HETEROGENEOUS GRAPH PRIOR-DATA FITTED FOUNDATION MODEL

Anonymous authors
Paper under double-blind review

ABSTRACT

Heterogeneous graphs are ubiquitous in real-world applications, yet their schema heterogeneity, relational diversity, and complex attribute distributions make it highly challenging to build foundation models with cross-task generalization. Existing graph foundation models are mostly designed for a single setting, such as homogeneous graphs or relational data, or rely on large-scale real-world pretraining data, and therefore face data scarcity and limited task adaptability in heterogeneous scenarios. To address this, we propose *HGraphFM*, a foundation model framework for heterogeneous graph node-level tasks, which enables unified modeling of diverse heterogeneous graph tasks through synthetic-data-driven pretraining. In a nutshell, we first design a general heterogeneous graph prior, which, together with the relational-data prior, provides large-scale, diverse pretraining data for the foundation model. HGraphFM extends a single-table model by introducing graph attention layers, which in turn enhance in-context learning attention, enabling a transition from processing single-table data to handling heterogeneous data. It combines both in-context learning and structure learning capabilities, and can jointly model heterogeneous attributes and relations that may be subject to temporal constraints. Furthermore, we introduce a feature augmentation mechanism, termed *Deep Feature Synthesis Based on Graph Band-Pass Filtering (DFS-BP)*, which effectively alleviates insufficient context utilization and the need for manual feature selection. The mechanism, reminiscent of "lazy learning," produces synthesized features that directly match or even surpass the performance of some existing pretrained models in classification tasks, and provides substantial gains in model generalization. After a two-stage training procedure based on the two types of priors, experiments on multiple real-world heterogeneous graph datasets show that our method outperforms existing approaches, including homogeneous-graph and relational foundation models, in both few-shot and fine-tuned settings, validating the effectiveness of the proposed heterogeneous graph foundation model paradigm. Our code is available at `https://anonymous.4open.science/r/HGraphFM-0F0F`.

1 INTRODUCTION

In recent years, inspired by the success of foundation models in many domains Achiam et al. (2023); Devlin et al. (2019); Brown et al. (2020); Dosovitskiy (2020); Radford et al. (2021), researchers have begun extending the foundation model paradigm to graph data Mao et al. (2024); Liu et al. (2025a). However, compared with text and images, graphs are far more diverse in schema, semantics, and data sources, making it difficult to directly transfer existing foundation-model recipes. This challenge arises mainly from two aspects: (1) graphs, whether natural or artificial, span a wide range of domains; in particular, heterogeneous graphs with cross-type interactions, compositional semantic patterns, and potential temporal or causal dependencies are difficult to pretrain at scale within a unified semantic space as in natural language, leading to limited cross-schema transferability; and (2) high-quality real-world graph data for pretraining are scarce, often originating from proprietary enterprise systems or specialized vertical domains, and are difficult to access publicly due to privacy and annotation constraints.

Despite these difficulties, recent progress in tabular learning has inspired a new line of graph foundation models based on Prior-Data Fitted Networks (PFNs) Müller et al. (2022). By carefully de-

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.

signing synthetic prior distributions, these methods can generate an effectively unlimited stream of diverse and controllable training tasks, enabling models pretrained purely on synthetic data to rapidly adapt to unseen datasets and tasks. Such approaches have already shown considerable promise on node-level prediction tasks. However, most existing methods are designed for homogeneous graphs, typically assuming a unified node space and edge semantics, and thus cannot naturally handle heterogeneous graphs Yang et al. (2023); Zhang et al. (2019) with multiple entity types and complex interactions. In addition, while some foundation models Fey et al. (2024); Dwivedi et al. (2026); Fey et al. (2023) for relational data can process multi-table or multi-relational information, their modeling scope is usually restricted to relational-database-specific input formats and task settings, rather than general-purpose heterogeneous graphs.

To address these limitations, we propose HGraphFM, a foundation model framework for heterogeneous graph node-level tasks. Instead of relying on hard-to-obtain large-scale real-world heterogeneous graph data, HGraphFM is pretrained on synthetic corpora generated from a heterogeneous graph prior that covers a broad space of schemas and task distributions. On top of this, we design a unified architecture with both in-context learning and structure learning capabilities, enabling unified modeling and fast adaptation across diverse heterogeneous graph tasks. Specifically, our contributions are as follows:

- A general heterogeneous graph prior. We design a general synthetic prior capable of generating heterogeneous graph instances with multiple node or edge types, and complex causal dependencies. It can be combined with existing relational priors, allowing the generated data to cover a broader range of relational structures and task contexts.
- A unified foundation model for heterogeneous graph node-level tasks. Building on the tabular foundation model TabICL Qu et al. (2026), we introduce graph attention to develop HGraphFM, a unified model for subgraph-wise heterogeneous graph node-level tasks. It simultaneously supports structure learning and in-context learning.
- Deep Feature Synthesis Based on Graph Band-Pass Filtering. We propose a feature augmentation method — DFS-BP, implemented through graph-filtering-driven feature generation. It fully exploits the available context without requiring manual feature selection. Interestingly, the synthesized features constructed at inference time in a "lazy learning" manner can, to some extent, directly approach or even surpass the performance of some existing pretrained models in classification tasks.

Empirical results on a variety of real-world heterogeneous graph datasets, including both general heterogeneous graphs and relational data, show that after a two-stage training procedure with the general heterogeneous prior and the relational prior, HGraphFM outperforms existing methods in both few-shot and fine-tuning settings, surpassing homogeneous graph foundation models, relational foundation models, and several task-specific models. Overall, our work demonstrates the feasibility of building heterogeneous graph foundation models from synthetic priors, and provides a new direction for unified representation learning and generalization on heterogeneous graphs.

2 PRELIMINARIES

This section provides background descriptions of the research object in this paper—heterogeneous graphs, and the inference setting.

2.1 HETEROGENEOUS GRAPHS

In this paper, we categorize heterogeneous graphs into two classes: general heterogeneous graphs and relational data. We present their preliminary definitions separately.

General Heterogeneous Graph. A general heterogeneous graph is formally defined as

$$\mathcal{G} = (\mathcal{V}, \mathcal{E}, \phi, \psi),$$

where $\mathcal{V}$ and $\mathcal{E}$ denote the sets of nodes and edges, respectively, $\phi : \mathcal{V} \to \mathcal{A}$ is the node type mapping function, and $\psi : \mathcal{E} \to \mathcal{R}$ is the edge type mapping function. Here, $\mathcal{A}$ and $\mathcal{R}$ denote the sets of node types and edge types, respectively. A graph is considered heterogeneous if it satisfies

$$|\mathcal{A}| + |\mathcal{R}| > 2.$$

Relational Data. Relational data can be viewed as a special form of heterogeneous graph, which is specified by a schema and an instance. The schema is a collection of tables

$$\mathcal{T} = \{T_1, T_2, \ldots, T_N\}.$$

Each table $T \in \mathcal{T}$ has a column set $C(T) = K_{\text{pk}}(T) \cup K_{\text{fk}}(T) \cup A(T)$, $K_{\text{pk}}(T)$, $K_{\text{fk}}(T)$, and $A(T)$ denote the sets of primary key columns, foreign key columns, and attribute columns, respectively. Each foreign key column $k \in K_{\text{fk}}(T)$ references a parent table $\text{Ref}(k) \in \mathcal{T}$.

An **instance** assigns a set of rows $\mathcal{V}(T)$ to each table T. Primary key values uniquely identify rows within a table, and foreign key values must match the primary key values in the referenced table.

The **schema** can be represented as a directed graph $\mathcal{G}_S = (\mathcal{V}_S, \mathcal{E}_S)$, where $\mathcal{V}_S = \mathcal{T}$ and

$$(T_p, T_c) \in \mathcal{E}_S \quad \Longleftrightarrow \quad \exists k \in K_{\text{fk}}(T_c) \text{ s.t. } \text{Ref}(k) = T_p.$$

Similarly, the whole database can be represented as a relational entity graph

$$\mathcal{G}_{\text{in}} = (\mathcal{V}_{\text{in}}, E_{\text{in}}), \mathcal{V}_{\text{in}} = \bigcup_{T \in \mathcal{T}} \mathcal{V}(T),$$

and an edge $(u, v) \in \mathcal{E}_{\text{in}}$ exists if row v contains a foreign key value referencing the primary key of row u. Compared with general heterogeneous graphs, relational data does not contain homogeneous edges, and activity tables are often associated with timestamps, which require special consideration in our model framework.

2.2 Inference Setting

Our model also follows the PFNs fashion. During inference, the In-Context Learning (ICL) paradigm Brown et al. (2020) typically requires a certain number of context samples. Some existing works refer to such a setting as *zero-shot learning* as pre-training is conducted on synthetic data and is independent of the target domain. To avoid ambiguity, let the number of context examples be denoted by c. In this paper, we refer to this setting as the c-shot setting. Although the model performs inference without updating its parameters, it still relies on labeled context samples, and we therefore categorize it as a few-shot learning setting.

3 Methodology

This section introduces the proposed heterogeneous graph foundation model — HGraphFM, which mainly consists of three components: a general heterogeneous graph synthetic prior, the model architecture, and a novel feature augmentation mechanism — DFS-BP.

3.1 General Heterogeneous Graph Prior

In existing single-table foundation models, structural causal models (SCMs) Pearl (2009); Peters et al. (2017) are widely used to synthesize row-level features. Building on this, stochastic block models (SBMs) Karrer & Newman (2011); Girvan & Newman (2002) and their variants can be adopted to generate intra-table topological structures, which are then integrated with node attributes to form a homogeneous graph prior. Extending this line of thought, we further consider the heterogeneous setting and propose a heterogeneous graph prior, establishing a progressive blueprint from single-table data to homogeneous graphs and then to heterogeneous graphs.

The proposed heterogeneous graph synthetic prior consists of three steps (in Figure 1):

1. **Table partition.** Based on the homogeneous graph prior GNN-SCM, we randomly partition a single table $\mathcal{G} = \{\mathbf{X} \in \mathbb{R}^{N \times D}, \mathcal{E}\}$ with topological structure into multiple smaller tables with different numbers of rows, i.e. $\mathcal{G} = \{(\mathbf{X}_1 \in \mathbb{R}^{n_1 \times D}, ..., \mathbf{X}_t \in \mathbb{R}^{n_t \times D}), \mathcal{E}\}, \sum_{i=1}^{t} n_i = N$.
2. **Latent variable encapsulation.** For each small table t, we discard the features $\mathbf{X}_i$ and targets $\mathbf{y}_i$ from Step 1, and concatenate the layer-wise hidden representations of GNN-SCM to form $\mathbf{L}$.
3. **Feature and target regeneration.** We use SCMs to sample more multilayer perceptrons (MLP-SCMs, not shared across tables), and regenerate features and targets $[\mathbf{X}'_i \in \mathbb{R}^{n_i \times D'_i}, \mathbf{y}'_i]$ by sampling the new latent variable from $\text{MLP}_t(\mathbf{L}_t)$ and $\mathbf{L}_t$.

