PROBING GRAPH REPRESENTATIONS

Mohammad Sadegh Akhondzadeh & Vijay Lingam & Aleksandar Bojchevski
CISPA – Helmholtz Center for Information Security
{mohammad.akhondzadeh,vijay.lingam,bojchevski}@CISPA.de

ABSTRACT

Today, we have a good theoretical understanding of the representational power of Graph Neural Networks (GNNs). For example, their limitations have been characterized in relation to a hierarchy of Weisfeiler-Lehman (WL) isomorphism tests. However, we do not know what is encoded in the learned representations. This is our main question. We answer it using a probing framework to quantify the amount of meaningful information captured in graph representations. Our findings on molecular datasets show the potential of probing for understanding the inductive biases of graph-based models. We compare different families of models and show that transformer-based models capture more chemically relevant information compared to models based on message passing. We also study the effect of different design choices such as skip connections and virtual nodes. We advocate for probing as a useful diagnostic tool for evaluating and developing graph-based models.

1 INTRODUCTION

In this paper we ask a deceptively simple question: *What is encoded in the representations learned by graph-based models?* For example, if we use a Graph Neural Network (GNN) to predict the toxicity of a molecule, will the hidden representation contain information about the number of hydrogen atoms or the presence of aromatic rings? Prior works put constraints on the set of possible answers. For a GNN that is only as expressive as a 1-Weisfeiler-Leman test (1-WL), the information necessary to distinguish between graphs higher in the WL-hierarchy is provably not encoded. This fact has inspired a series of works that aim to build even more powerful and more expressive GNNs (Xu et al., 2019; Morris et al., 2019). Nonetheless, just because a GNN is theoretically able to distinguish between two graphs, there is no guarantee that it will learn to do so in practice. Further limitations such as over-smoothing (Oono & Suzuki, 2020) and over-squashing (Alon & Yahav, 2021) stemming from the message passing paradigm impose additional constraints. Graph Transformer models (Ying et al., 2021; Kim et al., 2022a) aim to alleviate these limitations using a variety of structural and positional biases with impressive results on downstream tasks. It is natural to ask if their success is partly due to their ability to better capture the relevant (e.g. chemical) knowledge.

A trivial answer to our opening question is that the architecture, the dataset, the learning task, the optimization algorithm, and all other relevant design choices (e.g. data augmentation) jointly determine what the model learns to encode in the latent representations. Since theoretically characterizing the effect of these choices is challenging, we tackle the question with a rigorous empirical analysis. Specifically, we adopt the *probing* framework (Belinkov, 2022), which has proven useful in answering similar questions in the natural language processing domain (e.g. is the length of a sentence encoded in its representation). The idea behind probing is simple. If we can reliably extract a given property (e.g. the presence of aromatic rings) from a given representation (e.g. the output of the penultimate layer in a GNN), we can conclude that the information about that property is encoded in the representation. Defining what it means to reliably extract a property leads to different flavors of probing. In its simplest form, a so called linear probe evaluates whether a linear classifier can accurately predict the property given the fixed representations as input.

In this paper, we use this powerful framework to compare different families of models, e.g. traditional GNNs vs. Graph Transformers, and the effect of different modelling choices, e.g. using skip connections or virtual nodes. We focus on probing molecular representations since molecular prediction tasks are one of the main testbeds to benchmark graph-based models. We probe for chem-

ical knowledge such as information about the atoms, the presence of important functional groups, properties related to the molecule's 3D structure, and other high-level properties.

To obtain a more complete picture, we rely on two complementary probing strategies. We start with linear probing since the findings are the easiest to interpret. Next, we use a paired probing strategy to intervene on an input that directly changes a property of interest (e.g. by deleting a functional group) to isolate its effect on the representation. Additionally, we study the correlation between the richness of the representations space – quantified via probing – and the transferability of the representations to other downstream tasks.

Unsurprisingly, models that perform better tend to encode more relevant information in their representations. Nonetheless, it is helpful to quantify the effect of different design choices, e.g. we find that models with skip connections capture more information. Interestingly, we find that even randomly initialized models can extract useful features, e.g. they encode as much information about certain properties as pretrained models. This raises questions regarding the adequacy of common evaluation protocols.

We advocate for the use of probing as a diagnostic tool to better understand how and what is learned by our models. For example, if certain properties are reliably extractable, even for out-of-distribution graphs, our model might be able to generalize better. It is especially important to verify whether the model learns properties that are known to be important based on domain knowledge. We also investigate the acquisition of knowledge during training to understand which properties are more easily learned and how they relate to the downstream task. This highlights the potential of probing as a debugging tool to aid the development of better models. Similarly, probing can help us with the selection of (pretrained) models. Finally, even though it is not the focus of this work, probing can surface properties that might be necessary to solve a certain task. If all accurate models inevitably learn a certain property it may be causally related to the prediction.

2 Probing Methodology

Let $f : \boldsymbol{x} \mapsto y$ be a model that maps an input $\boldsymbol{x}$ to an output y. We consider functions f that generate intermediate (hidden) representations of $\boldsymbol{x}$. Let $f_l(\boldsymbol{x}) = \boldsymbol{z}, \boldsymbol{z} \in \mathbb{R}^d$ refer to the d-dimensional output of layer l in a neural network such as a GNN or a Graph Transformer. To simplify the exposition, we focus on the case where the input $\boldsymbol{x}$ is a molecule, but our approach is easily applicable to other tasks. Similarly, $\boldsymbol{z}$ may also refer to learned attention weights or any other intermediate output.

Our goal is to understand what kind of information is encoded in the representation $\boldsymbol{z}$. To do so we use a probing dataset $\mathcal{D} = \{f_l(\boldsymbol{x}_i), p_i\}_{i=1}^N = \{\boldsymbol{z}_i, p_i\}_{i=1}^N$ where p_i is a property of interest. For example, $\boldsymbol{x}_i$ is a molecule, $\boldsymbol{z}_i$ is the output of the second layer of a GNN, and p_i is the presence of aromatic rings. The probing dataset is independent of the datasets used to train and evaluate the original model f. We can efficiently compute many relevant properties, which means that given any unlabeled set of molecules we can automatically create a probing dataset $\mathcal{D}$. For other properties we will rely on existing annotated (labeled) datasets.

