.298	0.134	0.362	0.055	0.140	0.042	0.203	82.2%
23	Stable-AGDM	0.195	0.114	0.386	0.056	0.117	0.031	0.142	99.5%
24	Stable-AGDM+R	0.114	0.087	0.556	0.176	0.110	0.096	0.105	90.8%
25	DCRS	0.340	0.145	**0.278**[3]	0.043	0.115	0.021	0.178	81.1%
26	FINDER	0.153	0.092	0.400	0.051	0.117	0.040	0.135	100.0%
27	MIND	0.147	0.086	0.377	0.049	0.096	0.034	0.118	85.4%
28	LLM4CN	0.195	0.103	0.291	0.108	0.074	0.080	0.098	54.6%
29	CNT	0.771	0.398	0.351	0.024	0.396	0.008	0.463	77.3%
30	AGNN	0.392	0.192	0.347	0.061	0.252	0.043	0.275	81.1%

4 Experimental Results and Analysis

4.1 Effectiveness of Learning-based Methods (RQ1)

Settings. We rank methods on each network and average these failure-aware ranks, placing unsuccessful methods after all successful ones. Table 2 reports medians over successful networks together with completion coverage.

Results. BPD, CoreHD, and Min-Sum lead the target metric, while Explosive Immunization gives the best LCC-AUC rank. The strongest learning-based methods remain behind these baselines once failed runs are included. FINDER is the only learning-based method that completes all 185 networks; the others trade competitive successful-run results for lower coverage.

Key Insights for RQ1: Learning-based methods do not yet outperform strong non-learning methods consistently. Their best results are competitive, but no learning-based method combines leading dismantling performance with complete coverage across the 185 networks.

4.2 Variation across Network Domains and Scales (RQ2)

Settings. We compare failure-aware target ranks across eight application domains and four size ranges ($<$ 1K, 1K–10K, 10K–100K, and $\geq$ 100K nodes). Figure 3 reports method-level variation, while Appendix Table 10 provides fixed method-family representatives.

Results. The strongest methods remain stable across most domains, with BPD leading the fixed family representatives in seven of eight domains. Scale produces larger changes: Adaptive Betweenness performs well below 10K nodes but completes no network above 100K nodes, whereas BPD

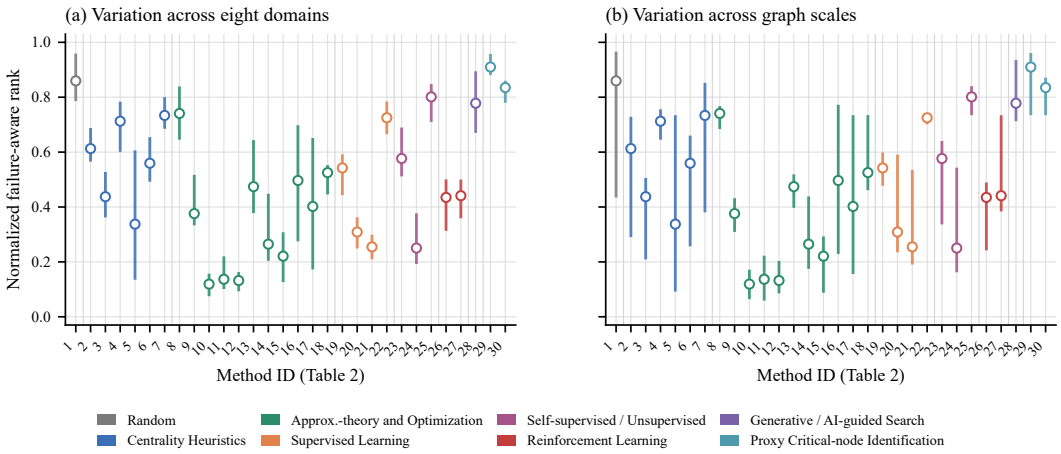


Figure 3: Method sensitivity to network domain and scale. Circles denote the overall normalized failure-aware target rank; vertical segments span the best and worst group-specific ranks across eight domains (left) or four size ranges (right). Lower is better. Method IDs refer to Table 2.

and CoreHD remain competitive throughout. FINDER covers all large networks at the cost of weaker dismantling quality on smaller ones.

Key Insights for RQ2: Scale matters more than domain. Some methods that perform well on small networks become impractical or fail completely as the network grows. Results on large networks should therefore be reported together with completion coverage.

4.3 Metric-dependent Assessment (RQ3)

Settings. Appendix Table 11 compares the five highest-ranked methods under the target criterion and two trajectory summaries, while Appendix Figure 5 reports all seven rank profiles. Transition diagnostics describe how fragmentation occurs rather than define independent measures of overall superiority.

Results. BPD, CoreHD, and Min-Sum rank highly by the target criterion but poorly by PairConn, whereas Static Betweenness, PageRank, and DCRS show the opposite pattern. Explosive Immunization performs well under both target- and trajectory-based evaluation, and the structural diagnostics distinguish gradual fragmentation from a few large component splits.

Key Insights for RQ3: Different metrics reward different removal sequences. Target and trajectory measures should be reported together, while structural diagnostics are most useful for explaining why their rankings differ.

4.4 Observed Computational Cost and Scalability (RQ4)

Settings. Figure 4 pairs failure-aware target rank with median successful-run time and peak host memory, retains coverage in marker area, and reports runtime across four size ranges. Runtime includes per-graph adaptation and search; heterogeneous CPU and GPU implementations support practical within-protocol comparisons rather than hardware-independent complexity claims. Trajectory-based results are provided in Appendix D.

Results. CoreHD and BPD combine strong target-based results with median running times below two seconds, while Explosive Immunization achieves the best trajectory rank in about one second. Among learning-based methods, FINDER is fast and complete, whereas methods with better dismantling quality lose coverage on larger networks. LLM4CN marks the opposite extreme, with a median runtime of 4,676 seconds and 54.6% coverage.

