ADAPTING TABULAR FOUNDATION MODELS FOR GRAPH NODE-LEVEL TASKS

Anonymous authors
Paper under double-blind review

ABSTRACT

While foundation models have revolutionized fields such as natural language processing and computer vision, their potential in graph machine learning remains largely unexplored. One of the key challenges in designing graph foundation models (GFMs) is handling diverse node features that can vary across different graph datasets. While many works on GFMs have focused exclusively on text-attributed graphs, the problem of handling arbitrary features of other types in GFMs has not been fully addressed. However, this problem is not unique to the graph domain, as it also arises in the field of machine learning for tabular data. In this work, motivated by the recent success of tabular foundation models (TFMs) like TabPFNv2 and LimiX, we propose G2T-FM, a simple framework that allows tabular foundation models to be applied to graph node-level tasks. Specifically, G2T-FM augments the original node features with neighborhood feature aggregation, adds structural embeddings, and then applies a TFM to the constructed node representations. Even in the in-context learning setting, our model achieves strong results when combined with a strong TFM, outperforming both prior GFMs and well-tuned GNNs trained from scratch. Moreover, after finetuning, G2T-FM consistently surpasses well-tuned GNN baselines, often by a significant margin. In summary, our paper reveals the potential of a previously overlooked direction: utilizing tabular foundation models for graph machine learning tasks. Our source code is available at https://anonymous.4open.science/r/8456b37f41900b.

1 INTRODUCTION

In recent years, foundation models have become a major breakthrough in deep learning. Foundation models are large machine learning models that are pretrained on diverse and extensive datasets. After this pretraining phase, they can be easily adapted to a variety of specific tasks with minimal additional training. Well-known examples include BERT (Devlin et al., 2019) and GPT (Brown et al., 2020) in natural language processing, as well as CLIP (Radford et al., 2021) in computer vision. The core principle behind foundation models is to learn general representations by leveraging large and varied data. These representations capture important patterns and semantics in the data, making the pretrained models highly transferable to different tasks. As a result, foundation models consistently achieve state-of-the-art results, while also improving efficiency and generalization. Furthermore, these models unify techniques across different fields, driving rapid progress and innovation in deep learning research and applications.

Despite their remarkable success in areas such as computer vision and natural language processing, the development of foundation models for graph data has not advanced to the same extent. The challenges of developing graph foundation models (GFMs) stem from the fact that graphs are not actually a single domain, but rather a way to represent data from different domains. These domains use graphs to represent very different structures, such as social networks, web networks, road networks, co-purchasing networks, molecules, connectomes, or even abstract objects and their relations. Thus, successful GFMs should be able to work with graphs from different domains where nodes represent very different objects and edges represent very different relations, which is a rather formidable task that requires overcoming many serious challenges. Two key challenges faced by GFMs are the ability to transfer to new feature spaces and target spaces. Graphs from different domains often have different node features and different targets, making it difficult to design GFMs that can work across

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.

various types of graphs. Some existing GFMs restrict themselves to text-attributed graphs (Wang et al., 2024b; He et al., 2025; Liu et al., 2024), which allows them to use pretrained text encoders. Another approach is to use simple dimensionality reduction methods like SVD and PCA (Xia & Huang, 2024; Zhao et al., 2024a; Wang et al., 2025a; Yu et al., 2025), which allow transforming all feature spaces to a space with a fixed predefined number of features. However, these approaches do not allow for the full and effective leveraging of arbitrary node features in graphs from new domains.

The challenges of transferring to new feature and target spaces are not, however, exclusive to graphs. Tabular data — one of the most widespread data modalities in machine learning — is similar to graph-structured data in that it does not constitute a single domain but is a way to represent data from different domains. Thus, tabular datasets come with different feature and target spaces, so tabular foundation models (TFMs) face issues similar to those faced by GFMs. While TFMs are not as developed as foundation models for language or vision, they have seen increased interest recently (Van Breugel & Van Der Schaar, 2024), with the first successful approaches being proposed (Hollmann et al., 2023; 2025; Mueller et al., 2025; Ma et al., 2025; Qu et al., 2025). For instance, TabPFNv2 (Hollmann et al., 2025) demonstrates strong performance in both in-context and finetuning settings, and it has recently gained significant attention from the community.

This parallel suggests that developers of GFMs can draw inspiration from TFMs, which have to deal with many of the same challenges. In this paper, we take a first step in this direction and show that tabular foundation models, such as TabPFNv2 (Hollmann et al., 2025) and LimiX (Zhang et al., 2025a), can be effectively adapted to graph datasets. We introduce a simple framework named Graph-to-Table Foundation Model (G2T-FM), which transforms graph node-level tasks into tabular ones and solves them with a tabular foundation model. More specifically, we augment the original features with neighborhood feature aggregations (Bazhenov et al., 2025), classical structure-based features (node degree, PageRank, and the eigenvectors of the graph Laplacian), and learnable structure-based encodings (Kanatsoulis et al., 2025). Then, we apply a tabular foundation model to the constructed node representations to get predictions.

Our empirical results indicate that this straightforward framework achieves strong results in a fully in-context learning setting, outperforming both prior GFMs and well-tuned GNNs trained from scratch. Moreover, after finetuning, G2T-FM surpasses well-tuned GNN baselines, with especially strong improvements obtained when G2T-FM uses LimiX as a tabular foundation model (see Table 2). These results highlight the potential of the proposed approach and the positive transfer brought by the usage of foundation models.

Importantly, G2T-FM can be directly applied to any existing or future TFM. Given the rapid pace of development of tabular foundation models (Grinsztajn et al., 2025; Qu et al., 2026), in terms of both downstream quality and scalability, G2T-FM can directly benefit from it, and thus we believe that G2T-FM will remain an important and up-to-date baseline even with further development of GFMs.

Our main contributions can be summarized as follows.

- We identify a promising and previously overlooked direction of applying tabular foundation models to graph machine learning tasks.
- As a proof of concept, we introduce G2T-FM, a simple framework that transforms graph node-level tasks into tabular ones and solves them with a tabular foundation model.
- We show that G2T-FM serves as a strong baseline for GFMs, outperforming both publicly available GFMs and well-tuned traditional GNNs even in the in-context learning setting, and that finetuning further improves performance.

2 Background

2.1 Graph foundation models for node classification

Graph foundation models (GFMs) have recently gained significant attention in the field of graph machine learning. The main purpose of GFMs is to enable effective transfer of knowledge across different graph datasets. In other words, they aim to learn knowledge from a variety of graph tasks that can be successfully applied to other graphs. In this work, we primarily focus on node-level tasks such as node classification and node regression. While many GFMs are limited to text-attributed

graphs (Wang et al., 2024b; He et al., 2025; Liu et al., 2024), graphs in many domains involve non-textual features. Therefore, we specifically discuss methods applicable to graphs with arbitrary numerical and categorical node features, which frequently appear in various real-world industrial applications.

In the following sections, we review the key design choices and considerations in the development of GFMs. In particular, we focus on pretraining objectives and data, as well as how GFMs handle graph structure and node features. For further details, see the survey by Wang et al. (2025b).

Pretraining objective Some graph foundation models use self-supervised learning (SSL) objectives to guide their pretraining process (Zhao et al., 2024a; Yu et al., 2025; Wang et al., 2025a), whereas others employ supervised learning strategies (Lachi et al., 2025; Finkelshtein et al., 2025). Notably, several works (Xia et al., 2024; Xia & Huang, 2024) reduce the node classification task to link prediction. Specifically, for each label in a downstream task, they create a virtual node that is connected to all training nodes of that class. Node classification thus becomes a task of predicting links to those virtual class nodes, for which the models are pretrained.

Pretraining data Collecting a sufficiently diverse collection of datasets for pretraining GFMs remains a significant challenge. To address this, some studies (Fey et al., 2025; Lachi et al., 2025; Xia et al., 2024) incorporate synthetic data — either together with real-world data or as an alternative. This includes the use of simple random graph models like stochastic block models (Lachi et al., 2025), as well as synthetic graphs generated by large language models (Xia et al., 2024). However, the majority of graph foundation models rely primarily on real-world datasets for pretraining. The number of datasets used for this purpose varies from as few as one graph (Finkelshtein et al., 2025), to larger collections of 2–10 datasets (Zhao et al., 2024a; Wang et al., 2025a), and up to several dozens in some studies (Xia & Huang, 2024; Lachi et al., 2025).

Handling features One of the key challenges for graph foundation models is handling heterogeneous features that can vary significantly across different datasets. Some approaches address this by focusing exclusively on text-attributed graphs, sometimes additionally converting non-textual features to text, and then applying a text encoder (Wang et al., 2024b; He et al., 2025; Liu et al., 2024). Methods aiming to deal with arbitrary features often rely on dimensionality reduction techniques such as SVD or PCA to obtain feature embeddings (Xia & Huang, 2024; Zhao et al., 2024a; Wang et al., 2025a; Yu et al., 2025). There are alternatives, such as learning dimension encoding modules that produce feature transformations (Zhao et al., 2024b), learning graph patches (Sun et al., 2025) or replacing node attribute values with their statistical dependencies (Shen et al., 2025), but these appear less common. We also highlight Finkelshtein et al. (2025), which constructs separate embeddings for each `(node, feature)` pair, enabling a more fine-grained representation of feature information.