2.1 Probing Properties

We focus on four different types of properties when probing molecular representations with the goal of evaluating whether the model learns chemically relevant information.

Atom Counting. For the first set of properties, we simply consider the number of different atoms in a molecule. We compute the number of Carbon, Nitrogen, and Oxygen atoms since they are the most common atoms in organic compounds. Therefore, here $p_i \in \mathbb{N}, p_i \geq 0$.

Meaningful Substructures. Second, we rely on chemical domain knowledge to find meaningful substructures in a molecule. We explore eight different substructures including: Aromatic Carbocycles, Aromatic Rings, Saturated Rings, Aniline, Benzene, Bicycle, Ketone, Methoxy, Para Hydroxylation, and Pyridine. These substructures are often referred to as functional groups since they usually have their own characteristic properties, regardless of the other atoms present in a molecule. For example, as the name suggests, the presence of aromatic rings changes the odor of a molecule. Using the RDKit tool (Landrum, 2016) we compute whether a molecule contains one of these functional groups, thus $p_i \in \{0, 1\}$. To illustrate the principle, we only probe for these eight properties,

but our approach scales to many more properties. Using RDKit, we can compute over 80 different functional groups. Similarly, RDKit can compute other chemically relevant properties such as the number of radical electrons or the number of rotatable bonds.

Molecular Properties. Next, we probe the model representations for high-level molecular properties that are different from the property/label used to train the original model f. For example, using the PCQM4Mv2 dataset (Hu et al., 2020), models are trained to predict the HOMO-LUMO energy gap of molecules (calculated with density functional theory) given their 2D molecular graphs. For probing we first consider the MoleculeNet benchmark (Wu et al., 2017) which contains multiple datasets with various tasks derived from different properties such as blood-brain permeability, toxicity, and lipophilicity. Some of the properties are continuous, $p_i \in \mathbb{R}$, others are binary or categorical $p_i \in \{0, \dots, C\}$ where C is the number of categories. We also probe for information about the smell of the molecule, e.g. sweet or woody (32 smells in total), using the Pyrfume data archive (see § A.3). In contrast to the two previous property types (atom counting and meaningful substructures), which we can compute for any molecule, these properties require labeled datasets. Therefore, this setup is closely related to transfer learning. Our probes can be seen as evaluating whether the representations trained on a source task transfer to a different (target) task.

3D Properties. Finally, since 3D geometric information plays an important role in determining the function of a molecule (Gilmer et al., 2017), we explore whether the learned representations capture any properties related to the 3D-structure of the molecules. Note, the models are trained using only the molecular graph as input without any 3D input. Using RDKit, we compute five 3D properties: Asphericity, Radius of Gyration, Sphericity Index, NPR1 (normalized first principal moments ratio), and PMI1 (smallest principal moment of inertia). All of these properties are continuous ($p_i \in \mathbb{R}$).

2.2 Probing Strategies

Linear Probing. The goal of probing is to evaluate the "extractability" or "readability" of a property from the representation. The standard approach is to train a separate model, called a probe, to *predict* the property p_i given the fixed representations $\boldsymbol{z}_i$. Specifically, the probing dataset $\mathcal{D}$ is split into a train and test set, the probe is trained on the train set, and its performance is evaluated on the test set. Good test performance is taken as evidence that the representation contains information about the property. Low performance indicates that the property is either not present in the representations or not usable. The idea of usability is prominent in the literature. The advocates of linear probing (Alain & Bengio, 2017) argue that the probe model should be simple, e.g. a logistic regression (or linear regression for continuous properties), since this means that the information can be easily extracted and used in subsequent processing (e.g. in subsequent layers). The advocates for more complex probes Pimentel et al. (2020b) argue they are better since the information about the property may be non-linearly encoded in the representation. Yet, the good performance of non-linear probes may come from overfitting (memorization of spurious correlations). To overcome these limitations, various control tasks have been proposed: comparing the performance to a majority baseline, random representations, randomization of the properties, or the use of minimum description length. Despite its limitations, in this paper we use linear probing since the results are more interpretable. If a linear probe has good performance there exists a hyperplane in representation space separating the inputs based on their properties.

Pairwise Probing. To better isolate the effect of a certain property on the representation we propose pairwise probing. The main idea is to create a probing dataset $\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{x}'_i)\}_{i=1}^{N}$ of pairs of molecules such that each pair is as similar as possible, differing *only* in the property of interest. We demonstrate this idea for properties corresponding to meaningful substructures (functional groups). Assume that the property $p \in \{0, 1\}$ is binary. First, we create a set $\mathcal{S}$ of molecules such that $p_i = 1$ for each $\boldsymbol{x}_i \in \mathcal{S}$. In other words, all of the molecules in the set S contain a given substructure (e.g. all molecules contain a Benzene ring). Then, for each $\boldsymbol{x}_i$ we create its corresponding pair molecule $\boldsymbol{x}'_i$ by removing the substructure of interest. We make sure that the resulting molecule is still valid and add hydrogen atoms to fill the capacity of the atom(s) previously connected to the functional group. This means that by design for each $\boldsymbol{x}'_i$ the property $p'_i = 0$, and that each pair $(\boldsymbol{x}_i, \boldsymbol{x}'_i)$ is minimally different except for the property of interest. We call $\boldsymbol{x}_i$ the source molecule and $\boldsymbol{x}'_i$ the target. Fig. 1 shows an example of this approach for the amide functional group.

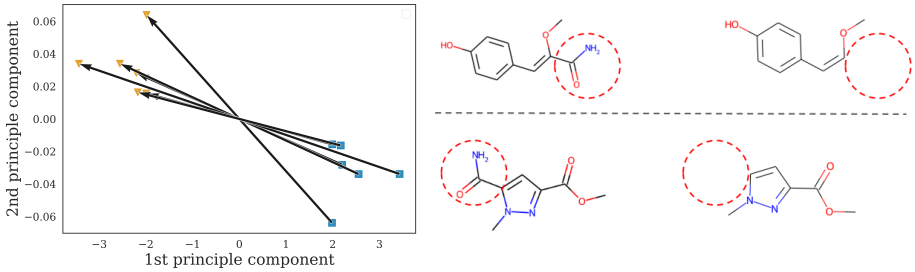

Figure 1: Pairwise probing when removing the amide functional group (dotted circle). Arrows connect each source molecule to its target. The directions are highly aligned. The first principal component is sufficient to separate molecules with (squares) and without (triangles) an amide group.