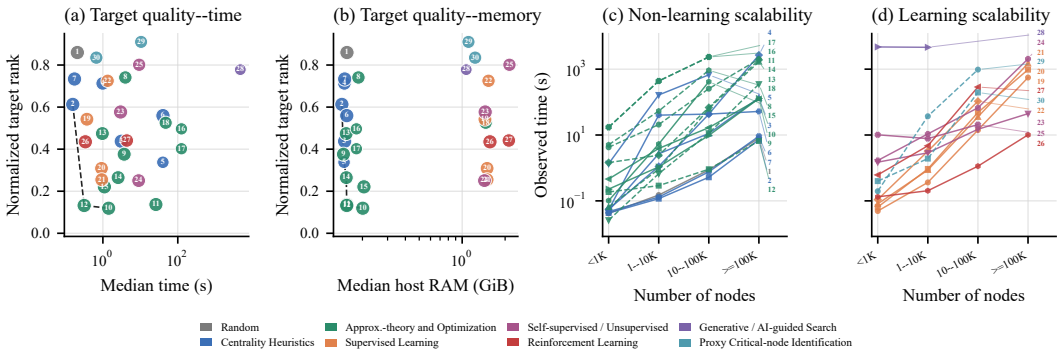


Figure 4: Observed target-quality cost trade-offs and runtime scaling over 185 networks. Panels (a) and (b) compare the target rank with median successful-run time and peak host memory; lower and further left is better, dashed lines mark non-dominated observations, and marker area encodes coverage. Panels (c) and (d) show median time by network size for non-learning and learning-based methods, respectively; absent size ranges are not connected. Method IDs refer to Table 2.

Key Insights for RQ4: More computation does not produce better dismantling performance in our experiments. Several non-learning methods are both faster and more effective than the learning-based alternatives. A low median runtime does not necessarily indicate good scalability when a method fails to complete many large networks. Runtime should therefore be reported together with completion coverage.

5 Conclusion and Future Directions

This paper presents NetBreaker, a comprehensive benchmark of 30 network dismantling methods on 185 networks under a common execution and evaluation protocol. NetBreaker converts different method outputs into node removal sequences, enabling direct comparison between non-learning and learning-based methods through the same evaluation procedure. The results show that current learning-based methods can be competitive in selected settings, but do not consistently outperform strong non-learning methods. Method rankings vary across evaluation metrics, while network scale affects both dismantling performance and completion coverage. These findings show that progress in network dismantling cannot be adequately assessed using a single metric or a limited collection of networks.

Our findings leads to some promising directions for future research:

Multi-objective learning. Target-based evaluation, trajectory-based evaluation, and structural transitions describe different properties of a removal sequence. Optimizing one property does not necessarily improve the others. Future learning-based methods should therefore explore learning frameworks that coordinate multiple dismantling objectives while preserving the distinctions among final intervention cost, connectivity throughout the removal process, and the structural process of fragmentation.

Reliable scaling and coverage. Performance degradation and incomplete execution on large networks remain important limitations of current methods. Future research should develop methods that generalize reliably across network scales and retain high completion coverage as network size increases. Promising directions include training on networks with diverse scales, learning representations invariant to graph size, and designing proper inductive inference procedures.

Efficiency-aware method development. Strong non-learning methods continue to provide favorable trade-offs between dismantling effectiveness and running time. Future learning-based methods should pursue clear improvements across effectiveness, completion coverage, running time, and memory consumption, rather than improving one aspect at the expense of the others. The NetBreaker codebase will provide a common basis for measuring these advances.

REFERENCES

Thomas Adler, Marco Grassia, Ziheng Liao, Giuseppe Mangioni, and Carlo Vittorio Cannistraci. Latent geometry-driven network automata for complex network dismantling. In *International Conference on Learning Representations*, 2026.

Oriol Artime, Marco Grassia, Manlio De Domenico, James P. Gleeson, Hernán A. Makse, Giuseppe Mangioni, Matjaž Perc, and Filippo Radicchi. Robustness and resilience of complex networks. *Nature Reviews Physics*, 6(2):114–131, 2024. doi: 10.1038/s42254-023-00676-y.

Alfredo Braunstein, Luca Dall'Asta, Guilhem Semerjian, and Lenka Zdeborová. Network dismantling. *Proceedings of the National Academy of Sciences*, 113(44):12368–12373, 2016.

Duan-Bing Chen, Hui Gao, Linyuan Lü, and Tao Zhou. Identifying influential nodes in large-scale directed networks: The role of clustering. *PLOS ONE*, 8(10):e77455, 2013. doi: 10.1371/journal.pone.0077455.

Liang Chen et al. Identifying influential nodes in complex networks via transformer. *Information Processing & Management*, 61:103775, 2024.

Peiyu Chen and Wenhui Fan. Identifying critical nodes via link equations and deep reinforcement learning. *Neurocomputing*, 562:126871, 2023. doi: 10.1016/j.neucom.2023.126871.

Pau Clusella, Peter Grassberger, Francisco J. Pérez-Reche, and Antonio Politi. Immunization and targeted destruction of networks using explosive percolation. *Physical Review Letters*, 117:208301, 2016.

Reuven Cohen, Shlomo Havlin, and Daniel ben Avraham. Efficient immunization strategies for computer networks and populations. *Physical Review Letters*, 91:247901, 2003. doi: 10.1103/PhysRevLett.91.247901.

Ye Deng, Dingrong Tan, Xiaoda Shen, Zhigang Wang, and Jun Wu. A hybrid approach to network disintegration: Integrating graph convolutional network and genetic algorithm. *Chaos, Solitons & Fractals*, 197:116488, 2025. doi: 10.1016/j.chaos.2025.116488.

Changjun Fan, Li Zeng, Yanghe Feng, Guangquan Cheng, Jincai Huang, and Zhong Liu. A novel learning-based approach for efficient dismantling of networks. *International Journal of Machine Learning and Cybernetics*, 11:2101–2111, 2020a. doi: 10.1007/s13042-020-01104-8.

Changjun Fan, Li Zeng, Yizhou Sun, and Yang-Yu Liu. Finding key players in complex networks through deep reinforcement learning. *Nature Machine Intelligence*, 2:317–324, 2020b.

Tianlong Fan, Linyuan Lü, Dinghua Shi, and Tao Zhou. Characterizing cycle structure in complex networks. *Communications Physics*, 4:272, 2021. doi: 10.1038/s42005-021-00781-3.

Jun Fu, Xiaojie Sun, Peng Zhang, and Rolf Findeisen. Exploring diverse solutions in network dismantling with generative model. *Chaos, Solitons & Fractals*, 202:117526, 2026. doi: 10.1016/j.chaos.2025.117526.

Marco Grassia and Giuseppe Mangioni. CoreGDM: Geometric deep learning network decycling and dismantling. In *Complex Networks XIV*, Springer Proceedings in Complexity, pp. 86–94. Springer, 2023. doi: 10.1007/978-3-031-28276-8_8.

