Turning Tabular Foundation Models into Graph Foundation Models

Dmitry Eremeev
HSE University, Yandex Research
`eremeev-d@yandex-team.ru`

Gleb Bazhenov
HSE University, Yandex Research
`gv-bazhenov@yandex-team.ru`

Oleg Platonov
HSE University, Yandex Research
`olegplatonov@yandex-team.ru`

Artem Babenko
Yandex Research, HSE University
`arbabenko@yandex-team.ru`

Liudmila Prokhorenkova
Yandex Research
`ostroumova-la@yandex-team.ru`

Abstract

While foundation models have revolutionized such fields 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 or LimiX, we propose G2T-FM, a simple framework for turning tabular foundation models into graph foundation models. 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 a fully in-context regime, our model achieves strong results, significantly outperforming publicly available GFMs and performing competitively with, and often better than, well-tuned GNNs trained from scratch. Moreover, after finetuning, G2T-FM surpasses well-tuned GNN baselines. In particular, when combined with LimiX, G2T-FM often outperforms the best GNN by a significant margin. In summary, our paper reveals the potential of a previously overlooked direction of utilizing tabular foundation models for graph machine learning tasks.[1]

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. Despite their

[1]Our source code is available at `https://github.com/yandex-research/G2T-FM`.

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: New Perspectives in Advancing Graph Machine Learning.

remarkable success in such areas as CV and NLP, the development of foundation models for graph data has been less advanced. 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 representing very different objects with nodes and very different relations with edges, 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 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 fully and effectively leveraging 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 face similar issues to GFMs. While tabular foundation models 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., 2024; Qu et al., 2025). For instance, TabPFNv2 (Hollmann et al., 2025) demonstrates strong performance in both in-context and finetuning regimes, and it has recently gained significant attention from the community. For a more detailed background on graph and tabular foundation models, please refer to Appendix B.

This parallel suggests that developers of GFMs can draw inspiration from tabular foundation models, as they 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., 2025), can be effectively adapted to graph datasets. We introduce a simple framework named Graph-to-Table Foundation Model (G2T-FM), which transforms graph 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 regime, significantly outperforming existing publicly available GFMs and performing competitively with, and often better than, 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.

Our main contributions are as follows:

- We identify a promising and previously overlooked direction of applying tabular foundation models to graph machine learning.
- As a proof of concept, we introduce G2T-FM, a simple framework that uses a tabular foundation model (TabPFNv2 and LimiX in our experiments) as the backbone of a graph foundation model.
- We show that, despite its simplicity, G2T-FM is a strong baseline for GFMs, substantially outperforming the existing publicly available GFMs.
- We further demonstrate that finetuned G2T-FM surpasses traditional GNNs trained from scratch.

More broadly, we hope that our work will promote the adoption of models and ideas from the tabular domain in developing generalizable and robust graph foundation models.

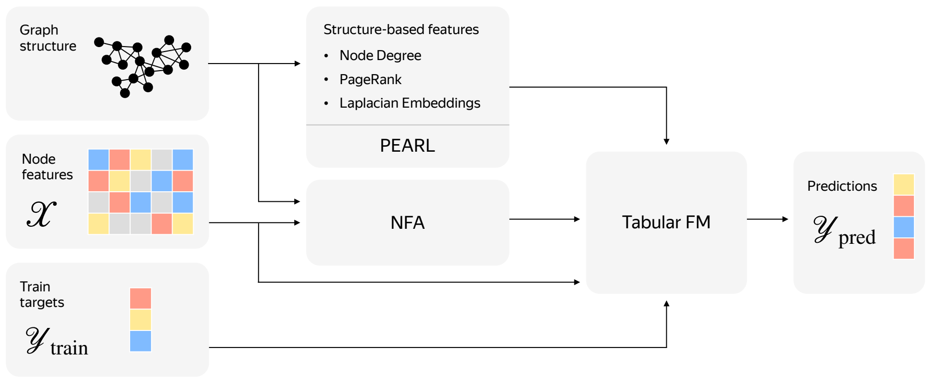

Figure 1: Overview of the proposed G2T-FM framework.

