Understanding Feature/Structure Interplay in Graph Neural Networks

Diana Gomes
IMEC, Leuven, Belgium
VUB, Brussels, Belgium
diana.gomes@imec.be

Ann Nowe
VUB, Brussels, Belgium
ann.nowe@vub.be

Peter Vrancx
IMEC, Leuven, Belgium
VUB, Brussels, Belgium
peter.vrancx@vub.be

Abstract

Graph neural networks (GNNs) have become a standard method to process graph data, but their performance is often poorly understood. It is easy to find examples in which a GNN is unable to learn useful graph representations, but generally hard to explain why. In this work, we analyse the effectiveness of graph representations learned by GNNs for input graphs with different structural properties and feature information. We expand on the failure cases by decoupling the impact of structural and feature information on the learning process. Our results indicate that GNNs' implicit architectural assumptions are tightly related to the structural properties of the input graph and may impair its learning ability. In case of mismatch, they can often be outperformed by structure-agnostic methods like multi-layer perceptrons.

1 Introduction

Graph neural networks (GNNs) have emerged as the default approach for graph representation learning in machine learning tasks. These models have clearly demonstrated their ability to learn from relational data and have achieved multiple successful applications in diverse domains such as drug design [1], biology [2] and complex physics simulations [3]. Despite this fact, one can easily find examples in which a GNN is unable to learn useful node/graph representations and underperform when compared to structure-agnostic counterparts [4, 5], such as multilayer perceptrons (MLPs).

GNNs make class predictions using two sources of information: the individual node properties described by the feature vectors associated with each node, and structural information represented by the relationships (edges) between the different nodes. The main idea behind using graph representations is that relational information can provide additional information to solve the target task, e.g. friendship links in a social network can be predictive of a person's interests. Given that we are augmenting the node properties with additional information from node relations, one might assume that GNNs always outperform feature-only methods that do not exploit any structural information. However, recent benchmark results [4, 5] show state-of-the-art GNNs performing on-par or worse than basic MLPs on a variety of node classification tasks. These results hint at the fact that the use of structural information by GNNs is not always helpful and may even be detrimental to classification performance, not only for the simplest network architectures but also for the more recent and complex ones.

While several authors have investigated the phenomena of oversmoothing and loss of expressivity in GNNs [5–8], these works typically focus on representational power as related to the depth of the graph neural networks and consider feature and structure information as a whole. In this work, we intend to decouple the influence of these two levels of information on graph learning to determine in which way the underlying assumptions of GNN architectures are key to determining their performance. To this end, we consider the problem of semi-supervised node classification and conduct a methodical, empiric investigation of the use of structural vs. feature information by different GNN methods. By applying our methodology, we show that, due to fundamental limitations, basic GNNs are unable

D. Gomes et al., Understanding Feature/Structure Interplay in Graph Neural Networks. *Proceedings of the Third Learning on Graphs Conference (LoG 2024)*, PMLR 269, Virtual Event, November 26–29, 2024.

to learn useful representations when local smoothing is not beneficial, either due to the structure or feature components, even when their depth is small. Furthermore, we provide evidence that more advanced GNN architectures can avoid this limitation, but may still fail to exploit structural information altogether and reduce to feature-only representations in case these cannot encode such structure productively.

The main contributions of this work are two-fold:

1. The proposition of an empirical, model-agnostic method for decoupling feature and structure influence on GNN node classification performance (Section 3);
2. The disclosure of novel empirical insights on how each of the two levels of graph information - structure and node features - can influence GNN classification performance, which can inspire future research directions towards improving graph learning methods (Sections 4 and 5), including the following:
    - *Simple GCNs need both feature and structural information to be meaningful to learn useful node representations.* If only one of these is present, the model cannot separate useful information from meaningless. However, GCNs can significantly outperform feature-only methods when the feature signal is weak but the structure is relevant for the final task.
    - *The absolute gains of using a certain GNN compared to feature-only methods can be measured in a model-agnostic way.* We show that these gains are a function of how much each model can leverage the graph structure and how relevant the features are for the final classification task.

1.1 Related Work

Causes of poor GNN performance. Many authors have investigated poor GNN performance due to apparent oversmoothing or loss of expressivity in deeper GNNs [5–10]. This phenomenon is typically explained as deeper networks excessively smoothing the features between distant graph nodes. Others have argued that poor results are not necessarily associated with oversmoothing, but are rather a consequence of the training difficulty of GNNs [11–13]. Oversquashing [14] and heterophily [5] have also been pointed out as causes for performance degradation. Despite their generalized acceptance within the GNN research community, these different causes/concepts are to some extent intertwined and it is difficult to detach one from the others and measure their individual contribution to the performance degradation witnessed downstream.

This fact could be the reason why it is currently possible to find contradictory information in theoretical studies regarding GNN learning behaviors [13, 15]. Therefore, there is a demand for relevant empirical studies addressing these questions. In Luan et al. [16], for example, the authors show how GNN's performance goes beyond homo-/heterophily alone through empirical analyses with dedicated quantitative metrics; however, no relation with feature distribution/quality was explored. Our work aims to fill this gap, bringing forward novel insights on the key conditions for effective GNN learning (both on structural and feature levels) through a dedicated set of experiments under full control of graph attributes.

Advanced GNN architectures. New GNN architectures have emerged in recent years aiming to address specific challenges/limitations of these networks, including improving expressivity [17, 18], decreasing computational cost [19, 20], handling graph oversmoothing [21, 22] and overcoming harmful node-level heterophily [5, 23, 24]. While these networks have proved successful in addressing some of the above limitations, their empirical gains for the classification task are not always well understood. It can also be difficult to fully compare these approaches, due to the higher complexity, the different base assumptions, and insufficient benchmarking analyses. This makes the identification of learning bottlenecks a non-trivial problem for GNNs. The adoption of more robust model-agnostic, experimental analyses, such as the ones proposed in this work, could help the community better understand these bottlenecks.

Feature/structure influence in graph learning. Decoupling feature and structure impact in GNNs has recently been investigated for transfer learning and graph generation purposes [25]. However, the authors assume node homophily and do not explore the cases where this condition is not met, which is one of the aspects in which our work differs. Other works also explore homo-/heterophilic settings and create advanced architectures to handle challenging scenarios [26]. Nevertheless, it is not clear whether these architectures lead to more useful node representations or if they solely overcome

its performance hindering impact, as we can see in recent benchmarks [4]. Our work intends to complement these efforts by providing insights on how structure can not only be harmful but also simply uninformative, leading to local smoothing operations that decrease feature expressivity, in which case one might simply resort to feature-only methods.