Marco Grassia, Manlio De Domenico, and Giuseppe Mangioni. Machine learning dismantling and early-warning signals of disintegration in complex systems. *Nature Communications*, 12:5190, 2021.

Jihui Han, Yupeng Shi, Daoying Huang, Ziqi Huang, and Anlin Zhang. Graph learning-based network dismantling via hierarchical feature fusion and margin-ranked optimization. *Journal of Complex Networks*, 13(4):cnaf015, 2025. doi: 10.1093/comnet/cnaf015.

Petter Holme, Beom Jun Kim, Chang No Yoon, and Seung Kee Han. Attack vulnerability of complex networks. *Physical Review E*, 65:056109, 2002. doi: 10.1103/PhysRevE.65.056109.

Li Hong, Mengqiao Xu, Yu Liu, Xuemei Zhang, and Changjun Fan. A deep learning dismantling approach for understanding the structural vulnerability of complex networks. *Chaos, Solitons & Fractals*, 194:116148, 2025. doi: 10.1016/j.chaos.2025.116148.

Ru Huang, Shuo Zhou, Pengfei Li, Kun Yang, Zijian Chen, Jianhua He, Xiaoli Chu, Zhengbing Zhou, and Guangtao Zhai. A sparse transformer-enhanced graph convolutional model for robust node importance ranking in complex networks. *IEEE Transactions on Network Science and Engineering*, 13:2350–2370, 2026. doi: 10.1109/TNSE.2025.3614439.

Karen E. Joyce, Paul J. Laurienti, Jonathan H. Burdette, and Satoru Hayasaka. A new measure of centrality for brain networks. *PLOS ONE*, 5(8):e12200, 2010. doi: 10.1371/journal.pone.0012200.

Maksim Kitsak, Lazaros K. Gallos, Shlomo Havlin, Fredrik Liljeros, Lev Muchnik, H. Eugene Stanley, and Hernán A. Makse. Identification of influential spreaders in complex networks. *Nature Physics*, 6:888–893, 2010. doi: 10.1038/nphys1746.

Jérôme Kunegis. KONECT: The koblenz network collection. In *Proceedings of the 22nd International Conference on World Wide Web Companion*, pp. 1343–1350, 2013. doi: 10.1145/2487788.2488173.

Jure Leskovec and Andrej Krevl. SNAP datasets: Stanford large network dataset collection. `https://snap.stanford.edu/data`, 2014.

Qingxia Liu and Bang Wang. Neural extraction of multiscale essential structure for network dismantling. *Neural Networks*, 154:99–108, 2022. doi: 10.1016/j.neunet.2022.07.015.

Siqi Ma, Weixin Zeng, Weidong Xiao, and Xiang Zhao. Dismantling complex networks with graph contrastive learning and multi-hop aggregation. *Information Sciences*, 676:120780, 2024. doi: 10.1016/j.ins.2024.120780.

Jinzhu Mao, Dongyun Zou, Li Sheng, Siyi Liu, Chen Gao, Yue Wang, and Yong Li. Identify critical nodes in complex network with large language models. *arXiv preprint arXiv:2403.03962*, 2024. URL `https://arxiv.org/abs/2403.03962`.

Flaviano Morone and Hernán A. Makse. Influence maximization in complex networks through optimal percolation. *Nature*, 524:65–68, 2015.

Sheila Mugisha and Hai-Jun Zhou. Identifying optimal targets of network attack by belief propagation. *Physical Review E*, 94:012305, 2016.

Sai Munikoti, Laya Das, and Balasubramaniam Natarajan. Scalable graph neural network-based framework for identifying critical nodes and links in complex networks. *Neurocomputing*, 468: 211–221, 2022. doi: 10.1016/j.neucom.2021.10.031.

Saeed Osat, Fragkiskos Papadopoulos, Andreia Sofia Teixeira, and Filippo Radicchi. Embedding-aided network dismantling. *Physical Review Research*, 5:013076, 2023. doi: 10.1103/PhysRevResearch.5.013076.

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The PageRank citation ranking: Bringing order to the web. Technical report, Stanford University, 1998.

Tianle Pu, Li Zeng, and Chao Chen. Deep reinforcement learning for network dismantling: A K-core based approach. *Mathematics*, 12:1215, 2024. doi: 10.3390/math12081215.

Hongbo Qu, Xu Wang, Yu-Rong Song, Wei Ni, Guo-Ping Jiang, and Quan Z. Sheng. GRLND: A graph reinforcement learning framework for network dismantling. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management*, pp. 2450–2459, 2025. doi: 10.1145/3746252.3761371.

Xiao-Long Ren and Nino Antulov-Fantulin. Ensemble approach for generalized network dismantling. *arXiv preprint arXiv:1909.09016*, 2019. URL `https://arxiv.org/abs/1909.09016`.

Xiao-Long Ren, Niels Gleinig, Dirk Helbing, and Nino Antulov-Fantulin. Generalized network dismantling. *Proceedings of the National Academy of Sciences*, 116(14):6554–6559, 2019.

Ryan A. Rossi and Nesreen K. Ahmed. The network data repository with interactive graph analytics and visualization. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, pp. 4292–4293, 2015.

Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Gallagher, and Tina Eliassi-Rad. Collective classification in network data. *AI Magazine*, 29(3):93–106, 2008.

Jinchen Shi, Jiuyang Tang, Zhen Tan, Yanli Hu, and Yang Fang. Complex network dismantling with node diffusion and node similarity. *Expert Systems with Applications*, 300:130433, 2026. doi: 10.1016/j.eswa.2025.130433.

Hao Tian, Pietro Ferraro, Robert Shorten, Mahdi Jalili, and Homayoun Hamedmoghadam. Learning network dismantling without handcrafted inputs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2026.

Sebastian Wandelt, Xiaoqian Sun, Daozhong Feng, Massimiliano Zanin, and Shlomo Havlin. A comparative analysis of approaches to network-dismantling. *Scientific Reports*, 8(1):13513, 2018. doi: 10.1038/s41598-018-31902-8.

Min Wu, Wu Shi, Fengwei Guo, Bitao Dai, Jianhong Mou, Suoyi Tan, Xin Lu, and Chaomin Ou. A reinforcement learning-enhanced meta-heuristic framework for network dismantling. *Journal of Physics: Complexity*, 6:035012, 2025. doi: 10.1088/2632-072X/adfff1.

Ying Xing, Hui Shu, and Fei Kang. PeerRemove: An adaptive node removal strategy for P2P botnet based on deep reinforcement learning. *Computers & Security*, 128:103129, 2023. doi: 10.1016/j.cose.2023.103129.