We can use the pairwise property in several ways. First, inspired by the idea of semantic analogy for word embeddings Mikolov et al. (2013), we evaluate whether the vectors connecting the representations of the source molecules to their targets, i.e. $\boldsymbol{v}_i = f(\boldsymbol{x}_i) - f(\boldsymbol{x}'_i) = \boldsymbol{z}_i - \boldsymbol{z}'_i$ are aligned with each other. To do so, we compute the cosine similarity between each $(\boldsymbol{v}_i, \boldsymbol{v}_j)$ pair. If the similarity is high on average it means that there is a consistent direction in the representation space that identifies the property of interest. Similar to linear probing, the hyperplane perpendicular to this direction can separate source from target molecules. We also perform a principal component analysis (PCA) to identify whether the directions of highest variation are aligned with the property (example in Fig. 1, details in § A.2). Another way to use the paired dataset is to discard the pairing information and use $\mathcal{D}$ for linear probing. The benefit is that, unlike before, the dataset is now balanced w.r.t. p_i, making the probe models easier to train.

As suggested by Amini et al. (2022), our paired probing strategy can be interpreted as causal probing by considering each $\boldsymbol{x}'_i$ as the counterfactual of $\boldsymbol{x}_i$ after intervention on the property. If we make assumptions for the Structural Causal Model (SCM) we can estimate the causal effect of the property. Similar to Yang et al. (2022), we assume our generative process comes from a causal DAG, where E, F, M, Z and R are random variables corresponding to the environment, the functional group, the molecule, the representation, and the remaining structure respectively. Using observational data to compute $p(Z \mid F = 1)$ we obtain biased estimates since there is a backdoor path $F \leftarrow E \rightarrow R \rightarrow M \rightarrow Z$ from F to Z. However, by intervening on F as we propose we can block the backdoor path. Then, using the paired dataset we can estimate the average treatment effect $p(Z \mid \text{do}(F = 1)) - p(Z \mid \text{do}(F = 0))$ computing $\boldsymbol{v}_{\text{ate}} = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{v}_i = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{z}_i - \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{z}'_i$.

To be clear, the SCM relies on strong assumptions and we do not argue that it necessarily represents the true data-generating distribution. Nonetheless, the implications given those assumptions are insightful – the average treatment effect $\boldsymbol{v}_{\text{ate}}$ captures the direction from molecules with a given property to molecules without it. We verify whether $\boldsymbol{v}_{\text{ate}}$ is aligned with the source-target directions by computing $C^{\text{pair}}_{\text{ate}}$ the average cosine similarity between $\boldsymbol{v}_{\text{ate}}$ and $\boldsymbol{v}_i$.

3 Probing for Molecular Properties

We study popular message-passing GNNs and Graph Transformer architectures.[1] For the message-passing models, we use GCN (Kipf & Welling, 2016) and GIN (Xu et al., 2019). GIN is theoretically more expressive. For the transformer models, we study Graphormer (Ying et al., 2021), GRPE (Park et al., 2022), GraphGPS (Rampášek et al., 2022), and TokenGT (Kim et al., 2022b). We train all models on the large scale PCQM4Mv2 dataset (Hu et al., 2021), and we probe their hidden representations for various properties. As a baseline, we also probe the raw features (using sum of input features across nodes as a representation), and Morgan fingerprints (Morgan, 1965) – manually crafted representations based on domain knowledge. We create the probing dataset $\mathcal{D}$ from the PCQM4Mv2 dataset. We randomly sample 100K molecules from the training and 20K molecules

[1] Our code is publicly available at `https://github.com/msadegh97/probing-graph-representation`.

Table 1: Linear probing performance (AUC score) across models for various functional groups.

	Arom. Carb.	Arom. Ring	Satur. Ring	Aniline	Benzene	Bicyclic	Ketone	Methoxy	ParaHydrox.	Pyridine	AVG
Raw	83.5	97.2	84.0	78.8	83.5	89.6	81.1	80.6	83.1	82.5	84.4
Morgan	83.0	85.5	76.7	71.4	83.0	72.6	68.0	72.9	70.9	70.3	75.4
GCN	98.1	99.3	87.8	89.9	98.1	90.1	98.1	82.3	91.1	90.4	92.5
GIN	87.6	96.1	84.6	71.3	87.6	80.2	73.8	62.6	77.4	77.7	79.9
Graphormer	99.5	99.8	90.6	98.6	99.5	95.5	99.5	86.7	96.9	98.4	96.5
GRPE	100.0	100.0	99.7	100.0	100.0	99.0	99.9	99.2	99.1	99.3	**99.6**
GraphGPS	99.6	100.0	92.3	98.2	99.6	96.0	99.0	93.1	97.0	95.8	97.1
TokenGT	99.8	100.0	97.6	99.8	99.8	98.7	99.9	99.1	98.6	99.3	99.3

Table 2: Linear probing performance (R2 score) across models for atom and 3D structure properties.

	3D Structure Properties					Counting Properties		
	Asphericity	NPR1	PMI3	RadiusOfGyration	SpherocityIndex	# Carbon	# Oxygen	# Nitrogen
Raw	0.152	0.131	0.734	0.827	0.174	0.033	-0.850	0.000
Morgan	0.178	0.160	0.628	0.706	0.155	0.031	0.000	0.000
GCN	N/A	N/A	N/A	N/A	N/A	0.586	N/A	-6.655
GIN	-52.922	-38.858	-278.344	-192.431	-0.016	0.555	-0.037	-0.016
Graphormer	0.376	0.364	0.469	0.489	0.386	0.627	0.536	0.631
GRPE	0.490	0.457	0.757	0.874	0.469	0.729	0.196	0.221
GraphGPS	0.268	0.275	0.803	0.845	0.332	0.898	0.746	0.806
TokenGT	0.487	0.490	0.757	0.779	0.433	0.710	0.665	0.769

from the validation set, thus maintaining a scaffold-split. For binary properties we report the ROC AUC score and for ordinal/continuous properties we report the R2 score. See § A.1 for more details.