2 Background

2.1 Graph Convolution

Let us represent our graph as $G = (V, \mathbf{A})$, where V is the set of all n vertices $\{v_1, ..., v_n\}$ and $\mathbf{A}$ is an adjacency matrix of size $n \times n$. Each element a_{ij} of $\mathbf{A}$ defines the edge between nodes v_i and v_j, assuming the value of 1 if v_i and v_j are connected, and 0 otherwise. Each node v_i is associated with a m-dimensional feature vector $\mathbf{x}_i$. A feature matrix $\mathbf{X} \in \mathbb{R}^{n \times m}$ can also be defined by stacking all n feature vectors.

We introduce Graph Convolutional Networks (GCNs) as proposed by Kipf & Welling [27] in the context of semi-supervised node classification. In this work, a normalized version of the adjacency matrix of G with added self-loops is defined by $\hat{\tilde{\mathbf{A}}} = \tilde{\mathbf{D}}^{-1/2}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-1/2}$, where $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}_n$, $\tilde{\mathbf{D}} = \mathbf{D} + \mathbf{I}_n$, and $\mathbf{D}$ is the diagonal degree matrix of G, where the degree of v_i can be defined as $d_i = \sum_j a_{ij}$.

$$\mathbf{H}^l = \sigma(\hat{\tilde{\mathbf{A}}}\mathbf{H}^{l-1}\mathbf{W}^l) \tag{1}$$

The initial node representations $\mathbf{H}^0 = \mathbf{X}$ are transformed by graph convolutions in the form of Equation 1, where $l \in \{1, ..., L\}$ represents the graph convolution layer, $\mathbf{W}^l$ consists of a matrix of learned weights and σ is a nonlinear activation function, such as ReLU. For node-level tasks, the final representation $\mathbf{H}^L$ is finally fed to a classification head consisting of a linear layer and an activation function, providing the final predictions $\mathbf{Y} \in \mathbb{R}^{n \times c}$, where c is the number of classes.

2.2 Graph Smoothing

By relying on consecutive local averaging operations, graph convolutions are local smoothers by nature. This smoothing can be beneficial, and has even been considered the major benefit of GCN layers [28]. For this reason, GCNs often perform outstandingly on homophilic graphs, i.e., graphs for which links between nodes of the same class are more frequent than inter-class links. In these graphs, same-class node clusters are common, making operations such as local averaging particularly useful, as these lead to a smoother graph signal in these regions. Thus, the resulting node representations often lead to evident empirical gains for the downstream task when compared to their non-smoothed versions, such as those used by feature-only methods.

However, the same is not necessarily true for heterophilic graphs [5, 29]; in this case, downstream benefits are not as evident, and local smoothing can even deteriorate overall performance. The consecutive aggregation of node representations by stacking multiple graph convolutional layers can also have dire consequences. This phenomena is known as oversmoothing, and can be defined as the exponential convergence of all node representations to the same constant [30], inevitably leading to the loss of the discriminative power of the network. These conflicting consequences of graph convolutions pose challenging obstacles to the establishment of GNNs as the one-size-fits-all approach for machine learning on graphs, and have been motivating the development of new, more complex network architectures and the definition of best practices in the field.

Assumption 1. Given this context, we propose to evaluate when the following assumption holds and when it fails: *the local smoothing obtained through a certain sequence of graph convolution operations leads to useful node embeddings, suitable for solving a downstream task.* We consider simple and advanced models paired with highly controlled synthetic and real-world tasks to perform cross-comparisons among said pairs and disclose new insights on the patterns of GNN learning.

3 Model-agnostic Feature/Structure Decoupling Method

In order to study when GNNs can and cannot learn useful node representations, we propose a new, model-agnostic method which intends to decouple the influence of feature and structure information

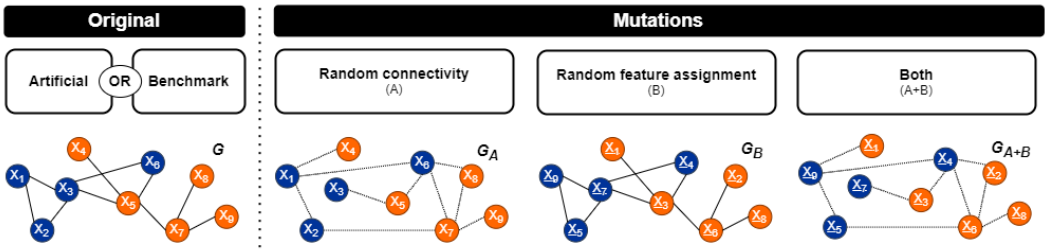

Figure 1: Graph transformations proposed in this work to methodically destroy structure and/or features while preserving the remaining graph attributes. For each original graph G, three combinations of feature/structure transformations are created (mutations).

in each node classification task and separately violate GNNs' assumptions of meaningful underlying structure and features. This method can provide a quantitative measure of how much the GNN is leveraging the structure vs. feature components, thus assisting the identification of learning bottlenecks.

Graph mutations. We implement two operations designed to derive a set of mutations for each original graph. These operations are set to methodically destroy structure and/or feature information while preserving the remaining graph attributes (Figure 1). Each input graph (artificial or real-world) is submitted to transformations on the structural and feature levels: (1) *Random connectivity*: shuffle columns of graph connectivity matrix; (2) *Random feature assignment*: randomize the attribution of feature vectors across all nodes. With these operations, we conceive up to three mutations (Figure 1) for each original graph whenever appropriate: a mutation with random connectivity but same feature distribution (G_A); another with same structural information but random feature assignment (G_B); and a final one with both transformations (G_{A+B}).