Y. Xiong, Z. Hu, C. Su, S.-M. Cai, and T. Zhou. Vital node identification in complex networks based on autoencoder and graph neural network. *Applied Soft Computing*, 163:111895, 2024.

Xiang Xu, Cheng Zhu, Qingyong Wang, Xianqiang Zhu, and Yun Zhou. Identifying vital nodes in complex networks by adjacency information entropy. *Scientific Reports*, 10:2691, 2020. doi: 10.1038/s41598-020-59616-w.

Lenka Zdeborová, Pan Zhang, and Hai-Jun Zhou. Fast and simple decycling and dismantling of networks. *Scientific Reports*, 6:37954, 2016.

Junhao Zhang and Biao Wang. Dismantling complex networks by a neural model trained from tiny networks. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2022.

Junhao Zhang and Biao Wang. Encoding node diffusion competence and role significance for network dismantling. In *Proceedings of the ACM Web Conference*, 2023.

Pan Zhang, Jin-Hua Fu, Xiao Sun, Tao Cheng, and Guanrong Chen. Neural algorithmic approach to network dismantling. *Science China Information Sciences*, 69:119203, 2026.

Xiuming Zhao, Hongtao Yu, Ruiyang Huang, Shuxin Liu, Nan Hu, and Xiaochun Cao. A novel higher-order neural network framework based on motifs attention for identifying critical nodes. *Physica A: Statistical Mechanics and its Applications*, 629:129194, 2023. doi: 10.1016/j.physa.2023.129194.

Wennan Zhou, Suoyi Tan, Yang Fang, Xin Lü, and Xiang Zhao. Dismantling complex networks based on higher-order graph neural network. *Communications Physics*, 9:181, 2026. doi: 10.1038/s42005-026-02601-y.

A Benchmark Dataset Details

A.1 The Taxonomy of Dataset Domains

Existing network dismantling studies use substantially inconsistent taxonomies for dataset domains. Some studies provide no explicit categorization, whereas others organize networks using either a few broad domains or multi-level schemes with dozens of types; even similarly named categories often differ in scope. Retaining these study-specific labels would therefore make domain coverage incomparable across studies. We consequently reclassify every real-world dataset according to the system represented by the network, rather than its source repository, file format, or edge type. Table 3 summarizes the schemes adopted in existing studies, and Table 4 defines the eight-domain hierarchy used in our benchmark. Each real-world network is assigned one primary domain and one sub-domain, while the synthetic reference is recorded separately and excluded from domain coverage.

Table 3: The Taxonomy of Dataset Domains Adopted in Existing Network Dismantling Studies.

Study	Categorization Scheme	Domain Categories
Wandelt et al. (Wandelt et al., 2018)	Not provided	—
Artime et al. (Artime et al., 2024)	Single-level, 4 categories	Biological; Information; Social; Technological
FINDER (Fan et al., 2020b)	Single-level, 5 categories	Criminal; Biological; Communication; Infrastructure; Social
GDM (Grassia et al., 2021)	Single-level, 16 categories	Infrastructure; Social; Metabolic; Hyperlink; Citation; Coauthorship; Neural; Economic; Communication; Power; Trophic; Software; Transport; Human Contact; Computer; Lexical
MIND (Tian et al., 2026)	Single-level, 4 categories	Biological; Social; Information; Technological
LGD-NA (Adler et al., 2026)	Two-level, 8 fields and 32 types	Biomolecular; Brain; Covert; Food Web; Infrastructure; Internet; Miscellaneous; Social

Table 4: The unified taxonomy of dataset domains and sub-domains used in this study.

Domains	Sub-domains
Biological and Ecological	Protein and molecular interaction; Gene regulation; Metabolic interaction; Neural connectivity; Drug interaction; Food webs; Animal social interaction
Social and Human Interaction	Friendship and online social interaction; Trust and reputation; Human communication; Physical contact; Professional and organizational interaction; Location-based interaction; Political and geopolitical interaction; Entertainment and cultural interaction; Sports interaction
Information, Knowledge, and Media	Web and hyperlinks; Online media; Knowledge and encyclopedia; Lexical and linguistic; Literary and fictional; Collaborative editing and voting
Scientific and Scholarly	Citation; Coauthorship; Scientific collaboration
Technological and Communication Infrastructure	Internet and autonomous systems; Peer-to-peer systems; Power grids; Software dependency; Electronic circuits
Transportation and Mobility	Roads; Airlines; Urban transit; Maritime shipping; Traffic and geographic mobility
Economic and Financial	Trade; Commerce and co-purchasing; Financial trust and transactions
Security, Crime, and Covert	Crime; Terrorism; Corruption; Botnets and malicious activity

A.2 Network Collection, Preprocessing, and Statistics

The benchmark contains 185 distinct processed graphs: 184 real-world networks organized under the eight-domain taxonomy in Table 4, together with one synthetic reference. Table 5 summarizes the number, processed scale, median graph-average degree, and size distribution of the networks in each

domain. The accompanying dataset registry records per-network identifiers, provenance, domain and sub-domain assignments, processed node and edge counts, preprocessing outcomes, and structural statistics.

Table 5: Dataset statistics for 184 real-world networks across eight unified domains and one synthetic reference. Node and edge counts refer to processed simple, undirected largest connected components. The four size counts correspond to $N < 10^3$, $10^3 \leq N < 10^4$, $10^4 \leq N < 10^5$, and $N \geq 10^5$, matching the scale analysis in Section 4.2. The degree column reports the median graph-average degree; k and M denote 10^3 and 10^6.

Domain	Graphs	Nodes: range	Edges: range	Med. $\langle k \rangle$	Size counts
Biological and Ecological	26	14–324.2k	20–2.64M	7.74	11/13/1/1
Social and Human Interaction	54	49–1.63M	107–22.30M	10.39	15/12/14/13
Information, Knowledge, and Media	22	77–855.8k	167–4.29M	7.52	5/9/4/4
Scientific and Scholarly	23	86–365.2k	124–1.72M	7.94	2/6/13/2
Technological and Communication Infrastructure	28	512–1.69M	819–11.09M	5.25	1/13/12/2
Transportation and Mobility	18	39–1.35M	129–1.88M	3.42	9/6/1/2
Economic and Financial	6	105–334.9k	441–925.9k	7.94	3/2/0/1
Security, Crime, and Covert	7	64–4,801	243–882.6k	7.59	5/2/0/0
Synthetic reference	1	84	775	18.45	1/0/0/0
All networks	185	14–1.69M	20–22.30M	7.35	52/63/45/25

All inputs are converted into simple, undirected, and unweighted graphs: self-loops are removed, duplicate or reciprocal edge records are merged, the largest connected component is retained, and node identifiers are relabeled consistently. Source-level and processed counts are both preserved to document these transformations. Networks whose meaning depends on edge direction, weights, temporal order, or multiple layers are excluded rather than treated as equivalent static inputs. Among 219 catalogue records, 213 were processed successfully; collapsing 28 provenance aliases yields 185 distinct graphs, while four failed records and two quarantined records are excluded.