2 Graph-to-Table Foundation Model

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 (Liu et al., 2024; Wang et al., 2024b; He et al., 2025). This method is highly effective for text-attributed graphs (TAGs), where the features are naturally represented as text. Nevertheless, many real-world graphs do not include solely text 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.

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 better graph foundation models, and, in particular, they can help handle different feature and target spaces. As a proof of concept, we introduce 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) or LimiX (Zhang et al., 2025). 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 neighbors. Namely, for each numerical feature, we compute its mean, maximum, and minimum values over the node's neighbors. For each categorical

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	N/A	no	tabular
avazu-ctr	76,269	10,984,077	260	288.0	N/A	no	tabular
city-roads-M	57,073	107,104	26	3.8	N/A	yes	tabular
twitch-views	168,114	6,797,557	4	80.9	N/A	no	tabular
artnet-views	50,405	280,348	50	11.1	N/A	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

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: the node degree captures strictly local information, while PageRank also captures global information. Then, we compute the first K eigenvectors of the graph 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 in Kanatsoulis et al. (2025). The basic 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 TFM. The resulting model can be applied in a fully in-context regime and, as we show below, it performs on par with GNNs that are well-tuned for a single dataset. In the finetuning regime, we jointly tune PEARL and TFM.

As a final remark, we discuss the symmetries of G2T-FM. Following Finkelshtein et al. (2025), a graph foundation model should satisfy three natural inductive biases: (i) feature permutation invariance; (ii) label permutation equivariance; and (iii) node permutation equivariance. In Appendix E, we show that G2T-TabPFNv2, namely, G2T-FM with TabPFNv2 backbone, satisfies these symmetries in distribution: its outputs depend on internal randomness, but the distribution of the outputs remains invariant or equivariant under the corresponding permutations.

3 Experimental setup

3.1 Datasets

In our experiments, we use TabPFNv2 as one of the backbones, which imposes specific constraints that direct our dataset selection. In particular, TabPFNv2 is suitable for both regression tasks and multi-class classification tasks, but the latter is 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 future developments in tabular foundation models may relax these constraints and enable experiments with a broader range of graph benchmarks.

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 homophilious and non-homophilous structure,[2] 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.[3] 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`.

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 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 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.

3.2 Methods

In our experiments with G2T-FM, we adopt TabPFNv2 (Hollmann et al., 2025) and LimiX (Zhang et al., 2025) as backbone models. We refer to these models as G2T-TabPFNv2 and G2T-LimiX respectively. 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).[4] 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

[2]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.

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

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

Table 2: Evaluation results on datasets with tabular features (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.

	tolokers-2	city-reviews	artnet-exp	hm-prices	avazu-ctr	city-roads-M	twitch-views	artnet-views	**AR**
LightGBM+NFA	56.34 ± 0.06	78.53 ± 0.01	46.13 ± 0.03	70.84 ± 0.04	31.71 ± 0.01	61.18 ± 0.03	60.14 ± 0.01	56.10 ± 0.02	**5.38**
GCN	56.27 ± 0.29	77.81 ± 0.14	44.86 ± 0.34	68.02 ± 0.40	32.00 ± 0.15	58.82 ± 0.24	75.51 ± 0.05	56.03 ± 0.24	**6.25**
GraphSAGE	54.43 ± 0.32	78.17 ± 0.09	45.14 ± 0.34	70.00 ± 0.70	31.44 ± 0.15	59.44 ± 0.26	66.29 ± 0.31	49.32 ± 0.86	**7.12**
GAT	57.41 ± 0.80	77.74 ± 0.20	45.06 ± 0.49	72.07 ± 1.16	32.63 ± 0.16	59.86 ± 0.19	72.89 ± 0.25	53.60 ± 0.23	**5.12**
GT	56.98 ± 0.53	77.34 ± 0.20	46.41 ± 0.68	69.44 ± 0.89	31.11 ± 0.47	59.55 ± 0.27	72.13 ± 0.13	53.37 ± 0.43	**6.62**
OpenGraph (ICL)	40.38 ± 1.13	59.09 ± 0.72	15.16 ± 0.83	N/A	N/A	N/A	N/A	N/A	**11.00**
AnyGraph (ICL)	28.75 ± 3.56	63.71 ± 1.45	12.84 ± 0.93	N/A	N/A	N/A	N/A	N/A	**12.33**
TS-GNN (ICL)	38.54 ± 0.94	43.46 ± 5.17	20.44 ± 1.05	N/A	N/A	N/A	N/A	N/A	**11.33**
GCOPE (FT)	28.81 ± 1.28	67.16 ± 0.98	14.92 ± 1.56	N/A	N/A	N/A	N/A	N/A	**11.33**
G2T-TabPFNv2 (ICL)	60.42 ± 0.27	77.46 ± 0.10	45.84 ± 0.03	66.68 ± 0.09	26.38 ± 0.07	60.47 ± 0.04	70.00 ± 0.06	58.75 ± 0.15	**6.38**
G2T-TabPFNv2 (FT)	57.65 ± 1.92	79.12 ± 0.21	47.31 ± 0.59	71.05 ± 0.91	28.52 ± 0.43	63.08 ± 0.28	74.06 ± 0.16	60.29 ± 0.13	**3.62**
G2T-LimiX (ICL)	61.48 ± 0.30	77.72 ± 0.54	48.43 ± 0.18	74.96 ± 0.06	32.39 ± 0.14	64.53 ± 0.07	71.08 ± 0.07	60.95 ± 0.10	**3.12**
G2T-LimiX (FT)	61.17 ± 0.49	80.13 ± 0.05	49.88 ± 0.13	76.32 ± 0.17	33.94 ± 0.34	65.87 ± 0.10	73.16 ± 0.40	62.12 ± 0.10	**1.38**

