A Practical Guide to Method Selection for Feature-Poor Bipartite Graphs: From Traditional Baselines to Graph Neural Networks

Pavel Prochazka[1,*], Daniil Buchko[1]

[1]Cisco Systems, Inc., Prague, Czech Republic

Abstract

Bipartite graphs naturally arise in structured data applications such as recommendation systems and knowledge graphs. While Graph Neural Networks (GNNs) excel when rich node features are available, many real-world scenarios lack such features, leaving only structural information. This raises a fundamental practical question: How should practitioners approach method selection for learning on feature-poor bipartite graphs? We develop a systematic framework grounded in five empirical hypotheses about method effectiveness, validated through comprehensive comparison of 16 methods across six real-world datasets. Our results reveal that method effectiveness is highly dataset-dependent, with no single approach dominating all scenarios. Traditional methods like Label Propagation provide excellent starting points due to their computational efficiency (2–3 orders of magnitude faster), while well-established GNNs like HGCN offer competitive performance when additional resources are available. Our framework provides systematic guidance through four tiers of increasing complexity, with clear decision points based on confident prediction assessment and resource allocation principles that address real–world deployment constraints.

Keywords

Practical Guidelines, Bipartite Graphs, Graph Representation Learning, Node Classification, Empirical Analysis

1. Introduction

Graph Neural Networks (GNNs) have demonstrated remarkable success for learning on graph-structured data when rich node features are available. However, many real-world bipartite graph scenarios—including recommendation systems, citation networks, and knowledge graphs—naturally lack meaningful node features, leaving only structural information for learning. This characteristic fundamentally challenges the conventional assumption that GNNs consistently outperform traditional graph learning methods, since their primary advantage of joint structure-feature learning is diminished. Moreover, GNNs introduce practical limitations that become pronounced in feature-poor scenarios: extensive computational requirements (2–3 orders of magnitude higher than traditional methods), complex hyperparameter optimization, and implementation challenges that may not be justified when their core advantage is reduced. This raises the fundamental question: *How should practitioners systematically approach method selection for learning on feature-poor bipartite graphs?*

We address this challenge by developing a systematic framework grounded in five empirical hypotheses about method effectiveness in feature-poor scenarios. Our comprehensive evaluation compares 16 methods across six real-world datasets, revealing that method effectiveness is highly dataset-dependent with no universal winner across scenarios. Traditional methods like Label Propagation provide excellent starting points due to their computational efficiency and reliable optimization, while established GNNs offer competitive performance when additional resources are justified. Our key contribution is a four-tier progression framework with clear decision points based on confident prediction assessment, enabling practitioners to balance performance requirements against computational constraints and implementation complexity in real-world deployment scenarios.

ITAT'25: Information Technologies – Applications and Theory, September 26–30, 2025, Telgárt, Slovakia

*Corresponding author.

paprocha@cisco.com (P. Prochazka); dabuchko@cisco.com (D. Buchko)

Paper Organization: We formalize the bipartite graph learning problem, then present our hypothesis-driven framework for systematic method selection. Related work provides tier-based method categorization and dataset context, followed by empirical evidence supporting our framework hypotheses and conclusions with future research directions.

2. Problem Formulation

2.1. Notation and Definitions

We consider bipartite graphs $G = (U, V, E)$ where U and V are disjoint node sets and $E \subseteq U \times V$ represents edges. The bipartite structure is characterized by the incidence matrix $H \in \{0,1\}^{|U|\times|V|}$, where $H_{ij} = 1$ if there exists an edge between $u_i \in U$ and $v_j \in V$. No additional node features are provided beyond structural information encoded in H.

Methods requiring unipartite graph input (such as GraphSAGE or label propagation) convert the bipartite graph to a unipartite representation using the adjacency matrix

$$A = \begin{bmatrix} 0 & H \\ H^T & 0 \end{bmatrix}.$$

For node $u_i \in U$, we denote its neighborhood in V as $N(u_i) = \{v_j \in V : H_{ij} = 1\}$. Feature-based methods first compute dense node representations through techniques such as matrix factorization or spectral embedding of H. Computation of dense representation introduces additional computational overhead compared to methods. These methods operate directly on the sparse structural information and they use one-hot feature matrix (H) representation.

We examine two fundamental learning tasks on these feature-poor bipartite graphs:

Classification Task: Given a training set $T \subset U$ with known labels, we predict labels for nodes in U. Performance is measured using classification accuracy, which assesses average performance across all predictions.

Retrieval Task: For datasets with multi-class labels, we formulate retrieval as a series of one-versus-rest binary classification problems. For each class $c \in \{1, 2, \ldots, C\}$, given a training set $T_c^+ \subset U$ containing only positive examples for class c, methods must retrieve the top-k nodes most likely to belong to class c. Performance is measured using Precision@k ($P@k$), which assesses performance where models are most confident. The final retrieval performance is computed as the average over all classes: $P@k = \frac{1}{C}\sum_{c=1}^{C} P@k_c$.

2.2. Problem Statement

The central research question addressed in this paper is: *How should practitioners systematically select methods for learning on feature-poor bipartite graphs, where Graph Neural Networks lose their primary advantage of joint structure-feature learning?*

This question becomes particularly relevant because:

- GNNs typically require dense feature representations, making them computationally expensive compared to methods that operate directly on sparse structural information
- Traditional graph methods may offer competitive performance with significantly lower computational overhead and greater reliability
- The absence of rich node features challenges the conventional assumption that GNNs consistently outperform traditional approaches

Rather than focusing on developing new algorithms or providing theoretical analysis of method capabilities, this paper addresses the practical challenge of **efficient method selection** in resource-constrained settings. Our goal is to provide systematic, evidence-based guidance that helps practitioners navigate the trade-offs between computational cost, implementation complexity, and performance across different method categories.

Scope and Contribution: This work focuses specifically on providing practical method selection guidance through systematic empirical analysis. We aim to share actionable insights that inform decision-making when computational budgets, implementation timelines, and optimization resources are limited—constraints that are common in real-world deployment scenarios but often overlooked in academic evaluations focused primarily on performance maximization.