Impact of the Architecture. In Table 1 we observe that on average transformer-based models tend to capture more information about the functional group properties – they learn more chemically relevant knowledge. Even though GIN is theoretically more expressive than GCN it performs significantly worse. The raw features perform surprisingly well. In Table 2 we report the R2 score for atom counting and 3D structure properties. Values closer to 1 are better and N/A indicates that the probe was not able to converge. Again, we see that the transformer-based models perform significantly better. Interestingly, some models are able to capture a good amount of 3D information even though no explicit 3D structure is provided during training.

We continue with linear probing for high-level molecular properties using the MoleculeNet benchmark. In Table 3 we show the average performance and a subset of the properties (see § A.3 for all, and additional results for odor properties). Again, transformer-based models perform better on average. This bolsters our hypothesis that they learn richer representations allowing them to better transfer to other tasks. Furthermore, in Fig. 2 we see a strong correlation between the average probing performance w.r.t. the functional groups and the average performance on MoleculeNet.

Training Dynamics. Next, we study how the amount of encoded information changes during training and across the representations learned in different layers. In Fig. 3 we plot a 2D heatmap for different epochs (horizontal) and layers (vertical). Darker shade is better. For both GCN and GraphGPS we see that already in the first layer we have significantly more information compared to the input features (layer 0). This tells us that accounting for the graph structure is beneficial. The information is maintained in the intermediate layers, and is slowly removed as we move towards the final layer.

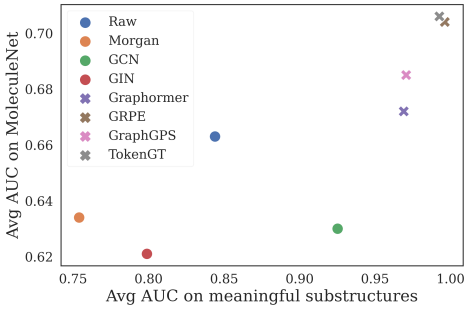

Figure 2: Substructures vs. high-level properties.

Table 3: Linear probing for high-level molecular properties.

Dataset	BBBP	ToxCast	Sider	ClinTox	BACE	AVG
Molecules	2,039	8,575	1,427	1,478	1,5113	
Tasks	1	617	27	2	1	
Raw	66.8	60.5	56.2	60.6	75.1	66.3
Morgan	63.2	54.1	60.5	57.4	75.0	63.4
GCN	56.5	57.0	55.2	64.9	71.1	63.0
GIN	58.8	55.7	53.2	57.0	77.1	62.1
Graphormer	59.6	59.2	57.5	78.5	63.0	67.2
GRPE	64.9	61.1	55.7	82.8	77.9	70.4
GraphGPS	63.9	60.0	60.7	72.7	74.5	68.5
TokenGT	57.1	59.3	58.4	88.5	75.2	**70.6**

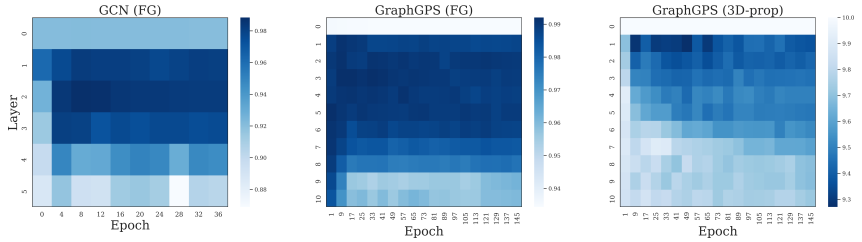

Figure 3: Average probing performance across epochs and layers for two models and two types of properties.

One reason for this is that the representations in the final layer are becoming more specialized in order to solve the original task.

The functional group properties (left and middle column) are learned right away and the performance does not significantly change over epochs. For GraphGPS the performance is high even for epoch 0, i.e. for the randomly initialized model before any training is done, while for GCN there is a noticeable increase from epoch 0 to epoch 1. For the 3D properties, we see that in the first layer the performance starts high and slightly decreases over time. However, the performance in the subsequent layers is significantly lower at the start and it increases over time. Again, the layers close to the output encode less information.

Randomly Initialized Models. We also compare the performance of representations obtained from randomly initialized models (without pretraining) and the models pretrained on PCQM4Mv2. In Table 4 we see that the untrained models are surprisingly good feature extractors. For the functional group properties they even perform on par or better than pretrained models. The random GraphGPS model also performs well for 3D properties. These results highlight the usefulness of the models' inductive biases.

Table 4: Average performance of random vs. pretrained models.

	FG (AUC)		3D-Prop (R2)	
Model	**Random**	**Train**	**Random**	**Train**
GCN	0.905	0.925	N/A	N/A
GIN	0.897	0.799	-0.552	-139.4
Graphormer	0.975	0.965	-0.279	0.417
GRPE	0.996	0.995	0.186	0.610
GraphGPS	0.981	0.971	0.541	0.505
TokenGT	0.963	0.993	0.083	0.589

Impact of the Pretraining Dataset. Next, we assess the impact of the pretraining dataset on the usability of the learned representations. For a fair comparison, we use the Molhiv and Molpcba datasets (Hu et al., 2020) for pretraining since both contain small molecules with similar molecule statistics (see § A.5). We also select a random subset of Molpcba with the same size as Molhiv (called Molpcba_s). This will help us disentangle the effect of the dataset size and the effect of multitask learning – in Molhiv we predict a single label and in Molpcba we predict 128 labels. We perform linear probing on the MoleculeNet benchmark and show the results in Table 5. We see that the performance when pretraining on Molpcba tends to be higher on average for most models. This is true even for the smaller subset and indicates that multitask learning might be beneficial since it results in a richer representation space.

Table 5: The effect of the pre-training dataset.