Handling structure Handling the structure is more straightforward, as graph neural networks are well-suited for this task, and they are inherently capable of processing arbitrary graph structures. Consequently, many GFMs simply adopt GNNs as their backbone to handle graph structure (Zhao et al., 2024a; Finkelshtein et al., 2025). In addition to GNN-based approaches, some methods apply matrix decomposition techniques like SVD to graph-derived matrices (for example, the normalized adjacency matrix or the sum of its powers), to encode structural information (Xia et al., 2024). However, while GNNs can in principle operate on any graph, their performance may still be limited due to varying graph structures. To address this, some works implement additional mechanisms specifically designed to handle structural differences (Yu et al., 2025; Wang et al., 2025a).

2.2 Limitations of existing GFMs

Focus on text-attributed graphs Many existing graph foundation models are specifically designed for text-attributed graphs, where nodes or edges have associated textual information (Wang et al., 2024b; He et al., 2025; Liu et al., 2024). These models typically leverage large language models or other text encoders to process textual attributes, integrating natural language representations with graph structures. While this approach can be effective for certain domains such as academic networks or knowledge graphs, it limits the applicability of GFMs across a broader range of graphs where such text attributes are not available. For instance, for graphs representing transportation networks, biological networks, or transaction networks (commonly used for fraud detection tasks), which often come with rich numerical and categorical features, the reliance exclusively on textual information restricts the model's usability and effectiveness. As a result, many current GFMs may

not generalize well to graphs with non-textual attributes, hindering their adoption across diverse real-world scenarios.

Limited support for regression tasks Most publicly available GFMs are designed and evaluated on classification tasks, where the goal is to predict categorical labels for nodes, edges, or entire graphs. To date, no publicly available GFMs, aside from TS-GNN (Finkelshtein et al., 2025), support regression tasks, where the output is a continuous value rather than a class label. This is a substantial limitation because many important graph-based applications require regression instead of classification. The lack of support for regression tasks reduces the practical applicability of current GFMs and highlights an important area for future research.

Misleading use of the "zero-shot" term Some recent studies on graph foundation models have described their methods as operating in a "zero-shot" setting (Xia & Huang, 2024; Xia et al., 2024). Typically, these approaches introduce virtual nodes that represent target classes and connect them to the corresponding real nodes with known class labels. Then, the node classification problem reduces to predicting links between the test nodes and the appropriate virtual nodes. This process makes it possible to perform evaluation on unseen graphs without additional finetuning. While inventive and interesting, this technique does not truly realize zero-shot learning. Strictly speaking, zero-shot learning means that no labeled examples of the target classes are available during evaluation. However, the described method requires labeled nodes to be connected to virtual class nodes for effective link prediction. Therefore, the correct term for this setup should be "in-context learning", since evaluation does not involve further finetuning but still depends on access to labeled training samples. This inconsistency in terminology may lead to misleading comparisons with baseline approaches. For instance, the aforementioned studies (Xia & Huang, 2024; Xia et al., 2024) compare their "zero-shot" performance against the one-shot and five-shot results of other baselines, yet they do not clearly report the number of training samples used in "zero-shot" evaluation of the proposed method, which makes the comparison harder to interpret.

2.3 Tabular foundation models

The field of tabular foundation models (TFMs) was pioneered by the TabPFN model (Hollmann et al., 2023), which was designed to address any tabular problem off-the-shelf. TabPFN employs a transformer-like architecture and works in the in-context learning regime, with the entire downstream training set serving as the prompt. The pretraining of TabPFN was performed on a large number of synthetic datasets designed to mimic typical tabular tasks. The more recent model, TabPFNv2 (Hollmann et al., 2025), employs a more powerful backbone architecture, pretraining on a broader spectrum of synthetic datasets, and advanced techniques of data preprocessing. Nowadays, new TFMs are emerging regularly (Ma et al., 2025; Mueller et al., 2025; Zhang et al., 2025b;a; Qu et al., 2025; 2026; Grinsztajn et al., 2025; 2026), and their success is exploited beyond the domain of pure tabular tasks, e.g., for time series forecasting (Hoo et al., 2025). In our work, we demonstrate that TFMs can also serve as a core building block for graph foundation models.

3 G2T-FM framework

As discussed above, one of the key challenges for graph foundation models is processing node features that can vary widely across different graphs and domains. To address this, previous approaches primarily rely on one of the following strategies. The first is to apply dimensionality reduction techniques, such as principal component analysis or singular value decomposition (Xia & Huang, 2024; Zhao et al., 2024a; Wang et al., 2025a; Yu et al., 2025). However, this approach can result in the loss of information, and, although it ensures that all feature vectors share the same dimension after reduction, it remains unclear whether these reduced features are transferable across different graphs. The second strategy employs text encoders to process node or edge features (Wang et al., 2024b; He et al., 2025; Liu et al., 2024). This method is highly effective for text-attributed graphs, where the features are naturally represented as text. Nevertheless, many real-world graphs do not include solely textual features (Ivanov & Prokhorenkova, 2021; Chen et al., 2022; Robinson et al., 2024; Wang et al., 2024a; Bazhenov et al., 2025). Using text encoders for non-text features can therefore be highly suboptimal, as it does not leverage the nature or structure of these features. Overall, neither of these strategies fully addresses the problem of handling diverse and heterogeneous features.

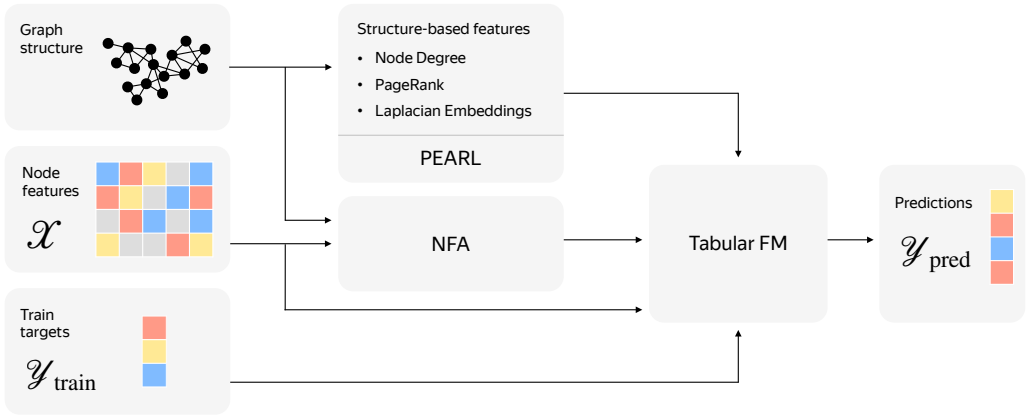

Figure 1: Overview of the proposed G2T-FM framework.

Similarly, the problem of adapting to different target spaces has also not been fully solved. Existing approaches, such as converting node classification to link prediction or using textual descriptions of node classes, only work with node classification tasks and not with node regression tasks, which frequently appear in real-world applications of graph machine learning.

However, these challenges are not unique to graph machine learning, and they also arise in tabular machine learning. Recent advances in tabular deep learning, such as the development of foundation models like TabPFNv2 (Hollmann et al., 2025), offer promising solutions for handling diverse feature and target spaces.

We argue that tabular foundation models can be used to create strong models for graph node-level tasks, and, in particular, they can help handle different feature and target spaces. As a proof of concept, we introduce the Graph-to-Table Foundation Model (G2T-FM) framework, which addresses the challenge of learning on graphs with diverse and heterogeneous node features and different targets. Figure 1 provides an overview of our method.

To process heterogeneous node features, we employ a tabular foundation model like TabPFNv2 (Hollmann et al., 2025), TabPFN-2.5 (Grinsztajn et al., 2025) or LimiX (Zhang et al., 2025a). TFMs are designed for tabular data only, so to make them applicable to graph-structured data, we introduce a graph-based preprocessing step that encodes graph information into the node features. Our goal is to capture the information about different aspects of the graph structure as well as the interplay between the graph structure and the node features. Hence, the new augmented feature vector consists of the original node features and the following graph-based components.

Neighborhood feature aggregation (NFA) Following Bazhenov et al. (2025), for each node, we compute aggregated feature statistics over its one-hop neighbors. Namely, for each numerical feature, we compute its mean, maximum, and minimum values over the node's neighbors. For each categorical feature, we first apply one-hot encoding and then compute the mean of the obtained binary features. The computation of NFA uses both the graph structure and node features. This component provides information about the features in the local neighborhood of the node, which is a valuable signal for many graph-related tasks.

Classic structure-based features (SF) We also include basic node structural characteristics — node degree, PageRank score, and Laplacian eigenvectors. The first two are classic node centrality measures that indicate how "important" a particular node is; however, they do so in different ways. Node degree captures strictly local information, while PageRank also captures global information. Then, we compute the first K non-trivial eigenvectors of the symmetric normalized Laplacian and consider the corresponding K-dimensional embeddings as additional feature vectors. Laplacian embeddings encode a node's position within the graph relative to other nodes. Thus, such node representations provide valuable information that supplements the centrality measures.