3. Method Selection Framework for Feature-Poor Bipartite Graphs

Based on our comprehensive empirical analysis across six bipartite graph datasets, we present a systematic framework for method selection that addresses the core challenge practitioners face: efficiently navigating the trade-offs between computational cost, implementation complexity, and performance when Graph Neural Networks lose their primary advantage of joint structure-feature learning.

3.1. Empirical Hypotheses

Our framework is grounded in five key hypotheses derived from systematic evaluation across diverse bipartite graph scenarios. These hypotheses form the theoretical foundation for our practical guidance, with confidence levels reflecting the strength of supporting evidence from our empirical study.

Hypothesis 1: *No Universal Winner*. Method effectiveness is fundamentally dataset-dependent in feature-poor bipartite graphs, with no single approach consistently dominating across all scenarios. This hypothesis provides the core motivation for systematic evaluation rather than assuming GNN superiority across all bipartite graph learning tasks.

Hypothesis 2: *Implementation Maturity Effects*. Implementation reliability and optimization stability often outweigh theoretical performance advantages, particularly for specialized methods lacking standard library support. Methods requiring extensive custom implementation frequently exhibit reliability issues that limit practical applicability.

Hypothesis 3: *Dense Feature Extraction Trade-off*. Dense feature extraction through matrix factorization or spectral embedding could provide performance benefits but at 2–3 orders of magnitude computational cost compared to methods operating directly on sparse structural information. This trade-off defines a natural complexity boundary in method selection.

Hypothesis 4: *Simple Method Failure Principle*. If simple methods fail to achieve reliable performance on confident predictions, complex methods rarely provide significant improvements. This principle guides the critical decision between method progression versus data quality improvement.

Hypothesis 5: *Optimization Budget Constraints*. With realistic hyperparameter optimization budgets, systematic progression through increasing complexity tiers typically provides better return on investment than directly applying the most sophisticated methods. Simpler methods benefit from more thorough optimization within fixed time constraints.

3.2. Four-Tier Method Selection Framework

Our framework organizes methods into four tiers of increasing implementation complexity and computational requirements. Each tier boundary is motivated by specific empirical hypotheses:

Tier 1 - *Structural Baselines*: Methods operating directly on bipartite graph structure using one-hot representations (CSP [1], Label Propagation, One-Hot Logistic Regression). Motivated by Hypothesis 3: These approaches avoid the computational overhead of dense feature extraction while providing rapid feasibility assessment with straightforward hyperparameter optimization.

Tier 2 - *Traditional ML Methods*: Traditional machine learning methods requiring dense feature extraction (Feature-based Logistic Regression, Random Forest, MLP) and simple single-layer neural networks (HGCN [2]). Motivated by Hypotheses 3 and 5: These methods accept the computational cost of feature preprocessing but maintain manageable configuration spaces that benefit from thorough optimization within realistic time constraints.

Tier 3 - *Established Graph Neural Networks*: Well-established GNN architectures with large configuration spaces (e.g. HGCN with multiple layers [2], HyperND [3], etc.). Motivated by Hypotheses 2 and 5: These methods offer sophisticated graph-aware learning through reliable implementations, but their extensive hyperparameter spaces require significant optimization resources that may limit practical advantages.

Tier 4 - *Specialized Experimental Methods*: Methods specifically designed for feature-poor scenarios but lacking mature implementations (VilLain [4], Subgradient methods [5]). Motivated by Hypothesis 2: These approaches may offer theoretical advantages but present implementation challenges and reliability concerns that limit practical deployment.

3.3. Systematic Selection Strategy

Our framework provides clear decision points based on the empirical hypotheses:

Phase 1 - *Initial Assessment*: Identify whether Tier 1 methods can handle "simple examples"—test instances that are very similar to training examples according to the given structural representation. If they demonstrate this fundamental capability, proceed to Phase 2 for systematic enhancement. If not, focus on data quality improvement by enhancing the representation (either through improved graph construction or additional features). This assessment can be operationalized through sufficient retrieval performance on confident predictions (start with $P@100 \leq 0.5$ for identification of insufficient performance, but adapt this threshold to your specific problem characteristics). This principle reflects Hypothesis 4: when simple methods fail on their most confident predictions, complex architectures rarely overcome fundamental task limitations.

Phase 2 - *Progressive Enhancement*: Based on Hypothesis 5: When Tier 1 methods demonstrate fundamental capability but insufficient performance for application requirements, progress systematically through higher tiers. Each tier transition should be justified by clear performance requirements that outweigh increased computational costs and optimization complexity.

Resource Allocation Guidance: Based on Hypotheses 2 and 5. Allocate optimization effort proportionally to method reliability - invest heavily in Tier 1-2 hyperparameter exploration, moderately in Tier 3 given large configuration spaces, and cautiously in Tier 4 given implementation uncertainties.

3.4. Framework Validation and Limitations

Our empirical evaluation demonstrates that this systematic approach effectively captures performance-complexity trade-offs across diverse bipartite graph scenarios. The framework successfully identifies when simple methods suffice (high confident prediction performance) versus when sophistication is warranted (clear progression benefits).

Confidence Assessment: Hypotheses 1-3 receive strong support across all evaluated datasets, while Hypotheses 4-5 show consistent but more limited evidence requiring broader validation. The framework principles appear robust, but specific thresholds and decision points may require adaptation for different domains or task characteristics.

Practical Impact: When the framework's assumptions hold, it enables efficient resource allocation and reduces the risk of over-engineering solutions for tasks where simple methods suffice. When assumptions fail, systematic evaluation through the tier structure still provides valuable comparative insights while avoiding premature commitment to complex approaches. If medium-confidence hypotheses (4-5) prove invalid in specific scenarios - for example, if complex methods succeed when simple methods fail on confident predictions, or if extensive optimization consistently improves complex method performance - practitioners should adapt resource allocation and decision thresholds accordingly while maintaining the systematic tier-based evaluation structure.

This framework transforms our empirical findings into actionable guidance while acknowledging the need for continued validation across broader domains and graph types. The systematic progression strategy addresses real-world constraints of limited computational budgets and optimization time while providing clear decision points for when to invest in increased method sophistication.