Comparison with baselines. Each original graph and respective mutations are then separately used to train two different types of models - a GNN, with no restrictions on complexity or architecture, and a structure-agnostic model (in this case, an MLP). These outputs can be interpreted on their own or yet be used to compute two metrics for measuring the separate contributions of structure (SC) and feature (FC) components, when appropriate. These are calculated by Equations 2 and 3, where s_G, s_{G_A}, s_{G_B}, $s_{G_{A+B}}$ refer to the scores of GNNs trained on the original graph, random connectivity, random feature assignment and both, respectively. Together, these metrics give an indication of how sensitive a certain type of GNN model is to the structure and feature information of a particular dataset (please refer to Appendix A.2 for further discussion on the interpretation of SC and FC).

$$\text{SC} = (s_G - s_{G_A}) + (s_{G_B} - s_{G_{A+B}}) \tag{2}$$

$$\text{FC} = (s_G - s_{G_B}) + (s_{G_A} - s_{G_{A+B}}) \tag{3}$$

Finally, the comparison with the MLP evidences whether structure encoding using the GNN architecture under analysis is generally successful or harmful, i.e., GNN performance on the original graph is superior to feature-only MLP or not, respectively.

4 Experiments

Our investigation of the cases when GNNs can and cannot learn useful node representations is conducted by applying our previously introduced features/structure decoupling method to artificial graphs with certain engineered properties. In particular, we manipulate homophily and edge density (structural properties) and feature signal-to-noise ratio (SNR), due to their direct relation with the implicit assumptions of message-passing graph models. This is a fully controlled scenario in which we can measure how GNNs respond to variations of the graph structure and the feature vector from an empirical perspective.

These experiments are further extended by extrapolating our insights to more challenging settings by means of several real-world datasets, commonly used as benchmarks for machine learning on graphs,

Table 1: Node classification performance (mean accuracy $\pm$ standard deviation) of GCN and MLP models on an artificial graph G with highly informative structure and features, and the mutations of G with random connectivity (uninformative structure), random feature assignment (uninformative features), and both (uninformative structure and features).

		Structure			
		Informative		Uninformative	
Features	Informative	GCN	0.96 ± 0.02	GCN	0.69 ± 0.02
		MLP	0.92 ± 0.02	MLP	0.90 ± 0.04
	Uninformative	GCN	0.61 ± 0.04	GCN	0.48 ± 0.03
		MLP	0.50 ± 0.05	MLP	0.50 ± 0.03

and more advanced network architectures. All details concerning experimental setup, including artificial graph generation, benchmark datasets and model selection, training and evaluation can be found in Appendix A.1.

4.1 Graph Convolution on Artificial Graphs with Engineered Properties

While GCNs seem to be an efficient and straightforward method to apply machine learning to graphs, the fact that they perform on-par or worse than basic MLPs on certain tasks makes us question whether GCNs are able to encode graph structure productively for all tasks. *Is there always a practical benefit in encoding both feature and structure information for node representation learning with GCNs? And, if not, can we identify under which conditions is the exploitation of both levels of information detrimental for the classification task?* In this subsection, we provide an answer to this question by decoupling the influence of feature/structure in GCN's performance on artificial graphs with engineered properties using our newly proposed method (as introduced in Section 3).

4.1.1 Informative graph

Let us consider the simple case of binary node classification in a graph with both highly informative features (SNR = 1.5) and structure (homophily = 0.95; edge density = 0.06). Table 1 compares the performance of a 2-layer vanilla-GCN with that of a structure-agnostic model (MLP) on the original version of this graph and its mutations (uninformative versions).

While GCN exhibits adequate performance (superior to the MLP) when both structural and feature information are present, results show an evident drop when either is lost. Despite the fact that features are highly informative, GCN does not seem able to fully leverage them when the structure of the input graph is meaningless towards its inherent assumptions, leading to a significant loss of performance even relatively to its structure-agnostic counterpart. A similar drop is verified in the scenario where structure is preserved but feature information is lost, as GCNs are not able to aggregate neighborhood information in meaningful node representations, despite the highly informative structure of the input graph. We verify this behavior using shallow models of 2 message-passing layers, for which graph oversmoothing does not occur, as the appropriate performance in the informative scenario corroborates.

These results suggest that GCN models need both feature and structural information to be meaningful in order to learn useful node representations. When only one of these is present, the model does not seem to be able to separate the useful from meaningless information. This result makes sense given the intuition of GCN as a smoothing operator [9]. Blindly aggregating either features of dissimilar nodes due to lack of structural information or combining non-informative features of similar nodes does not extract useful node descriptions.

4.1.2 Graphs with different properties

Given the empirical verification that GCN performance can be closely related to feature information and structural properties of the input graph, we consider these attributes in separate methodical studies. Let us take a base graph with fixed characteristics (homophily = 0.8; edge density = 0.03; feature SNR = 1.2). Tables 2 and 3 present the node classification results on several versions of this graph that correspond to assigning it different connectivity matrices (and respective mutations). These

Table 2: Node classification performance (mean accuracy $\pm$ standard deviation) of GCN and MLP (baseline) models on artificial graphs with fixed edge density (0.03) and features (SNR = 1.2) and different homophily (H). Results are shown for each original graph G and the respective mutations of G with random connectivity, random feature assignment, and both. Given its feature-only nature, a single MLP result is shown per feature transformation.

	H	Structure		MLP (baseline)
		Original	Random	
Original Features	0.2	0.78 ± 0.08	0.58 ± 0.03	0.71 ± 0.03
	0.5	0.60 ± 0.02	0.61 ± 0.03	
	0.8	0.81 ± 0.03	0.59 ± 0.03	
Random Feature Assignment	0.2	0.51 ± 0.05	0.48 ± 0.04	0.49 ± 0.03
	0.5	0.53 ± 0.04	0.50 ± 0.03	
	0.8	0.49 ± 0.03	0.49 ± 0.02	

Table 3: Node classification performance (mean accuracy $\pm$ standard deviation) of GCN and MLP (baseline) models on artificial graphs with fixed homophily (0.8) and features (SNR = 1.2) and different edge density (D_e). Results are shown for each original graph G and the respective mutations of G with random connectivity, random feature assignment, and both. Given its feature-only nature, a single MLP result is shown per feature transformation.