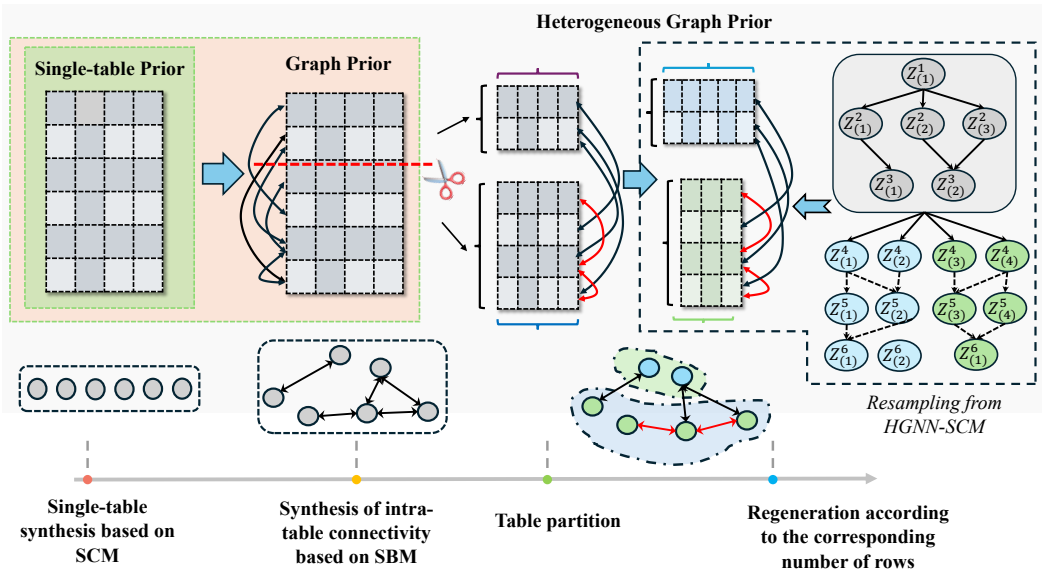


Figure 1: General Heterogeneous Graph Prior. The figure illustrates the progressive relationship among three types of priors: single-table, homogeneous graph, and heterogeneous graph.

Specifically, we build on the GNN-SCM prior of Eremeev et al. (2026) — a SCM over a sampled graph $\mathcal{G}$. Such a prior yields a *homogeneous* graph $(\mathbf{X}, \mathbf{y}, \mathcal{E})$. To synthesize *heterogeneous multi-table* databases, we retain what the readout discards: the full latent $\mathbf{L} = \text{z-score}\left([\mathbf{H}^{(1)}; \cdots; \mathbf{H}^{(L)}]\right) \in \mathbb{R}^{N \times W}$. We partition the N nodes into T tables with Dirichlet-drawn sizes, keeping all edges (which now cross table boundaries), and equip each table t with a *different, freshly sampled MLP-SCM* f_t. Pushing the table's rows $\mathbf{L}_t \in \mathbb{R}^{n_t \times W}$ of the shared latent through f_t yields per-layer activations $\mathbf{Z}_t^{(1)}, \ldots, \mathbf{Z}_t^{(K_t)}$ with fresh noise at each layer. As for table t, its features and label are then *re-sampled jointly* by drawing m_t feature columns and one label column at random from the concatenated activations of them, $(\mathbf{X}'_t, \mathbf{y}'_t) = \text{read}_t\left([\mathbf{H}_t; \mathbf{Z}_t^{(1)}; \cdots; \mathbf{Z}_t^{(K_t)}]\right)$. Co-reading makes $\mathbf{X}'_t$ and $\mathbf{y}'_t$ effects of one shared common cause, $\mathbf{X}'_t \Leftarrow \mathbf{L}_t \Rightarrow \mathbf{y}'_t$: the label stays predictable from the features, while the per-table mechanisms, index maps, and column counts m_t make the tables schema-heterogeneous yet coupled through $\mathbf{L}$ and $\mathcal{G}$—a random continuation of the GNN-SCM causal process rather than a re-encoding of $(\mathbf{X}, \mathbf{y})$. More details and analysis are provided in D.

Afterward, by combining the proposed prior with the existing synthetic prior for relational data, we obtain the complete synthetic corpus for model pre-training. The pre-training process is divided into two stages. In the first stage, the general heterogeneous graph prior is used. In the second stage, the model is further trained on the relational data prior. After these two stages of pre-training, the model acquires the ability to generalize across tasks on heterogeneous graphs.

3.2 Model Architecture

HGraphFM is built upon TabICL Qu et al. (2026), while introducing multi-head graph attention into the ICL attention layers. The design aims to achieve three objectives: (1) unified representation learning for heterogeneous features; (2) explicit graph structure encoding; and (3) enhanced in-context learning. Benefiting from the four learnable `[CLS]` token in the row-attention layers with RoPE Su et al. (2024), HGraphFM can naturally map inputs of different dimensions into a unified representation space. The graph attention layers are responsible for explicitly encoding structural relations. As shown in Figure 2.

Notably, the input of HGraphFM is no longer a single table (a single type of nodes/edges), but rather heterogeneous features together with a per-query sampled subgraph, with each subgraph serving as the computational unit. The sampled subgraph may contain context nodes

from the task table. Together with the randomly sampled context instances from the training set, these nodes jointly form the context set used by the ICL attention layers. Such a combination of random context and structural context further strengthens the ICL capability. In addition, subgraph sampling for relational data takes timestamps into account to avoid information leakage.

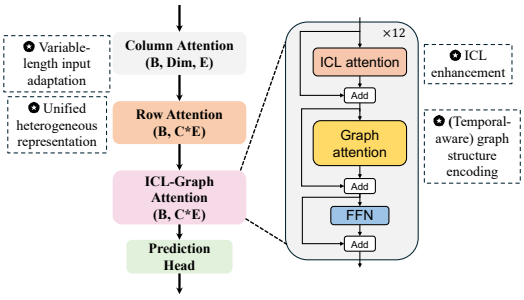


Figure 2: Model Architecture of HGraphFM.

3.3 Deep Feature Synthesis Based on Graph Band-Pass Filtering

This section introduces the proposed feature augmentation method, Deep Feature Synthesis based on Graph Band-Pass Filtering (DFS-BP). DFS-BP is applied for relational data at the inference stage: the original features are first augmented, and the augmented features are then fed into the model for forward prediction. DFS-BP consists of three steps:

1. **Constructing an intra-task-table subgraph.** We determine whether two rows in the task table are connected according to the heterogeneous relational links. If they are connected, an edge is established between them. The subgraph contains both context nodes and query nodes, with timestamps also taken into account to prevent future information leakage.
2. **Designing graph filters.** We initialize the parameters of graph band-pass filters $f^*_{\alpha,\beta}(\boldsymbol{L})$ and apply them to the original features to obtain filtered representations, $\boldsymbol{L}$ denotes the normalized Laplacian operator of the intra-task-table subgraph. The coefficients α and β control the frequency band: when $\alpha > \beta$, the filter tends to be low-pass; otherwise, it behaves as a high-pass filter. We initialize a series of parameters$\{[0, \alpha+\beta], [1, \alpha+\beta-1], \ldots, [\alpha+\beta, 0]\}$ to cover the entire frequency range Zhang et al. (2025a); Tang et al. (2022).
3. **Selecting informative features.** We first concatenate the original features with the newly generated features from DFS Kanter & Veeramachaneni (2015) to form the *base*, which is then fed into the filters in Step 2 to obtain a pool of candidate features. For each candidate column, we compute its metric on the training set and select the top-k dimensions with the highest scores and low redundancy. The selected dimensions are then concatenated with the *base* to produce the augmented features.

DFS-BP replaces manual feature selection with training-set-based scoring, while leveraging all training samples, thereby alleviating the limitation of ICL at inference time caused by the context window size. Interestingly, the features selected by training-set scores achieve test performance comparable to, and in some cases even better than, well-trained models. This phenomenon not only demonstrates the effectiveness of the augmented features, but also suggests that they can serve as a task-specific performance baseline for evaluating the model's generalization ability on a given dataset. The detailed procedure is provided in Algorithm 1.

3.4 Pretraining

Unlike some existing PFNs-based graph foundation models, HGraphFM is trained from scratch without using any pretrained weights. A single checkpoint is shared across both classification and regression tasks, without requiring separate training for each task type. In the two-stage pre-training procedure, we adopt the relational synthetic prior from RDBPFN Wang et al. (2026a) in the second stage. Notably, to preserve the model's multiclass capability acquired in the first stage, we further augment this relational prior with multiclass tasks, where the number of classes is no greater than 10. Specifically, continuous numerical columns are discretized into bins, and the model is trained to predict the corresponding bin indices.

Loss. Different decoders allow the model to optimize task-specific objectives for different prediction settings. In this work, we randomly sample a target column from each table, currently limited to numerical and boolean columns, and split the corresponding samples into context and query sets

for ICL training. For classification tasks, including both binary and multiclass classification, we use the cross-entropy loss, while for regression tasks we adopt Huber loss Gokcesu & Gokcesu (2021). Each batch in both pre-training stages contains a mixture of the three task types, and the overall loss is computed as their average to update the model parameters. The task ratio is approximately $1/3 : 1/3 : 1/3$. In addition, we pre-generate a pool of relational prior datasets and shuffle them before sampling training tasks, which helps the model better learn database-level data and achieve faster convergence Kothapalli et al. (2026).

4 EXPERIMENTS

This section presents a comprehensive evaluation of the proposed method. Specifically, ❶ we examine the in-context learning capability and fine-tuned results of HGraphFM on both general heterogeneous graphs and relational data under few-shot settings, covering node-level multiclass classification, binary classification, and regression tasks. ❷ We further verify that DFS-BP consistently improves performance, and even compare it independently against pretrained models to assess its effectiveness and adaptability across different datasets. ❸ Additional descriptions of certain datasets and baseline methods (A, B), along with related experiments including runtime efficiency (E.1), parameter sensitivity analysis (E.3), and other supplementary results, are provided in the Appendix.