Table 1
Method categorization and key characteristics driving tier assignment

Method	Tier	Key Characteristic	Tier Rationale
CSP [1]	1	Parameter-free, direct structure	No optimization needed
Label Propagation		One-hot, small config space	Fast, reliable optimization
One-hot Logistic Regression		One-hot, standard ML	Well-understood hyperparams
One-hot Naive Bayes		One-hot, standard ML	Well-understood hyperparams
Feature-based Logistic Regression	2	Dense features, manageable config	Feature extraction + simple ML
Feature-based Random Forest		Dense features, moderate config	Moderate complexity increase
Feature-based Naive Bayes		Dense features, manageable config	Feature extraction + simple ML
MLP		Dense features, neural baseline	Structure-agnostic neural
HGCN (1-layer)		Simple GNN, small config space	Limited architectural complexity
HGCN (multi-layer)	3	Complex GNN, large config space	Extensive hyperparameter tuning
GraphSAGE		Established GNN, reliable impl.	Large config, reliable library
HCHA		Established GNN, reliable impl.	Large config, reliable library
UniGCNII		Established GNN, reliable impl.	Large config, reliable library
EHGNN		Advanced GNN, extensive config	Sophisticated but well-implemented
HyperND		Advanced GNN, extensive config	Sophisticated but well-implemented
AllSetTransformer		Advanced GNN, extensive config	Sophisticated but well-implemented
AllSetTransformerNormalized		Advanced GNN, extensive config	Sophisticated but well-implemented
AllDeepSets		Advanced GNN, extensive config	Sophisticated but well-implemented
AllDeepSetsNormalized		Advanced GNN, extensive config	Sophisticated but well-implemented
VilLain	4	Custom implementation required	Implementation challenges
Subgradient		No standard library support	Reliability concerns

4. Related Work

Having established our problem formulation and framework, we position our work within the broader landscape of bipartite graph learning methods and systematic approach selection for machine learning tasks.

4.1. Evolution of Bipartite Graph Learning Methods

The development of bipartite graph learning methods follows a natural progression of increasing sophistication that aligns with our tier-based framework, as summarized in Table 1.

Tier 1 - *Structural Methods:* Early collaborative filtering approaches [6] operated directly on bipartite user-item structures without requiring dense feature representations. Classical label propagation [7] extended semi-supervised learning to graph structures through iterative message passing on converted unipartite representations. CSP [1] recently introduced a parameter-free hypergraph variant that works natively on bipartite structures, implementable with simple SQL operations while avoiding hyperparameter optimization entirely.

Tier 2 - *Traditional ML Methods:* Traditional machine learning methods leveraging dense representations emerged with matrix factorization techniques [8] and spectral methods for collaborative filtering. These approaches accept the computational overhead of feature extraction while maintaining manageable configuration spaces. Simple neural baselines like MLPs provide structure-agnostic learning on dense features, while single-layer graph neural networks like HGCN [2] offer limited architectural complexity with small configuration spaces.

Tier 3 - *Established Graph Neural Networks:* The rise of Graph Neural Networks marked a paradigm shift in graph-based learning. Dai et al. [9] demonstrated that GNNs can be viewed as learned message passing algorithms, bridging classical approaches with modern deep learning such as GraphSAGE [10]. Extensions to hypergraph and bipartite scenarios include HGCN [2], UniGCNII [11], HyperND [3], EHGNN [12], HCHA [13], and AllSetTransformer [14]. While these methods demonstrate sophisticated

architectural designs, they require extensive hyperparameter tuning and significant computational resources.

Tier 4 - *Specialized Experimental Methods:* Recent approaches like VilLain [4] and subgradient hypergraph classifiers [5] specifically target feature-poor scenarios through novel optimization procedures. However, these methods lack mature implementations in standard libraries, presenting deployment challenges and reliability concerns that limit practical applicability.

4.2. Method Selection and Systematic Evaluation Approaches

Our work contributes to the broader challenge of systematic method selection in machine learning, particularly for graph-structured data where method choice significantly impacts both performance and computational requirements.

AutoML and Method Selection: The automated machine learning community has developed systematic approaches for algorithm selection [15], typically focusing on tabular data with extensive feature engineering. However, these approaches rarely address the unique challenges of graph-structured data where structural representation choices fundamentally affect method applicability.

Graph Learning Method Comparisons: Prior comparative studies in graph learning have primarily focused on rich-feature scenarios where GNNs demonstrate clear advantages [16]. Systematic evaluation of traditional versus modern methods specifically for feature-poor bipartite graphs remains underexplored, with most studies assuming GNN superiority rather than providing decision frameworks for method selection.

Practical Guidelines in Machine Learning: The machine learning community increasingly recognizes the need for practical guidance that considers implementation constraints alongside theoretical performance [17]. Our tier-based framework contributes to this direction by providing systematic decision points based on computational budgets, optimization reliability, and implementation maturity rather than focusing solely on performance maximization.

Empirical Methodology for Graph Learning: Recent work has begun questioning universal GNN superiority, particularly when their primary advantage of joint structure-feature learning is diminished [18]. Our systematic empirical approach builds on this trend by providing evidence-based method selection guidance specifically for feature-poor bipartite scenarios.

4.3. Bipartite Graph Datasets and Evaluation Domains

Our empirical evaluation spans six real-world bipartite graph datasets that naturally lack rich node features, requiring methods to learn exclusively from structural information. These datasets represent two primary domains where feature-poor bipartite graphs commonly arise in practice.

Academic Citation Networks: Five datasets derive from scholarly publication databases, each offering different structural perspectives on academic relationships. Cora provides dual bipartite representations connecting papers to authors (Cora-CA) and papers to citation contexts (Cora-CC), demonstrating how the same underlying data can yield fundamentally different bipartite structures [19]. CiteSeer and PubMed follow citation-based approaches in computer science and biomedical domains respectively [20, 21], while DBLP represents author-paper collaborations across computer science publications [22]. These datasets exhibit substantial structural diversity, with node counts ranging from 2,708 to 41,302 and varying degrees of connectivity.

E-commerce Domain: The Walmart dataset represents product co-purchase relationships, where bipartite edges indicate items frequently bought together. This domain introduces different structural properties compared to citation networks, with higher edge density (460,629 edges across 88,860 products) and distinct degree distributions that stress-test method generalization across application domains.