	D_e	Structure		MLP (baseline)
		Original	Random	
Original Features	0.003	0.78 ± 0.03	0.63 ± 0.05	0.71 ± 0.03
	0.03	0.81 ± 0.03	0.59 ± 0.03	
	0.15	0.79 ± 0.02	0.57 ± 0.02	
Random Feature Assignment	0.003	0.49 ± 0.06	0.48 ± 0.05	0.49 ± 0.03
	0.03	0.49 ± 0.03	0.49 ± 0.02	
	0.15	0.52 ± 0.04	0.49 ± 0.04	

matrices define structures of different homophily, while keeping density constant (and vice-versa, for Table 2 and Table 3, respectively). Analogously, Table 4 displays node classification results for versions of the base graph with different feature information, measured by its SNR; results for the respective random connectivity mutations are also shown.

Homophily. The inspection of GCN's response to different homophily conditions (Table 2) reveals its adequate performance on the most and least homophilic original graphs. While adequate performance in the most heterophilic scenario might seem surprising at first glance, as GCN's limitations in dealing with such settings are well-know, it is not unexpected in our experiment. This behavior relates to the fact that our learning problem only considers two distinguishable types of nodes and has also been recently reported in other works [29]. Nodes are able to encode meaningful representations through neighborhood aggregation, despite most of their neighbors belonging to a different class, due to its consistency. While this outcome may not hold under different conditions (such as some multi-class problems), it also draws attention to the potential insufficiency of solely resorting to homophily-related assumptions to steer GNN architecture research endeavors, as discussed by recent works [16, 29].

Similar to the previous highly informative artificial graph, we verify a significant loss of performance when structure and/or feature information of original graphs are destroyed, except for when homophily is close to 0.5 (mediocre performance on the original graph, on-par with the random structure mutation). This means that when nodes are equally likely to be connected to nodes of a different class as to those of the same class, this produces an uninformative structure. A GCN will use this structure to perform local smoothing operations that will decrease feature expressivity and lead to poor node representations. These results verify that simply attributing a GCN's poor performance to graph heterophily may be insufficient, as some heterophilous graphs can encode relevant structure information while others do not.

Table 4: Node classification performance (mean accuracy $\pm$ standard deviation) of GCN and MLP (baseline) models on artificial graphs with fixed structure (homophily = 0.8; edge density = 0.03) and different feature signal-to-noise ratio (SNR). Results are shown for each original graph G and the respective mutations of G with random connectivity.

SNR	Structure		MLP (baseline)
	Original	Random	
1.0	0.57 ± 0.05	0.51 ± 0.03	0.51 ± 0.05
1.2	0.81 ± 0.03	0.59 ± 0.03	0.71 ± 0.03
1.5	0.92 ± 0.02	0.68 ± 0.02	0.91 ± 0.02
2.0	0.99 ± 0.01	0.72 ± 0.01	1.00 ± 0.00

Edge density. Varying edge density appears to have a minimal effect in overall performance under the tested conditions (Table 3). In theory, we would expect to see better node representations for graphs with fewer connections when structure is irrelevant, as these would lead to a minimal smoothing effect. However, while the average performance for random structure decreases as graphs become more densely connected, these results cannot be deemed statistically significant.

Feature SNR. Table 4 shows the impact of considering different levels of separability of node features when structure encodes useful information and when it does not. The results indicate a significant loss of performance for the random connectivity mutation in comparison with the original version, even in scenarios when base features are easily separable by a feature-only method. A meaningful graph structure can, however, greatly assist the classification task in cases when features are not easily separable, as we can verify by the significant performance gains of using GCN with the original structure in comparison with the feature-only MLP for SNR $\in [1.0, 1.2]$. This scenario is thus the more evident sweet spot for representation learning tasks through graph convolutions.

4.2 Feature/Structure Decoupling in Complex Scenarios

The previous subsection shows evidence of a certain level of co-dependence between feature/structure information when deriving node representations with the simplest form of GCNs. This subsection extends the insights of the previous by stepping out of the controlled scenarios and discussing the impact of applying the same feature/structure decoupling methodology to more advanced GNN architectures and real-world benchmarks.

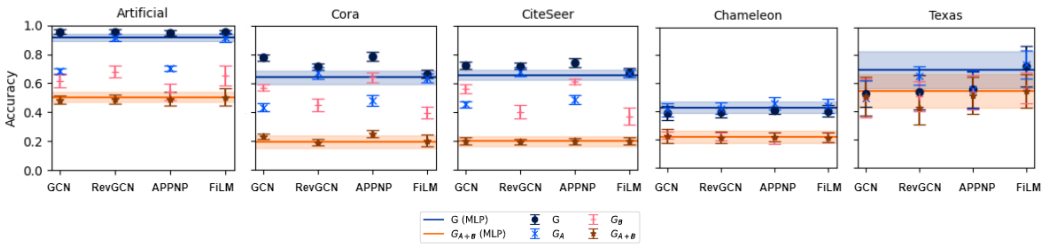

Figure 2: Node classification performance (accuracy) of different GNN architectures and MLP (baseline) models on real-world benchmarks. Results are shown in table and graphical form for each original graph G and the respective mutations of G with random connectivity (G_A), random feature assignment (G_B), and both (G_{A+B}). A single MLP (feature-only) result is shown per feature transformation. Shaded areas (MLP) and error bars (GNNs) represent standard deviation.

The results depicted in Figure 2 show that models that are generally more feature-dependent (i.e., those that respond more to changes of the feature distribution than to structural changes) are particularly robust in handling meaningless structural information and overcoming its potentially harmful effect. However, the same models can also underperform compared to vanilla-GCN for cases in which encoding graph structure through graph convolutions is particularly beneficial (e.g., Cora and CiteSeer with original structure). This is the case for RevGCN and FiLM. Their feature-preserving quality can be particularly observed in their overlap with MLP performance for random connectivity scenarios (see consistent overlap between RevGCN/FiLM and MLP performance for G_A mutation in Figure 2).