4.1 EXPERIMENTAL SETUP

Datasets. All evaluations are conducted on general heterogeneous graph datasets (ACM Wang et al. (2019), DBLP Fu et al. (2020), and IMDB Fu et al. (2020)) and the relational benchmark RelBench Robinson et al. (2024). Specifically, ACM and DBLP are citation-network datasets, while IMDB focuses on movies and television programs. All three datasets contain multiple types of nodes and edges (no fewer than three types), on which we evaluate few-shot capabilities under different shot settings. RelBench covers a wide range of scenarios, task types, and scales. Following the context-window setting used by the baselines, we adopt a fixed few-shot evaluation protocol (1024-shot) on RelBench. For all datasets, the context instances are drawn from the training set, and inference is performed on query instances from the test set.

Baselines. For general heterogeneous graphs, the compared baselines include classical heterogeneous graph neural networks and graph foundation models, including HAN Wang et al. (2019), HGT Hu et al. (2020), HetGNN Zhang et al. (2019), and HiGPT Tang et al. (2024); general graph foundation models pretrained on real-world data, including OpenGraph Xia et al. (2024), AnyGraph Xia & Huang (2026), and TS-GNN Finkelshtein et al. (2025); as well as PFN-based general graph foundation models, including NodePFN Choi et al. (2026), Node4All Lee & Yoo (2026), and GraphPFN v1.2/v1.3 Eremeev et al. (2026). For the relational benchmark, the baselines include classical tree-based models, such as Random Forest Breiman (2001) and XGBoost Chen & Guestrin (2016); the AutoML framework AutoGluon Erickson et al. (2020); a series of PFN-based single-table models, including TabICL Qu et al. (2025; 2026), LimiX Zhang et al. (2025b); Wang et al. (2026b), and TabPFN Hollmann et al. (2023); Grinsztajn et al. (2025); Hollmann et al. (2025); and relational foundation models, including Griffin Wang et al. (2025), RT Ranjan et al. (2026), PluRel Kothapalli et al. (2026), and RDBPFN Wang et al. (2026a). This work mainly focuses on unified generalization over heterogeneous graphs, and therefore does not consider methods involving domain-specific continued pretraining or other specialized adaptations.

The checkpoints obtained after Stage-1 and Stage-2 are denoted as $\text{HGraphFM}_{\text{stage}-1}$ and $\text{HGraphFM}_{\text{stage}-2}$, respectively. We report the average results over five different random seeds. Moreover, for our method, we report the inference performance from a single forward pass only, without using ensemble-style evaluation as in TabPFN Hollmann et al. (2025), in order to better reflect the model's cross-task generalization ability.

4.2 ICL PERFORMANCE

General heterogeneous graphs. Table 1 reports the few-shot node multiclass classification results under 10-shot, 20-shot, and 40-shot settings. Overall, $\text{HGraphFM}_{\text{stage}-1}$ achieves highly competitive performance across all three benchmark datasets, and shows clear advantages in most settings.

Table 1: Few-shot node multi-class classification results on ACM, DBLP, and IMDB under 10-shot, 20-shot, and 40-shot settings. Mi-F1 and Ma-F1 denote Micro-F1 and Macro-F1, respectively. The best and second-best results are marked in bold and underline, respectively.

Group	Method	ACM						DBLP						IMDB					
		10-shot		20-shot		40-shot		10-shot		20-shot		40-shot		10-shot		20-shot		40-shot	
		Mi-F1	Ma-F1	Mi-F1	Ma-F1	Mi-F1	Ma-F1	Mi-F1	Ma-F1	Mi-F1	Ma-F1	Mi-F1	Ma-F1	Mi-F1	Ma-F1	Mi-F1	Ma-F1	Mi-F1	Ma-F1
Heterogeneous GNNs	HAN	0.373	0.282	0.320	0.245	0.277	0.239	0.266	0.112	0.274	0.116	0.272	0.113	–	–	–	–	–	–
	HGT	0.319	0.201	0.355	0.210	0.356	0.273	0.249	0.117	0.285	0.121	0.301	0.127	–	–	–	–	–	–
	HetGNN	0.355	0.312	0.288	0.273	0.321	0.317	0.244	0.101	0.241	0.100	0.209	0.098	–	–	–	–	–	–
	HiGPT	0.455	**0.423**	0.507	0.491	0.469	0.410	0.312	0.201	0.385	0.316	0.330	0.232	–	–	–	–	–	–
General GFMs	OpenGraph	**0.474**	0.399	0.468	0.355	0.607	0.559	**0.353**	**0.340**	0.392	0.373	0.434	0.425	0.334	0.326	0.347	0.343	0.355	0.351
	AnyGraph	0.465	0.395	0.542	0.510	0.595	0.568	0.269	0.222	0.277	0.224	0.284	0.242	0.317	0.285	0.345	0.322	0.347	0.322
	TS-GNN	0.329	0.255	0.403	0.345	0.545	0.521	0.275	0.140	0.290	0.138	0.288	0.141	0.355	0.263	0.347	0.255	0.362	0.269
	NodePFN	0.339	0.200	0.368	0.298	0.417	0.354	0.251	0.194	0.240	0.207	0.286	0.270	0.355	0.317	0.364	0.330	0.368	0.314
	Node4All	0.374	0.319	0.366	0.325	0.616	0.614	0.255	0.186	0.327	0.280	0.420	0.412	0.329	0.267	0.356	0.281	0.352	0.299
	GraphPFNv1.2	0.340	0.184	0.401	0.297	0.691	0.689	0.284	0.176	0.298	0.261	**0.501**	0.481	0.278	0.155	0.339	0.285	0.345	0.274
	GraphPFNv1.3	0.337	0.168	0.332	0.304	0.692	0.690	0.259	0.212	0.299	0.294	0.323	0.311	0.285	0.183	0.360	0.293	0.352	0.308
Ours	HGraphFM$_{\text{stage}-1}$	0.424	0.402	**0.569**	**0.553**	**0.698**	**0.696**	0.287	0.258	**0.394**	**0.390**	0.486	**0.482**	**0.369**	**0.337**	**0.406**	**0.395**	**0.432**	**0.427**

Table 2: Performance comparison of different methods on binary classification tasks in RelBench. The last column reports the average score across all datasets. Higher is better (AUROC).

Method	`Rel − trial` `study − outcome`	`Rel − hm` `user − churn`	`Rel − stack` `user − engagement`	`Rel − amazon` `user − churn`	`Rel − avito` `user − clicks`	`Rel − avito` `user − visits`	**Avg.**
Random Forest	0.572	0.623	0.824	0.612	0.597	0.517	0.624
XGBoost	0.589	0.605	0.817	0.597	0.541	0.593	0.624
XGBoost-tuned	0.592	0.603	0.803	0.607	0.593	0.617	0.636
AutoGluon-Medium	0.575	0.637	0.750	0.595	0.537	0.597	0.615
AutoGluon-Best	0.594	0.637	0.775	0.599	0.519	0.610	0.622
TabICL v1.1	0.608	0.666	0.854	0.648	0.618	0.644	0.673
LimiX-2M	0.626	0.663	0.853	0.643	0.616	0.640	0.674
LimiX-16M	0.610	0.661	0.843	0.639	0.614	0.646	0.669
TabPFN v2	0.605	0.669	0.857	0.648	0.629	0.642	0.675
TabPFN v2.5	0.626	0.668	0.853	0.645	0.633	0.617	0.674
Griffin	0.510	0.602	0.735	0.623	0.459	0.607	0.589
PluRel	0.538	0.637	0.824	0.644	0.459	0.634	0.623
RT	0.518	0.628	0.757	0.640	0.595	0.618	0.626
RDBPFN	0.616	0.665	0.866	0.648	0.627	**0.655**	0.680
HGraphFM$_{\text{stage}-2}$	**0.667**	**0.675**	**0.877**	**0.655**	**0.638**	0.646	**0.694**
HGraphFM$^{*}_{\text{stage}-2}$	**0.688**	**0.690**	**0.882**	**0.677**	**0.678**	**0.654**	**0.712**

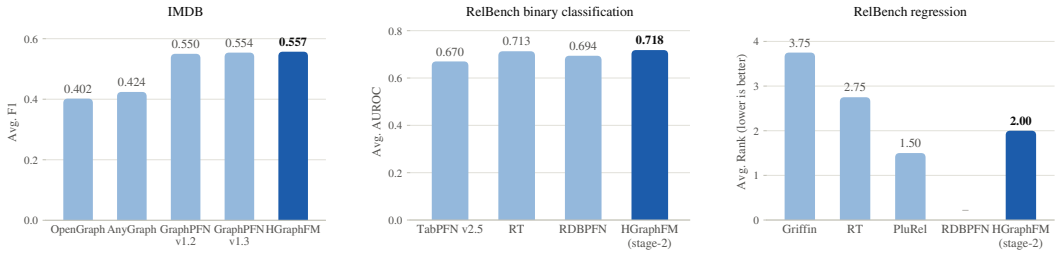
On **ACM**, although OpenGraph and HiGPT slightly outperform our method on Mi-F1 and Ma-F1 under the 10-shot setting, respectively, HGraphFM remains highly competitive and exhibits better scalability as the number of context samples increases. Notably, on **DBLP**, compared with methods pretrained on real-world graph corpora, such as OpenGraph, our synthetic-prior-based model still maintains strong competitiveness and even surpasses them under the medium-shot setting. Overall, these results validate the effectiveness of the stage-1 pretraining objective and the proposed heterogeneous graph prior.

Relational binary classification. Table 2 presents the results on binary classification tasks in RelBench. HGraphFM$_{\text{stage}-2}$ achieves the best average AUROC, outperforming all baselines with an average score of 0.694. Compared with the strongest relational PFN baseline, RDBPFN, our method improves the average AUROC from 0.680 to 0.694. The gains are particularly notable on `Rel-trial/study-outcome` (0.667 vs. 0.616) and `Rel-stack/user-engagement` (0.877 vs. 0.866). These improvements indicate that explicitly modeling heterogeneous structures and subgraph-level context provides additional predictive signals beyond those available from flattened tabular views alone. In addition, initializing the model with a general heterogeneous graph prior before further adapting it to relational synthetic data is beneficial for relational reasoning tasks. Compared with relational architectures such as Griffin, RT, and PluRel, the results further support the effectiveness of the proposed two-stage pretraining paradigm.