Dataset Characteristics and Challenges: Our collection exhibits natural diversity in structural complexity, from sparse citation networks to dense co-purchase graphs. The presence of isolated nodes varies dramatically (from 0 in DBLP to 15,877 in PubMed), providing systematic evaluation of how

methods handle nodes with no structural information. This diversity enables assessment of method robustness across different bipartite graph characteristics while maintaining the common constraint of feature-poor scenarios. Complete dataset statistics and experimental protocols are detailed in the appendix.

Our contribution addresses the gap between theoretical method development and practical deployment guidance by providing systematic, tier-based evaluation that considers the full spectrum of implementation and optimization constraints faced by practitioners working with feature-poor bipartite graphs.

5. Empirical Evidence for Framework Hypotheses

Table 2
Performance Summary: Method categories achieving top and competitive performance across tasks. #Top Perf. indicates datasets where any Tier method achieves best performance. Within 5% best indicates datasets where any Tier method performs within 5% of the optimal result (excluding methods already counted as top performers).

Method Category	Classification		Retrieval	
	# Top Perf.	# within 5% of best	# Top Perf.	# within 5% of best
Tier 1	1	1	1	1
Tier 2	2	3	2	4
Tier 3	2	5	3	4
Tier 4	1	1	0	0

Table 3
Retrieval performance (P@100) measuring confident predictions on bipartite graphs $G = (U, V, E)$. Bold indicates best performance per dataset; underlined indicates performance within 5% of optimum. Methods above midrule are neural/GNN approaches; below are traditional methods.

Tier	Method	Cora-CA	Cora-CC	CiteSeer-CC	PubMed-CC	DBLP-CA	Walmart
	Random	0.144	0.134	0.150	0.350	0.168	0.084
1	CSP	0.721	0.446	0.387	0.747	0.867	0.134
	Label Propagation	0.731	0.686	**0.503**	0.730	0.878	0.202
	One-hot Naive Bayes	0.714	0.443	0.382	0.767	0.950	0.204
	One-hot Logistic Regression	0.726	0.447	0.405	0.783	0.963	0.286
2	Feature-based Logistic Regression	0.710	0.681	0.473	0.810	0.825	0.109
	Feature-based Random Forest	0.709	0.684	0.412	0.807	0.960	**0.350**
	HGCN_one_layer	0.760	**0.823**	0.433	0.857	0.970	0.113
	MLP	0.767	0.789	0.427	0.883	0.740	0.123
3	HGCN_multiple_layers	0.760	0.759	0.480	**0.913**	0.727	0.105
	EHGNN	**0.770**	0.693	0.380	0.823	0.182	0.0936
	HyperND	0.756	0.763	0.485	0.740	**0.980**	0.0864
	GraphSAGE	0.744	0.770	0.462	0.893	0.502	0.125
	HCHA	0.720	0.660	0.435	0.450	0.167	0.0855
	AllDeepSetsNormalized	0.744	0.727	0.363	0.620	0.775	0.131
	AllDeepSets	0.734	0.659	0.460	0.767	0.938	0.122
	AllSetTransformerNormalized	0.577	0.727	0.427	0.907	0.375	0.127
	AllSetTransformer	0.673	0.763	0.347	0.843	0.912	0.146
	UniGCNII	0.619	0.461	0.363	0.543	0.225	0.0918
4	VilLain	0.274	0.511	0.417	0.690	N/A	N/A
	Subgradient	0.116	0.109	0.150	N/A	N/A	N/A

Table 4
Classification accuracy measuring average performance on bipartite graphs $G = (U, V, E)$. Bold indicates best performance per dataset; underlined indicates performance within 5% of optimum. Methods above midrule are neural/GNN approaches; below are traditional methods.

Tier	Method	Cora-CA	Cora-CC	CiteSeer-CC	PubMed-CC	DBLP-CA	Walmart
	Random	0.192	0.190	0.185	0.364	0.166	0.113
1	CSP	0.501	0.238	0.229	0.407	0.501	0.313
	Label Propagation	**0.551**	0.358	0.267	0.433	0.641	0.366
	One-hot Naive Bayes	0.468	0.215	0.247	0.264	0.407	0.356
	One-hot Logistic Regression	0.479	0.241	0.258	0.264	0.412	0.359
2	Feature-based Logistic Regression	0.446	0.425	**0.319**	0.458	0.681	0.359
	Feature-based Random Forest	0.425	0.430	0.277	0.400	0.680	0.371
	HGCN_one_layer	0.496	0.496	0.283	**0.465**	0.596	0.351
	MLP	0.477	0.399	0.283	0.455	0.610	0.350
3	HGCN_multiple_layers	0.465	**0.542**	0.249	0.465	0.652	0.360
	EHGNN	0.449	0.298	0.0712	0.249	0.221	0.359
	HyperND	0.515	0.509	0.245	0.453	0.666	0.365
	GraphSAGE	0.442	0.302	0.273	0.453	0.496	0.352
	HCHA	0.507	0.472	0.189	0.443	0.647	**0.374**
	AllDeepSetsNormalized	0.451	0.389	0.253	0.382	0.527	0.362
	AllDeepSets	0.347	0.370	0.221	0.440	0.387	0.371
	AllSetTransformerNormalized	0.484	0.517	0.307	0.447	0.187	0.372
	AllSetTransformer	0.283	0.291	0.282	0.442	0.630	0.295
	UniGCNII	0.389	0.368	0.258	0.406	0.549	0.227
4	VilLain	0.355	0.535	0.260	0.393	**0.696**	N/A
	Subgradient	0.155	0.311	0.214	0.208	N/A	N/A

Our experimental evaluation across six bipartite graph datasets provides systematic evidence for the five hypotheses underlying our method selection framework. The results demonstrate clear patterns that validate our tier-based approach while revealing the practical trade-offs that guide effective method selection.

5.1. Hypothesis 1: No Universal Winner

Tables 3 and 4 provide compelling evidence that method effectiveness is fundamentally dataset-dependent in feature-poor bipartite graphs. No single method achieves top performance across all datasets for either task. Traditional methods achieve best performance on multiple datasets: Label Propagation excels on Cora-CA classification, while Feature-based Random Forest leads on Walmart retrieval. Conversely, GNN methods dominate other scenarios, with HGCN variants achieving top performance on PubMed and Cora-CC tasks respectively.