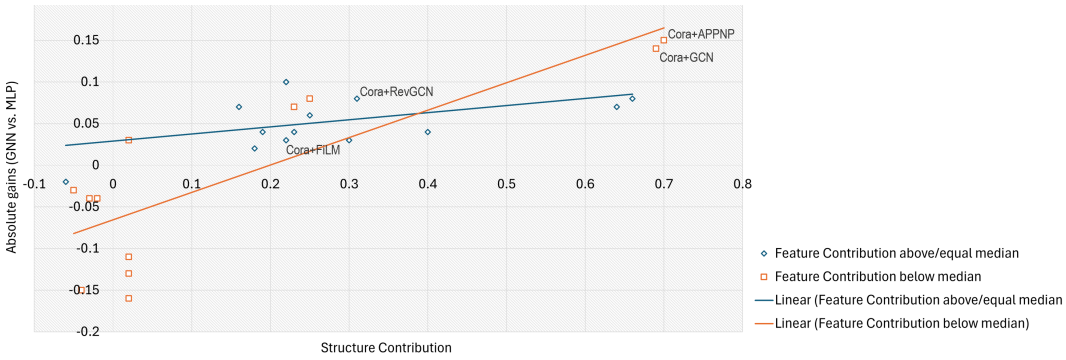

Figure 3: Structure/Feature contribution interaction plot. Each data point represents the metrics for a specific dataset+GNN pair (see Table 9 of the Appendix). For small values of feature contribution (below median), there is a strong positive relation between structure contribution and absolute gains; this relation is much weaker for high feature contribution (above/equal median).

GCN and APPNP seem to lack this feature-preserving quality. For these models, feature and structure information are both highly important. In cases in which either of these components holds irrelevant information for the downstream task, we can see significant performance drops compared to the original scenario (see GCN and APPNP performance for G_A and G_B mutations on the informative artificial graph, Cora and CiteSeer).

By decoupling feature/structure influence, we show evidence of how the several levels of graph information can impact the performance of different architectures for the same classification task. Our results support that new, advanced architectures (robust to some GNN limitations, like harmful smoothing) are not necessarily superior to vanilla methods, as it all depends on how the assumptions of each network architecture match the input graph's properties.

4.3 Measuring the Interplay between Feature and Structure Components

The results in the previous subsections strongly hint at the fact that there is an interplay between feature and structure components when using a GNN to solve a graph-related task. But can we make this conclusion stronger? In particular, *can we find quantitative evidence of the role of feature and structure interaction in successful GNNs?*

To provide an answer to this question, let us consider the absolute gains of a certain GNN model over a feature-agnostic counterpart (i.e., the MLP) as a measure of how much the GNN model is successfully exploiting all levels of graph information. Let us yet take the metrics introduced in Equations 2 and 3 (Section 3) as a measure of the separate contribution of the structure (SC) and feature (FC) components, respectively, in the classification task. Our hypothesis is that SC and FC can be adequate predictors of the quantitative benefits of using a certain GNN to solve a task and that a statistically relevant model using these predictors to estimate such gains can disclose new insights on the dynamics between structure and features in a GNN model. We tested this hypothesis through regression modelling. Our results show that we can make a valuable prediction of the gains of a GNN model using SC and FC as predictors. This prediction is particularly accurate and statistically significant if we include an interaction coefficient between the two variables in the form of SC$\times$FC, instead of considering their independent contributions alone (see details in Appendix A.2.1).

Figure 3 shows how this interaction manifests: when feature contribution is small for a certain dataset+GNN pair, the relationship between structure contribution and the effective gains of using that GNN is strongly positive; on the other hand, if feature contribution is high, the relationship between structure contribution and GNN gains is much weaker. This outcome advises for caution when dataset+GNN pairs resolve to more feature-preserving encoding, as their results may be sub-optimal in cases when the input graph structure leads to beneficial smoothing (at least when considering the same number of graph convolution steps), as per the example annotated in Figure 3 where feature preserving methods (in this case, RevGCN and FiLM) are outperformed by vanilla-GCN on Cora.

5 Discussion

Engineering the Structural Properties of the Input Graph. Our experiments indicate that GNNs can lead to poor representation learning if they are unable to leverage structure to usefully aggregate neighbouring feature vectors. These results encourage caution against blindly applying GNNs, as they may be outperformed by structure-agnostic methods.

Engineering the structural properties of the input graph to help GNNs learn better representations is rather common in the literature. Among common strategies are the creation of virtual nodes [31], graph rewiring algorithms [32], and the interchangeable definition of edges as directed/undirected for message-passing [24]. The idea behind these approaches is improving the flows of information through message-passing. However, it is hard to measure whether the new, engineered graph properties better fit GNNs' assumptions. Most rewiring efforts are motivated by either intelligible, domain-specific knowledge or some constraints on the graph representation, which can be relevant but yet insufficient. Our feature/structure decoupling method can assist this process of graph problem re-engineering from an empirical angle, providing a quantitative metric of the influence of the tested structures in GNN performance. This shall help guarantee that the rewiring efforts are also taking into account GNN's assumptions, potentially leading to more powerful models.

Denoising Feature Component via Graph Convolutions. Our results show that GCNs are very effective at feature denoising when the graph problem can be defined by structural information that promotes beneficial local smoothing. This means that they can bring major performance boosts when compared to feature-only methods in cases in which the feature signal is not particularly strong but the structural information is relevant. As such, we consider this *the sweet spot* for graph representation learning with GCNs.

The method proposed in Section 3 to decouple feature/structure influence in GNN classification can help researchers decide on whether a certain structure is relevant or not towards solving a graph problem. This is an important contribution since defining what is an informative structure is not straightforward and one can find conflicting points of view in the literature regarding GNNs' response to some graph properties, such as homo-/hetero-phily [5, 16]. We expect that our insights can complement those of other works [16] towards better understanding under which conditions should we expect GNNs to perform exceedingly well and when these models might not be the most appropriate to solve our downstream task.

Need for More Expressive GNNs. The experiment with more advanced GNN architectures on real-world benchmarks suggests an evident need for more expressive and general graph operations. This need has also been noted by several other authors [33, 34], as the field has been overflowing with new methods and layer types that only bring minor incremental gains or are crafted to fit specific applications. For example, new GNN architectures that allow deeper models without oversmoothing have been proposed in [20, 35–37]. Unfortunately, these models typically do not see the performance benefits of deeper networks seen in traditional deep learning [10]. We also evaluated some of these models in the scope of this work and our results suggest that some of these advanced methods are able to rely more on basic node features rather than on network structure. While they preserve feature information, they do not remedy the fact that GCN-like operations may not extract more useful features, which is an evidence of the demand for more expressive message-passing methods.