Table 3: Evaluation results on datasets with text-based features. For each column, we highlight first, second, and third best results with a color.

	pubmed	facebook	amazon-r.	questions	wiki-cs	**AR**
GCN	85.46 ± 0.18	91.26 ± 0.19	41.43 ± 0.46	15.42 ± 0.63	81.74 ± 0.20	**5.60**
GraphSAGE	86.04 ± 0.26	91.12 ± 0.21	40.07 ± 0.50	16.55 ± 0.61	81.50 ± 0.26	**6.00**
GAT	84.81 ± 0.22	92.61 ± 0.20	40.67 ± 0.53	16.75 ± 0.63	82.25 ± 0.26	**4.20**
GT	84.95 ± 0.18	91.71 ± 0.21	41.56 ± 0.38	14.03 ± 0.86	82.54 ± 0.20	**5.00**
OpenGraph (ICL)	70.30 ± 2.67	75.27 ± 5.05	29.36 ± 1.24	3.77 ± 0.65	75.66 ± 0.39	**10.80**
AnyGraph (ICL)	65.31 ± 6.26	61.17 ± 8.64	33.49 ± 3.44	4.27 ± 0.66	65.17 ± 2.51	**11.00**
TS-GNN (ICL)	64.41 ± 5.11	77.87 ± 2.73	43.00 ± 0.13	5.00 ± 0.48	46.25 ± 9.77	**9.60**
GCOPE (FT)	79.35 ± 0.70	85.08 ± 0.17	39.90 ± 0.43	6.59 ± 0.43	59.13 ± 1.20	**9.60**
G2T-TabPFNv2 (ICL)	88.80 ± 0.25	90.56 ± 0.12	40.63 ± 0.19	16.49 ± 0.16	76.61 ± 0.57	**6.60**
G2T-TabPFNv2 (FT)	90.46 ± 0.11	91.73 ± 0.28	44.71 ± 0.32	19.07 ± 0.53	79.70 ± 0.31	**3.00**
G2T-LimiX (ICL)	88.96 ± 0.18	91.29 ± 0.14	44.10 ± 0.16	15.31 ± 0.77	79.99 ± 0.28	**4.80**
G2T-LimiX (FT)	89.91 ± 0.48	92.16 ± 0.18	45.67 ± 0.35	20.19 ± 0.30	82.24 ± 0.31	**1.80**

(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), which is used in the finetuning (FT) regime. For all these methods, we use the original implementations provided by the authors. Further, we were only able to evaluate these methods on datasets with node classification tasks, as none of them support node regression.[5]

For all the methods, we run experiments 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.