Table 2 quantifies this diversity, showing that each tier contributes top performers across both tasks. This dataset-dependent performance validates our systematic evaluation approach rather than assuming universal GNN superiority. The variation in optimal methods across structurally different datasets (sparse citation networks vs. dense co-purchase graphs) demonstrates that bipartite graph characteristics fundamentally influence method effectiveness.

5.2. Hypothesis 2: Implementation Maturity Effects

The reliability challenges of Tier 4 methods provide strong evidence for implementation maturity effects. VilLain and Subgradient methods failed to execute on multiple datasets despite theoretical sophistication, with ”N/A” entries in Tables 3 and 4 indicating memory constraints that prevented execution. When

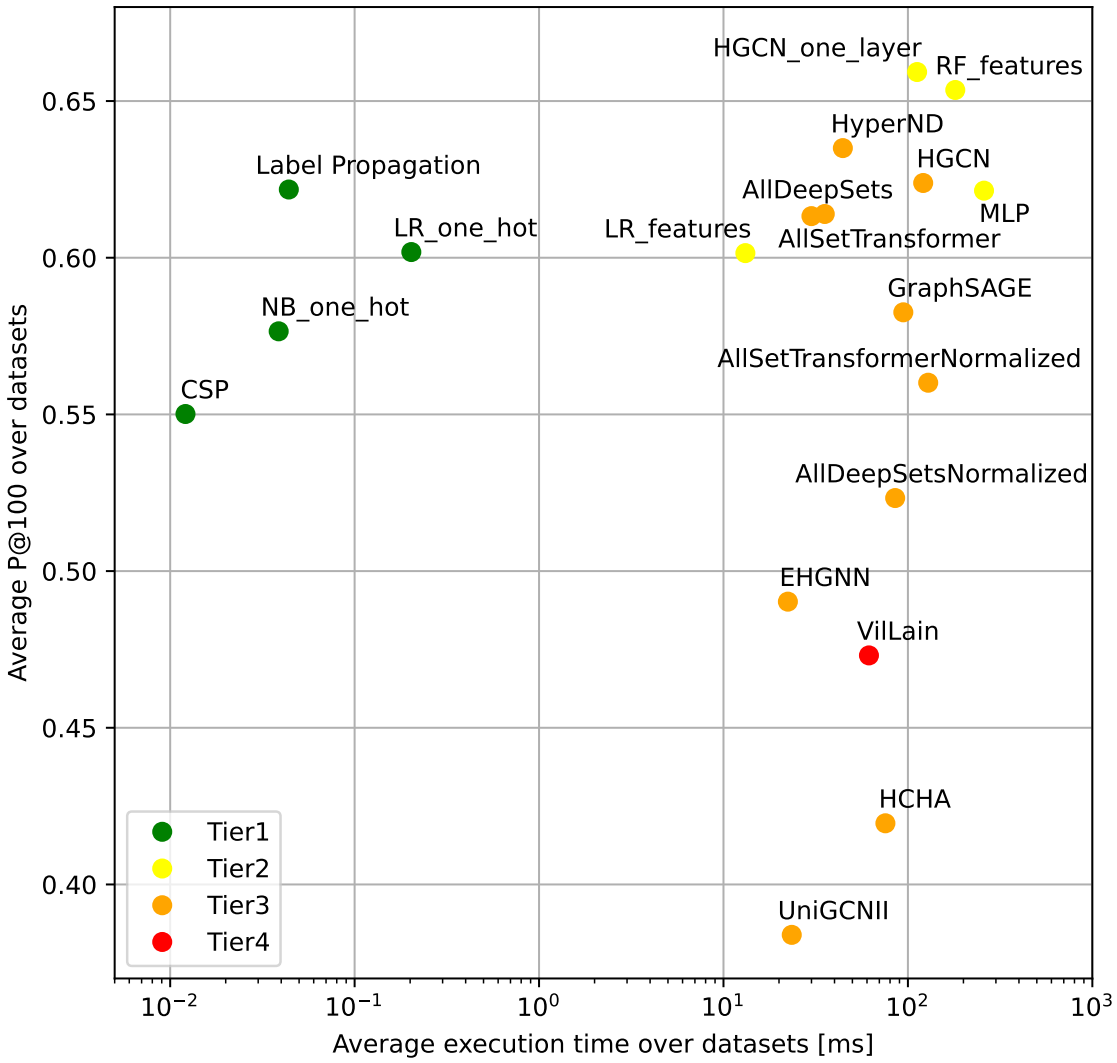

Figure 1: Retrieval performance vs. computational complexity averaged across all datasets. Methods operating on one-hot features (bottom-left cluster) achieve 2–3 orders of magnitude faster execution than dense representation methods while maintaining competitive performance.

these methods did execute, they often underperformed simpler alternatives, suggesting that resource requirements compromise their practical value.

In contrast, well-established methods in Tiers 1-3 demonstrated consistent execution across all datasets with reliable performance patterns. This reliability difference validates our framework's emphasis on implementation maturity alongside theoretical performance, supporting the progressive tier structure that prioritizes proven methods before exploring experimental approaches.

Even if Tier 4 method can achieve a competetive performance – VilLain in classification task – the execution times one order of magnitude higher than other methods demonstrating downsite of their not-mature implementation for practical use.

5.3. Hypothesis 3: Dense Feature Extraction Trade-off

Figures 1 and 2 dramatically illustrate the computational cost of dense feature extraction, showing 2–3 orders of magnitude differences between Tier 1 methods (green) operating on one-hot representations and higher tiers requiring dense features. The bottom-left clusters in both figures represent Tier 1 methods achieving competitive performance with minimal computational overhead.