For each network, we retain the unified domain label and processed node count used in Sections 4.2 and 4.4, together with edge count, average degree, degree coefficient of variation, average clustering, and degree assortativity for reproducibility and structural diagnostics. The synthetic graph is labeled explicitly as a reference rather than presented as a systematic synthetic benchmark. Dataset-level provenance and any reported license information follow the original documentation and are not inferred from file availability.

B Benchmark Algorithms

B.1 Method Taxonomy and Inventory

This appendix first summarizes the taxonomy and implementation provenance of the 30 evaluated methods, and then introduces each method according to the same family organization used throughout the benchmark. Methods reviewed during benchmark construction but not included in the evaluation are listed separately at the end of the appendix.

Table 6 provides the family-level taxonomy, while Tables 7 and 8 enumerate the individual methods and variants. Each "link" points to the implementation used or reviewed for the corresponding method. Availability and licensing were checked at the time of the literature review and will be re-verified before release; a locally retained copy does not by itself establish complete provenance or an explicit license.

Table 6: The 30 evaluated methods by family. IDs refer to Table 2; "+R" denotes a reinsertion variant. Full descriptions and implementation sources are in Appendix B.

Family	Methods (ID)	n	Removal order formed by	Learned from
Random control	Random (1)	1	seeded permutation	–
Centrality heuristics	Degree and Betweenness (static, adaptive), PageRank, k-core (2–7)	6	one-pass score, or recomputation after each removal	–
Approx. theory and optimization	Cycle Ratio, CI, BPD, Min-Sum, CoreHD, GND(+R), Explosive Imm., Ensemble-GND(+R), LGD-NA CND (8–18)	11	cycle analysis, message passing, decycling, or partition search	–
Supervised learning	GDM(+R), CoreGDM, NIRM (19–22)	4	GNN score; +R adds greedy reinsertion	small graphs with optimal labels
Self-/unsupervised	Stable-AGDM(+R), DCRS (23–25)	3	graph scoring or per-graph adaptation	unlabeled graph objectives
Reinforcement learning	FINDER, MIND (26–27)	2	sequential policy on the residual graph	synthetic graphs
Generative search	LLM4CN (28)	1	LLM-evolved scoring program per graph	target graph
Proxy critical-node	CNT, AGNN (29–30)	2	one-pass influence ranking	SIR spreading labels

Table 7: Evaluated non-learning methods and variants (18 entries). Modes are one-pass (static), adaptive, or global search.

ID	Method / variant	Subcategory	Mode	Output	Implementation source
1	Random	negative control	one-pass	sequence	link
2	Static Degree	centrality	one-pass	score	link
3	Adaptive Degree	centrality	adaptive	sequence	link
4	Static Betweenness	centrality	one-pass	score	link
5	Adaptive Betweenness	centrality	adaptive	sequence	link
6	PageRank	centrality	one-pass	score	link
7	k-core/k-shell	centrality	one-pass	score	link
8	Cycle Ratio	centrality	one-pass	score	link
9	Collective Influence	approximate theory	adaptive	sequence	link
10	BPD	approximate theory	adaptive	sequence	link
11	Min-Sum	optimization	global search	set	link
12	CoreHD	approximate theory	adaptive	sequence	link
13	GND	optimization	global search	set	link
14	GNDR	optimization + reinsertion	global search	sequence	link
15	Explosive Immunization	approximate theory	adaptive	sequence	link
16	Ensemble-GND	optimization ensemble	global search	set	link
17	Ensemble-GNDR	ensemble + reinsertion	global search	sequence	link
18	LGD-NA CND	latent-geometry optimization	adaptive	sequence	link

Table 8: Evaluated learning-based and AI-guided methods and variants (12 entries). CoreGDM uses a third-party implementation, and the available LLM4CN artifact has unresolved provenance and completeness.

ID	Method / variant	Role	Training signal	Mode	Output	Reviewed availability
19	GDM	direct	supervised	one-pass	score	link
20	GDM+R	direct	supervised	one-pass + reinsertion	sequence	link
21	CoreGDM	direct hybrid	supervised	adaptive	sequence	link
22	NIRM	direct	supervised	one-pass	score	link
23	Stable-AGDM	direct	supervised	one-pass	score	link
24	Stable-AGDM+R	direct	supervised	one-pass + reinsertion	sequence	link
25	DCRS	direct	self/unsupervised	per-graph adaptation + one-pass	score	link
26	FINDER	direct	reinforcement	adaptive policy	sequence	link
27	MIND	direct	reinforcement	adaptive policy	sequence	link
28	LLM4CN	direct	generative search	program search	score	link
29	CNT	proxy	supervised SIR labels	one-pass	score	link
30	AGNN	proxy	supervised SIR ranking	per-graph adaptation + one-pass	score	link

B.2 Method Descriptions

B.2.1 Non-learning Methods

The non-learning methods are organized into a random control, centrality heuristics, and approximate-theory or optimization-based approaches.

Random control and centrality heuristics.

- **Random** is included as a negative-control dismantling strategy. It generates a seeded random permutation of all nodes and therefore uses neither structural information nor residual-graph adaptation.
- **Static Degree** (Holme et al., 2002) ranks nodes once by their degree in the original network. It represents the simplest local-centrality strategy and does not update scores after removals.
- **Adaptive Degree** (Holme et al., 2002) repeatedly removes a highest-degree node from the current residual network. By recomputing degrees after each intervention, it captures structural changes that the static ranking ignores.
- **Static Betweenness** (Holme et al., 2002) ranks nodes once according to shortest-path betweenness in the original network. It targets nodes that mediate many shortest paths, but does not revise their importance after the topology changes.
- **Adaptive Betweenness** (Holme et al., 2002) recomputes shortest-path betweenness on the residual network after each removal. It therefore follows the migration of shortest-path bottlenecks during dismantling at a substantially higher computational cost than the static variant.
- **PageRank** (Page et al., 1998) assigns node importance through the stationary distribution of a random walk with teleportation. The resulting scores are computed once and converted into a descending removal order.
- **k-core/k-shell** (Kitsak et al., 2010) decomposes a network into nested shells and prioritizes nodes with larger core numbers. It captures membership in densely connected regions while producing a one-pass ranking.
- **Cycle Ratio** (Fan et al., 2021) measures node importance from participation in elementary cycles and the overlap among those cycles. It prioritizes nodes whose removal disrupts cyclic connectivity and produces a one-pass ranking.