Learnable structure-based encodings (PEARL) These encodings have been proposed by Kanatsoulis et al. (2025). The idea of PEARL is to generate a random value for each node and then apply a GNN using these values as node features. Such random initialization increases the expressive power of GNNs by breaking the structural symmetries. This procedure is repeated M times (each with new random node features) and the resulting node embeddings are averaged so that node permutation equivariance still holds in the limit. Kanatsoulis et al. (2025) propose using this as a learnable module, so that the encodings are trained to improve downstream performance. However, we noticed that PEARL encodings are also useful without training, i.e., when we use a randomly initialized GNN to obtain node embeddings. Thus, this module can produce both non-learnable and learnable representations, where non-learnable representations do not involve parameter optimization (training).

These new features allow us to provide the tabular foundation model with information about many properties of the graph underlying the given dataset. We concatenate them with the original node attributes and use the resulting augmented feature vector as an input to a TFM. The resulting model can be applied in a fully in-context learning regime and, as we show below, it outperforms well-tuned GNNs trained from scratch. In the finetuning regime, we jointly tune PEARL and TFM.

4 Experimental setup

4.1 Datasets

In our experiments, we use TabPFNv2 as one of the backbones, which imposes specific constraints that guide our dataset selection. In particular, TabPFNv2 is suitable for both regression tasks and multi-class classification tasks, but the latter are restricted to at most 10 classes. Additionally, it is designed for small-to-medium-scale datasets, requiring no more than roughly 10,000 training samples. As such, we select datasets that fit within these boundaries.

We note that further advances in TFMs have alleviated the 10,000-training-sample limit (Grinsztajn et al., 2025), and error-correcting output codes (Dietterich & Bakiri, 1994) make it possible to handle more than 10 classes even with TabPFNv2. However, we keep the dataset selection within these boundaries in order to evaluate a broader range of models in their default setups.

To comprehensively assess the capabilities of our method, we construct two collections of datasets. The first focuses on graphs with non-textual node features (our primary setting), while the second contains well-known graph benchmarks with text-based node features. Across these datasets, our selection covers both regression and classification tasks, and includes graphs with both homophilous and non-homophilous structure,[1] as summarized in Table 1.

Datasets with non-textual features This collection comprises eight datasets from the GraphLand benchmark (Bazhenov et al., 2025), all featuring diverse tabular node features.[2] The datasets include: social networks `artnet-exp`, `artnet-views`, and `twitch-views`; a network of workers from a crowdsourcing platform `tolokers-2`; a network of users of a review service `city-reviews`; a co-purchasing network `hm-prices`; a network of devices `avazu-ctr`; and a road network `city-roads-M`. Importantly, compared to many classical graph datasets, GraphLand provides more diverse and realistic tasks, and thus we mostly focus on this collection in our paper.

Datasets with text-based features This collection includes five classical graph datasets where node features are derived from textual descriptions: a network of users of a question-answering website `questions` (Platonov et al., 2023b); a citation network `pubmed` (Yang et al., 2016); a social network `facebook` (Rozemberczki & Sarkar, 2020); a co-purchasing network `amazon-ratings` (Platonov et al., 2023b); and a network of Wikipedia pages `wiki-cs` (Mernyei & Cangea, 2020).

While specialized graph foundation models that directly process raw text may yield better results for text-attributed graphs, including these datasets allows us to test the generalization of our approach. Also, we explicitly exclude datasets with bag-of-words (BoW) node features, as BoW is less relevant

[1] A graph is called homophilous if its edges tend to connect nodes with similar labels, see Newman (2003); Platonov et al. (2023a); Mironov & Prokhorenkova (2024) for details.

[2] By tabular features, we mean a mixture of numerical and categorical features with different distributions.

Table 1: The key statistics of the considered graph datasets.

name	# nodes	# edges	# features	mean degree	# classes	homophily	feature type
tolokers-2	11,758	519,000	16	88.3	2	no	tabular
city-reviews	148,801	1,165,415	37	15.7	2	yes	tabular
artnet-exp	50,405	280,348	75	11.1	2	no	tabular
hm-prices	46,563	10,730,995	41	460.9	–	no	tabular
avazu-ctr	76,269	10,984,077	260	288.0	–	no	tabular
city-roads-M	57,073	107,104	26	3.8	–	yes	tabular
twitch-views	168,114	6,797,557	4	80.9	–	no	tabular
artnet-views	50,405	280,348	50	11.1	–	no	tabular
pubmed	19,717	44,324	500	4.5	3	yes	text-based
facebook	22,470	170,823	128	15.2	4	yes	text-based
amazon-ratings	24,492	93,050	300	7.6	5	no	text-based
questions	48,921	153,540	301	6.3	2	no	text-based
wiki-cs	11,701	215,603	300	36.9	10	yes	text-based

to modern text processing, while introducing additional challenges like high-dimensionality and sparsity.

For all datasets, we employ a standardized data splitting protocol, allocating 10% of the nodes to training, 10% to validation, and the remaining 80% to testing. For the GraphLand datasets, we use the official RL (random low) splits. For the remaining datasets, we employ the random stratified splitting, which ensures consistent class distributions across the splits. All the experiments are run in a transductive setting, which is standard for node property prediction in the graph domain. For binary classification tasks, we report average precision. For multiclass classification tasks, we report accuracy. For regression tasks, we report R^2. For all metrics, higher is better.

4.2 Methods

In our experiments with G2T-FM, we adopt TabPFNv2 (Hollmann et al., 2025), TabPFN-2.5 (Grinsztajn et al., 2025) and LimiX (Zhang et al., 2025a) as backbone models. We refer to these models as G2T-TabPFNv2, G2T-TabPFNv2.5, and G2T-LimiX, respectively. These models are evaluated either in an in-context learning (ICL) or finetuning (FT) settings. Following a common practice in tabular PFNs (Hollmann et al., 2025; Qu et al., 2026), we use inference-time ensembling for G2T-FM via averaging predictions from 10 forward passes with different random seeds, see Appendix C for details. For comparison, we also evaluate the following baseline methods.

First, we include traditional supervised baselines trained from scratch for each dataset. These include four classic GNNs: GCN (Kipf & Welling, 2017), GraphSAGE (Hamilton et al., 2017), GAT (Veličković et al., 2018), and neighborhood-attention Graph Transformer (GT) (Shi et al., 2021).[3] For all GNNs, we use the modifications from Platonov et al. (2023b) that augment GNNs with residual connections (He et al., 2016), layer normalization (Ba et al., 2016), and MLP blocks, which often significantly improve their performance. For the GraphLand datasets with tabular features, we also use LightGBM-NFA as a strong baseline — which is a popular implementation of gradient-boosted decision trees (Ke et al., 2017) with input features augmented with graph neighborhood information via NFA (Bazhenov et al., 2025). The implementation of these models closely resembles that from GraphLand (Bazhenov et al., 2025) in terms of both model architecture and hyperparameter tuning, see Appendix C for more details.

Second, we employ several publicly available graph foundation models. Despite significant interest in developing GFMs recently, most of the research is focused exclusively on text-attributed graphs (as discussed above), so we were able to find only a few openly available models that support node property prediction in graphs with arbitrary node feature spaces. Specifically, we employ AnyGraph (Xia & Huang, 2024), OpenGraph (Xia et al., 2024), and TS-GNN (Finkelshtein et al., 2025), which are used in the in-context learning (ICL) regime, as well as GCOPE (Zhao et al., 2024a), SAMGPT (Yu et al., 2025) and MDGFM (Wang et al., 2025a), which are used in the finetuning (FT) regime. For all these methods, we use the original implementations provided by the authors. Further,

[3]Neighborhood-attention GT uses only local attention to node's neighbors, in contrast to graph transformers with global all-to-all attention.