Relational regression. In Table 3, we report the results on regression tasks in RelBench. HGraphFM$_{\text{stage}-2}$ achieves the best overall average rank (1.5), obtaining the lowest MAE on two of the four datasets while remaining highly competitive on the other two. Unlike many single-table PFN methods that require separate handling for classification and regression, HGraphFM uses a unified checkpoint for both task types and still maintains strong performance in heterogeneous relational regression settings. In addition, RDBPFN is not designed for regression tasks, and our

Table 3: Performance comparison on regression tasks in RelBench. We report MAE on each dataset, where lower is better. The last column reports the Rank across all datasets.

Method	`Rel − trial study − adverse`	`Rel − trial site − success`	`Rel − avito ad − ctr`	`Rel − stack post − votes`	**Rank**
TabPFN v2.5	55.187	0.469	0.044	0.104	3
Griffin	78.232	0.463	0.050	0.092	3.75
RT	73.999	0.455	0.058	0.110	4.25
PluRel	64.058	**0.355**	**0.041**	0.092	1.75
RDBPFN	N/A	N/A	N/A	N/A	N/A
$\text{HGraphFM}_{\text{stage}-2}$	**54.405**	0.436	0.044	**0.089**	**1.50**
$\text{HGraphFM}^{*}_{\text{stage}-2}$	**53.813**	**0.419**	**0.040**	**0.089**	-



(a) IMDB dataset (Avg. F1-score, ↑).

(b) RelBench binary classification (Avg. AUC, ↑).

(c) RelBench regression (Avg. Rank, ↓).

Figure 3: Overall comparison on IMDB and RelBench tasks.

method fills this gap. Compared with other relational baselines pretrained purely on synthetic data, HGraphFM provides a better balance across both classification and regression tasks.

4.3 Fine-tuned Results

The fine-tuning overall results are shown in Figure 3. For different scenarios, we fine-tune checkpoints from different stages to reflect the adaptability of each stage of training to specific application settings. As real supervision is introduced, the performance gaps among different methods begin to narrow on both general heterogeneous graphs (IMDB) and relational data. HGraphFM still achieves the best performance on classification tasks and ranks second only to PluRel on regression tasks. From the perspective of unified heterogeneous graph representation learning, our method demonstrates strong adaptability to specific scenarios and downstream tasks, thereby going beyond the performance ceiling of ICL. More detailed settings and discussions can be found in E.2.

4.4 Ablation Study

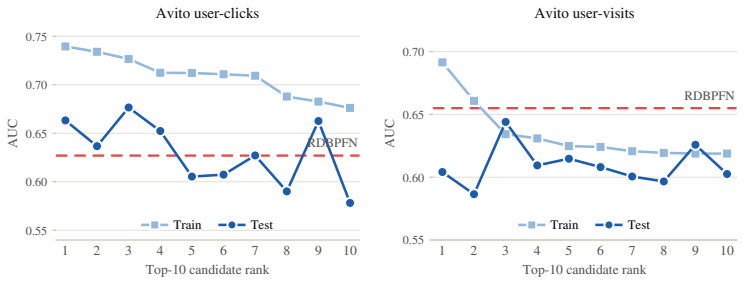
We conduct an ablation study on DFS-BP, denoted by $\text{HGraphFM}^{*}_{\text{stage}-2}$, to investigate its enhancement effect on relational data, including multi-table and cross-table settings. According to the results in Tables 2 and 3, DFS-BP consistently improves HGraphFM to varying degrees, with especially notable gains on classification tasks, where the average improvement reaches 2.6%. The improvements on regression tasks are more modest. For example, on `Rel-stack/post-votes`, the difference is only on the order of 10^{-4}. One possible reason is that graph filtering is more naturally suited to classification, where the relative magnitudes in the filtered representations can better capture target signals sparsely distributed across different channels. In contrast, regression often requires modeling strong one-to-one correlations, which is more challenging.

Notably, Figure 4 shows that only a subset of the top ten individual features selected by DFS-BP is already sufficient to approach (`user-visits`), or even surpass (`user-clicks`), the ICL performance of RDBPFN. This observation is consistent with the gains brought by DFS-BP to HGraphFM on the corresponding tasks in Table 2. In addition, we further analyze the strong adaptability of DFS-BP across different tasks; more details can be found in E.4.

5 Related Work

5.1 Foundation Models for Structured Data

Recently, foundation models for structured data have attracted increasing attention. One of the most representative directions is tabular foundation models based on Prior-Data Fitted Networks (PFNs) Müller et al. (2022), such as TabPFN Hollmann et al. (2023; 2025), Limix Zhang et al. (2025b); Wang et al. (2026b) and TabICL Qu et al. (2025; 2026). These methods achieve in-context learning on new tabular tasks by pre-training on synthetic tasks, demonstrating that synthetic priors can serve as a viable alternative to large-scale pre-training on real-world data. Inspired by this line of research, recent studies have begun to extend tabular foundation models to node-level graph tasks, including approaches based on graph-to-table transformation Eremeev et al. (2025) and hand-crafted structural features Hayler et al. (2025); Choi et al. (2025), as well as PFN-style graph foundation models that further incorporate graph priors and graph-aware architectures Choi et al. (2026); Lee & Yoo (2026); Eremeev et al. (2026). These works verify the potential of the PFN paradigm in graph learning, but they generally rely on manual features or specific preprocessing pipelines and are still mainly designed for homogeneous graphs. In contrast, this paper focuses on the more general setting of heterogeneous graphs, further extending the foundation-model paradigm for structured data to this domain.



(a) Avito user-clicks. (b) Avito user-visits.

Figure 4: AUC of the top-10 ranked candidates on two rel-avito tasks against the RDBPFN baseline (dashed).

5.2 Heterogeneous Graph Foundation Models

Compared with homogeneous graph foundation models Xia et al. (2024); Xia & Huang (2026); Finkelshtein et al. (2025), existing heterogeneous graph foundation models can be roughly divided into two categories: relational GNNs or foundation models Chen et al. (2025); Dwivedi et al. (2026); Wang et al. (2025) and traditional heterogeneous graph learning methods. The former focuses on relational databases. For example, works such as PluRel Kothapalli et al. (2026) and relational transformer Ranjan et al. (2026) mainly explore synthetic generation for relational data and relation-aware model architectures. RDB-PFN Wang et al. (2026a) adopts a single-table PFN backbone and enables relational modeling through pre-training on a synthetic relational prior. However, relational data usually depends on specific table schemas, primary–foreign key constraints, and task formulations, and is therefore not equivalent to general heterogeneous graphs. The latter category includes a large body of heterogeneous graph neural networks Wang et al. (2019); Zhang et al. (2019) and heterogeneous graph Transformers Hu et al. (2020); Tang et al. (2024), which are capable of modeling interactions among multiple node types and relation types, but are typically designed for specific datasets or rely on real-world data. In this paper, we focus on unified pre-training and transfer on more general heterogeneous graphs without relying on real-world data.

6 Conclusion

In this paper, we introduce HGraphFM, a foundation model for unified heterogeneous graph representation learning. As a novel PFN-based approach, HGraphFM benefits from the proposed general heterogeneous graph synthetic prior, the two-stage pretraining strategy, and the feature augmentation method DFS-BP. Extensive empirical evidence demonstrates that HGraphFM exhibits strong generalization ability and adaptability across heterogeneous graph settings, including both general heterogeneous graphs and relational data. Our method marks a significant step forward for foundation models on structured data, while also highlighting the importance of synthetic priors, pretraining protocols, and feature augmentation in further unlocking the potential of PFNs.

AI USE STATEMENT

In this work, we used generative AI tools to assist with translation and to implement methods. We have not used generative AI tools to generate synthetic data sets, to help develop theoretical models or conceptual frameworks, to propose or refine hypotheses, to design or provide feedback on research methodology or experiments, to clean or reformat data sets, or to interpret results, and formulating mathematical claims, providing critical ingredients for proving mathematical claims, assisting in the writing of proofs, and supporting qualitative and thematic data analysis are not applicable to this work. Additionally, we used generative AI tools to source and search for information, and to edit the research paper to improve readability.

We have reviewed all AI-assisted work. Every translated passage was reviewed by the authors against the original-language draft, and all technical terminology and claims were checked for fidelity. Implementation assistance was limited to coding the proposed methods as designed and specified by the authors; all AI-generated code was reviewed line by line and tested for correctness by the authors, and all experimental results reported in this paper were produced and validated by the authors. Information-sourcing assistance was used only to locate candidate references and documentation; every cited source was independently retrieved, read, and verified by the authors. Readability edits were reviewed by the authors to confirm that no technical content, claim, or attribution was altered.

We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Tianlang Chen, Charilaos Kanatsoulis, and Jure Leskovec. Relgnn: Composite message passing for relational deep learning, 2025.

Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pp. 785–794, 2016.

Jeongwhan Choi, Woosung Kang, Minseo Kim, Jongwoo Kim, and Noseong Park. Can tabpfn compete with gnns for node classification via graph tabularization? *arXiv preprint arXiv:2512.08798*, 2025.

Jeongwhan Choi, Jongwoo Kim, Woosung Kang, and Noseong Park. Learning posterior predictive distributions for node classification from synthetic graph priors. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=NodePFN.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pp. 4171–4186, 2019.

Alexey Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Vijay Prakash Dwivedi, Sri Jaladi, Yangyi Shen, Federico López, Charilaos I Kanatsoulis, Rishi Puri, Matthias Fey, and Jure Leskovec. Relational graph transformer. *International Conference on Learning Representations*, 2026.

Dmitry Eremeev, Gleb Bazhenov, Oleg Platonov, Artem Babenko, and Liudmila Prokhorenkova. Turning tabular foundation models into graph foundation models. *arXiv preprint arXiv:2508.20906*, 2025.

Dmitry Eremeev, Oleg Platonov, Gleb Bazhenov, Artem Babenko, and Liudmila Prokhorenkova. GraphPFN: A prior-data fitted graph foundation model. In *International Conference on Machine Learning*, 2026.

Nick Erickson, Jonas Mueller, Alexander Shirkov, Hang Zhang, Pedro Larroy, Mu Li, and Alexander Smola. Autogluon-tabular: Robust and accurate automl for structured data. *arXiv preprint arXiv:2003.06505*, 2020.

Matthias Fey, Weihua Hu, Kexin Huang, Jan Eric Lenssen, Rishabh Ranjan, Joshua Robinson, Rex Ying, Jiaxuan You, and Jure Leskovec. Relational deep learning: Graph representation learning on relational databases. *arXiv preprint arXiv:2312.04615*, 2023.

Matthias Fey, Weihua Hu, Kexin Huang, Jan Eric Lenssen, Rishabh Ranjan, Joshua Robinson, Rex Ying, Jiaxuan You, and Jure Leskovec. Position: Relational deep learning - graph representation learning on relational databases. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 13592–13607. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/fey24a.html.