4 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

[5]While TS-GNN in theory supports node regression, its current official implementation, to the best of our knowledge, does not allow running experiments for regression tasks.

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 different datasets and larger 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 *On datasets with tabular features, G2T-FM evaluated in the in-context learning mode performs competitively with, and often better than, traditional GNNs trained from scratch.*

In particular, in terms of the average rank on datasets with tabular features, G2T-LimiX outperforms all baselines trained from scratch, while G2T-TabPFNv2 performs on par with them. Results of G2T-FM on individual datasets are also strong. For example, on `tolokers-2`, `artnet-exp`, `hm-prices`, `city-roads-M`, and `artnet-views`, G2T-LimiX surpasses all traditional baselines, often by more than two percentage points.

At the same time, in-context performance of both G2T-TabPFNv2 and G2T-LimiX on datasets with text-based features is less impressive. In particular, G2T-TabPFNv2 is outperformed by all traditional GNNs. We attribute this to TFMs being trained on tabular data: non-tabular features appear to be out-of-domain, and good performance therefore requires adaptation via finetuning.

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-FM is also strong. In particular, G2T-LimiX achieves the best average rank and often brings noticeable improvements over traditional GNNs. We note, however, that specialized models that process raw text more directly may achieve higher accuracy on text-attributed graphs. Though, comparing against such models is outside the scope of this work.

Finally, we refer to Appendix D for the ablation analysis. Our results show that the gains of G2T-FM come from the synergy between the TFM backbone and our graph-to-table components.

5 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 initial features with graph information, and then applies a tabular foundation model. Our empirical results show the strong performance of G2T-FM. In particular, G2T-LimiX outperforms all well-tuned GNN baselines in terms of the average rank even in the in-context learning regime. 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 shows the potential of creating truly generalizable GFMs that can achieve strong results across diverse tasks and real-world applications of graph machine learning. Importantly, G2T-FM can be combined with any tabular foundation model. Thus, future advancements in TFMs can be easily transferred to the graph ML domain. Despite its strong performance, G2T-FM framework is only a first step towards utilizing models and ideas from the tabular domain for developing GFMs. Hence, our work has several noteworthy limitations that we discuss in Appendix A.

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.

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.

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.

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*, pp. 4171–4186, 2019.

Matthias Fey and Jan E. Lenssen. Fast graph representation learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019.

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. *arXiv preprint arXiv:2506.14291*, 2025.

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

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, 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. *International Conference on Learning Representations (ICLR)*, 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 *The Thirteenth 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. *International Conference on Learning Representations (ICLR)*, 2017.

Divyansha Lachi, Mehdi Azabou, Vinam Arora, and Eva Dyer. GraphFM: A scalable framework for multi-graph pretraining. *arXiv preprint arXiv:2407.11907*, 2024.

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 *The Twelfth 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, 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*. PMLR, 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. *arXiv preprint arXiv:2410.18164*, 2024.

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 *The Third Learning on Graphs Conference*, 2024.

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

Mark EJ 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. In *Advances in Neural Information Processing Systems*, volume 36, pp. 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.

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, 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 *Forty-second International Conference on Machine Learning*, 2025.

Yunsheng Shi, Zhengjie Huang, Shikun Feng, Hui Zhong, Wenjing Wang, and Yu Sun. Masked label prediction: Unified message passing model for semi-supervised classification. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, pp. 1548–1554, 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*. PMLR, 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 *Forty-second 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 *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 2365–2379, 2024.

Zhilin Yang, William Cohen, and Ruslan Salakhudinov. Revisiting semi-supervised learning with graph embeddings. In *International Conference on Machine Learning*, pp. 40–48. PMLR, 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 on 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*, 2025.

Xiyuan Zhang and Danielle Maddix Robinson. Mitra: Mixed synthetic priors for enhancing tabular foundation models. `https://www.amazon.science/blog/mitra-mixed-synthetic-priors-for-enhancing-tabular-foundation-models`, 2025.

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.

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 the new TFMs can be directly used within the G2T-FM framework.

B Background

B.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 (TAGs) (Wang et al., 2024b; He et al., 2025; Liu et al., 2024), they are not truly general as 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., 2024; 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., 2024; 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., 2024), 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 more moderate 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., 2024).