Table 2: Evaluation results on datasets with tabular features, i.e., datasets from the GraphLand benchmark under the RL (random low) data split. For each column, we highlight first, second, and third best results with a color. We also report average rank over classification datasets in the 'AR (cls)' column and over all datasets in the 'AR (all)' column. During the computation of 'AR (all)' we exclude all methods that do not support regression tasks, so that ranks are computed only over tables with no missing entries.

	artnet-exp	city-reviews	tolokers-2	artnet-views	avazu-ctr	city-roads-M	hm-prices	twitch-views	AR (cls)	AR (all)
LightGBM-NFA	46.13 ± 0.04	78.53 ± 0.01	56.34 ± 0.06	56.10 ± 0.02	31.71 ± 0.01	61.18 ± 0.03	70.84 ± 0.04	60.14 ± 0.01	**6.33**	**6.62**
GCN	44.86 ± 0.36	77.81 ± 0.15	56.27 ± 0.31	56.03 ± 0.25	32.00 ± 0.16	58.82 ± 0.25	68.02 ± 0.42	75.51 ± 0.05	**8.67**	**7.38**
GraphSAGE	45.14 ± 0.36	78.17 ± 0.10	54.43 ± 0.34	49.32 ± 0.91	31.44 ± 0.16	59.44 ± 0.27	70.00 ± 0.74	66.29 ± 0.32	**8.00**	**8.38**
GAT	45.06 ± 0.52	77.74 ± 0.21	57.41 ± 0.85	53.60 ± 0.24	32.63 ± 0.17	59.86 ± 0.20	72.07 ± 1.22	72.89 ± 0.27	**7.67**	**6.50**
GT	46.41 ± 0.71	77.34 ± 0.21	56.98 ± 0.55	53.37 ± 0.46	31.11 ± 0.49	59.55 ± 0.28	69.44 ± 0.94	72.13 ± 0.13	**7.00**	**7.50**
OpenGraph (ICL)	15.16 ± 0.83	59.09 ± 0.72	40.38 ± 1.13	–	–	–	–	–	**12.67**	–
AnyGraph (ICL)	12.84 ± 0.93	63.71 ± 1.45	28.75 ± 3.56	–	–	–	–	–	**14.67**	–
TS-GNN (ICL)	20.44 ± 1.05	43.46 ± 5.17	38.54 ± 0.94	–	–	–	–	–	**12.67**	–
GCOPE (FT)	14.92 ± 1.56	67.16 ± 0.98	28.81 ± 1.28	–	–	–	–	–	**13.67**	–
SAMGPT (FT)	16.20 ± 0.93	45.91 ± 2.68	36.93 ± 1.07	–	–	–	–	–	**13.33**	–
MDGFM (FT)	16.99 ± 1.75	31.24 ± 8.50	31.97 ± 1.34	–	–	–	–	–	**14.00**	–
G2T-TabPFNv2 (ICL)	45.73 ± 0.03	77.28 ± 0.25	60.39 ± 0.19	59.72 ± 0.14	28.00 ± 0.36	60.12 ± 0.03	65.75 ± 0.02	70.21 ± 0.04	**6.67**	**7.38**
G2T-TabPFNv2.5 (ICL)	49.33 ± 0.12	79.99 ± 0.04	61.21 ± 0.12	60.64 ± 0.09	33.06 ± 0.11	63.53 ± 0.07	74.32 ± 0.09	71.64 ± 0.06	**2.00**	**3.00**
G2T-LimiX (ICL)	48.42 ± 0.78	78.98 ± 0.44	61.60 ± 0.18	61.58 ± 0.08	32.70 ± 0.14	65.16 ± 0.07	76.14 ± 0.08	71.31 ± 0.06	**2.33**	**3.00**
G2T-TabPFNv2 (FT)	47.53 ± 0.62	78.69 ± 0.30	57.54 ± 0.93	60.60 ± 0.16	33.38 ± 0.41	63.05 ± 0.39	72.58 ± 0.40	74.24 ± 0.11	**4.33**	**3.75**
G2T-LimiX (FT)	50.39 ± 0.19	80.65 ± 0.05	59.75 ± 1.14	63.24 ± 0.07	34.09 ± 0.37	66.29 ± 0.12	77.37 ± 0.17	74.91 ± 0.06	**2.00**	**1.50**

we were only able to evaluate these methods on datasets with node classification tasks, as none of them support node regression.[4]

For all the methods, we run each experiment 10 times (5 times for GFMs from prior literature) and report the mean and standard deviation of the model performance, since the results are affected by stochasticity during model training and inference: some of the considered baselines have stochastic inference by design, while others require training or finetuning with different random states.

5 Experimental Results

Table 2 contains the evaluation results on graph datasets with tabular features, and Table 3 contains the additional results on datasets with text-derived features. In addition to the results on individual datasets, we also report the average ranks (AR). Below, we summarize and discuss our key observations.

Observation 1. *In our evaluation, the existing publicly available graph foundation models perform substantially worse than well-tuned traditional GNNs trained from scratch.*

This observation holds true across both collections of datasets we evaluated. The sole exception we identified is the performance of TS-GNN on the amazon-ratings dataset, where it surpassed the GNNs trained from scratch. In all other cases, the performance of the existing GFMs was significantly lower than that of our GNN baselines.

While many GFM publications report outperforming traditional GNNs, our findings suggest otherwise. We hypothesize that this discrepancy stems from several key differences in the evaluation protocol. First, unlike some GFM studies that focus on few-shot benchmarks, we use larger, more realistic, training splits (10%/10%/80%) that allow GNNs to be trained effectively from scratch. Second, we ensure our GNN baselines are highly competitive by performing a thorough hyperparameter optimization and using the GNN architectures from Platonov et al. (2023b) that include established performance-enhancing features like residual connections and normalization (Luo et al., 2024; 2025), which are often absent in simpler baselines.

Observation 2. *When endowed with a strong TFM backbone, G2T-FM evaluated in the in-context learning mode performs better on average than traditional GNNs trained from scratch.*

In particular, in terms of the average rank on both collections of datasets, G2T-LimiX and G2T-TabPFNv2.5 outperform all baselines trained from scratch, while G2T-TabPFNv2 performs on par

[4]While TS-GNN can in theory perform node regression, its current official implementation does not support that.

Table 3: Evaluation results on classic graph datasets with text-based features. For each column, we highlight first, second, and third best results with a color. We also report average rank (AR) in the last column.

	amazon-ratings	facebook	pubmed	questions	wiki-cs	AR
GCN	41.43 ± 0.49	91.26 ± 0.21	85.46 ± 0.19	15.42 ± 0.67	81.74 ± 0.21	**7.00**
GraphSAGE	40.07 ± 0.53	91.12 ± 0.22	86.04 ± 0.27	16.55 ± 0.64	81.50 ± 0.27	**7.60**
GAT	40.67 ± 0.55	92.61 ± 0.21	84.81 ± 0.24	16.75 ± 0.67	82.25 ± 0.27	**5.80**
GT	41.56 ± 0.40	91.71 ± 0.22	84.95 ± 0.19	14.03 ± 0.90	82.54 ± 0.21	**6.20**
OpenGraph (ICL)	29.36 ± 1.24	75.27 ± 5.05	70.30 ± 2.67	3.77 ± 0.65	75.66 ± 0.39	**12.60**
AnyGraph (ICL)	33.49 ± 3.44	61.17 ± 8.64	65.31 ± 6.26	4.27 ± 0.66	65.17 ± 2.51	**13.20**
TS-GNN (ICL)	43.00 ± 0.13	77.87 ± 2.73	64.41 ± 5.11	5.00 ± 0.48	46.25 ± 9.77	**12.00**
GCOPE (FT)	39.90 ± 0.43	85.08 ± 0.17	79.35 ± 0.70	6.59 ± 0.43	59.13 ± 1.20	**11.20**
SAMGPT (FT)	21.20 ± 1.90	80.61 ± 0.93	71.93 ± 1.45	8.24 ± 0.48	63.82 ± 2.35	**12.00**
MDGFM (FT)	21.37 ± 1.34	70.30 ± 3.59	67.54 ± 1.94	10.55 ± 0.50	58.71 ± 3.74	**13.00**
G2T-TabPFNv2 (ICL)	41.99 ± 0.12	91.27 ± 0.12	89.39 ± 0.15	18.46 ± 0.58	80.09 ± 0.26	**6.00**
G2T-TabPFNv2.5 (ICL)	43.90 ± 0.17	91.56 ± 0.08	89.06 ± 0.23	21.31 ± 0.12	81.68 ± 0.13	**4.40**
G2T-LimiX (ICL)	45.03 ± 0.10	91.72 ± 0.10	89.25 ± 0.14	20.97 ± 0.52	80.93 ± 0.13	**4.40**
G2T-TabPFNv2 (FT)	45.19 ± 0.25	92.53 ± 0.05	90.52 ± 0.21	18.91 ± 1.06	81.46 ± 0.29	**3.60**
G2T-LimiX (FT)	46.09 ± 0.23	92.83 ± 0.12	90.94 ± 0.09	22.98 ± 0.27	83.38 ± 0.17	**1.00**

with them. Results of G2T-FM on individual datasets are also strong. For example, on `artnet-exp`, `tolokers-2`, `artnet-views`, `city-roads-M`, and `hm-prices` G2T-LimiX surpasses all traditional baselines with more than two percentage points margin.

Observation 3. *After finetuning, G2T-FM outperforms on average all traditional baselines trained from scratch on both collections of datasets with G2T-LimiX achieving especially strong improvements.*

On datasets with tabular features, G2T-LimiX achieves not only the best average rank, but also the best results on six out of eight datasets, often yielding notable improvements of more than two percentage points compared to the best GNN result.

On datasets with text-based features, finetuned G2T-LimiX is also strong, achieving the best results on all the considered datasets, often with noticeable improvements over traditional GNNs. We note, however, that specialized models that directly process raw text may achieve higher accuracy on text-attributed graphs. Though, comparing against such models is outside the scope of this work.

Ablation Finally, we refer to Appendix B for the ablation analysis. There, we first show that removing any graph-to-table component (NFA, structural features or PEARL) negatively affects average performance. Second, we compare G2T-FM against supervised baselines with exactly the same augmented features as G2T-FM, showing that G2T-FM still outperforms them on average.