Approximate-theory and optimization-based methods.

- **Collective Influence** (Morone & Makse, 2015) is proposed to approximate optimal percolation through a finite-radius score that combines a node's excess degree with the degrees on its boundary. Scores are updated on the residual network to construct an adaptive dismantling sequence.
- **BPD** (Mugisha & Zhou, 2016) is proposed to identify a small feedback vertex set using belief-propagation marginals and decimation. Tree breaking and reinsertion then convert the decycling solution into a dismantling sequence.
- **Min-Sum** (Braunstein et al., 2016) formulates network decycling as a message-passing optimization problem. It first identifies a decycling set and then breaks the remaining forest and reinserts redundant removals to obtain an ordered dismantling solution.
- **CoreHD** (Zdeborová et al., 2016) is proposed as a fast decycling heuristic that repeatedly removes a highest-degree node from the current 2-core. The resulting forest is subsequently broken and processed by reinsertion to form the final dismantling sequence.
- **GND** (Ren et al., 2019) recursively partitions a network through weighted spectral bisection and approximates a vertex cover over the cut edges. The union of selected separator nodes forms its global dismantling set.
- **GNDR** (Ren et al., 2019) extends GND with a reinsertion stage that removes redundant nodes and orders the retained solution. This refinement converts the global candidate set into a sequential dismantling strategy.
- **Explosive Immunization** (Clusella et al., 2016) constructs a removal order by reversing an explosive-percolation growth process. Its score is designed to delay the emergence of a giant connected component and thereby identify nodes important for global connectivity.
- **Ensemble-GND** (Ren & Antulov-Fantulin, 2019) aggregates multiple randomized GND solutions to reduce dependence on a single spectral partition. Candidate dismantling sets are combined into an ensemble solution without the final reinsertion refinement.

- **Ensemble-GNDR** (Ren & Antulov-Fantulin, 2019) applies reinsertion to the ensemble GND solution, eliminating redundant removals and producing an ordered sequence. It is evaluated separately because the reinsertion stage changes both solution size and trajectory.
- **LGD-NA CND** (Adler et al., 2026) uses the CND variant of the latent-geometry-driven neural automaton to construct an adaptive dismantling sequence. We evaluate the released procedure as a complete optimization method through the benchmark's common removal-sequence interface.

B.2.2 LEARNING-BASED AND AI-GUIDED METHODS

The learning-based methods are grouped by their training or search mechanism: supervised learning, self-supervised or unsupervised learning, reinforcement learning, generative search, and proxy critical-node identification.

Supervised learning.

- **GDM** (Grassia et al., 2021) is proposed to learn transferable dismantling scores from high-quality solutions on small training graphs. It uses graph neural representations to rank nodes in an unseen network with a single forward scoring stage.
- **GDM+R** (Grassia et al., 2021) augments the raw GDM ranking with greedy reinsertion. The refinement removes redundant selected nodes and produces a sequence whose terminal set differs from that of the direct score order.
- **CoreGDM** (Grassia & Mangioni, 2023) combines the supervised GDM scorer with a CoreHD-style decycling pipeline. It repeatedly applies learned scores within the current 2-core, followed by tree breaking and reinsertion.
- **NIRM** (Zhang & Wang, 2022) is proposed to learn transferable node-importance scores from dismantling targets generated on small graphs. It propagates target labels and combines local and global encoders to represent both neighborhood structure and network-level context.
- **Stable-AGDM** (Zhang et al., 2026) is proposed to improve the stability of learned dismantling scores across different network topologies. It combines supervised message passing with feature selection and outputs a one-pass node ranking.
- **Stable-AGDM+R** (Zhang et al., 2026) applies the released refinement and reinsertion procedure to the Stable-AGDM ranking. The post-processing stage removes redundant interventions and changes the resulting dismantling trajectory.

Self-supervised and unsupervised learning.

- **DCRS** (Zhang & Wang, 2023) is proposed to learn critical-node scores without dismantling labels. It encodes diffusion competence and structural-role significance and fuses them through unlabeled, graph-specific adaptation.

Reinforcement learning.

- **FINDER** (Fan et al., 2020b) is proposed to learn an adaptive dismantling policy through deep reinforcement learning. It combines structure2vec representations with an n-step Q-function trained on synthetic graphs and applies the learned policy sequentially to unseen networks.
- **MIND** (Tian et al., 2026) is proposed to optimize sequential network dismantling through a discrete soft actor–critic framework. It couples graph message passing with an adaptive policy trained against accumulated LCC behavior.

Generative search and proxy critical-node identification.

- **LLM4CN** (Mao et al., 2024) is proposed to search for interpretable node-scoring programs through language-model-guided crossover and mutation. Candidate programs are evaluated by dismantling fitness, and the selected program is then applied to generate node scores. Its provenance and snapshot completeness remain pending verification.

- **CNT** (Chen et al., 2024) is proposed to identify influential nodes by learning from SIR-based supervision. It uses Transformer self-attention to model global node interactions and outputs a one-pass influence ranking, making it a proxy critical-node method rather than a direct LCC-dismantling optimizer.
- **AGNN** (Xiong et al., 2024) is proposed to identify vital nodes by combining an autoencoder, graph message passing, and ranking supervision derived from SIR simulations. It produces a one-pass influence ranking and is therefore evaluated as a proxy critical-node method. The reviewed snapshot has incomplete data and checkpoint dependencies.

B.3 Methods Reviewed but Not Evaluated

Table 9 records additional methods considered during benchmark construction. These methods are kept separate from the evaluated inventory so that literature coverage is not conflated with executable benchmark coverage.

Table 9: Additional methods covered by the literature survey but not evaluated in this study. "No verified executable implementation" means that the recorded review did not identify a runnable implementation of the proposed method; it is not a claim that no implementation exists.