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 (TAGs), 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 simple 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 particularly 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; Yu et al., 2025; Wang et al., 2025a). In addition to GNN-based approaches, some methods use matrix decomposition techniques such as SVD applied 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).

B.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 popular 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.

B.3 Tabular foundation models

The field of tabular foundation models (TFMs) was pioneered by the TabPFN model (Hollmann et al., 2023) that 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 (Mueller et al., 2025; Ma et al., 2024; Qu et al., 2025; Zhang & Robinson, 2025; Zhang et al., 2025), 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.

C Implementation details

In this section, we describe our main implementation choices. Additional information and the full code are available in our repository.

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}.

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

PEARL For the GNN backbone within PEARL, our implementation is based on the GraphLand implementation (Bazhenov et al., 2025). However, we remove layer normalization and residual connections, based on preliminary experiments that showed improved results without these components.

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 D). For the finetuning experiments, we jointly finetune PEARL and the TFM backbone.

Label shuffling To ensure that our framework is equivariant to permutations of class labels in multiclass classification tasks, we employ a label shuffling procedure. During each forward pass, class numerical labels are randomly shuffled, so that the average predictions remain independent of the original numerical label assignment.

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 D, 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

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

	`tolokers-2`	`city-reviews`	`artnet-exp`	`hm-prices`	`avazu-ctr`	`city-roads-M`	`twitch-views`	`artnet-views`	**AR**
G2T-TabPFNv2 (ICL)	60.42 ± 0.27	77.46 ± 0.10	45.84 ± 0.03	66.68 ± 0.09	26.38 ± 0.07	60.47 ± 0.04	70.00 ± 0.06	58.75 ± 0.15	**5.88**
w/o NFA (ICL)	60.28 ± 0.34	76.98 ± 0.08	43.57 ± 0.13	62.90 ± 0.09	25.54 ± 0.33	58.68 ± 0.23	69.61 ± 0.05	58.86 ± 0.22	**7.62**
w/o SF & PEARL (ICL)	56.17 ± 0.12	76.87 ± 0.02	45.90 ± 0.01	67.22 ± 0.03	26.78 ± 0.02	60.08 ± 0.01	58.94 ± 0.06	55.30 ± 0.02	**7.88**
w/o SF (ICL)	57.48 ± 0.18	77.07 ± 0.02	45.92 ± 0.02	67.11 ± 0.03	26.73 ± 0.01	60.01 ± 0.07	67.80 ± 0.02	56.23 ± 0.03	**6.88**
w/o PEARL (ICL)	60.56 ± 0.32	77.47 ± 0.10	45.84 ± 0.03	66.81 ± 0.06	26.33 ± 0.09	60.53 ± 0.03	65.30 ± 0.07	58.36 ± 0.24	**6.12**
G2T-TabPFNv2 (FT)	57.65 ± 1.92	79.12 ± 0.21	47.31 ± 0.59	71.05 ± 0.91	28.52 ± 0.43	63.08 ± 0.28	74.06 ± 0.16	60.29 ± 0.13	**2.25**
w/o NFA (FT)	59.75 ± 0.76	78.61 ± 0.21	45.10 ± 0.35	67.10 ± 0.74	25.98 ± 0.99	61.40 ± 0.44	73.28 ± 0.15	59.93 ± 0.18	**5.38**
w/o SF & PEARL (FT)	57.19 ± 1.15	78.68 ± 0.18	47.05 ± 0.39	71.19 ± 0.61	27.99 ± 0.36	62.67 ± 0.17	60.98 ± 0.14	57.02 ± 0.45	**5.25**
w/o SF (FT)	57.44 ± 0.48	78.61 ± 0.13	47.17 ± 0.40	71.77 ± 0.53	28.31 ± 0.58	59.72 ± 0.70	73.10 ± 0.21	58.29 ± 0.10	**4.62**
w/o PEARL (FT)	60.18 ± 0.49	79.18 ± 0.21	47.57 ± 0.43	70.64 ± 0.83	28.19 ± 0.31	63.28 ± 0.25	66.90 ± 0.19	60.26 ± 0.12	**2.88**

Table 5: Comparison of G2T-FM against the baselines that are enhanced with the same components as G2T-FM. M stands for Modified and means that we add NFA, classic structure-based features, and PEARL encodings to their features.