6 Conclusion

In this work, we have shown that tabular foundation models can be successfully employed for solving graph problems since they are able to process heterogeneous feature and target spaces. To show this, we proposed a simple G2T-FM framework, which converts a graph task into a tabular task by augmenting the node features with graph information, and then applies a tabular foundation model. Our empirical results show the strong performance of G2T-FM, both in the in-context learning and finetuning regimes. In particular, G2T-LimiX and G2T-TabPFNv2.5 outperform all well-tuned GNN baselines in terms of the average rank even in the in-context learning mode. After finetuning, G2T-LimiX achieves the best results on most of the datasets, often outperforming the best GNN by a significant margin. Our approach is simple, but it provides significant advantages over prior attempts at developing GFMs and shows the potential of creating truly generalizable GFMs that can achieve strong results across diverse tasks and real-world applications of graph machine learning, regardless of the inherent feature and target spaces. Importantly, G2T-FM can be combined with any tabular foundation model. Thus, future advancements in TFMs can be easily transferred to the graph domain, providing a strong baseline for future GFM studies. Despite its strong performance, the G2T-FM framework is only a first step towards utilizing models and ideas from the tabular domain for developing truly generalizable GFMs. Hence, our work has several noteworthy limitations that suggest future research directions, which we discuss in Appendix A.

AI USE STATEMENT

In this work, we used generative AI tools to aid and polish writing. We have not used generative AI tools for proposing or refining hypotheses, designing or providing feedback on research methodology or experiments, implementing methods, cleaning and reformatting datasets, supporting qualitative and thematic data analysis, and interpreting results. The following tasks with required disclosure are not applicable to our work: generating synthetic datasets, helping developing theoretical models or conceptual frameworks, formulating mathematical claims, providing critical ingredients for proving mathematical claims, assisting in the writing of proofs, assisting with translation. We have reviewed all AI-assisted work and take responsibility for the final content of this work, including text, claims and artifacts produced with the aid of AI.

REFERENCES

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Gleb Bazhenov, Oleg Platonov, and Liudmila Prokhorenkova. GraphLand: Evaluating graph machine learning models on diverse industrial data. *Advances in Neural Information Processing Systems*, 2025.

Maya Bechler-Speicher, Ben Finkelshtein, Fabrizio Frasca, Luis Müller, Jan Tönshoff, Antoine Siraudin, Viktor Zaverkin, Michael M. Bronstein, Mathias Niepert, Bryan Perozzi, Mikhail Galkin, and Christopher Morris. Position: Graph learning will lose relevance due to poor benchmarks. In *International Conference on Machine Learning*, 2025.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33:1877–1901, 2020.

Jiuhai Chen, Jonas Mueller, Vassilis N Ioannidis, Soji Adeshina, Yangkun Wang, Tom Goldstein, and David Wipf. Does your graph need a confidence boost? Convergent boosted smoothing on graphs with tabular node features. In *International Conference on Learning Representations*, 2022.

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.

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*, pp. 4171–4186, 2019.

Thomas G Dietterich and Ghulum Bakiri. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*, 2:263–286, 1994.

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.

Matthias Fey, Vid Kocijan, Federico Lopez, Jan Eric Lenssen, and Jure Leskovec. KumoRFM: A foundation model for in-context learning on relational data, 2025.

Ben Finkelshtein, İsmail İlkan Ceylan, Michael Bronstein, and Ron Levie. Equivariance everywhere all at once: A recipe for graph foundation models. *Advances in Neural Information Processing Systems*, 2025.

Léo Grinsztajn, Klemens Flöge, Oscar Key, Felix Birkel, Philipp Jund, Brendan Roof, Benjamin Jäger, Dominik Safaric, Simone Alessi, Adrian Hayler, Mihir Manium, Rosen Yu, Felix Jablonski, Shi Bin Hoo, Anurag Garg, Jake Robertson, Magnus Bühler, Vladyslav Moroshan, Lennart Purucker, Clara Cornu, Lilly Charlotte Wehrhahn, Alessandro Bonetto, Bernhard Schölkopf, Sauraj Gambhir, Noah Hollmann, and Frank Hutter. TabPFN-2.5: Advancing the state of the art in tabular foundation models. *arXiv preprint arXiv:2511.08667*, 2025.

Léo Grinsztajn, Klemens Flöge, Oscar Key, Felix Birkel, Philipp Jund, Brendan Roof, Mihir Manium, Shi Bin Hoo, Magnus Bühler, Anurag Garg, Dominik Safaric, Jake Robertson, Benjamin Jäger, Simone Alessi, Adrian Hayler, Vladyslav Moroshan, Lennart Purucker, Philipp Singer, Alan Arazi, Julien Siems, Jan Hendrik Metzen, Georg Grab, Nick Erickson, Siyuan Guo, Eliott Kalfon, Simon Bing, David Salinas, Clara Cornu, Lilly Charlotte Wehrhahn, Diana Kriuchkova, Kursat Kaya, Lydia Sidhoum, Marie Salmon, Jerry Chen, Madelon Hulsebos, Yann LeCun, Samuel Müller, Bernhard Schölkopf, Sauraj Gambhir, Noah Hollmann, and Frank Hutter. TabPFN-3: Technical report. *arXiv preprint arXiv:2605.13986*, 2026.

Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in Neural Information Processing Systems*, 30, 2017.

Adrian Hayler, Xingyue Huang, İsmail İlkan 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.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778. IEEE, 2016.

Yufei He, Yuan Sui, Xiaoxin He, and Bryan Hooi. UniGraph: Learning a unified cross-domain foundation model for text-attributed graphs. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 448–459, 2025.

Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter. TabPFN: A transformer that solves small tabular classification problems in a second. In *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*, 637(8045):319–326, 2025.

Shi Bin Hoo, Samuel Müller, David Salinas, and Frank Hutter. From tables to time: How TabPFN-v2 outperforms specialized time series forecasting models. *arXiv preprint arXiv:2501.02945*, 2025.

Sergei Ivanov and Liudmila Prokhorenkova. Boost then convolve: Gradient boosting meets graph neural networks. In *International Conference on Learning Representations*, 2021.

Charilaos Kanatsoulis, Evelyn Choi, Stefanie Jegelka, Jure Leskovec, and Alejandro Ribeiro. Learning efficient positional encodings with graph neural networks. In *International Conference on Learning Representations*, 2025.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. LightGBM: A highly efficient gradient boosting decision tree. *Advances in Neural Information Processing Systems*, 30, 2017.

Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017.

Divyansha Lachi, Mehdi Azabou, Vinam Arora, and Eva L Dyer. GraphFM: A generalist graph transformer that learns transferable representations across diverse domains. *Transactions on Machine Learning Research (TMLR)*, 2025.

Hao Liu, Jiarui Feng, Lecheng Kong, Ningyue Liang, Dacheng Tao, Yixin Chen, and Muhan Zhang. One for all: Towards training one graph model for all classification tasks. In *International Conference on Learning Representations*, 2024.

Yuankai Luo, Lei Shi, and Xiao-Ming Wu. Classic GNNs are strong baselines: Reassessing GNNs for node classification. *Advances in Neural Information Processing Systems*, 37:97650–97669, 2024.

Yuankai Luo, Lei Shi, and Xiao-Ming Wu. Can classic GNNs be strong baselines for graph-level tasks? Simple architectures meet excellence. In *International Conference on Machine Learning*, 2025.

Junwei Ma, Valentin Thomas, Rasa Hosseinzadeh, Hamidreza Kamkari, Alex Labach, Jesse C. Cresswell, Keyvan Golestan, Guangwei Yu, Maksims Volkovs, and Anthony L. Caterini. TabDPT: Scaling tabular foundation models on real data. *Advances in Neural Information Processing Systems*, 2025.

Péter Mernyei and Cătălina Cangea. Wiki-CS: A Wikipedia-based benchmark for graph neural networks. *arXiv preprint arXiv:2007.02901*, 2020.

Mikhail Mironov and Liudmila Prokhorenkova. Revisiting graph homophily measures. In *Learning on Graphs Conference*, 2024.

Andreas C Mueller, Carlo A Curino, and Raghu Ramakrishnan. MotherNet: Fast training and inference via hyper-network transformers. In *International Conference on Learning Representations*, 2025.

Mark E. J. Newman. Mixing patterns in networks. *Physical Review E*, 67(2), 2003.

Oleg Platonov, Denis Kuznedelev, Artem Babenko, and Liudmila Prokhorenkova. Characterizing graph datasets for node classification: Homophily-heterophily dichotomy and beyond. *Advances in Neural Information Processing Systems*, 36:523–548, 2023a.

Oleg Platonov, Denis Kuznedelev, Michael Diskin, Artem Babenko, and Liudmila Prokhorenkova. A critical look at the evaluation of GNNs under heterophily: Are we really making progress? In *International Conference on Learning Representations*, 2023b.

Jingang Qu, David Holzmüller, Gaël Varoquaux, and Marine Le Morvan. TabICL: A tabular foundation model for in-context learning on large data. In *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. In *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, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pp. 8748–8763, 2021.