Model	**Dataset**	**BBBP**	**ToxCast**	**Sider**	**ClinTox**	**BACE**	**AVG**
GCN	**Molhiv**	62.5	59.4	60.1	68.9	79.6	**66.1**
	Molpcba_s	65.8	61.0	59.1	79.0	73.9	**67.8**
	Molpcba	65.0	61.4	59.4	71.4	71.7	65.8
GIN	**Molhiv**	56.4	54.3	55.2	63.7	62.5	58.4
	Molpcba_s	62.0	63.0	60.3	69.9	71.1	65.3
	Molpcba	62.1	63.5	60.0	62.7	74.5	64.5
V-GCN	**Molhiv**	59.5	56.8	58.5	68.1	74.4	63.5
	Molpcba_s	64.7	60.1	58.4	69.9	72.2	65.1
	Molpcba	62.5	60.3	60.2	71.1	69.6	64.8
V-GIN	**Molhiv**	55.4	54.8	56.0	69.2	66.7	60.4
	Molpcba_s	62.8	62.7	58.8	67.8	77.4	65.9
	Molpcba	64.7	65.2	61.4	69.2	78.0	**67.7**

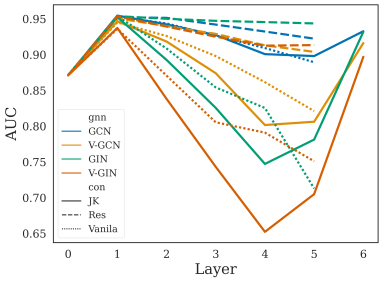

Figure 4: Design choices' impact: jumping knowledge (JK) / residual connections (Res).

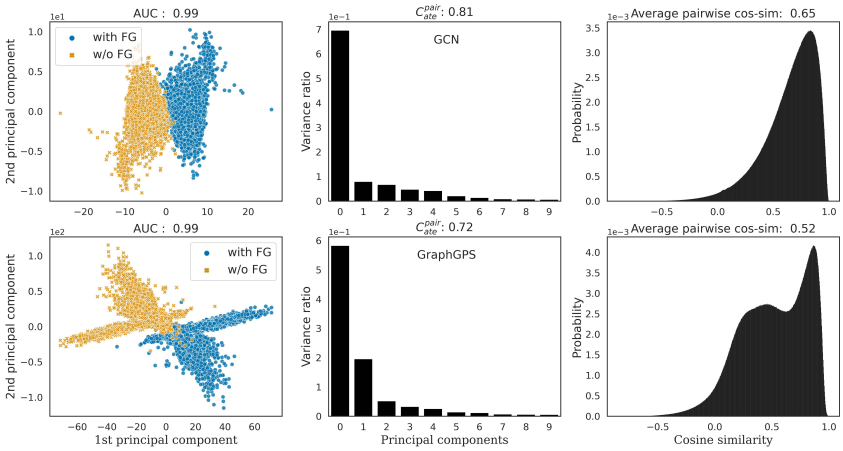

Figure 5: Pairwise probing for GCN (top) and GraphGPS (bottom). We show the projection on the top two principal components (left), the ratio of explained variance per component (middle), and the pairwise cosine similarity (right).

Impact of Design Choices. We study the impact of different design choices for the architecture of a GNN model. First, we consider residual connections (Res) and jumping knowledge (JK). For this experiment we pretrained on Molhiv. Fig. 4 shows the average linear probing performance for all the meaningful substructure properties for networks trained with and without JK, and with and without Res. We consider the representations learned in each layer. We see that that using Res (dashed lines) helps preserve more information about the property across different layers. Compared to the vanilla model (dotted lines) where the information steadily drops with each layer (see also findings on training dynamics). The representations learned by the models using JK (solid lines) encode less information than the respective vanilla models, except at the last layer. This is expected since the last layer is a concatenation of all previous layers. For all models and variants the first layer (after one round of message passing) encodes the most information about the property. We also study the impact of two pooling strategies, namely the default mean pooling (GCN, GIN) vs. virtual nodes (V-GCN, V-GIN). We see that on average mean pooling results in representations that encode significantly more information about functional groups compared to the representations of the virtual node.

Pairwise Probing. Finally, we perform our pairwise probing strategy with the goal of explicitly isolating the effect of a certain property on the learned representations. Here we focus on the presence of meaningful substructures (functional groups). We select one representative model from each family, namely GCN (top) and GraphGPS (bottom), and we show the results in Fig. 5 for the nitro group. See § A.7 for other models and properties. In the left column we set the projection of the representations onto the top two principal components (PCs). We see that even a single principal component is sufficient to separate the source molecules (with nitro) from their counterparts (without nitro). This means that the direction of highest variance in the representation space is highly aligned with the property of interest. The middle column in Fig. 5 shows the ratio of the captured variance for each PC. We see that the first PC captures more than half of the total variance.

We also compute the cosine similarity between all pairs of vectors connecting the representations of source and target molecules (all $\boldsymbol{v}_i, \boldsymbol{v}_j$) and we plot a histogram of the similarity distribution (right column in Fig. 5). We see that the source-target directions in the representation space are highly aligned. This is also reflected in the average pairwise similarity shown on top of the plot.

Next, we replicate our linear probing strategy, this time using the pairwise data instead of randomly sampled data to train the linear probe. We show the AUC on top of the first column. Compared to standard linear probing we get higher performance (see side-by-side comparison in § A.7). The main reason is that the pairwise probing dataset is balanced by construction making the linear probe easier to train. In contrast, if we randomly sample the probing dataset as before the ratio of molecules with and without a given functional group is highly skewed. Nonetheless, both probing strategies mostly show the same trends when comparing different models and properties. Under the causal interpretation of pairwise probing, we can compute the average treatment effect $\boldsymbol{v}_{\text{ate}}$. We show that it is also highly aligned with the source-target directions as evidenced by the large average cosine

similarity (see $C_{\text{ate}}^{\text{pair}}$ on top of the middle column in Fig. 5). We also study the average treatment effect of removing a functional group on other high-level molecular properties. The results for odor properties are given in § A.8 due to lack of space. We are able to recover well-known findings about the relations between certain substructures and the smell of a molecule.

4 Related Work

We give a brief overview of probing. For a comprehensive survey, including a discussion of its limitations, see Belinkov (2022). We also discuss GNN explainability since probing can provide valuable insights on the inductive biases of the models, which can be seen as explanations.