	`tolokers-2`	`city-reviews`	`artnet-exp`	`hm-prices`	`avazu-ctr`	`city-roads-M`	`twitch-views`	`artnet-views`	**AR**
LightGBM+NFA	56.34 ± 0.06	78.53 ± 0.01	46.13 ± 0.03	70.84 ± 0.04	31.71 ± 0.01	61.18 ± 0.03	60.14 ± 0.01	56.10 ± 0.02	**5.75**
GCN	56.27 ± 0.29	77.81 ± 0.14	44.86 ± 0.34	68.02 ± 0.40	32.00 ± 0.15	58.82 ± 0.24	75.51 ± 0.05	56.03 ± 0.24	**7.75**
GraphSAGE	54.43 ± 0.32	78.17 ± 0.09	45.14 ± 0.34	70.00 ± 0.70	31.44 ± 0.15	59.44 ± 0.26	66.29 ± 0.31	49.32 ± 0.86	**8.62**
GAT	57.41 ± 0.80	77.74 ± 0.20	45.06 ± 0.49	72.07 ± 1.16	32.63 ± 0.16	59.86 ± 0.19	72.89 ± 0.25	53.60 ± 0.23	**5.75**
GT	56.98 ± 0.53	77.34 ± 0.20	46.41 ± 0.68	69.44 ± 0.89	31.11 ± 0.47	59.55 ± 0.27	72.13 ± 0.13	53.37 ± 0.43	**8.38**
LightGBM+NFA (M)	57.16 ± 0.70	78.68 ± 0.04	45.57 ± 0.19	70.25 ± 0.14	31.31 ± 0.08	60.86 ± 0.16	65.17 ± 0.04	57.53 ± 0.04	**5.88**
GCN (M)	58.71 ± 0.45	77.07 ± 0.27	43.44 ± 0.32	70.73 ± 0.26	31.10 ± 0.22	57.91 ± 0.22	77.11 ± 0.09	56.14 ± 0.24	**7.62**
GraphSAGE (M)	59.59 ± 0.51	77.95 ± 0.09	44.31 ± 0.53	70.50 ± 0.47	31.51 ± 0.41	59.66 ± 0.09	75.93 ± 0.19	55.39 ± 0.32	**6.12**
GAT (M)	57.76 ± 0.70	77.47 ± 0.14	44.36 ± 0.50	72.46 ± 0.49	31.97 ± 0.23	59.57 ± 0.43	77.20 ± 0.18	56.51 ± 0.35	**4.88**
GT (M)	58.79 ± 0.76	76.43 ± 0.10	43.03 ± 0.60	71.84 ± 0.64	29.86 ± 0.67	59.85 ± 0.41	76.15 ± 0.11	56.39 ± 0.31	**6.75**
G2T-TabPFNv2 (ICL)	60.42 ± 0.27	77.46 ± 0.10	45.84 ± 0.03	66.68 ± 0.09	26.38 ± 0.07	60.47 ± 0.04	70.00 ± 0.06	58.75 ± 0.15	**6.62**
G2T-TabPFNv2 (FT)	57.65 ± 1.92	79.12 ± 0.21	47.31 ± 0.59	71.05 ± 0.91	28.52 ± 0.43	63.08 ± 0.28	74.06 ± 0.16	60.29 ± 0.13	**3.88**

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 Ablation

In our ablation study, we focus on G2T-TabPFNv2, selecting it as the representative model because TabPFNv2 is a widely adopted and well-established tabular foundation model.

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, the following observation holds.

Observation 4 *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.

Observation 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, on some datasets it outperforms all other methods by a wide margin.*

Summary To sum up, 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, 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.

E Symmetries: equivariance and invariance

For graph problems, it is typically assumed that node IDs can be relabeled, feature columns can be reordered, and class IDs can be renamed without changing the underlying task. Hence, a model should not depend on these arbitrary choices. This motivates three symmetries for graph foundation models, as advocated in Finkelshtein et al. (2025): (i) feature permutation invariance; (ii) label permutation equivariance; and (iii) node permutation equivariance.