Joshua Robinson, Rishabh Ranjan, Weihua Hu, Kexin Huang, Jiaqi Han, Alejandro Dobles, Matthias Fey, Jan Eric Lenssen, Yiwen Yuan, Zecheng Zhang, et al. RelBench: A benchmark for deep learning on relational databases. *Advances in Neural Information Processing Systems*, 37:21330–21341, 2024.

Benedek Rozemberczki and Rik Sarkar. Characteristic functions on graphs: Birds of a feather, from statistical descriptors to parametric models. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pp. 1325–1334, 2020.

Ivan Rubachev, Akim Kotelnikov, Nikolay Kartashev, and Artem Babenko. On finetuning tabular foundation models. *arXiv preprint arXiv:2506.08982*, 2025.

Yangyi Shen, Jincheng Zhou, Beatrice Bevilacqua, Joshua Robinson, Charilaos Kanatsoulis, Jure Leskovec, and Bruno Ribeiro. Zero-shot generalization of GNNs over distinct attribute domains. In *International Conference on Machine Learning*, 2025.

Yunsheng Shi, Zhengjie Huang, Shikun Feng, Hui Zhong, Wenjin Wang, and Yu Sun. Masked label prediction: Unified message passing model for semi-supervised classification. In *Proceedings of the 30th International Joint Conference on Artificial Intelligence*, 2021.

Yifei Sun, Yang Yang, Xiao Feng, Zijun Wang, Haoyang Zhong, Chunping Wang, and Lei Chen. Handling feature heterogeneity with learnable graph patches. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 1313–1324, 2025.

Boris Van Breugel and Mihaela Van Der Schaar. Why tabular foundation models should be a research priority. In *International Conference on Machine Learning*, 2024.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018.

Minjie Wang, Quan Gan, David Wipf, Zhenkun Cai, Ning Li, Jianheng Tang, Yanlin Zhang, Zizhao Zhang, Zunyao Mao, Yakun Song, et al. 4DBInfer: A 4D benchmarking toolbox for graph-centric predictive modeling on relational DBs. *Advances in Neural Information Processing Systems*, 37, 2024a.

Shuo Wang, Bokui Wang, Zhixiang Shen, Boyan Deng, and Zhao Kang. Multi-domain graph foundation models: Robust knowledge transfer via topology alignment. In *International Conference on Machine Learning*, 2025a.

Zehong Wang, Zheyuan Zhang, Nitesh Chawla, Chuxu Zhang, and Yanfang Ye. GFT: Graph foundation model with transferable tree vocabulary. *Advances in Neural Information Processing Systems*, 37:107403–107443, 2024b.

Zehong Wang, Zheyuan Liu, Tianyi Ma, Jiazheng Li, Zheyuan Zhang, Xingbo Fu, Yiyang Li, Zhengqing Yuan, Wei Song, Yijun Ma, et al. Graph foundation models: A comprehensive survey. *arXiv preprint arXiv:2505.15116*, 2025b.

Lianghao Xia and Chao Huang. AnyGraph: Graph foundation model in the wild. *arXiv preprint arXiv:2408.10700*, 2024.

Lianghao Xia, Ben Kao, and Chao Huang. OpenGraph: Towards open graph foundation models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 2024.

Zhilin Yang, William Cohen, and Ruslan Salakhudinov. Revisiting semi-supervised learning with graph embeddings. In *International Conference on Machine Learning*, pp. 40–48, 2016.

Xingtong Yu, Zechuan Gong, Chang Zhou, Yuan Fang, and Hui Zhang. SAMGPT: Text-free graph foundation model for multi-domain pre-training and cross-domain adaptation. In *Proceedings of the ACM Web Conference 2025*, pp. 1142–1153, 2025.

Xingxuan Zhang, Gang Ren, Han Yu, Hao Yuan, Hui Wang, Jiansheng Li, Jiayun Wu, Lang Mo, Li Mao, Mingchao Hao, Ningbo Dai, Renzhe Xu, Shuyang Li, Tianyang Zhang, Yue He, Yuanrui Wang, Yunjia Zhang, Zijing Xu, Dongzhe Li, Fang Gao, Hao Zou, Jiandong Liu, Jiashuo Liu, Jiawei Xu, Kaijie Cheng, Kehan Li, Linjun Zhou, Qing Li, Shaohua Fan, Xiaoyu Lin, Xinyan Han, Xuanyue Li, Yan Lu, Yuan Xue, Yuanyuan Jiang, Zimu Wang, Zhenlei Wang, and Peng Cui. LimiX: Unleashing structured-data modeling capability for generalist intelligence. *arXiv preprint arXiv:2509.03505*, 2025a.

Xiyuan Zhang, Danielle C. Maddix, Junming Yin, Nick Erickson, Abdul Fatir Ansari, Boran Han, Shuai Zhang, Leman Akoglu, Christos Faloutsos, Michael Mahoney, Tony Hu, Huzefa Rangwala, George Karypis, and Bernie Wang. Mitra: Mixed synthetic priors for enhancing tabular foundation models. *Advances in Neural Information Processing Systems*, 2025b.

Haihong Zhao, Aochuan Chen, Xiangguo Sun, Hong Cheng, and Jia Li. All in one and one for all: A simple yet effective method towards cross-domain graph pretraining. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 4443–4454, 2024a.

Jitao Zhao, Di Jin, Meng Ge, Lianze Shan, Xin Wang, Dongxiao He, and Zhiyong Feng. FUG: Feature-universal graph contrastive pre-training for graphs with diverse node features. *Advances in Neural Information Processing Systems*, 37:4003–4034, 2024b.

A LIMITATIONS AND FUTURE WORK

Despite its strong performance, the proposed G2T-FM framework is only a first step towards utilizing models and ideas from the tabular domain for developing truly generalizable GFMs. Hence, our work has several noteworthy limitations that suggest future research directions.

First, the present version of G2T-FM uses only basic methods for processing graph structures. Future research can bring more graph-specific components into the framework, such as more complex aggregation mechanisms (including learnable and multi-hop aggregations) and cross-graph pretraining. We believe these extensions could enable the model to better capture graph-specific information and transfer knowledge across different graphs. We also note that our preliminary experiments with multi-hop NFA did not provide consistent improvements.

Second, G2T-FM inherits several restrictions from its TFM backbone. In particular, when endowed with TabPFNv2, G2T-FM cannot handle classification datasets with more than 10 classes, and its training set size is limited to at most 10,000 samples. These limitations currently make it difficult to apply G2T-FM to large-scale datasets. However, future research on tabular foundation models can alleviate these limitations and new TFMs can be directly used within the G2T-FM framework.

Finally, the scope of our work is limited to node-level prediction tasks. Potentially, G2T-FM can be applied to link-level tasks, but its performance on such problems requires further analysis. Regarding graph-level tasks, adapting G2T-FM is less straightforward.

B ABLATIONS

G2T-FM components First, we provide an ablation of the G2T-FM components by removing them from G2T-FM and comparing performance. Table 4 shows the results of this ablation, from which we conclude that all the components are critical for the performance of G2T-FM. In particular, neighborhood feature aggregation (NFA) and classic structure-based features (SF) improve the overall performance of G2T-FM, while PEARL allows one to drastically improve performance in rare cases where standard augmented features are not sufficient.

Augmenting baselines with the same components Second, one may argue that the performance improvements of G2T-FM come solely from the fact that it employs augmented features that are not accessible to the GNN and LightGBM baselines. To verify this, we provide the baselines with exactly the same features as G2T-FM. The results are presented in Table 5. While some improvements achieved by G2T-FM can be explained by its access to the features that are not used by traditional GNNs, G2T-FM shows strong performance even against the enhanced baselines. In particular, G2T-LimiX achieves better average rank, and on some datasets it outperforms all other methods by a notable margin.

Evaluation without ensembling The default evaluation setup of G2T-FM uses an ensemble over 10 forward passes. For completeness, Table 5 reports the results obtained without ensembling. Overall, ensembling yields small but consistent gains, with larger improvements in certain cases, e.g., G2T-TabPFNv2.5 (ICL) on `artnet-exp` and `hm-prices` or G2T-LimiX (FT) on `artnet-views`. Nevertheless, G2T-FM remains competitive without ensembling, still outperforming traditional supervised baselines on average even in an in-context learning regime.

Summary The gains of G2T-FM come from the synergy between the TFM backbone and our graph-to-table components. The ablations show that removing any component degrades the performance on average, and providing the baselines with the same augmented features does not close the gap. Hence, both the backbone and the proposed components are necessary for the strong performance.

C IMPLEMENTATION DETAILS

In this section, we describe additional implementation details. Further information and the full code are available in our repository: https://anonymous.4open.science/r/8456b37f41900b.