Ben Finkelshtein, Ismail Ilkan Ceylan, Michael Bronstein, and Ron Levie. Equivariance everywhere all at once: A recipe for graph foundation models. *Advances in Neural Information Processing Systems*, 38:30377–30434, 2025.

Xinyu Fu, Jiani Zhang, Ziqiao Meng, and Irwin King. Magnn: Metapath aggregated graph neural network for heterogeneous graph embedding. In *Proceedings of the web conference 2020*, pp. 2331–2341, 2020.

Michelle Girvan and Mark EJ Newman. Community structure in social and biological networks. *Proceedings of the national academy of sciences*, 99(12):7821–7826, 2002.

Kaan Gokcesu and Hakan Gokcesu. Generalized huber loss for robust learning and its efficient minimization for a robust statistics, 2021. URL https://arxiv.org/abs/2108.12627.

Léo Grinsztajn, Klemens Flöge, Oscar Key, Felix Birkel, Philipp Jund, Brendan Roof, Benjamin Jäger, Dominik Safaric, Simone Alessi, Adrian Hayler, et al. Tabpfn-2.5: Advancing the state of the art in tabular foundation models. *arXiv preprint arXiv:2511.08667*, 2025.

Adrian Hayler, Xingyue Huang, Ismail Ilkan Ceylan, Michael Bronstein, and Ben Finkelshtein. Bringing graphs to the table: Zero-shot node classification via tabular foundation models. *arXiv preprint arXiv:2509.07143*, 2025.

Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter. Tabpfn: A transformer that solves small tabular classification problems in a second. *International Conference on Learning Representations*, 2023.

Noah Hollmann, Samuel Müller, Lennart Purucker, Arjun Krishnakumar, Max Körfer, Shi Bin Hoo, Robin Tibor Schirrmeister, and Frank Hutter. Accurate predictions on small data with a tabular foundation model. *Nature*, 01 2025. doi: 10.1038/s41586-024-08328-6. URL https://www.nature.com/articles/s41586-024-08328-6.

Ziniu Hu, Yuxiao Dong, Kuansan Wang, and Yizhou Sun. Heterogeneous graph transformer. In *Proceedings of the web conference 2020*, pp. 2704–2710, 2020.

James Max Kanter and Kalyan Veeramachaneni. Deep feature synthesis: Towards automating data science endeavors. In *2015 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pp. 1–10, 2015. doi: 10.1109/DSAA.2015.7344858.

Brian Karrer and Mark EJ Newman. Stochastic blockmodels and community structure in networks. *Physical Review E—Statistical, Nonlinear, and Soft Matter Physics*, 83(1):016107, 2011.

Vignesh Kothapalli, Rishabh Ranjan, Valter Hudovernik, Vijay Prakash Dwivedi, Johannes Hoffart, Carlos Guestrin, and Jure Leskovec. Plurel: Synthetic data unlocks scaling laws for relational foundation models. In *Forty-third International Conference on Machine Learning*, 2026.

Dooho Lee and Jaemin Yoo. Node4all: Learning node representation beyond datasets. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pp. 2461–2472, 2026.

Jiawei Liu, Cheng Yang, Zhiyuan Lu, Junze Chen, Yibo Li, Mengmei Zhang, Ting Bai, Yuan Fang, Lichao Sun, Philip S Yu, et al. Graph foundation models: Concepts, opportunities and challenges. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(6):5023–5044, 2025a.

Jingyuan Liu, Jianlin Su, Xingcheng Yao, Zhejun Jiang, Guokun Lai, Yulun Du, Yidao Qin, Weixin Xu, Enzhe Lu, Junjie Yan, Yanru Chen, Huabin Zheng, Yibo Liu, Shaowei Liu, Bohong Yin, Weiran He, Han Zhu, Yuzhi Wang, Jianzhou Wang, Mengnan Dong, Zheng Zhang, Yongsheng Kang, Hao Zhang, Xinran Xu, Yutao Zhang, Yuxin Wu, Xinyu Zhou, and Zhilin Yang. Muon is scalable for llm training, 2025b. URL https://arxiv.org/abs/2502.16982.

Haitao Mao, Zhikai Chen, Wenzhuo Tang, Jianan Zhao, Yao Ma, Tong Zhao, Neil Shah, Mikhail Galkin, and Jiliang Tang. Position: Graph foundation models are already here. *arXiv preprint arXiv:2402.02216*, 2024.

Samuel Müller, Noah Hollmann, Sebastian Pineda Arango, Josif Grabocka, and Frank Hutter. Transformers can do bayesian inference. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=KSugKcbNf9.

Judea Pearl. *Causality*. Cambridge university press, 2009.

Jonas Peters, Dominik Janzing, and Bernhard Schölkopf. *Elements of causal inference: foundations and learning algorithms*. The MIT press, 2017.

Jingang Qu, David Holzmüller, Gaël Varoquaux, and Marine Le Morvan. Tabicl: A tabular foundation model for in-context learning on large data. *International Conference on Machine Learning*, 2025.

Jingang Qu, David Holzmüller, Gaël Varoquaux, and Marine Le Morvan. Tabiclv2: A better, faster, scalable, and open tabular foundation model. *International Conference on Machine Learning*, 2026.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PmLR, 2021.

Rishabh Ranjan, Valter Hudovernik, Mark Znidar, Charilaos Kanatsoulis, Roshan Upendra, Mahmoud Mohammadi, Joe Meyer, Tom Palczewski, Carlos Guestrin, and Jure Leskovec. Relational Transformer: toward zero-shot foundation models for relational data. In *The Fourteenth International Conference on Learning Representations*, 2026.

Joshua Robinson, Rishabh Ranjan, Weihua Hu, Kexin Huang, Jiaqi Han, Alejandro Dobles, Matthias Fey, Jan Eric Lenssen, Yiwen Yuan, Zecheng Zhang, Xinwei He, and Jure Leskovec. Relbench: A benchmark for deep learning on relational databases. In *Advances in Neural Information Processing Systems*, 2024.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomput.*, 568(C), February 2024. ISSN 0925-2312. doi: 10.1016/j.neucom.2023.127063. URL https://doi.org/10.1016/j.neucom.2023.127063.

Jiabin Tang, Yuhao Yang, Wei Wei, Lei Shi, Long Xia, Dawei Yin, and Chao Huang. Higpt: Heterogeneous graph language model. In *Proceedings of the 30th ACM SIGKDD conference on knowledge discovery and data mining*, pp. 2842–2853, 2024.

Jianheng Tang, Jiajin Li, Ziqi Gao, and Jia Li. Rethinking graph neural networks for anomaly detection. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 21076–21089. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/tang22b.html.

Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu. Heterogeneous graph attention network. In *The world wide web conference*, pp. 2022–2032, 2019.

Yanbo Wang, Xiyuan Wang, Quan Gan, Minjie Wang, Qibin Yang, David Wipf, and Muhan Zhang. Griffin: Towards a graph-centric relational database foundation model. *International Conference on Machine Learning*, 2025.

Yanbo Wang, Jiaxuan You, Chuan Shi, and Muhan Zhang. Relational in-context learning via synthetic pre-training with structural prior. *International Conference on Machine Learning*, 2026a.

Yuanrui Wang, Xingxuan Zhang, Han Yu, Mingchao Hao, Gang Ren, Hao Yuan, Li Mao, Yunjia Zhang, Chun Yuan, and Peng Cui. Limix-2m: Mitigating low-rank collapse and attention bottlenecks in tabular foundation models. *International Conference on Machine Learning*, 2026b.

Lianghao Xia and Chao Huang. Anygraph: Graph foundation model in the wild. In *Findings of the Association for Computational Linguistics: ACL*, pp. 882–896, 2026.

Lianghao Xia, Ben Kao, and Chao Huang. Opengraph: Towards open graph foundation models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 2365–2379, 2024.

Xiaocheng Yang, Mingyu Yan, Shirui Pan, Xiaochun Ye, and Dongrui Fan. Simple and efficient heterogeneous graph neural network. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pp. 10816–10824, 2023.

Chuxu Zhang, Dongjin Song, Chao Huang, Ananthram Swami, and Nitesh V Chawla. Heterogeneous graph neural network. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 793–803, 2019.

Jingyuan Zhang, Xin Wang, Lei Yu, Zhirong Huang, Li Yang, and Fengjun Zhang. Restricted global-aware graph filters bridging gnns and transformer for node classification. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 162403–162429. Curran Associates, Inc., 2025a. doi: 10.52202/085713-5423. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/edbcb7583fd8921dad78adecfe06a99b-Paper-Conference.pdf.

Xingxuan Zhang, Gang Ren, Han Yu, Hao Yuan, Hui Wang, Jiansheng Li, Jiayun Wu, Lang Mo, Li Mao, Mingchao Hao, et al. Limix: Unleashing structured-data modeling capability for generalist intelligence. *arXiv preprint arXiv:2509.03505*, 2025b.

A Dataset Statistics

Table 4: Statistics of the heterogeneous graph datasets used in our experiments. The target node type for classification in each dataset is underlined.

	ACM	**DBLP**	**IMDB**
# node types	4	4	3
# edge types	4	3	2
Nodes	Paper: 3,025 Author: 5,959 Subject: 56 Term: 1,902	Author: 4,057 Paper: 14,328 Term: 7,723 Conference: 20	Movie: 4,278 Director: 2,081 Actor: 5,257
Edges	P–A: 9,949 P–S: 3,025 P–T: 255,619 P–P: 5,343	P–A: 19,645 P–C: 14,328 P–T: 85,810	M–D: 4,278 M–A: 12,828
Target node	Paper	Paper	Movie
# classes	3	4	3

We evaluate our method on three widely used heterogeneous graph benchmarks: **IMDB** Fu et al. (2020), **DBLP** Fu et al. (2020), and **ACM** Wang et al. (2019). **IMDB**[1] is a heterogeneous movie network containing three types of nodes, i.e., movie, director, and actor. Following the standard benchmark setting, each movie is associated with one of three classes: *Action*, *Comedy*, or *Drama*. **DBLP**[2] is an academic heterogeneous graph collected from the DBLP computer science bibliography, consisting of authors, papers, terms, and publication venues. The target nodes are categorized into four research areas, including *Database*, *Data Mining*, *Artificial Intelligence*, and *Information Retrieval*. **ACM**[3] is another academic heterogeneous graph that contains papers, authors, and subjects (and term information in the processed benchmark). Its target papers are classified into three categories: *Database*, *Wireless Communication*, and *Data Mining*.