The performance-complexity trade-off varies by task and dataset. For retrieval tasks, several Tier 1

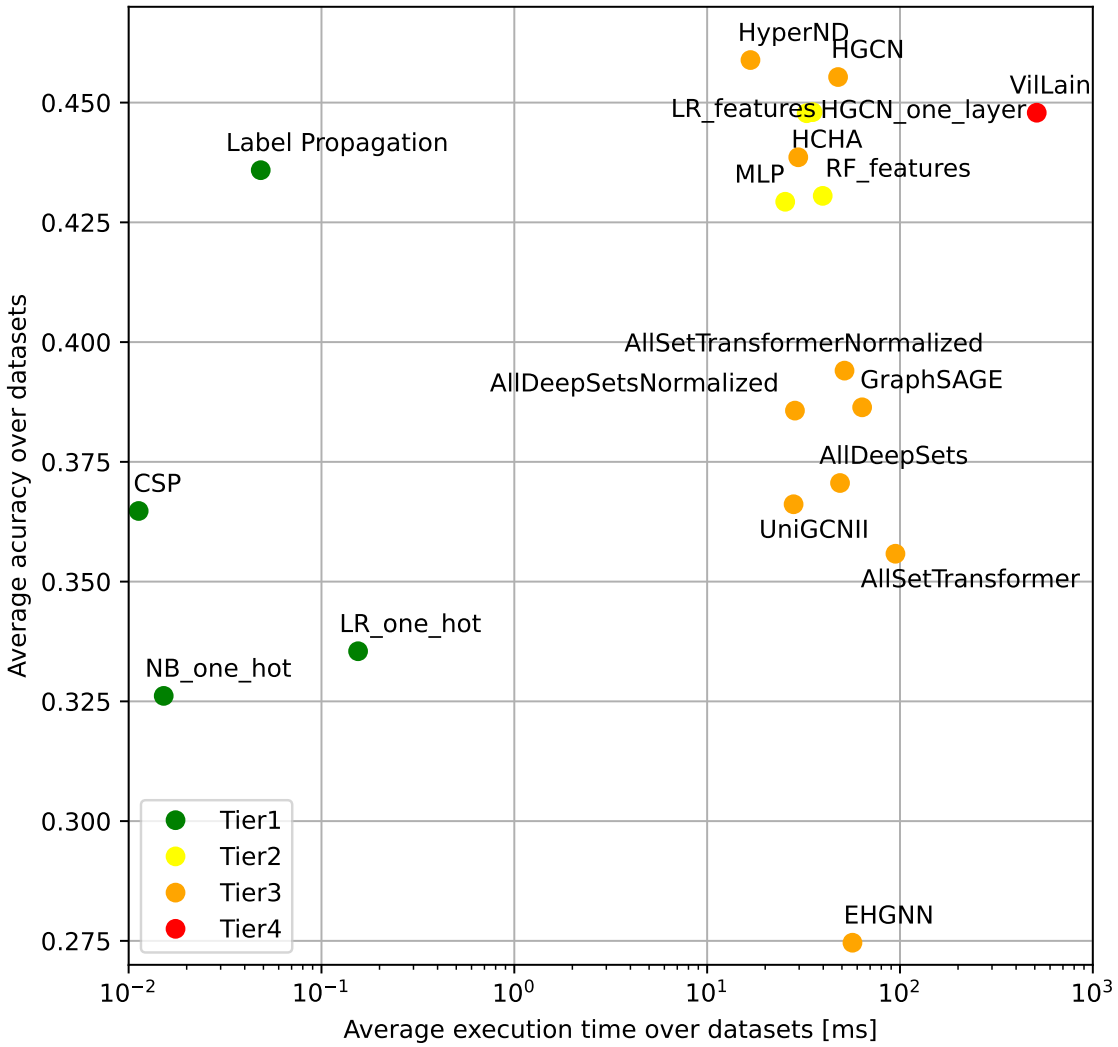

Figure 2: Classification performance vs. computational complexity averaged across all datasets. Dense representation methods (right side) show improved accuracy over one-hot approaches but at significant computational cost, with traditional methods often matching GNN performance.

methods achieve performance within 5% of the best while maintaining the fastest execution times. Classification tasks show more pronounced benefits from dense representations, but often at computational costs that may not justify modest performance improvements. This trade-off validates our tier boundary between structural methods (Tier 1) and dense representation approaches (Tiers 2-3).

5.4. Hypothesis 4: Simple Method Failure Principle

When simple methods cannot reliably identify the most obvious positive examples in a dataset, complex architectures rarely overcome these fundamental limitations. This principle provides a critical decision point: rather than immediately progressing to sophisticated methods when Tier 1 performance appears insufficient, practitioners should first assess whether the limitation stems from method inadequacy or fundamental task characteristics that no learning approach can easily overcome.

We operationalize this assessment through confident prediction performance, where $P@100$ proves more suitable than accuracy for evaluating performance on "simple examples"—instances where methods should excel if they can capture fundamental data patterns. Our empirical analysis reveals that higher-tier methods improve performance meaningfully only when Tier 1 methods achieve sufficient confident prediction performance. Based on our results, $P@100 \leq 0.5$ serves as a practical indicator for this

threshold, though this value may require adjustment for different domains.

This pattern is evident across our datasets: scenarios where even the strongest Tier 1 methods achieve limited confident prediction performance (CiteSeer-CC at 0.503, Walmart at 0.286-0.350) consistently show traditional methods matching or outperforming sophisticated GNNs. Conversely, datasets with higher $P@100$ baselines (Cora variants, PubMed, DBLP with values above 0.7) demonstrate clearer benefits from method progression, validating our Phase 1 assessment strategy.

This hypothesis connects to our broader framework by providing a systematic decision point for resource allocation. Rather than automatically progressing to higher tiers when Tier 1 performance appears insufficient, practitioners should first assess whether the limitation stems from method inadequacy or fundamental task characteristics that no learning approach can easily overcome.

Confidence Assessment: We evaluate this hypothesis from multiple perspectives. The specific threshold of 0.5 receives low-to-medium confidence, being based on only six datasets in our evaluation. However, the $P@k$ metric as an indicator of fundamental task failure appears quite reasonable for tasks with reasonably uniform label distributions, where confident predictions should reflect the model's ability to identify the most obvious positive examples. The underlying assumption that test examples range from easy to hard—while intuitively reasonable—lacks direct empirical confirmation in our study. This assumption should be carefully considered when applying the framework, as violations could explain cases where the framework's guidance proves ineffective. Future work should validate both the threshold values and the easy/hard example assumption across broader domains to strengthen this hypothesis.