Probing. The idea of using probing tasks to quantify the information encoded in the representations of a computational system was first proposed by neuroscientists (Cox & Savoy, 2003; Kamitani & Tong, 2005; Mitchell et al., 2008) and was later employed in machine learning research, especially for the study of large language models (Alain & Bengio, 2016; Adi et al., 2016; Conneau et al., 2018; Liu et al., 2019). One line of research is focused on probing via prediction. As we discussed in § 2.2, selecting the expressive power of the probe is an important choice, with various arguments for linear vs. non-linear probes. Pimentel et al. (2020a) propose pareto probing to take into account the trade-off between probing accuracy and complexity. Ivanova et al. (2021) argue that selecting the probe and its complexity depends on the research goal, e.g. do we aim for usability or predictability. Another line of research studies probing from an information theory perspective. Pimentel et al. (2020c) suggest that probing measures the mutual information between the property and the representations. Voita & Titov (2020) propose a framework based on minimum description length to study how easy is to extract the property. To measure feature usability Hewitt et al. (2021) introduce conditional probing based on $\mathcal{V}$-information (usable information) Xu et al. (2020). Later Pimentel & Cotterell (2021) extend the notion of usable information and introduce Bayesian probing. Finally, Zhou & Srikumar (2021) propose a direct probe that does not rely on classifiers. They consider the geometry of the representation space by using hierarchical clustering.

The overwhelming majority of the probing literature, including the works above, is within the natural language processing domain. There are a few works that study models in computer vision (Alain & Bengio, 2016; Resnick et al., 2019; Caron et al., 2021) and biology (Villegas-Morcillo et al., 2021; Rives et al., 2021; Elnaggar et al., 2020). The preprint by Wang et al. (2022) is the only work that considers graph-based models. In contrast to our work, they focus only on self-supervised learning, they do not consider transformers-based models, and they rely mostly on probing via prediction.

Explainability. Most Graph Neural Networks make black box predictions. As they are used for real-world problems, understanding and explaining their predictions becomes crucial. There are two types of explainability methods. Instance-level methods include gradients, feature attribution, perturbations, decomposition, and surrogate models which studied well before (see Yuan et al. (2020b) for a comprehensive survey). Model-level explanation which is not studied well. One exception is Yuan et al. (2020a) where they employ a generative model to generate the explanation. Zhang et al. (2021) proposes a prototype layer that makes the GNNs self-explainable. There are also works on counterfactual explanations (Lucic et al., 2021; Numeroso & Bacciu, 2021). Probing can be seen as a model-level explanation since we quantify the information about a certain property encoded in the learned representations.

5 Conclusion

To tackle our main question "What is encoded in the representations learned by graph-based models?" we performed an extensive empirical analysis using the probing framework. We studied four types of properties related to: atoms, meaningful substructures (functional groups), molecular properties (different functions and odor), and 3D structure. We employed several complementary probing strategies, including a pairwise probing strategy that aims at directly isolating the effect of the property. Our findings show that transformer-based models learn richer representations that capture more (chemically) relevant information compared to GNNs that use message-passing. Surprisingly, randomly-initialized untrained models also provide useful representations with performance on par with some trained models. We also showed the effect of some design choices, e.g. including skip connections increases the encoded information. We advocate for probing as an explainability tool.

REFERENCES

Yossi Adi, Einat Kermany, Yonatan Belinkov, Ofer Lavi, and Yoav Goldberg. Fine-grained analysis of sentence embeddings using auxiliary prediction tasks. *arXiv preprint arXiv:1608.04207*, 2016.

Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes. *arXiv preprint arXiv:1610.01644*, 2016.

Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=HJ4-rAVtl.

Uri Alon and Eran Yahav. On the bottleneck of graph neural networks and its practical implications. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=i80OPhOCVH2.

Afra Amini, Tiago Pimentel, Clara Meister, and Ryan Cotterell. Naturalistic causal probing for morpho-syntax. *ArXiv*, abs/2205.07043, 2022.

Yonatan Belinkov. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48:207–219, 2022.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pp. 9630–9640. IEEE, 2021. doi: 10.1109/ICCV48922.2021.00951. URL https://doi.org/10.1109/ICCV48922.2021.00951.

Alexis Conneau, German Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. What you can cram into a single vector: Probing sentence embeddings for linguistic properties. *arXiv preprint arXiv:1805.01070*, 2018.

David D Cox and Robert L Savoy. Functional magnetic resonance imaging (fmri)"brain reading": detecting and classifying distributed patterns of fmri activity in human visual cortex. *Neuroimage*, 19(2):261–270, 2003.

Yanai Elazar, Shauli Ravfogel, Alon Jacovi, and Yoav Goldberg. Amnesic probing: Behavioral explanation with amnesic counterfactuals. *Trans. Assoc. Comput. Linguistics*, 9:160–175, 2021. doi: 10.1162/tacl_a_00359. URL https://doi.org/10.1162/tacl_a_00359.

Ahmed Elnaggar, Michael Heinzinger, Christian Dallago, Ghalia Rihawi, Yu Wang, Llion Jones, Tom Gibbs, Tamas Feher, Christoph Angerer, Martin Steinegger, Debsindhu Bhowmik, and Burkhard Rost. Prottrans: Towards cracking the language of life's code through self-supervised deep learning and high performance computing. *CoRR*, abs/2007.06225, 2020. URL https://arxiv.org/abs/2007.06225.

Amir Feder, Nadav Oved, Uri Shalit, and Roi Reichart. Causalm: Causal model explanation through counterfactual language models. *Comput. Linguistics*, 47(2):333–386, 2021. doi: 10.1162/coli_a_00404. URL https://doi.org/10.1162/coli_a_00404.

Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1263–1272. PMLR, 2017. URL http://proceedings.mlr.press/v70/gilmer17a.html.

John Hewitt, Kawin Ethayarajh, Percy Liang, and Christopher D Manning. Conditional probing: measuring usable information beyond a baseline. *arXiv preprint arXiv:2109.09234*, 2021.

Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. *arXiv preprint arXiv:2005.00687*, 2020.