Table 4: Ablation of the components of G2T-FM. 'SF' stands for Structure-based Features, which include degree, PageRank, and Laplacian eigenvectors.

	artnet-exp	city-reviews	tolokers-2	artnet-views	avazu-ctr	city-roads-M	hm-prices	twitch-views	**AR**
LightGBM-NFA	46.13 ± 0.04	78.53 ± 0.01	56.34 ± 0.06	56.10 ± 0.02	31.71 ± 0.01	61.18 ± 0.03	70.84 ± 0.04	60.14 ± 0.01	**12.25**
GCN	44.86 ± 0.36	77.81 ± 0.15	56.27 ± 0.31	56.03 ± 0.25	32.00 ± 0.16	58.82 ± 0.25	68.02 ± 0.42	75.51 ± 0.05	**12.12**
GraphSAGE	45.14 ± 0.36	78.17 ± 0.10	54.43 ± 0.34	49.32 ± 0.91	31.44 ± 0.16	59.44 ± 0.27	70.00 ± 0.74	66.29 ± 0.32	**13.50**
GAT	45.06 ± 0.52	77.74 ± 0.21	57.41 ± 0.85	53.60 ± 0.24	32.63 ± 0.17	59.86 ± 0.20	72.07 ± 1.22	72.89 ± 0.27	**11.00**
GT	46.41 ± 0.71	77.34 ± 0.21	56.98 ± 0.55	53.37 ± 0.46	31.11 ± 0.49	59.55 ± 0.28	69.44 ± 0.94	72.13 ± 0.13	**12.50**
G2T-LimiX (ICL)	48.42 ± 0.78	78.98 ± 0.44	61.60 ± 0.18	61.58 ± 0.08	32.70 ± 0.14	65.16 ± 0.07	76.14 ± 0.08	71.31 ± 0.06	**6.12**
w/o NFA (ICL)	47.17 ± 0.11	79.18 ± 0.14	61.30 ± 0.20	61.24 ± 0.07	32.14 ± 0.11	63.11 ± 0.10	71.27 ± 0.18	70.63 ± 0.06	**8.38**
w/o SF & PEARL (ICL)	48.33 ± 0.50	79.24 ± 0.28	58.57 ± 0.14	58.44 ± 0.06	32.87 ± 0.08	65.64 ± 0.08	76.24 ± 0.11	60.77 ± 0.04	**7.88**
w/o SF (ICL)	48.26 ± 0.60	79.24 ± 0.20	58.29 ± 0.12	59.22 ± 0.05	33.14 ± 0.07	65.99 ± 0.07	76.03 ± 0.11	68.64 ± 0.02	**7.38**
w/o PEARL (ICL)	48.40 ± 0.77	78.87 ± 0.52	61.72 ± 0.21	61.34 ± 0.09	32.58 ± 0.15	64.85 ± 0.07	76.33 ± 0.08	66.58 ± 0.05	**7.00**
G2T-LimiX (FT)	50.39 ± 0.19	80.65 ± 0.05	59.75 ± 1.14	63.24 ± 0.07	34.09 ± 0.37	66.29 ± 0.12	77.37 ± 0.17	74.91 ± 0.06	**2.75**
w/o NFA (FT)	47.24 ± 0.38	80.02 ± 0.06	60.52 ± 0.67	62.53 ± 0.09	33.84 ± 0.23	64.18 ± 0.29	73.28 ± 0.34	74.36 ± 0.08	**6.00**
w/o SF & PEARL (FT)	49.88 ± 0.16	80.26 ± 0.10	58.16 ± 0.65	60.51 ± 0.08	34.90 ± 0.23	66.25 ± 0.09	77.54 ± 0.26	62.40 ± 0.06	**5.62**
w/o SF (FT)	50.01 ± 0.14	80.48 ± 0.03	58.05 ± 0.49	60.84 ± 0.14	34.23 ± 0.48	66.38 ± 0.10	77.29 ± 0.41	73.24 ± 0.16	**4.38**
w/o PEARL (FT)	50.13 ± 0.14	80.66 ± 0.07	60.62 ± 0.90	63.25 ± 0.07	34.92 ± 0.17	66.29 ± 0.13	77.38 ± 0.32	67.92 ± 0.05	**2.88**

Table 5: Extended evaluation of G2T-FM. 'M' stands for Modified and means that we add NFA, classic structure-based features, and PEARL encodings to their features. 'L' (Light) indicates that G2T-FM was evaluated without ensembling.

	artnet-exp	city-reviews	tolokers-2	artnet-views	avazu-ctr	city-roads-M	hm-prices	twitch-views	**AR**
LightGBM-NFA	46.13 ± 0.04	78.53 ± 0.01	56.34 ± 0.06	56.10 ± 0.02	31.71 ± 0.01	61.18 ± 0.03	70.84 ± 0.04	60.14 ± 0.01	**13.12**
GCN	44.86 ± 0.36	77.81 ± 0.15	56.27 ± 0.31	56.03 ± 0.25	32.00 ± 0.16	58.82 ± 0.25	68.02 ± 0.42	75.51 ± 0.05	**14.38**
GraphSAGE	45.14 ± 0.36	78.17 ± 0.10	54.43 ± 0.34	49.32 ± 0.91	31.44 ± 0.16	59.44 ± 0.27	70.00 ± 0.74	66.29 ± 0.32	**16.25**
GAT	45.06 ± 0.52	77.74 ± 0.21	57.41 ± 0.85	53.60 ± 0.24	32.63 ± 0.17	59.86 ± 0.20	72.07 ± 1.22	72.89 ± 0.27	**12.75**
GT	46.41 ± 0.71	77.34 ± 0.21	56.98 ± 0.55	53.37 ± 0.46	31.11 ± 0.49	59.55 ± 0.28	69.44 ± 0.94	72.13 ± 0.13	**15.12**
LightGBM-NFA (M)	45.57 ± 0.20	78.68 ± 0.05	57.16 ± 0.74	57.53 ± 0.05	31.31 ± 0.09	60.86 ± 0.17	70.25 ± 0.15	65.17 ± 0.04	**13.38**
GCN (M)	43.44 ± 0.34	77.07 ± 0.28	58.71 ± 0.48	56.14 ± 0.25	31.10 ± 0.23	57.91 ± 0.24	70.73 ± 0.28	77.11 ± 0.10	**14.25**
GraphSAGE (M)	44.31 ± 0.56	77.95 ± 0.10	59.59 ± 0.54	55.39 ± 0.34	31.51 ± 0.43	59.66 ± 0.10	70.50 ± 0.50	75.93 ± 0.20	**12.62**
GAT (M)	44.36 ± 0.53	77.47 ± 0.15	57.76 ± 0.74	56.51 ± 0.37	31.97 ± 0.24	59.57 ± 0.45	72.46 ± 0.51	77.04 ± 0.08	**11.75**
GT (M)	43.03 ± 0.63	76.43 ± 0.11	58.79 ± 0.80	56.39 ± 0.33	29.86 ± 0.71	59.85 ± 0.43	71.84 ± 0.68	76.15 ± 0.12	**13.62**
G2T-TabPFNv2 (ICL, L)	45.69 ± 0.05	77.19 ± 0.26	60.25 ± 0.21	59.69 ± 0.18	28.06 ± 0.19	60.09 ± 0.10	65.74 ± 0.04	69.91 ± 0.10	**14.38**
G2T-TabPFNv2.5 (ICL, L)	47.89 ± 0.08	79.34 ± 0.04	60.79 ± 0.32	60.01 ± 0.10	32.69 ± 0.07	62.65 ± 0.24	72.59 ± 0.20	70.78 ± 0.03	**7.38**
G2T-LimiX (ICL, L)	48.44 ± 0.23	77.29 ± 0.54	61.13 ± 0.21	60.95 ± 0.09	32.41 ± 0.12	64.53 ± 0.09	75.41 ± 0.04	71.08 ± 0.07	**7.25**
G2T-TabPFNv2 (ICL)	45.73 ± 0.03	77.28 ± 0.25	60.39 ± 0.19	59.72 ± 0.14	28.00 ± 0.36	60.12 ± 0.03	65.75 ± 0.02	70.21 ± 0.04	**13.62**
G2T-TabPFNv2.5 (ICL)	49.33 ± 0.12	79.99 ± 0.04	61.21 ± 0.12	60.64 ± 0.09	33.06 ± 0.11	63.53 ± 0.07	74.32 ± 0.09	71.64 ± 0.06	**4.88**
G2T-LimiX (ICL)	48.42 ± 0.78	78.98 ± 0.44	61.60 ± 0.18	61.58 ± 0.08	32.70 ± 0.14	65.16 ± 0.07	76.14 ± 0.08	71.31 ± 0.06	**4.88**
G2T-TabPFNv2 (FT, L)	46.68 ± 0.71	78.82 ± 0.14	57.85 ± 1.30	60.59 ± 0.12	32.77 ± 0.54	63.02 ± 0.38	72.50 ± 0.72	74.25 ± 0.15	**7.62**
G2T-LimiX (FT, L)	49.79 ± 0.21	80.13 ± 0.13	60.82 ± 0.57	62.08 ± 0.12	34.03 ± 0.35	65.87 ± 0.11	76.72 ± 0.20	74.31 ± 0.13	**2.88**
G2T-TabPFNv2 (FT)	47.53 ± 0.62	78.69 ± 0.30	57.54 ± 0.93	60.60 ± 0.16	33.38 ± 0.41	63.05 ± 0.39	72.58 ± 0.40	74.24 ± 0.11	**7.38**
G2T-LimiX (FT)	50.39 ± 0.19	80.65 ± 0.05	59.75 ± 1.14	63.24 ± 0.07	34.09 ± 0.37	66.29 ± 0.12	77.37 ± 0.17	74.91 ± 0.06	**2.50**

C.1 G2T-FM

Finetuning For the finetuning experiments, we follow the procedure outlined by Rubachev et al. (2025). Rather than using parameter-efficient finetuning, we opt for full model finetuning, as previous work indicates this yields better performance. We search for the optimal learning rate over the logarithmic grid of 10 values, ranging from 5×10^{-6} to 5×10^{-4}.