6 Conclusion

This work expanded on the cases when GNNs may not be better than feature-only methods for node classification on graphs. We propose a method that provides new insights on GNN learning behavior by decoupling how much we can learn from features vs. structure for each task. This can be an important outcome towards GNN explainability and effectively assist the identification of learning bottlenecks. Our results suggest that GCNs may lead to poor node representations when the input graph does not fit the inherent assumptions of their architectures, even without oversmoothing. While some advanced architectures can avoid this limitation, we verify that when they cannot leverage structural information, these mostly refrain from exploiting it and ultimately resort to a feature-preserving encoding, similar to feature-only methods, which can lead to sub-optimal results. This conclusion supports that GNNs should not be considered a one-size-fits-all approach for machine learning on graphs, but rather demand careful inspection of all levels of graph information prior to their application.

References

[1] Felix Wong, Erica J Zheng, Jacqueline A Valeri, Nina M Donghia, Melis N Anahtar, Satotaka Omori, Alicia Li, Andres Cubillos-Ruiz, Aarti Krishnan, Wengong Jin, et al. Discovery of a structural class of antibiotics with explainable deep learning. *Nature*, 626(7997):177–185, 2024. 1

[2] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, 2021. 1

[3] Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter Battaglia. Learning to simulate complex physics with graph networks. In *International conference on machine learning*, pages 8459–8468. PMLR, 2020. 1

[4] Francesco Di Giovanni, James Rowbottom, Benjamin P Chamberlain, Thomas Markovich, and Michael M Bronstein. Graph neural networks as gradient flows. *arXiv preprint arXiv:2206.10991*, 2022. 1, 3

[5] Yujun Yan, Milad Hashemi, Kevin Swersky, Yaoqing Yang, and Danai Koutra. Two sides of the same coin: Heterophily and oversmoothing in graph convolutional neural networks. In *2022 IEEE International Conference on Data Mining (ICDM)*, pages 1287–1292, 2022. doi: 10.1109/ICDM54844.2022.00169. 1, 2, 3, 9

[6] Chen Cai and Yusu Wang. A note on over-smoothing for graph neural networks. *arXiv preprint arXiv:2006.13318*, 2020.

[7] Muhammet Balcilar, Renton Guillaume, Pierre Héroux, Benoit Gaüzère, Sébastien Adam, and Paul Honeine. Analyzing the expressive power of graph neural networks in a spectral perspective. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

[8] Kenta Oono and Taiji Suzuki. Graph neural networks exponentially lose expressive power for node classification. *arXiv preprint arXiv:1905.10947*, 2019. 1

[9] Qimai Li, Zhichao Han, and Xiao-Ming Wu. Deeper insights into graph convolutional networks for semi-supervised learning. In *Thirty-Second AAAI conference on artificial intelligence*, 2018. 5

[10] Diana Gomes, Kyriakos Efthymiadis, Ann Nowe, and Peter Vrancx. Depth scaling in graph neural networks: Understanding the flat curve behavior. *Transactions on Machine Learning Research*, 2024. 2, 9

[11] Sitao Luan, Mingde Zhao, Xiao-Wen Chang, and Doina Precup. Training matters: Unlocking potentials of deeper graph convolutional neural networks. *arXiv preprint arXiv:2008.08838*, 2020. 2

[12] Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay Pande, and Jure Leskovec. Strategies for pre-training graph neural networks. *arXiv preprint arXiv:1905.12265*, 2019.

[13] Weilin Cong, Morteza Ramezani, and Mehrdad Mahdavi. On provable benefits of depth in training graph convolutional networks. *Advances in Neural Information Processing Systems*, 34:9936–9949, 2021. 2

[14] Jake Topping, Francesco Di Giovanni, Benjamin Paul Chamberlain, Xiaowen Dong, and Michael M Bronstein. Understanding over-squashing and bottlenecks on graphs via curvature. *arXiv preprint arXiv:2111.14522*, 2021. 2

[15] Nicolas Keriven. Not too little, not too much: a theoretical analysis of graph (over) smoothing. *Advances in Neural Information Processing Systems*, 35:2268–2281, 2022. 2

[16] Sitao Luan, Chenqing Hua, Minkai Xu, Qincheng Lu, Jiaqi Zhu, Xiao-Wen Chang, Jie Fu, Jure Leskovec, and Doina Precup. When do graph neural networks help with node classification? investigating the homophily principle on node distinguishability. *Advances in Neural Information Processing Systems*, 36, 2024. 2, 6, 9

[17] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rJXMpikCZ. 2

[18] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2018. 2

[19] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017. 2

[20] Guohao Li, Matthias Müller, Bernard Ghanem, and Vladlen Koltun. Training graph neural networks with 1000 layers. In *International conference on machine learning*, pages 6437–6449. PMLR, 2021. 2, 9, 12, 13

[21] Ming Chen, Zhewei Wei, Zengfeng Huang, Bolin Ding, and Yaliang Li. Simple and deep graph convolutional networks. In *International conference on machine learning*, pages 1725–1735. PMLR, 2020. 2

[22] Lingxiao Zhao and Leman Akoglu. Pairnorm: Tackling oversmoothing in gnns. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=rkecl1rtwB. 2

[23] Hongbin Pei, Bingzhe Wei, Kevin Chen-Chuan Chang, Yu Lei, and Bo Yang. Geom-gcn: Geometric graph convolutional networks. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=S1e2agrFvS. 2

[24] Cristian Bodnar, Francesco Di Giovanni, Benjamin Chamberlain, Pietro Lio, and Michael Bronstein. Neural sheaf diffusion: A topological perspective on heterophily and oversmoothing in gnns. *Advances in Neural Information Processing Systems*, 35:18527–18541, 2022. 2, 9

[25] Duong Chi Thang, Hoang Thanh Dat, Nguyen Thanh Tam, Jun Jo, Nguyen Quoc Viet Hung, and Karl Aberer. Nature vs. nurture: Feature vs. structure for graph neural networks. *Pattern Recognition Letters*, 159:46–53, 2022. 2

[26] Jiong Zhu, Yujun Yan, Lingxiao Zhao, Mark Heimann, Leman Akoglu, and Danai Koutra. Beyond homophily in graph neural networks: Current limitations and effective designs. *Advances in Neural Information Processing Systems*, 33:7793–7804, 2020. 2

[27] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017. 3, 12

[28] Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. Simplifying graph convolutional networks. In *International conference on machine learning*, pages 6861–6871. PMLR, 2019. 3