Weihua Hu, Matthias Fey, Hongyu Ren, Maho Nakata, Yuxiao Dong, and Jure Leskovec. Ogb-lsc: A large-scale challenge for machine learning on graphs. *arXiv preprint arXiv:2103.09430*, 2021.

Anna A Ivanova, John Hewitt, and Noga Zaslavsky. Probing artificial neural networks: insights from neuroscience. *arXiv preprint arXiv:2104.08197*, 2021.

Yukiyasu Kamitani and Frank Tong. Decoding the visual and subjective contents of the human brain. *Nature neuroscience*, 8(5):679–685, 2005.

Jinwoo Kim, Tien Dat Nguyen, Seonwoo Min, Sungjun Cho, Moontae Lee, Honglak Lee, and Seunghoon Hong. Pure transformers are powerful graph learners. *ArXiv*, abs/2207.02505, 2022a.

Jinwoo Kim, Tien Dat Nguyen, Seonwoo Min, Sungjun Cho, Moontae Lee, Honglak Lee, and Seunghoon Hong. Pure transformers are powerful graph learners. *arXiv preprint arXiv:2207.02505*, 2022b.

Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

Greg Landrum. Rdkit: Open-source cheminformatics software. 2016.

Nelson F Liu, Matt Gardner, Yonatan Belinkov, Matthew E Peters, and Noah A Smith. Linguistic knowledge and transferability of contextual representations. *arXiv preprint arXiv:1903.08855*, 2019.

Ana Lucic, Maartje ter Hoeve, Gabriele Tolomei, Maarten de Rijke, and Fabrizio Silvestri. Cf-gnnexplainer: Counterfactual explanations for graph neural networks. *arXiv preprint arXiv:2102.03322*, 2021.

Julian Michael, Jan A. Botha, and Ian Tenney. Asking without telling: Exploring latent ontologies in contextual representations. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (eds.), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pp. 6792–6812. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.emnlp-main.552. URL https://doi.org/10.18653/v1/2020.emnlp-main.552.

Tomás Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger (eds.), *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pp. 3111–3119, 2013. URL https://proceedings.neurips.cc/paper/2013/hash/9aa42b31882ec039965f3c4923ce901b-Abstract.html.

Tom M Mitchell, Svetlana V Shinkareva, Andrew Carlson, Kai-Min Chang, Vicente L Malave, Robert A Mason, and Marcel Adam Just. Predicting human brain activity associated with the meanings of nouns. *science*, 320(5880):1191–1195, 2008.

Harry L Morgan. The generation of a unique machine description for chemical structures-a technique developed at chemical abstracts service. *Journal of chemical documentation*, 5(2):107–113, 1965.

Christopher Morris, Martin Ritzert, Matthias Fey, William L. Hamilton, Jan Eric Lenssen, Gaurav Rattan, and Martin Grohe. Weisfeiler and leman go neural: Higher-order graph neural networks. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pp. 4602–4609. AAAI Press, 2019. doi: 10.1609/aaai.v33i01.33014602. URL https://doi.org/10.1609/aaai.v33i01.33014602.

Danilo Numeroso and Davide Bacciu. Meg: Generating molecular counterfactual explanations for deep graph networks. *arXiv preprint arXiv:2104.08060*, 2021.

Kenta Oono and Taiji Suzuki. Graph neural networks exponentially lose expressive power for node classification. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=S1ldO2EFPr.

Wonpyo Park, Woong-Gi Chang, Donggeon Lee, Juntae Kim, et al. Grpe: Relative positional encoding for graph transformer. In *ICLR2022 Machine Learning for Drug Discovery*, 2022.

Tiago Pimentel and Ryan Cotterell. A bayesian framework for information-theoretic probing. *arXiv preprint arXiv:2109.03853*, 2021.

Tiago Pimentel, Naomi Saphra, Adina Williams, and Ryan Cotterell. Pareto probing: Trading off accuracy for complexity. *arXiv preprint arXiv:2010.02180*, 2020a.

Tiago Pimentel, Josef Valvoda, Rowan Hall Maudslay, Ran Zmigrod, Adina Williams, and Ryan Cotterell. Information-theoretic probing for linguistic structure. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel R. Tetreault (eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pp. 4609–4622. Association for Computational Linguistics, 2020b. doi: 10.18653/v1/2020.acl-main.420. URL https://doi.org/10.18653/v1/2020.acl-main.420.

Tiago Pimentel, Josef Valvoda, Rowan Hall Maudslay, Ran Zmigrod, Adina Williams, and Ryan Cotterell. Information-theoretic probing for linguistic structure. *arXiv preprint arXiv:2004.03061*, 2020c.

Ladislav Rampášek, Mikhail Galkin, Vijay Prakash Dwivedi, Anh Tuan Luu, Guy Wolf, and Dominique Beaini. Recipe for a general, powerful, scalable graph transformer. *arXiv preprint arXiv:2205.12454*, 2022.

Abhilasha Ravichander, Yonatan Belinkov, and Eduard H. Hovy. Probing the probing paradigm: Does probing accuracy entail task relevance? In Paola Merlo, Jörg Tiedemann, and Reut Tsarfaty (eds.), *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, EACL 2021, Online, April 19 - 23, 2021*, pp. 3363–3377. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.eacl-main.295. URL https://doi.org/10.18653/v1/2021.eacl-main.295.

Cinjon Resnick, Zeping Zhan, and Joan Bruna. Probing the state of the art: A critical look at visual representation evaluation. *CoRR*, abs/1912.00215, 2019. URL http://arxiv.org/abs/1912.00215.

Alexander Rives, Joshua Meier, Tom Sercu, Siddharth Goyal, Zeming Lin, Jason Liu, Demi Guo, Myle Ott, C. Lawrence Zitnick, Jerry Ma, and Rob Fergus. Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *Proc. Natl. Acad. Sci. USA*, 118(15):e2016239118, 2021. doi: 10.1073/pnas.2016239118. URL https://doi.org/10.1073/pnas.2016239118.