Formally, let G be a group acting on inputs $\mathcal{X}$ and outputs $\mathcal{Y}$. A mapping $f : \mathcal{X} \to \mathcal{Y}$ is G-equivariant if $f(g \cdot x) = g \cdot f(x)$ for all $g \in G$, $x \in \mathcal{X}$. It is G-invariant if $f(g \cdot x) = f(x)$ for all g. In our context, relevant groups include node permutations $S_{|V|}$, feature (column) permutations S_d, and label permutations $S_{|C|}$. Here, S_n denotes the symmetric group on a set of n elements (i.e., the group of all permutations), $|V|$ is the number of nodes, d is the number of features, and $|C|$ is the number of classes.

Modern models may also include stochastic components (e.g., random positional encodings (Kanatsoulis et al., 2025)). In such cases, one may require symmetries to hold *in distribution*: after the relevant permutation, the distribution of outputs is unchanged (for invariance) or transformed accordingly (for equivariance), even if a single stochastic realization is not exactly symmetric.

Below, we discuss the symmetries of G2T-FM. In particular, we show that all components added to TFM are equivariant in distribution, so the resulting symmetries of G2T-FM depend on the symmetries of the TFM backbone. As an example, we analyze the symmetries of TabPFNv2.

G2T-FM components The symmetries of G2T-FM rely on the symmetries of the chosen TFM. Let us show that if the TFM has feature permutation invariance, label permutation equivariance and sample permutation equivariance (in distribution), then G2T-FM also has all the desired symmetries. Indeed, G2T-FM employs several components: NFA, PEARL, and simple structure-based features (degree, PageRank, Laplacian eigenvectors). NFA is node- and feature-equivariant. Structural features such as node degree, PageRank, and Laplacian eigenvectors are all node-equivariant by construction. The PEARL framework, which we also use, is node-equivariant *in distribution*. The combination of these components makes G2T-FM node permutation equivariant *in distribution*. Thus, all the desired symmetries are preserved.

TabPFNv2 backbone TabPFNv2 is sample permutation equivariant and feature permutation invariant *in distribution*. This property holds because the model applies random positional encodings to its input features. As these encodings are sampled independently and identically, the output distribution is unaffected by the order of the feature columns. By default, TabPFNv2 is not label permutation equivariant. However, this can be achieved with a simple modification that we incorporate into our implementation. Specifically, during each forward pass, we randomly permute the ordinal encodings assigned to the class labels. This procedure ensures that the model becomes label permutation equivariant *in distribution*.

F Additional Details

F.1 Computational Resources

Every individual experiment with a fixed dataset and method required only a single Tesla A100 80GB GPU. We employed several GPUs from an internal cluster to run experiments in parallel (for example, by assigning experiments for different datasets to different GPUs). The longest experiment took 16 hours to complete. Overall, our main experiments required 953 GPU-hours.

F.2 Datasets

We employ PyG (Fey & Lenssen, 2019) to access all datasets except for the GraphLand datasets. To access the GraphLand datasets, we use the official repository. We were unable to find a license for the `pubmed` dataset. However, we cite the original paper, where this dataset was introduced. To the best of our knowledge, the `amazon-ratings`, `questions`, and `wiki-cs` datasets are under an MIT license, the `facebook` dataset is under a GNU license, while the GraphLand datasets are under an Apache 2.0 license.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The main claims made in the abstract and introduction clearly reflect the content of the paper and are supported by the experimental results.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitations of our work in Appendix A.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

Justification: Our work does not include theoretical results.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We describe all needed information in Section 3 and Appendix C. We also release the code to make our experiments reproducible.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We release the code and use publicly available datasets.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We describe all needed information in Section 3 and Appendix C. We also release the code to make our experiments reproducible.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We run our experiments with several random seeds and report mean and standard deviations.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We describe the compute resources in Appendix F.1.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conforms, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: The paper poses no such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We cite the original papers for all used datasets. The corresponding licenses are provided in Appendix F.2.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: Our repository is well-documented and contains a license.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (https://neurips.cc/Conferences/2025/LLM) for what should or should not be described.