[29] Yao Ma, Xiaorui Liu, Neil Shah, and Jiliang Tang. Is homophily a necessity for graph neural networks? In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=ucASPPD9GKN. 3, 6

[30] T Konstantin Rusch, Michael M Bronstein, and Siddhartha Mishra. A survey on oversmoothing in graph neural networks. *arXiv preprint arXiv:2303.10993*, 2023. 3

[31] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *International conference on machine learning*, pages 1263–1272. PMLR, 2017. 9

[32] Yu Rong, Wenbing Huang, Tingyang Xu, and Junzhou Huang. Dropedge: Towards deep graph convolutional networks on node classification. In *International Conference on Learning Representations*, 2019. 9

[33] Deyu Bo, Xiao Wang, Chuan Shi, and Huawei Shen. Beyond low-frequency information in graph convolutional networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 3950–3957, 2021. 9

[34] Cristian Bodnar, Francesco Di Giovanni, Benjamin Paul Chamberlain, Pietro Liò, and Michael M Bronstein. Neural sheaf diffusion: A topological perspective on heterophily and oversmoothing in gnns. *arXiv preprint arXiv:2202.04579*, 2022. 9

[35] Johannes Klicpera, Aleksandar Bojchevski, and Stephan Günnemann. Predict then propagate: Graph neural networks meet personalized pagerank. *arXiv preprint arXiv:1810.05997*, 2018. 9, 12, 13

[36] Hanqing Zeng, Muhan Zhang, Yinglong Xia, Ajitesh Srivastava, Andrey Malevich, Rajgopal Kannan, Viktor Prasanna, Long Jin, and Ren Chen. Decoupling the depth and scope of graph neural networks. *Advances in Neural Information Processing Systems*, 34:19665–19679, 2021.

[37] T Konstantin Rusch, Benjamin Paul Chamberlain, Michael W Mahoney, Michael M Bronstein, and Siddhartha Mishra. Gradient gating for deep multi-rate learning on graphs. *ICLR*, 9:25, 2023. 9

[38] John Palowitch, Anton Tsitsulin, Brandon Mayer, and Bryan Perozzi. Graphworld: Fake graphs bring real insights for gnns. *arXiv preprint arXiv:2203.00112*, 2022. 12

[39] Zhilin Yang, William Cohen, and Ruslan Salakhudinov. Revisiting semi-supervised learning with graph embeddings. In *International conference on machine learning*, pages 40–48. PMLR, 2016. 12

[40] Benedek Rozemberczki, Carl Allen, and Rik Sarkar. Multi-scale attributed node embedding. *Journal of Complex Networks*, 9(2):cnab014, 2021. 12

[41] Hongbin Pei, Bingzhe Wei, Kevin Chen-Chuan Chang, Yu Lei, and Bo Yang. Geom-gcn: Geometric graph convolutional networks. *arXiv preprint arXiv:2002.05287*, 2020. 12

[42] Marc Brockschmidt. Gnn-film: Graph neural networks with feature-wise linear modulation. In *International Conference on Machine Learning*, pages 1144–1152. PMLR, 2020. 12, 13

[43] Matthias Fey and Jan E. Lenssen. Fast graph representation learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019. 13

[44] Jiaxuan You, Zhitao Ying, and Jure Leskovec. Design space for graph neural networks. *Advances in Neural Information Processing Systems*, 33:17009–17021, 2020. 13

A Appendix

A.1 Experimental Details

Artificial Graphs Generation. Artificial graphs are generated using the stochastic block model (SBM) as implemented by Palowitch et al. [38], which enables control of certain graph properties, namely edge density, homophily, and feature SNR (a metric of feature homogeneity between classes, which equals to 1 in homogeneous scenarios and increases with heterogenity, i.e. as features become more separable). Please refer to [38] for more framework-specific details. We generate graphs with 1000 nodes. Each node is assigned a label to create a class-balanced binary node classification problem, i.e., each graph comprises 500 nodes of each class. All artificially generated graphs are available as supplementary material[1] for repeatability.

Benchmarks. Four real-world datasets with different properties are also selected to extend our experiments and for benchmarking purposes. Cora [39] and CiteSeer [39] are homophilic citation networks; Chameleon [40] and Texas [41] are heterophilic graphs, where nodes correspond to web pages and edges to hyperlinks between them. A complete description of these datasets, including relevant properties, can be found in Table 5.

Table 5: Graph properties of benchmark datasets.

Dataset	#N	#E	#C	H	D_e
Cora [39]	2485	10138	7	0.81	0.002
CiteSeer [39]	2120	7358	6	0.74	0.002
Chameleon [40]	2277	65019	5	0.23	0.013
Texas [41]	183	558	5	0.11	0.017

N: Nodes; E: Edges; C: Classes; H: Homophily; D_e: Edge density

Models and Evaluation. We consider the GCN [27] as a base for all experiments. More complex layer and model types are also used to extend our analyses: reversible GCN (RevGCN) [20], due to its robustness to oversmoothing even for deep GNNs; APPNP [35] and FiLM convolution [42] which seem to perform adequately for input graphs that belong to different parts of the graph properties spectrum proposed in [38], where vanilla-GCNs do not. Reported results refer to performance (accuracy) on the test sets (best epoch on the validation set, averaged over 10 runs). Configuration files are available as supplementary material for repeatability.

[1] https://github.com/dsg95/decoupling-graph-info

Table 6: Graph convolution layers of RevGCN, APPNP and FiLM. f_θ and g_θ correspond to learnable functions.

Method	Layer
RevGCN [20]	$\mathbf{H}^l = \hat{\tilde{\mathbf{A}}} \cdot \mathit{Dropout}(\mathit{ReLU}(\mathit{Norm}(\mathbf{H}^{l-1}))) \cdot \mathbf{W}^l$
APPNP [35]	$\mathbf{Z}^l = (1-\alpha)\hat{\tilde{\mathbf{A}}}\mathbf{Z}^{l-1} + \alpha\mathbf{H}, \quad \mathbf{Z}^0 = \mathbf{H} = f_\theta(\mathbf{X}), \quad \alpha \in (0, 1]$
FiLM [42]	$\mathbf{H}^l = \sigma(\gamma^{l-1}\hat{\tilde{\mathbf{A}}}\mathbf{H}^{l-1}\mathbf{W}^l + \beta^{l-1}), \quad \beta^{l-1}, \gamma^{l-1} = g_\theta(\mathbf{H}^{l-1})$

Implementation. All neural network architectures consist of a single linear layer for feature transformation into 8 channels, followed by 2 message-passing layers (or fully connected layers for the MLP baseline), and a node classification head (linear layer). We use the PyG library [43] and all experiments are run using GraphGym framework [44], which we extended to include more advanced architectures (new layer types or models) when needed. We performed mini-batch training using neighbourhood sampling (batch size of 32) and considered a train-validation-test split of 80%-10%-10%.