5.5. Hypothesis 5: Optimization Budget Constraints

The consistent performance of Tier 1 and Tier 2 methods across both tasks provides evidence for optimization budget effects. These methods benefit from manageable hyperparameter spaces that enable thorough exploration within realistic time constraints. Table 2 shows Tier 2 methods achieving the highest number of competitive performances (within 5% of best), suggesting that moderate complexity with reliable optimization often outperforms sophisticated architectures with optimization uncertainty.

Our time-boxed optimization approach reveals that Tier 3 methods with extensive configuration spaces may not achieve their full potential within practical constraints. This limitation actually reinforces our framework's value: when simpler methods achieve competitive performance with reliable optimization, the additional complexity and optimization uncertainty of sophisticated architectures becomes a practical disadvantage rather than merely a theoretical concern.

5.6. Integration of Evidence Supporting Framework Design

The empirical evidence collectively validates our tier-based progression strategy. Tier 1 methods like Label Propagation and CSP provide excellent starting points with minimal computational overhead and reliable optimization. Tier 2 methods offer systematic enhancement through dense representations while maintaining manageable complexity. Tier 3 GNNs demonstrate sophisticated capabilities but with optimization challenges that limit their practical advantages. Tier 4 methods show implementation reliability issues that constrain their applicability.

The dataset-dependent performance patterns, combined with clear computational and optimization trade-offs, support our systematic evaluation approach over universal method preferences. The framework successfully identifies when simple methods suffice versus when sophistication is warranted, providing practitioners with evidence-based decision points that balance performance requirements against practical constraints.

6. Conclusions and Future Work

Our systematic framework for method selection in feature-poor bipartite graphs provides evidence-based guidance through comprehensive evaluation of 16 methods across six real-world datasets. We

established five empirical hypotheses about method effectiveness and validated a four-tier progression strategy that balances performance requirements against computational constraints and implementation complexity. Our results demonstrate that method effectiveness is fundamentally dataset-dependent, with traditional approaches like Label Propagation often achieving competitive performance at 2–3 orders of magnitude lower computational cost than Graph Neural Networks. The framework's key insight is the confident prediction assessment principle: when simple methods achieve poor performance on their most confident predictions (roughly $P@100 \leq 0.5$), complex architectures rarely provide significant improvements, suggesting data quality enhancement as more effective than method sophistication. This systematic approach enables practitioners to efficiently navigate the performance-complexity trade-off space while avoiding over-engineering solutions for tasks where simple methods suffice.

Limitations and Future Work: Our framework validation is based on six datasets primarily from academic citation networks, requiring broader domain validation to strengthen generalizability claims. The framework's core assumption—that test examples contain a range of difficulties from very simple to very hard—underlies our confident prediction assessment strategy and may not hold universally across all domains. Future work should focus on systematic validation across diverse domains to identify when this difficulty distribution assumption fails and how practitioners should adapt their approach accordingly. The main challenge for framework adaptation lies in identifying appropriate metrics for detecting "simple examples" and establishing domain-specific thresholds for confident prediction failure. For highly imbalanced tasks like threat detection, the $P@100 \leq 0.5$ threshold would likely need adjustment to much lower values, while other domains may require entirely different confidence metrics to assess fundamental task tractability. Additionally, extending our tier-based evaluation protocol to incorporate new architectures as they emerge will ensure the framework remains current and actionable for practitioners facing feature-poor bipartite graph learning challenges.

Declaration on Generative AI

The authors have not employed any Generative AI tools.

References

[1] P. Procházka, M. Dědič, L. Bajer, Convolutional signal propagation: A simple scalable algorithm for hypergraphs, arXiv preprint arXiv:2409.17628 (2024).

[2] S. Bai, F. Zhang, P. H. Torr, Hypergraph convolution and hypergraph attention, Pattern Recognition 110 (2021) 107637.

[3] F. Tudisco, A. R. Benson, K. Prokopchik, Nonlinear higher-order label spreading, in: Proceedings of the Web Conference 2021, 2021, pp. 2402–2413.

[4] G. Lee, S. Y. Lee, K. Shin, Villain: Self-supervised learning on homogeneous hypergraphs without features via virtual label propagation, in: Proceedings of the ACM Web Conference 2024, 2024, pp. 594–605.

[5] C. Zhang, S. Hu, Z. G. Tang, T. H. Chan, Re-revisiting learning on hypergraphs: confidence interval and subgradient method, in: International Conference on Machine Learning, PMLR, 2017, pp. 4026–4034.

[6] J. B. Schafer, D. Frankowski, J. Herlocker, S. Sen, Collaborative filtering recommender systems, in: The adaptive web: methods and strategies of web personalization, Springer, 2007, pp. 291–324.

[7] X. Zhuĺ, Z. Ghahramaniĺн, Learning from labeled and unlabeled data with label propagation, ProQuest number: information to all users (2002).

[8] Y. Koren, R. Bell, C. Volinsky, Matrix factorization techniques for recommender systems, Computer 42 (2009) 30–37.

[9] H. Dai, B. Dai, L. Song, Discriminative embeddings of latent variable models for structured data, in: International conference on machine learning, PMLR, 2016, pp. 2702–2711.