Table 5: Overview of RelBench datasets with all entity-level prediction tasks.

Dataset	**Domain**	**#Tables**	**#Rows**	**#Cols**	**Task Name**	**Task Type**
rel-amazon	E-commerce	3	15,000,713	15	user-churn	Classification
rel-avito	E-commerce	8	20,679,117	42	user-visits user-clicks ad-ctr	Classification Classification Regression
rel-hm	E-commerce	3	16,664,809	37	user-churn	Classification
rel-stack	Social	7	4,247,264	52	user-engagement post-votes	Classification Regression
rel-trial	Medical	15	5,434,924	140	study-outcome study-adverse site-success	Classification Regression Regression

We also evaluate our method on RelBench datasets, a benchmark of real-world relational databases spanning e-commerce, social, sports, and medical domains. As summarized in Table 5, It contains diverse entity-level prediction tasks, including both classification and regression settings, over databases with substantial variation in schema size and scale.

rel-amazon (*E-commerce*).

[1] https://www.imdb.com/
[2] https://dblp.org/
[3] https://www.acm.org/

- **user-churn** (*classification*): For each user, predict whether the user will not review any product in the next 3 months.

rel-avito (*E-commerce*).

- **user-visits** (*classification*): For each user, predict whether the user will visit more than one ad in the next 4 days.
- **user-clicks** (*classification*): For each user, predict whether the user will click more than one ad in the next 4 days.
- **ad-ctr** (*regression*): For each ad, predict its click-through rate in the next 4 days.

rel-hm (*E-commerce*).

- **user-churn** (*classification*): Predict whether a customer will churn, i.e., make no transaction, in the next week.

rel-stack (*Social*).

- **user-engagement** (*classification*): Predict whether a user will make any votes, posts, or comments in the next 3 months.
- **post-votes** (*regression*): Predict how many votes a post will receive in the next 3 months.

rel-trial (*Medical*).

- **study-outcome** (*classification*): Predict whether a clinical trial will achieve its primary outcome in the next 1 year.
- **study-adverse** (*regression*): Predict the number of affected patients with severe adverse events or death in the next 1 year.
- **site-success** (*regression*): Predict the success rate of a trial site in the next 1 year.

B Baseline Configurations & Analysis

B.1 Graph learning & Foundation Models

We include several representative heterogeneous graph learning methods as baselines. **HAN**[4] adopts a hierarchical attention mechanism over metapath-based neighbors, allowing the model to capture both node-level and semantic-level importance in heterogeneous graphs. **HGT**[5] extends transformer-based message passing to heterogeneous graphs by introducing node-type- and edge-type-specific parameters for attention and feature transformation. **HetGNN**[6] learns node representations by combining heterogeneous random walk sampling with content- and neighbor-aware aggregation modules. **HiGPT**[7] further incorporates large language models into heterogeneous graph learning through an in-context graph tokenizer and heterogeneous graph instruction tuning, thereby improving generalization to unseen heterogeneous graphs and downstream tasks.

As for graph foundation models, **OpenGraph**[8] is a graph foundation model designed for open-domain graph learning, aiming to improve cross-dataset transferability through large-scale graph pretraining. **AnyGraph**[9] is a graph foundation model intended for graph learning in the wild, with an emphasis on handling diverse graph structures and feature spaces across domains. **TS-GNN**[10] is a graph foundation model that studies the role of equivariant and transferable graph representations for general-purpose node-level prediction. **Node4All**[11] learns reusable node representations across

[4] https://github.com/Jhy1993/HAN
[5] https://github.com/acbull/pyHGT
[6] https://github.com/chuxuzhang/KDD2019_HetGNN
[7] https://github.com/HKUDS/HiGPT
[8] https://github.com/HKUDS/OpenGraph.
[9] https://github.com/hkuds/anygraph.
[10] https://github.com/benfinkelshtein/EquivarianceEverywhere.
[11] https://github.com/dooho00/node4all

arbitrary graph datasets through a Channel Graph Transformer and synthetic-graph self-supervised pretraining. **NodePFN**[12] extends prior-data fitted networks to node classification by training on synthetic graph priors and enabling in-context prediction on unseen graphs. **GraphPFN**[13] is a prior-data fitted graph foundation model pretrained on millions of synthetic attributed graphs generated from a carefully designed graph prior, and supports both in-context learning and finetuning for node-level graph tasks.

B.2 Relational Deep Learning

Among classical baselines, **Random Forest**[14] is an ensemble of decision trees trained with bagging, serving as a strong non-neural baseline for structured data. **XGBoost**[15] is a gradient-boosted decision tree framework widely used for tabular prediction due to its accuracy and scalability. **AutoGluon**[16] is an AutoML system that combines multiple tabular learners with automated model selection and ensembling.

TabPFNv2[17] is a prior-data fitted tabular foundation model pretrained on millions of synthetic datasets, enabling highly effective in-context prediction on small tabular datasets without task-specific gradient-based training. For single-table foundation model baselines, **TabPFN v2.5**[18] is a PFN-based tabular foundation model pretrained on synthetic classification tasks for fast in-context prediction on small tabular datasets. **TabICL v1.1**[19] is a tabular in-context learning model designed for larger tabular datasets through scalable PFN-style pretraining. **LimiX 16M**[20] is a large-scale tabular foundation model for structured-data modeling that serves as a strong PFN-style baseline. **LimiX-2M**[21] is a compact tabular foundation model that improves efficiency through radial-basis value embeddings and a reordered bidirectional attention block, achieving a strong accuracy–efficiency trade-off on tabular benchmarks.

For relational foundation model references, **Griffin**[22] is a graph-centric relational database foundation model that focuses on transfer learning and fine-tuning over relational corpora. **RT**[23] is a relational transformer for zero-shot prediction on relational data, operating by incorporating label information into relational context rather than using the strict global-context ICL protocol adopted by RDB-PFN. **PLUREL**[24] is a synthetic relational database generation framework that models schemas, inter-table primary–foreign key connectivity, and table features in a staged manner, and is mainly used to study synthetic-data scaling laws for relational foundation models. **RDB-PFN**[25] is a relational foundation model trained purely on synthetic relational databases generated from a relational prior, and performs prediction via in-context learning over DFS-linearized relational features without requiring gradient-based adaptation at inference time.

C Implementation

For the general heterogeneous graph benchmarks, we apply principal component analysis (PCA) to reduce high-dimensional heterogeneous node attributes to 100 dimensions. If the attribute dimensions are still inconsistent after dimensionality reduction, we use the largest dimension as the reference and pad the remaining attributes with zeros, so as to satisfy the input requirement of

[12] https://github.com/jeongwhanchoi/NodePFN

[13] https://github.com/yandex-research/graphpfn.

[14] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

[15] https://github.com/dmlc/xgboost

[16] https://github.com/autogluon/autogluon

[17] https://github.com/PriorLabs/TabPFN

[18] https://github.com/PriorLabs/TabPFN

[19] https://github.com/microsoft/tabicl

[20] https://github.com/microsoft/limix

[21] https://github.com/limix-ldm-ai/LimiX.

[22] https://github.com/MuLabPKU/Griffin

[23] https://github.com/snap-stanford/relational-transformer

[24] https://star-project.stanford.edu/plurel/.

[25] https://github.com/MuLabPKU/RDBPFN

Table 6: HGraphFM two-stage pre-training setup.

	Stage 1	**Stage 2**
Steps	200,000	200,000
Initialization	from scratch	Stage-1 weights (model-only)
Batch size (global)	64	64
Micro-batch size	8	4
GPUs	16	16
Optimizer	Muon (momentum 0.9, Nesterov)	
Peak LR / end LR	8×10^{-4} / 10^{-7} (cosine, 1 cycle)	
Warmup / weight decay	2,000 steps (1%) / 0.01 (grad clip 10.0)	
Precision	fp32+AMP, FA3 off	fp32+AMP, FA3 on
Model	MultiTabICL, 40.7M params (d_{model}=128; col/row/ICL = 3/3/12 blocks, 8 heads; graph attn 4 heads)	
Task mix (per DB)	$\frac{1}{3}$ binary / $\frac{1}{3}$ $\leq$10-class / $\frac{1}{3}$ regression	
Loss	CE + Huber (δ=1), weights 1:1	
Prior	GNNSCM heterographs (graph_hetero, online synth.)	Relational DBs (pool, 1024 pre-generated DBs)
Graph size per sample	heterograph, $\leq$100k nodes	$\leq$1024 nodes (+suppl. 1024) DFS depth-1 (context-only, $\leq$100 slots)

homogeneous graph foundation models. For relational data, all pretrained foundation models are equipped with deep feature synthesis. For regression tasks, following RT Ranjan et al. (2026), we use the pretrained language model `all-MiniLM-L6-v2`[26] to encode rich text fields and augment the original features with the resulting embeddings.

Fine-tuning experiments on general heterogeneous graphs are conducted on IMDB, since the other two datasets do not provide validation sets. We select three representative and strongest baselines from different categories, namely OpenGraph, GraphPFN v1.2 and GraphPFN v1.3. For a fair comparison, we directly adopt the fine-tuning pipeline of GraphPFN v1.3, perform 10 rounds of hyperparameter tuning, and run 3 independent trials with different random seeds, reporting the average performance. We fine-tune all model parameters using only the official training split, tune hyperparameters on the validation set, and report the final results on the test set.

For relational data, the underlying databases are large in scale, often containing tens of millions of nodes. For computational efficiency, we train our method for at most 20 epochs and apply early stopping based on the validation set with a patience of 5. The evaluation protocol is kept consistent with the ICL setting. We initialize the model with different random seeds and report the average score over five independent runs.

We train the model for a total of 400,000 steps in two stages, with 200,000 steps in each stage. The model is optimized using Muon Liu et al. (2025b) with a learning rate of 8×10^{-4}, and all parameters are randomly initialized. All experiments are conducted on 16 PPU-810E GPUs. More parameter settings can be found in Table 6.

D Heterogeneous Graph Prior Analysis

D.1 Details of HGNN-SCM Prior

We build our pre-training prior on the GNN-SCM of Eremeev et al. (2026), extending it from single-table graphs to multi-table relational data: one synthesized database becomes a *heterograph* whose nodes are rows, whose edges are row-level relations, and whose tables are row groups with heterogeneous column sets. Each database is generated by:

- **Row graph.** Sampling the number of nodes $N \sim \log\mathcal{U}\{1500, 2000\}$ and the average degree from a two-component log-uniform mixture over $[2, 50]$ and $[50, 100]$ (probability

[26] `https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2`

0.5 each), and drawing the row graph from a hierarchical stochastic block model with a preferential-attachment component (2–8 first-level subgraphs, each split into 2–8 node groups).