A.2 Component Interplay Measurement Details

Interpretation of SC and FC. The intuition behind SC and FC is to provide simple and compact measures of the contribution of each component, by quantifying how much we gain/lose with the original/random versions. Thus, we measure SC (Equation 2) by summing how much we lose by destroying structure (compared to the original graph) with how much we gain from the original structure (compared to the analogous random graph). Similarly for FC, in Equation 3, the first coefficient measures how much lose by destroying the feature signal and the second measures how much we gain from the original feature distribution (using the original and random graphs as references, respectively).

A.2.1 Regression Results

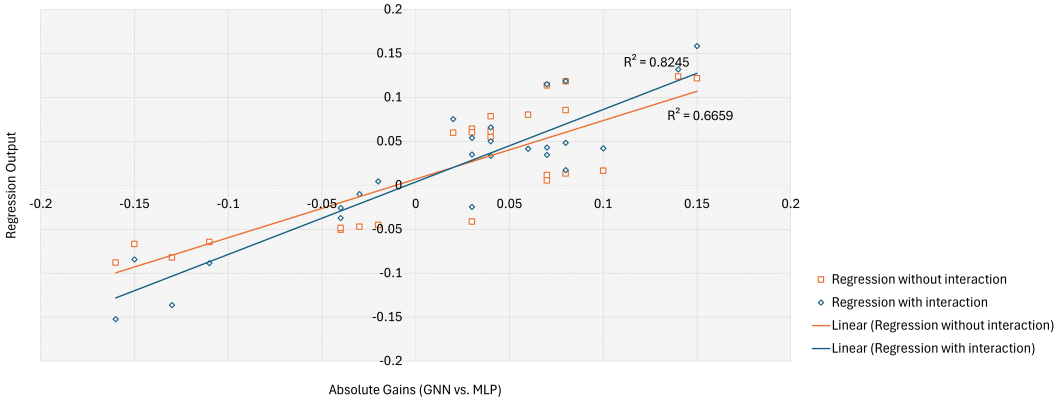

Figure 4: Regression plots with and without considering variable interaction (SC×FC). A significant better fit is achieved when interaction is considered. Further details on the statistical significance of these curves can be found in Tables 7 and 8.

Table 7: Regression without interaction: summary of statistics.

Regression Statistics		
	F	*Significance F*
F-Statistic	22.9	3.35e-6
	Coefficients	**P-value**
Intercept	-0.0955	6.93e-4
Structure Component (SC)	0.231	3.21e-5
Feature Component (FC)	0.146	8.14e-3

Table 8: Regression with interaction: summary of statistics.

Regression Statistics		
	F	*Significance F*
F-Statistic	34.4	1.73e-8
	Coefficients	**P-value**
Intercept	-0.179	6.61e-7
Structure Component (SC)	0.930	7.74e-6
Feature Component (FC)	0.435	7.58e-6
Interaction (SC $\times$ FC)	-1.80	1.97e-4

Table 9: Regression: data points (structure and feature components) and target (absolute gains of GNN compared to MLP).

Dataset	Model	SC	FC	Absolute Gains
Artificial (informative)	GCN	0.40	0.56	0.07
Artificial (homophily=0.2)	GCN	0.23	0.37	0.04
Artificial (homophily=0.5)	GCN	0.02	0.18	-0.11
Artificial (homophily=0.8)	GCN	0.22	0.42	0.10
Artificial (density=0.003)	GCN	0.16	0.44	0.07
Artificial (density=0.03)	GCN	0.22	0.42	0.10
Artificial (density=0.15)	GCN	0.25	0.35	0.08
Cora	GCN	0.69	0.41	0.14
CiteSeer	GCN	0.64	0.42	0.07
Chameleon	GCN	-0.02	0.34	-0.04
Texas	GCN	0.02	-0.02	-0.16
Artificial (informative)	RevGCN	0.23	0.71	0.04
Cora	RevGCN	0.31	0.75	0.08
CiteSeer	RevGCN	0.25	0.81	0.06
Chameleon	RevGCN	-0.03	0.37	-0.04
Texas	RevGCN	-0.04	0.26	-0.15
Artificial (informative)	APPNP	0.30	0.62	0.03
Cora	APPNP	0.70	0.38	0.15
CiteSeer	APPNP	0.66	0.42	0.08
Chameleon	APPNP	-0.06	0.44	-0.02
Texas	APPNP	0.02	0.06	-0.13
Artificial (informative)	FiLM	0.19	0.73	0.04
Cora	FiLM	0.22	0.72	0.03
CiteSeer	FiLM	0.18	0.78	0.02
Chameleon	FiLM	-0.05	0.41	-0.03
Texas	FiLM	0.02	0.34	0.03

A.3 Limitations and Future Work

Our results suggest that we should not only explore feature/structure co-dependence but also how models respond to certain combinations of graph properties. This scenario was not yet explored in our controlled experiments. We also relied on artificial graphs for which we occasionally make an assumption of how meaningful is their information based on theoretical criteria, which can limit

the empirical conclusions drawn upon those graphs. These limitations are tied to the fact that we only experimented with extreme scenarios. Conducting more experiments could further validate our assumptions by considering more demanding and diverse conditions, for both artificial graphs and benchmarks. Furthermore, our method for destroying structure information does not destroy all structural properties (e.g. edge density remains the same); this fact should not affect in the conclusions of this work, but can limit further applications.

As future work, we must deepen our insights on how models respond to graph properties by extending our method to more complex scenarios, including combinations of structural properties (e.g. simultaneous variation of sets of properties) and feature information. We shall also explore the potential of the feature/structure decoupling method as an empirical indication of how informative graph structures are to a certain network architecture, as such methods are still in demand.