Methods	Role/category	Evaluated?	Reason not evaluated
Leverage (Joyce et al., 2010), ClusterRank (Chen et al., 2013), adjacency entropy (Xu et al., 2020), acquaintance (Cohen et al., 2003)	non-learning heuristics	No	outside the executable suite
Embedding-aided dismantling (Osat et al., 2023), AMH (Wu et al., 2025)	non-learning optimization	No	not integrated into the executable suite
GA–GCN (Deng et al., 2025), GLND (Han et al., 2025)	supervised direct	No	no verified executable implementation
NEES (Liu & Wang, 2022), GASC (Hong et al., 2025), CLMA (Ma et al., 2024), CPDS (Shi et al., 2026), SPR (Zhou et al., 2026)	self/unsupervised direct	No	no verified executable implementation
CoreGQN (Fan et al., 2020a), SmartCore (Pu et al., 2024), PeerRemove (Xing et al., 2023), GRLND (Qu et al., 2025), MHRL (Wu et al., 2025)	reinforcement direct	No	no verified executable implementation
GFD (Fu et al., 2026)	generative direct	No	no verified executable implementation
DeepELE (Chen & Fan, 2023), HONNMA (Zhao et al., 2023), STGCN (Huang et al., 2026)	proxy critical-node	No	no verified executable implementation
ILGR (Munikoti et al., 2022)	proxy critical-node	No	public implementation identified but not integrated

C Further Discussion

C.1 Implications for Learning-based Network Dismantling

The comparison does not indicate that learning is unsuitable for network dismantling. Instead, it shows that current learning-based methods have not yet established a consistent advantage over mature non-learning approaches. Several learning-based methods obtain competitive results on the networks they complete, but this advantage often weakens when completion coverage and computational cost are considered. FINDER illustrates a different trade-off: it completes all 185 networks, but does not achieve the best dismantling quality. These observations suggest that effectiveness, applicability, and efficiency should be considered together when developing learning-based dismantling methods.

The differences among evaluation metrics also have direct implications for model design. A method trained to minimize the removal set required to reach a prescribed fragmentation level is not necessarily optimized to suppress connectivity throughout the removal trajectory. Likewise, a sharp structural transition does not by itself imply a better dismantling solution. Future learning-based methods should therefore state the intended intervention objective explicitly and evaluate whether the training objective is consistent with that goal. Multi-objective formulations may be useful when both the final intervention cost and the intermediate connectivity of the network are important.

Network scale presents another important challenge. Strong performance on small and medium-sized networks does not guarantee that a method can be applied to larger networks, particularly when inference requires repeated graph processing, graph-specific adaptation, or additional search. Future research should report completion coverage across network sizes and examine which components of

the method limit scalability. Improving coverage should not be treated only as an implementation issue, because a method that cannot return a valid solution has limited value in practical dismantling tasks.

C.2 Scope of the Empirical Evidence

NetBreaker focuses on static node removal from undirected, unweighted simple graphs. Directed, weighted, temporal, and multilayer networks are outside the present scope. The benchmark also evaluates naturally occurring differences among the collected networks rather than performing controlled interventions on individual graph properties. The domain- and scale-specific results should therefore be interpreted as empirical comparisons within the assembled dataset collection, not as causal claims about how a particular graph property determines method performance.

The benchmark evaluates complete and reproducible methods in the forms made available by their authors. The included learning-based methods differ in training data, released checkpoints, optimization procedures, and post-processing strategies. We retain these differences because they are part of the methods that can currently be reproduced and used. The results consequently compare deployable method pipelines rather than neural architectures retrained under an identical optimization budget. A controlled comparison of learning architectures would answer a related but different research question.

C.3 Limitations and Benchmark Extensions

The reported computational costs are observations under the execution protocol used in this study. The evaluated methods rely on heterogeneous CPU and GPU implementations, and their recorded running times may include different combinations of inference, graph-specific adaptation, search, and post-processing. These results support practical comparisons under the stated protocol, but should not be interpreted as hardware-independent measures of algorithmic complexity. Peak host-memory measurements also do not capture all device-memory consumption. Runtime and memory statistics over successful runs must therefore be interpreted together with completion coverage.

Several extensions would broaden the scope of the benchmark. Controlled profiling across hardware environments would provide a more detailed account of computational cost. Retraining selected learning-based methods under a common data and optimization budget could separate architectural capability from differences in existing implementations. The dataset collection could also be extended to directed, weighted, temporal, and multilayer networks, while controlled synthetic networks could be used to study the effect of individual structural properties. These extensions complement the present benchmark and provide directions for examining aspects of network dismantling that are not covered by the current setting.

D Additional Results

D.1 Detailed Domain- and Scale-specific Results

Table 10 complements the method-level ranges in Figure 3 with fixed method-family representatives selected once from the overall target-based ranking.

D.2 Complete Metric-dependent Rank Profiles

Table 11 summarizes the five highest-ranked methods under the target criterion and two trajectory summaries. Figure 5 extends this comparison to all methods and all seven evaluation views. The target and trajectory criteria measure dismantling quality, whereas the transition diagnostics characterize when and how the residual network fragments.

D.3 Trajectory-based Quality–Cost Trade-offs

Figure 6 complements the target-based cost analysis in Figure 4 by evaluating the connectivity retained throughout the removal trajectory.

Table 10: Sensitivity of fixed method-family representatives to network domain and scale. For each family, the representative is selected once by its overall $\rho_{0.10}$ failure-aware rank and then held fixed across all conditions. Entries are normalized failure-aware ranks (lower is better). Gap columns quantify condition sensitivity. Bold marks the best representative in each condition.

	Network domain									Number of nodes				
Method family (fixed representative)	Bio/Eco	Econ/Fin	Info/Media	Science	Security	Social	Tech/Comm	Transport	Δ_{domain}	$<$ 1K	1K–10K	10K–100K	$\geq$ 100K	Δ_{scale}
Random (Random)	0.959	0.891	0.865	0.849	0.946	0.786	0.873	0.897	0.173	0.966	0.965	0.825	0.434	0.531
Centrality (Adaptive Betweenness)	0.146	0.230	0.297	0.606	0.281	0.398	0.353	**0.135**	0.471	**0.153**	**0.092**	0.675	0.734	0.643
Approx./Optimization (BPD)	**0.121**	**0.086**	**0.099**	**0.075**	**0.101**	**0.146**	**0.102**	0.157	0.082	0.172	0.104	**0.064**	**0.147**	0.107
Supervised (CoreGDM)	0.223	0.299	0.279	0.265	0.234	0.293	0.209	0.217	0.089	0.213	0.224	0.190	0.535	0.345
Self-/Unsupervised (Stable-AGDM+R)	0.207	0.368	0.263	0.211	0.377	0.286	0.192	0.240	0.185	0.243	0.203	0.162	0.543	0.381
Reinforcement (FINDER)	0.494	0.313	0.441	0.501	0.335	0.381	0.454	0.489	0.188	0.457	0.489	0.440	0.242	0.247
Generative Search (LLM4CN)	0.704	0.701	0.747	0.828	0.670	0.776	0.895	0.745	0.225	0.713	0.737	0.936	0.734	0.223
Proxy Critical-node (AGNN)	0.860	0.833	0.841	0.780	0.837	0.833	0.841	0.858	0.080	0.868	0.871	0.802	0.734	0.137

Table 11: Top-5 methods under the target-based criterion and the two trajectory-based criteria. Entries report failure-aware mean ranks over 185 networks; lower is better.