- **Graph-coupled SCM.** Sampling the depth ℓ_{scm} and hidden width h_{scm} from the discretized noisy log-normal distributions of Hollmann et al. (2023), activations from 19 candidates (tanh, LeakyReLU, ELU, identity, ...), noise scales from noisy log-normals, and block-dropout rates from a beta distribution; each layer is a mixed linear operator that blends attribute mixing with graph convolution over the row graph, with the convolution type sampled from $\{\text{GCN}, \text{SAGE (mean/min/max)}, \text{graph-transformer}\}$ and a mixing ratio $r \in \{0.0, 0.1, \ldots, 1.0\}$ controlling how strongly the label depends on neighborhood aggregation versus own-row attributes.
- **Feature/label readout.** Sampling exogenous causes ($\{\text{normal}, \text{mixed}, \text{uniform}\}$, optionally augmented with node degree and PageRank features), propagating them through the SCM, and selecting $d \sim \mathcal{U}\{4, 100\}$ feature columns ($p_{\text{cat}} = 0.2$ made categorical, cardinality ≤ 256) and one label column from the intermediate layer outputs, with the label acting as cause or effect of the features with probability 0.5 each. The full stack of per-layer hidden states is retained as a latent $\mathbf{L} \in \mathbb{R}^{N \times W}$.
- **Table partition.** Partitioning the N rows into $T \sim \mathcal{U}\{3, 8\}$ tables with Dirichlet-distributed row counts (≥ 50 rows each); global row ids are preserved, so all row-graph edges become inter- and intra-table relations.
- **Per-table regeneration.** Instantiating, for each table t, a fresh MLP-SCM with newly drawn weights whose exogenous input is the table's rows of the latent $\mathbf{L}$, and co-reading $m_t \sim \mathcal{U}\{3, \min(20, d)\}$ feature columns and the new label y'_t from the *combined* per-layer outputs of the base GNN-SCM and the fresh MLP-SCM. Features and label are thus two readouts of one shared latent, inducing the common-cause structure $X'_t \leftarrow \mathbf{L} \rightarrow y'_t$ as a causal continuation of the base SCM rather than an arbitrary stitching of graph and labels.
- **Label-column autocomplete.** Rank-binning y'_t into C classes for classification or keeping it real-valued for regression, with the task type rolled per database uniformly among binary classification (C=2), multiclass classification ($C \sim \mathcal{U}\{3, 10\}$), and regression. The label is stored as a reserved column present on every table; at training time every table is queried simultaneously, its rows split into context (label visible, serving as in-context demonstrations) and query (label masked, prediction target) with query fraction $q \sim \mathcal{U}[0.5, 0.95]$.

Unlike label-conditioned graph priors such as contextual SBMs, which couple topology and labels through edge probabilities, HGNN-SCM couples them through the causal mechanism itself (the graph-convolution mixture r). Because each table's label column is spliced out at its query rows, a query label can be inferred only from in-context demonstrations or from propagation over inter-table edges, forcing the model to combine in-context learning and graph reasoning on every training sample.

D.2 Theoretical Analysis

The PFN framework (Müller et al., 2022) approximates posterior predictive distributions (PPDs) by minimizing the prior-data negative log-likelihood. The identity behind this — expected loss = entropy + KL divergence between the true PPD and its approximation (Müller et al., 2022, Insight 1) — is variational, uses no exchangeability, and carries over unchanged to structured conditioning objects. What does not transfer is the approximation side — that unbounded permutation-equivariant transformers are rich enough, which is proved for exchangeable example sequences (Müller et al., 2022, Corollary 1.2) — since our demonstrations are coupled through the graph. We therefore state conditions rather than transfer guarantees. The conditioning object is a heterogeneous graph — row-nodes partitioned into tables with table-dependent column sets, one untyped edge set linking rows across tables, context labels, and a query indicator — with query labels held out, while the label columns of non-target tables remain visible features. Exchangeability of examples is replaced by joint invariance under node relabeling, required of the prior and the architecture, and paired with an expressivity assumption, an idealization, and a scope qualification:

(i) Prior: the prior's law is relabeling-invariant in distribution — i.i.d. node-level causes, orientation-free edge sampling, an exchangeable table partition with independent per-table label readouts, a

Table 7: Runtime comparison of different models on ACM, DBLP, and IMDB (s).

Model	ACM	DBLP	IMDB
AnyGraph	17.6	33.0	11.4
OpenGraph	17.7	49.4	14.7
NodePFN	13.8	14.7	2.6
Node4All	19.8	39.9	3.7
GraphPFN	60.4	52.3	12.6
HGraphFM	38.9	54.6	10.6

uniformly chosen target table, and uniformly drawn query rows.[27] *(ii) Architecture:* the network is relabeling-equivariant — the in-context branch is self-attention over nodes, with keys/values restricted to context rows of the query-bearing table and no positional encoding on the node axis; context labels enter only as additive encodings on context rows (never on query rows), and query rows are excluded from key/value positions; per-table encoders summarize columns by inducing set attention over rows and apply rotary phases on the column axis only; table membership acts through presence masks, encoder scoping, connectivity, and the target-table role — never through an identifier embedding or ordering; the graph branch aggregates by commutative scatter-add after per-destination softmax, fused by residual addition. *(iii) Expressivity and idealization:* at unbounded capacity the architecture class is *assumed* dense among relabeling-equivariant predictors — the step Müller et al. (2022) prove only for exchangeable sequences — and the optimum is attained only in the infinite-data, exact-minimization limit. *(iv) Scope:* all statements are relative to the synthetic prior; transfer to real relational data is a prior-coverage question.

Under (i)–(iii), the minimizer of the population prior-data risk over relabeling-equivariant predictors is the PPD of the query label given the observed graph, and prior-data training minimizes this risk within the model class. Of the output heads, only the classification branch (softmax, cross-entropy) targets the PPD itself; the regression branch is trained either with a pinball loss on a quantile grid — yielding a predictive distribution on a finite grid, though not via a likelihood — or with the Huber loss, whose optimum is the Bayes act under that specific risk. We thus claim PPD approximation for classification only.

The graph branch is, finally, a closure requirement rather than an enrichment — relative to our prior. Labels of different tables are read out by independent per-table mechanisms from one shared latent whose dimensions mix each node's own transformation with neighbor aggregation at a per-dataset probability γ (drawn from a grid containing zero), optionally with degree- and PageRank-type inputs. Because the readouts of non-target tables are observable context, the exact PPD of a query label depends on other nodes' labels and on graph structure whenever the query's label involves neighbor-mixed latent coordinates — which fails, e.g., if $\gamma = 0$ with structural inputs off, if the query has no neighbor, or if its readout happens to select only unmixed coordinates. On that event, any predictor restricted to the query's own attributes and same-table demonstrations is suboptimal: its excess risk — the conditional KL between the PPD and the best such predictor — is strictly positive. The claim is qualitative; no quantitative gap is claimed.

E Additional Experimental Results

E.1 Computational Efficiency

According to the model architecture, the main computational cost comes from the ICL attention layers and the graph attention layers. The complexity of *Column Attention* is reduced from $O(N^2)$ to $O(Nk)$ through the two-stage induced attention design, where k denotes the number of inducing points. Overall, the total complexity of HGraphFM is $O(N^2 + Nd^2) + O(|\mathcal{E}|)$, where $|\mathcal{E}|$ is the number of edges in the graph. In general, graphs are sparse, so the ICL attention still dominates the computational cost. We further report the wall-clock time of full-graph transductive inference on graphs with over ten thousand nodes and hundreds of thousands of edges, as shown in Table 7. The

[27] Approximate: order-dependent partition repairs and statistically randomized class permutations remain; occasional column sorting affects column order, which our claims do not require.

Table 8: Fine-tuning performance comparison on the **IMDB** dataset. We report **Micro-F1** and **Macro-F1**.

Method	Micro-F1	Macro-F1
OpenGraph	0.406	0.397
AnyGraph	0.433	0.414
GraphPFNv1.2	0.557	0.542
GraphPFNv1.3	**0.560**	0.548
HGraphFM	0.558	**0.556**

Table 9: Fine-tuning performance comparison of the SOTA methods on binary classification tasks in RelBench. The last column reports the average score across all datasets. Higher is better (AUROC).

Method	Rel − trial study − outcome	Rel − hm user − churn	Rel − stack user − engagement	Rel − amazon user − churn	Rel − avito user − clicks	Rel − avito user − visits	Avg.
TabPFN v2.5	0.626	0.669	0.852	0.648	0.588	0.635	0.670
RT	0.686	**0.699**	**0.900**	**0.705**	0.636	0.650	0.713
RDBPFN	0.643	0.674	0.883	0.658	0.646	**0.660**	0.694
$\text{HGraphFM}_{\text{stage}-2}$	**0.709**	0.697	0.887	0.693	**0.667**	0.656	**0.718**

computational overhead of HGraphFM is comparable to that of existing graph foundation models, while maintaining high efficiency alongside strong generalization ability.

E.2 Fine-tuned

We select the corresponding state-of-the-art baselines for each task or dataset and conduct fine-tuning comparisons. The results are reported in Tables 8, 9, and 10. During fine-tuning, the training split is used to construct the context and query sets, the validation split is used for hyperparameter tuning and early stopping, and the final performance is reported on the test split. Overall, HGraphFM achieves the best performance on average. Although it does not consistently obtain the top score on every dataset or task, it still demonstrates strong adaptability across diverse data domains and task types. This is closely related to the generalization ability acquired during pretraining. In the fine-tuning setting, as real supervision signals are introduced, the performance gaps among baselines become smaller, suggesting that their differences in encoder-level structural modeling ability are limited. Instead, the key factors lie in the aspects emphasized in this work: prior design, pretraining protocols, and feature augmentation.