Ensembling By default, we evaluate G2T-FM using an ensemble of 10 members. Ensemble members share the same model weights, but apply possibly different preprocessing schemes to the input features (e.g., column permutation) and targets (e.g., label shuffling for multiclass classification tasks). We employ ensembling in both in-context learning and finetuning settings. For each optimization step during finetuning, we sample a single random ensemble member (i.e., one preprocessing scheme) to compute the gradient. This strategy yields the advantages of ensembling while limiting computational costs. Specifically, because intermediate evaluations during finetuning occur only once every 10 gradient steps, evaluating the full ensemble requires just 9 additional forward passes for every 10 forward-backward passes. Therefore, the maximum time overhead of our ensembling approach with this strategy is at most double that of standard single-model finetuning.

PCA On certain datasets, applying G2T-FM directly on the original features results in out-of-memory errors. To address this, we apply principal component analysis (PCA) on `avazu-ctr`, `city-reviews`, `amazon-ratings` and `questions` datasets to reduce the feature dimensionality. PCA is performed separately on the original features and the neighborhood feature aggregations, reducing each to 64 dimensions.

Table 6: Comparison of downstream performance between G2T-FM and TAG.

	amazon-ratings	artnet-exp	city-reviews	facebook	pubmed	questions	tolokers-2	wiki-cs
LightGBM-NFA	43.05 ± 0.11	46.13 ± 0.04	78.53 ± 0.01	88.48 ± 0.07	89.31 ± 0.05	14.85 ± 0.23	56.34 ± 0.06	80.43 ± 0.06
GCN	41.43 ± 0.49	44.86 ± 0.36	77.81 ± 0.15	91.26 ± 0.21	85.46 ± 0.19	15.42 ± 0.67	56.27 ± 0.31	81.74 ± 0.21
GraphSAGE	40.07 ± 0.53	45.14 ± 0.36	78.17 ± 0.10	91.12 ± 0.22	86.04 ± 0.27	16.55 ± 0.64	54.43 ± 0.34	81.50 ± 0.27
GAT	40.67 ± 0.55	45.06 ± 0.52	77.74 ± 0.21	92.61 ± 0.21	84.81 ± 0.24	16.75 ± 0.67	57.41 ± 0.85	82.25 ± 0.27
GT	41.56 ± 0.40	46.41 ± 0.71	77.34 ± 0.21	91.71 ± 0.22	84.95 ± 0.19	14.03 ± 0.90	56.98 ± 0.55	82.54 ± 0.21
TAG-TabPFNv2 (ICL)	43.97 ± 0.52	47.87 ± 0.39	77.38 ± 0.29	93.11 ± 0.17	87.80 ± 0.20	15.07 ± 1.32	59.33 ± 1.00	82.59 ± 0.12
G2T-TabPFNv2 (ICL)	41.99 ± 0.12	45.73 ± 0.03	77.28 ± 0.25	91.27 ± 0.12	89.39 ± 0.15	18.46 ± 0.58	60.39 ± 0.19	80.09 ± 0.26
TAG-LimiX (ICL)	44.89 ± 0.15	50.19 ± 0.46	78.87 ± 0.13	92.85 ± 0.14	88.09 ± 0.18	15.53 ± 1.99	59.46 ± 1.05	82.64 ± 0.17
G2T-LimiX (ICL)	45.03 ± 0.10	48.42 ± 0.78	78.98 ± 0.44	91.72 ± 0.10	89.25 ± 0.14	20.97 ± 0.52	61.60 ± 0.18	80.93 ± 0.13

Table 7: Runtime in seconds per single seed.

	amazon-ratings	artnet-exp	city-reviews	facebook	pubmed	questions	tolokers-2	wiki-cs
G2T-TabPFNv2 (ICL)	169.04	153.41	150.14	56.75	236.52	481.67	9.38	57.66
TAG-TabPFNv2 (ICL)	736.37	1,407.83	2,308.25	656.29	568.63	1,414.72	150.83	291.5
G2T-LimiX (ICL)	116.28	103.24	279.41	48.82	151.02	317.75	18.00	60.05
TAG-LimiX (ICL)	825.71	1,532.35	2,479.54	686.38	600.54	1,539.22	150.04	310.86

PEARL and SF For Laplacian positional encodings (LapPE), we set the dimension to $k = 14$ so that the total number of structure-based features (specifically, LapPE, degree, and PageRank) is 16. For the GNN backbone within PEARL, we use a 3-layer GCN with GELU activations, hidden dimension 512 and output dimension 16. However, we do not use layer normalization and residual connections, based on preliminary experiments that showed improved results without these components. In PEARL, we average GNN embeddings over 8 random inputs.

For the in-context learning experiments, we utilize a randomly initialized PEARL model whose weights are shared across all datasets. Interestingly, even without explicit training, this untrained PEARL model still produces useful representations for some datasets, as demonstrated in our ablation studies (Appendix B). For the finetuning experiments, we jointly finetune PEARL and the TFM backbone.

C.2 GNNs and LightGBM

GNNs Our GNN setup closely follows the architecture and hyperparameter optimization procedure from GraphLand (Bazhenov et al., 2025), with two main differences. First, we introduce early stopping with a patience of 100 steps to accelerate training. Second, we use unified hyperparameter search spaces for both feature and target preprocessing, rather than dataset-specific spaces used in GraphLand. See our code for more details. Note that this latter modification only affects the preprocessing hyperparameters, while the search grids for learning rate and dropout remain identical to those in GraphLand.

PEARL integration In the ablation studies described in Appendix B, we evaluate the effect of integrating PEARL with both GNN and LightGBM models. For GNNs, we concatenate PEARL outputs with the initial node features, and train the combined model end-to-end. For LightGBM, due to the challenge of end-to-end training with PEARL, we use the outputs from the same randomly initialized PEARL as in our G2T-FM (ICL) experiments.

D Concurrent and follow-up works

Shortly after the release of this manuscript, several independent works explored applying TFMs and PFNs to graph node-level tasks. The closest concurrent methods are TAG (Hayler et al., 2025) and TabPFN-GN (Choi et al., 2025). Like G2T-FM, they augment original features with hand-crafted graph-aware features (e.g., neighborhood aggregations or node degrees) and subsequently apply a TFM. However, their specific feature sets and implementations differ. For completeness, we discuss the key differences and provide an empirical comparison with TAG.[5]

[5]We do not compare with TabPFN-GN because, at the time of writing, its code has not been publicly released.

Key differences Our work differs from Hayler et al. (2025); Choi et al. (2025) in the following key ways. First, we focus on real-world datasets from the recently introduced GraphLand benchmark (Bazhenov et al., 2025) which contains a variety of classification and regression tasks covering more realistic problems compared to classic graph benchmarks that have been recently criticized in Bechler-Speicher et al. (2025). Second, we evaluate TFMs in both in-context learning and finetuning regimes, and we show that finetuning substantially improves performance. Finally, we use well-tuned GNNs with hyperparameter tuning and architectural enhancements that serve as strong baselines for GFMs.

Evaluation We evaluate TAG in our setup (i.e., on datasets and splits used in our main experiments) using the official implementation of TAG provided by the authors. The only change we make is to replace the accuracy metric with average precision during ensemble selection and evaluation on binary classification datasets, because average precision is the metric used throughout our study. We restrict the comparison to classification datasets because the current TAG implementation does not support regression tasks at the time of writing. Table 6 reports the results. Overall, the two methods achieve comparable performance. However, our preliminary runtime benchmarks in Table 7 indicate that G2T-FM is typically 4-10 times faster.

Follow-up work Other relevant works are GraphPFN (Eremeev et al., 2026) and NodePFN (Choi et al., 2026). Building on the ideas and results of G2T-FM, GraphPFN replaces hand-crafted features with learnable message-passing layers embedded into the LimiX TFM. These layers are pretrained on synthetic datasets using the PFN framework. NodePFN adopts a similar approach but pretrains the entire model from scratch rather than initializing from an existing TFM.

Importantly, while methods such as GraphPFN and NodePFN may yield more expressive graph-specific models, they lack G2T-FM's plug-and-play capability to easily incorporate any off-the-shelf TFM without additional pretraining. Given the rapid pace of TFM development in terms of both downstream quality and scalability (Grinsztajn et al., 2026; Qu et al., 2026), we believe this plug-and-play property is an important advantage of G2T-FM, allowing it to remain a strong and up-to-date baseline despite the presence of more advanced and complex approaches.