	Target-based		Trajectory-based			
Rank	$\rho_{0.10}$		LCC-AUC		PairConn	
	Method	Mean rank ↓	Method	Mean rank ↓	Method	Mean rank ↓
1	**BPD**	**4.45**	**Explosive Immunization**	**3.37**	**Static Betweenness**	**8.46**
2	CoreHD	4.83	Collective Influence	6.45	PageRank	8.59
3	Min-Sum	4.97	BPD	9.33	DCRS	9.01
4	Explosive Immunization	7.41	MIND	9.70	Collective Influence	9.34
5	Stable-AGDM+R	8.26	Adaptive Betweenness	9.84	Random	9.74

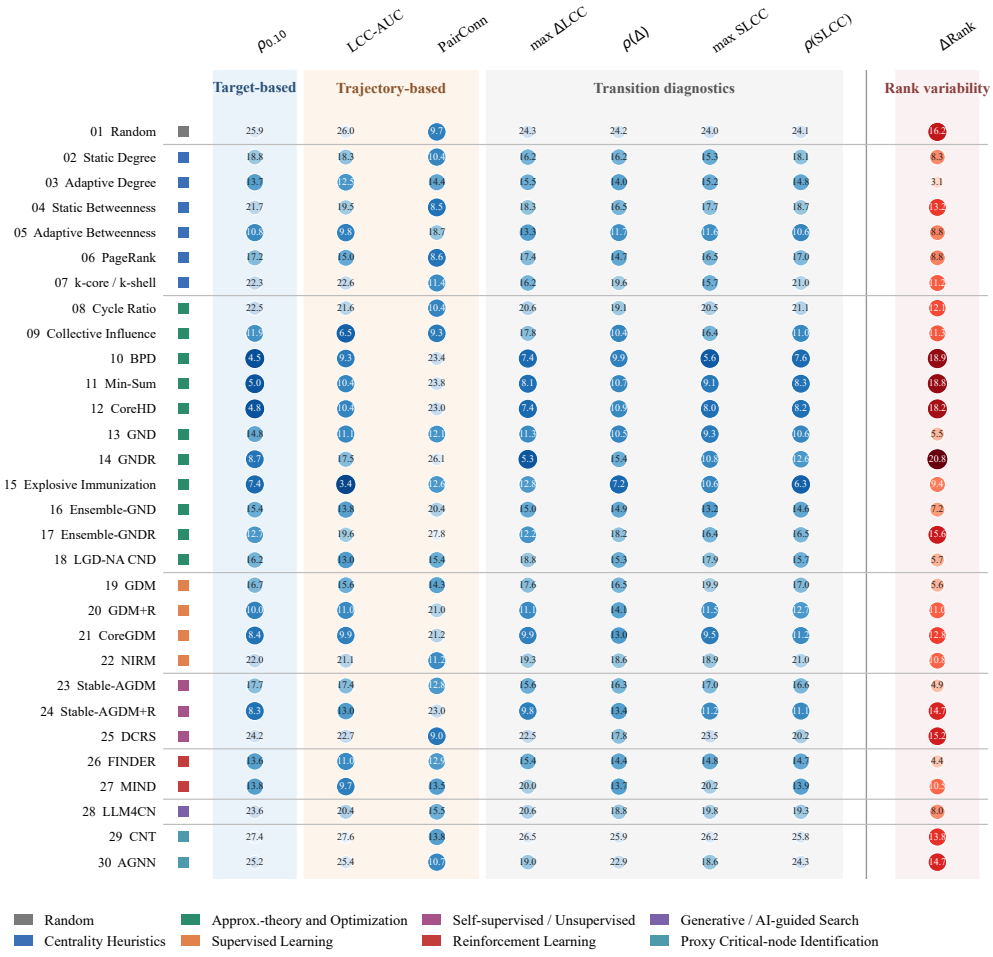


Figure 5: Complete failure-aware rank profiles across evaluation perspectives. Darker and larger blue circles indicate better mean ranks; the red column reports the range between each method's best and worst rank. Transition magnitudes characterize the fragmentation mechanism rather than an independent overall objective.

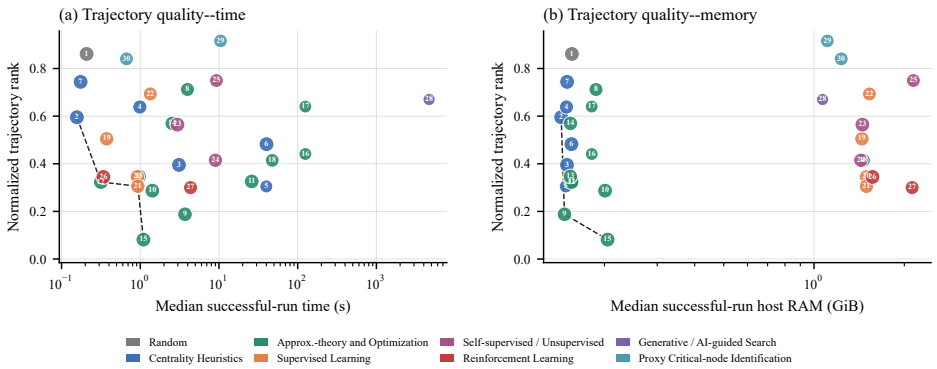


Figure 6: Observed trajectory-quality cost trade-offs over 185 networks. Panels (a) and (b) compare the failure-aware LCC-AUC rank with median successful-run time and peak host memory, respectively. Lower and further left is better; dashed lines mark non-dominated observations, marker area encodes completion coverage, and method IDs refer to Table 2.