E.3 Sensitivity Analysis

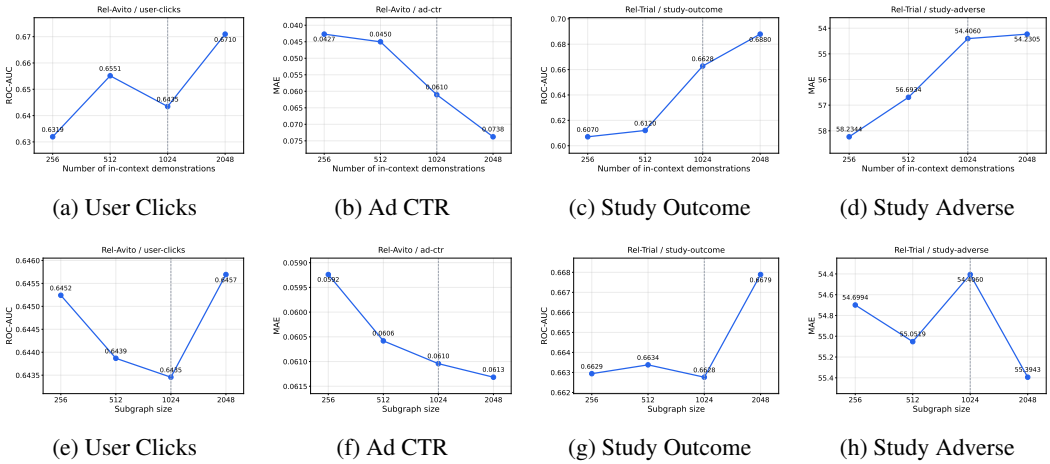
We conduct a sensitivity analysis on two important hyperparameters, namely the number of context samples and the sampled subgraph size. The results are shown in Figure 5 and cover both classification and regression tasks. As can be seen from the figure, classification tasks exhibit relatively clear trends with respect to both hyperparameters: larger subgraphs and more context samples are generally positively correlated with inference performance. Regression tasks, however, behave differently. Introducing more information does not necessarily lead to further gains, which is closely related to the nature of the task itself. For example, when the target variable has high variance or contains outliers, excessive contextual information may introduce noisy or anomalous signals that distort the data distribution and consequently harm inference performance. In practice, it is advisable to start from relatively small values and increase them gradually until the marginal benefits diminish.

E.4 Spectral Adaptability of DFS-BP

This section demonstrates the flexible adaptability of DFS-BP in complex relational data. As shown in Figure 6, greener colors indicate higher scores. The target signals of `User Clicks` and `User Visits` are mainly located in the low-frequency (concentrated) and mid-to-low-frequency (more dispersed) bands, respectively. By computing the metric on the training set (the left column), DFS-BP selects enhanced features whose corresponding patterns are also broadly preserved on the test

Table 10: Fine-tuning performance comparison of different methods on regression tasks in RelBench. We report MAE on each dataset, where lower is better. The last column reports the Rank across all datasets.

Method	`Rel − trial study − adverse`	`Rel − trial site − success`	`Rel − avito ad − ctr`	`Rel − stack post − votes`	**Rank**
Griffin	59.107	0.380	0.043	0.139	3.75
RT	44.288	<u>0.349</u>	0.039	0.077	2.75
PluRel	**42.050**	**0.274**	<u>0.038</u>	0.076	**1.5**
RDBPFN	N/A	N/A	N/A	N/A	N/A
HGraphFM$_{\text{stage}-2}$	<u>44.280</u>	0.413	**0.036**	**0.073**	<u>2.0</u>



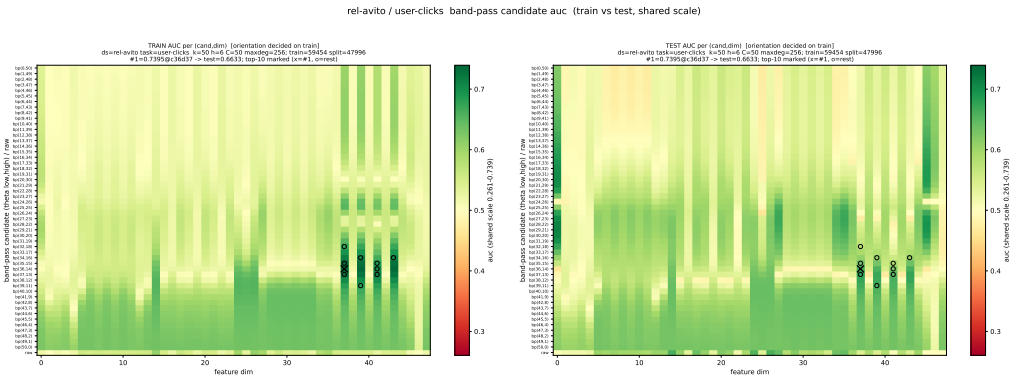
(a) User Clicks (b) Ad CTR (c) Study Outcome (d) Study Adverse

(e) User Clicks (f) Ad CTR (g) Study Outcome (h) Study Adverse

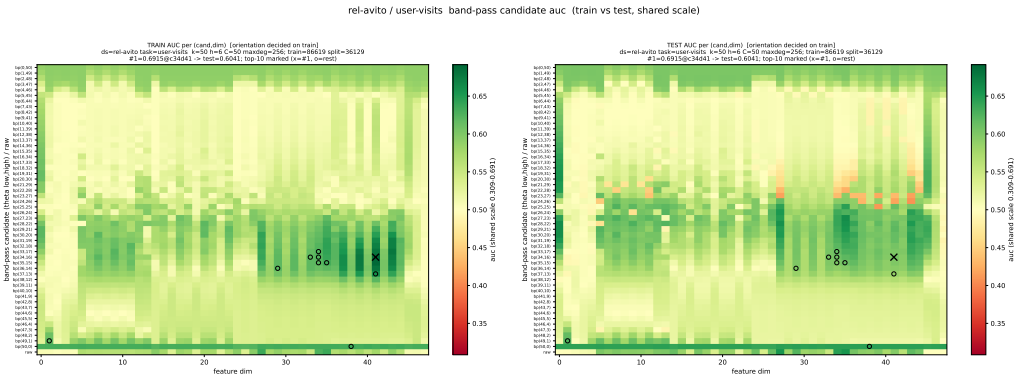
Figure 5: Parameter sensitivity analysis. Top row (a–d): sensitivity to the number of demonstrations (contexts); bottom row (e–h): sensitivity to sampled subgraph size, across four tasks.

set (the right column). This shows that DFS-BP can flexibly adapt to different datasets without requiring manual feature selection based on human expertise.

F Pseudocode of DFS-BP



(a) User Clicks



(b) User Visits

Figure 6: DFS-BP-based attribution: AUC per dimension on the two Rel-Avito tasks. Circles and crosses indicate the selected top 10 features, with crosses denoting the highest-ranked one.

Algorithm 1 Deep feature synthesis based on graph band-pass filtering on the relational task graph

Require: Task-table rows $i \in [n]$ with timestamps t_i and roles $\rho_i \in \{\mathrm{tr}, \mathrm{q}\}$ (train / held-out query); features $\mathbf{X} = [\,\mathbf{X}_{\mathrm{orig}} \mid \mathbf{X}_{\mathrm{dfs}}\,]$ with target column j_y; train labels $y = \mathbf{X}_{\mathcal{I}_{\mathrm{tr}}, j_y}$; hyperparameters h (hop radius), k (fan-out), b (degree cap), C (bank size), K (budget), τ (corr. threshold), c (clip)

Ensure: $\tilde{\mathbf{X}} = [\,\mathbf{X}_{\mathrm{orig}} \mid \mathbf{X}_{\mathrm{dfs}} \mid \mathbf{Z}\,]$, $\mathbf{Z} \in \mathbb{R}^{n \times K_{\mathrm{eff}}}$, $K_{\mathrm{eff}} \le K$, new columns on task rows only
$\mathbf{Z}_{\mathrm{tr}}$: column standardization with train statistics; $L(\boldsymbol{U}){=}\mathbf{Z}_{\mathrm{tr}}(\boldsymbol{A}\boldsymbol{U}{+}\boldsymbol{U})$, $G(\boldsymbol{U}){=}\mathbf{Z}_{\mathrm{tr}}(\boldsymbol{U}{-}\boldsymbol{A}\boldsymbol{U})$*;* corr*: |Pearson| on train rows.*

$\mathcal{P} \leftarrow \{\{i, j\} : i, j$ task rows joined through ≥ 1 non-task table within h FK hops$\}$ ▷ seeded k-fanout sampling, degrees $\le b$
for $\{i, j\} \in \mathcal{P}$ **with both** t **defined do**
  if $t_i = t_j$: $\mathcal{E} \leftarrow \mathcal{E} \cup \{i \to j,\ j \to i\}$; **else**, with p the earlier row and f the later: $\mathcal{E} \leftarrow \mathcal{E} \cup \{p \to f\}$, plus $\{f \to p\}$ if $\big[\rho_f{=}\mathrm{tr} \ \vee\ (\text{all queries share one } t \ \wedge\ \rho_p{=}\mathrm{tr})\big]$ ▷ a query never receives from a later query
end for
$\boldsymbol{A} \leftarrow \boldsymbol{D}_{\mathrm{in}}^{-1/2}\, \tilde{\boldsymbol{A}}\, \boldsymbol{D}_{\mathrm{out}}^{-1/2}$, $\tilde{\boldsymbol{A}}_{uv} = \mathbf{1}[v \to u \in \mathcal{E}]$ ▷ rows = receivers
$\mathbf{H}^{(\ell)} \leftarrow G^{\,C-\ell}\big(L^{\,\ell}(\mathbf{X}^{(0)})\big)$ for $\ell = 0, \ldots, C$, $\mathbf{X}^{(0)} \leftarrow \mathbf{Z}_{\mathrm{tr}}(\hat{\mathbf{X}})$, $\hat{\mathbf{X}} = \mathbf{X}$ minus column j_y ▷ bank $\theta^{(\ell)}{=}(\ell, C{-}\ell)$
$a_{\ell j} \leftarrow \mathrm{AUC}\big(y,\ \mathbf{H}^{(\ell)}_{\mathcal{I}_{\mathrm{tr}}, j}\big)$; $\phi_{\ell j} \leftarrow \mathbf{1}[a_{\ell j} < \frac{1}{2}]$, $s_{\ell j} \leftarrow \max(a_{\ell j},\ 1 - a_{\ell j})$; rank all (ℓ, j) by s desc.
greedy: walking the ranking, add (ℓ, j) to $\mathcal{S}$ while $|\mathcal{S}| < K$ iff $v = (-1)^{\phi_{\ell j}} \mathbf{H}^{(\ell)}_{\mathcal{I}_{\mathrm{tr}}, j}$ has $|\mathrm{corr}(v, u)| \le \tau$ for every column u of $\mathbf{X}^{(0)}_{\mathcal{I}_{\mathrm{tr}}}$ and every selected column
$\mathbf{Z} \leftarrow \big[\mathrm{clip}\big((-1)^{\phi_{\ell j}} \mathbf{H}^{(\ell)}_{:, j},\ \pm c\big)\big]_{(\ell, j) \in \mathcal{S}}$ (rank order);
return $\tilde{\mathbf{X}}$