[10] W. Hamilton, Z. Ying, J. Leskovec, Inductive representation learning on large graphs, Advances in neural information processing systems 30 (2017).
[11] J. Huang, J. Yang, Unignn: a unified framework for graph and hypergraph neural networks, arXiv preprint arXiv:2105.00956 (2021).
[12] J. Jo, J. Baek, S. Lee, D. Kim, M. Kang, S. J. Hwang, Edge representation learning with hypergraphs, Advances in Neural Information Processing Systems 34 (2021) 7534–7546.
[13] S. Bai, F. Zhang, P. H. Torr, Hypergraph convolution and hypergraph attention, Pattern Recognition 110 (2021) 107637.
[14] E. Chien, C. Pan, J. Peng, O. Milenkovic, You are allset: A multiset function framework for hypergraph neural networks, arXiv preprint arXiv:2106.13264 (2021).
[15] X. He, K. Zhao, X. Chu, Automl: A survey of the state-of-the-art, Knowledge-Based Systems 212 (2021) 106622.
[16] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, S. Y. Philip, A comprehensive survey on graph neural networks, IEEE transactions on neural networks and learning systems 32 (2020) 4–24.
[17] D. Sculley, G. Holt, D. Golovin, E. Davydov, T. Phillips, D. Ebner, V. Chaudhary, M. Young, J.-F. Crespo, D. Dennison, Hidden technical debt in machine learning systems, Advances in neural information processing systems 28 (2015).
[18] J. You, Z. Ying, J. Leskovec, Design space for graph neural networks, Advances in Neural Information Processing Systems 33 (2020) 17009–17021.
[19] A. K. McCallum, K. Nigam, J. Rennie, K. Seymore, Automating the construction of internet portals with machine learning, Information Retrieval 3 (2000) 127–163.
[20] C. L. Giles, K. D. Bollacker, S. Lawrence, Citeseer: An automatic citation indexing system, in: Proceedings of the third ACM conference on Digital libraries, 1998, pp. 89–98.
[21] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, T. Eliassi-Rad, Collective classification in network data, AI magazine 29 (2008) 93–93.
[22] J. Tang, J. Zhang, L. Yao, J. Li, L. Zhang, Z. Su, Arnetminer: extraction and mining of academic social networks, in: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '08, Association for Computing Machinery, New York, NY, USA, 2008, p. 990–998. URL: https://doi.org/10.1145/1401890.1402008. doi:10.1145/1401890.1402008.
[23] P. Wang, S. Yang, Y. Liu, Z. Wang, P. Li, Equivariant hypergraph diffusion neural operators, arXiv preprint arXiv:2207.06680 (2022).
[24] Q. Huang, H. He, A. Singh, S.-N. Lim, A. R. Benson, Combining label propagation and simple models out-performs graph neural networks, arXiv preprint arXiv:2010.13993 (2020).

A. Implementation Details and Reproducibility

This appendix provides essential implementation information for reproducing the presented results. All methods were evaluated using identical experimental protocols with time-boxed hyperparameter optimization to ensure fair comparison.

A.1. Dataset Construction and Preprocessing

Bipartite Graph Construction: Each dataset constructs bipartite graphs $G = (U, V, E)$ where U represents primary entities (papers/products) and V represents secondary entities (authors/citations/co-purchases). Citation networks derive V from unique cited papers (CC variants) or individual authors (CA variants). Walmart constructs V from co-purchase relationships.

Feature Extraction: Matrix Factorization uses the implicit library with 50 iterations and confidence weighting. Spectral Embedding performs eigenvalue decomposition using scipy.sparse.linalg.eigsh on the normalized Laplacian. One-hot encoding uses incidence matrix rows directly.

Data Splitting: Natural dataset splits when available, otherwise stratified random splitting ensuring class balance. Training sets range from 53 nodes (PubMed) to 2,932 nodes (Walmart). Cross-validation

employed for limited training data scenarios.

A.2. Method Implementation Sources

Tier 1 Methods: CSP implemented via custom SQL operations. Label Propagation uses sklearn.semi_supervised with bipartite-to-unipartite conversion. One-hot methods use sklearn.linear_model and sklearn.naive_bayes directly on incidence matrix rows.

Tier 2 Methods: Feature-based methods combine embedding techniques with sklearn implementations. MLP uses implementation from [23] that adapted the implementation from [24]. HGCN single-layer implements a single HyperConv from the PyTorch Geometric library.

Tier 3 Methods: HGCN multi-layer uses PyTorch Geometric implementation with 1-5 hidden layers. GraphSAGE uses PyTorch Geometric implementation with previously applied bipartite-to-unipartite conversion. HyperND, EHGNN, HCHA, UniGCNII, and AllSet variants use implementations provided by [23], the ED-HNN method proposed by was not included in results, because of consistently bad performance with the current settings during the initial tests.

Tier 4 Methods: VilLain uses an implementation from its authors [4]. And subgradient method uses a custom implementation without standard library support based on the provided by its authors pseudocode algorithms [5], explaining execution failures on memory-constrained data

A.3. Hyperparameter Configuration Spaces

Traditional Methods: CSP is used in its parameter-free form [1]. Label Propagation searches alpha in the range [0, 1.0] in 0.1 increments, num_layers [1-5]. Logistic Regression varies C [0.1, 1, 10], max_iter [1000, 3000, 5000], solver [saga, liblinear, newton-cg]. Feature-based methods add embedding_dim [60, 120, 240, 480] and embedding type [mf, se], standing for matrix factorization and spectral embedding correspondingly.

Neural Methods: All neural methods utilize embeddings computed by non-negative matrix factorization or spectral embedding methods with dimensionality in the range [32-512]. MLP adds layers [1-5], hidden size [8-1024], dropout [0.0-0.8]. Multi-layer GNNs include layers [1-5], hidden [8-1024], dropout [0.0-0.8]. Advanced methods like EHGNN feature extensive configuration including classifier_hidden [8-256], edconv_type [EquivSet, JumpLink, MeanDeg], activation [Id, relu, prelu]. VilLain uses specialized parameters dim [128, 256, 512], steps [1-9], max_iter [10, 100, 1000].

Optimization Protocol: Time-boxed random grid search with equal computational budget per method. Neural methods use early stopping on validation performance. Traditional methods use optimal hyperparameters directly. Configuration spaces range from tens of combinations (traditional) to thousands (neural) – (see above), justifying the extensive optimization effort and explaining computational cost differences between tiers. We consider 1 hour time budget for each method/dataset for finding best hyperparameter setup (evaluated on validation set).

Computational Environment: All experiments conducted on identical hardware with consistent memory and time constraints. The hardware specificaiton is Amazon EC2 G4.xlarge instances with 4 vCPUs, 16 GB RAM, and NVIDIA T4 GPU. All timing measurements exclude data loading and preprocessing to focus on algorithm-specific computational costs. ”N/A” entries indicate memory limitations preventing method execution rather than implementation failures.

A.4. Results Presentation

Performance Analysis Methodology: Methods achieving performance within 5% of the dataset-specific optimum are considered competitive and highlighted in results tables. For each dataset and task, we compute the threshold as (best_performance $\times$ 0.95), with all methods meeting or exceeding this threshold receiving equal consideration for practical deployment decisions.