Amelia Villegas-Morcillo, Stavros Makrodimitris, Roeland C. H. J. van Ham, Angel M. Gomez, Victoria E. Sánchez, and Marcel J. T. Reinders. Unsupervised protein embeddings outperform hand-crafted sequence and structure features at predicting molecular function. *Bioinform.*, 37(2): 162–170, 2021. doi: 10.1093/bioinformatics/btaa701. URL https://doi.org/10.1093/bioinformatics/btaa701.

Elena Voita and Ivan Titov. Information-theoretic probing with minimum description length. *arXiv preprint arXiv:2003.12298*, 2020.

Hanchen Wang, Jean Kaddour, Shengchao Liu, Jian Tang, Matt J. Kusner, Joan Lasenby, and Qi Liu. Evaluating self-supervised learning for molecular graph embeddings. *CoRR*, abs/2206.08005, 2022. doi: 10.48550/arXiv.2206.08005. URL https://doi.org/10.48550/arXiv.2206.08005.

Zhenqin Wu, Bharath Ramsundar, Evan N. Feinberg, Joseph Gomes, Caleb Geniesse, Aneesh S. Pappu, Karl Leswing, and Vijay S. Pande. Moleculenet: A benchmark for molecular machine learning. *arXiv: Learning*, 2017.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=ryGs6iA5Km.

Yilun Xu, Shengjia Zhao, Jiaming Song, Russell Stewart, and Stefano Ermon. A theory of usable information under computational constraints. *arXiv preprint arXiv:2002.10689*, 2020.

Nianzu Yang, Kaipeng Zeng, Qitian Wu, Xiaosong Jia, and Junchi Yan. Learning substructure invariance for out-of-distribution molecular representations. 2022.

Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. Do transformers really perform badly for graph representation? *Advances in Neural Information Processing Systems*, 34:28877–28888, 2021.

Hao Yuan, Jiliang Tang, Xia Hu, and Shuiwang Ji. Xgnn: Towards model-level explanations of graph neural networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 430–438, 2020a.

Hao Yuan, Haiyang Yu, Shurui Gui, and Shuiwang Ji. Explainability in graph neural networks: A taxonomic survey. *arXiv preprint arXiv:2012.15445*, 2020b.

Zaixi Zhang, Qi Liu, Hao Wang, Chengqiang Lu, and Cheekong Lee. Protgnn: Towards self-explaining graph neural networks. *arXiv preprint arXiv:2112.00911*, 2021.

Yichu Zhou and Vivek Srikumar. Directprobe: Studying representations without classifiers. In Kristina Toutanova, Anna Rumshisky, Luke Zettlemoyer, Dilek Hakkani-Tür, Iz Beltagy, Steven Bethard, Ryan Cotterell, Tanmoy Chakraborty, and Yichao Zhou (eds.), *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pp. 5070–5083. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.naacl-main.401. URL https://doi.org/10.18653/v1/2021.naacl-main.401.

A Appendix

The appendix is structured as follows. First, we provide details on how our baselines are trained. In Section A.1, we provide additional details on how we train our linear probes for different probing tasks. In Section A.2, additional details on pairwise probing are discussed. In Sections A.3 and A.4, we extend our probing framework to the additional tasks and provide further insights. In Section A.5, we provide the statistics for our pretraining datasets. In Section A.6, we analyze the effect of over-smoothing phenomenon for different baselines and present our observations. Sections A.7, A.8 and A.9 reflect detailed studies on pairwise probing, causal effect of functional groups, and overlap between pre-training dataset respectively. Finally, in Section A.10 we discuss the limitations of our work.

Hyperparameters and Reproducibility. To pretrain all models used in our experiments, we rely on the authors code and the checkpoints released by them. For models without checkpoints, we use the sweep over the hyperparameters ranges suggested by the authors. Our probes utilize linear regression models. Additional information on our probes can be found in the next section. For reproducibility of our reported results, we provide our code in the supplementary material.

A.1 Training and Evaluating Linear Probes

We train a logistic regression model for categorical properties where $p_i \in \{0, \ldots, C\}$ where C is the number of categories, e.g., high-level molecular prediction, and meaningful substructure. We standardize the representation of each model by removing the mean and scaling it to unit variance to increase the convergence rate of the logistic regression. For tasks that contain positive integers as labels ($p_i \in \mathbb{N}, p_i \geq 0$.), e.g. counting atom properties, we train linear Poisson regression probes. Since the 3D structure properties are continuous we train a standard linear regression probe. For pooling, all our baseline models use virtual nodes unless mentioned otherwise.

A.2 Additional Details on Pairwise Probing

Before running PCA, we center the representations with $\hat{\boldsymbol{z}}_i = \boldsymbol{z}_i - \boldsymbol{m}_i$, and $\hat{\boldsymbol{z}}'_i = \boldsymbol{z}'_i - \boldsymbol{m}_i$, where $\boldsymbol{m}_i = 0.5 \cdot (\boldsymbol{z}_i + \boldsymbol{z}'_i)$. This has the effect of moving the representation of each molecule such that all difference vectors $\boldsymbol{v}_i$ cross the origin at the midpoint. This centering helps to better isolate the main source of variation. Then we run standard PCA on the matrix $\boldsymbol{Z} = [\hat{\boldsymbol{z}}_i; \hat{\boldsymbol{z}}'_i] \in \mathbb{R}^{2N \times d}$ containing all centered representations. This approach is illustrated on Fig. 1. We visualize the projection of the representations onto the top two principal components, isolating the relevant subspace for the property. We see that even a single principal component is sufficient to separate the molecules that contain the *amide* functional group from those that do not.

A.3 Odor Probing

The objective of this experiment is to probe for the presence of informative signals about the smell (odor) of a molecule, e.g. sweet or woody (32 smells in total) in the learned representations, using the Pyrfume data archive [2]. As before, we pretrain models on the PCQM4Mv2 dataset and we apply linear probing. We split the dataset into 80% training and 20% testing using the stratified split for each smell. We tabulate and present the results in Table 6. Again, we see that on average the representations learned by transformer-based models encode more information about odors. GIN is outperformed by GcN even though it is theoretically more expressive. Certain odors (e.g. sulfurous) are reliably encoded by all models.

Table 6: Linear probing performance (AUC score) for different odors.

Model	apple	balsamic	burnt	caramellic	cheesy	citrus	earthy	ethereal	fatty	fermented	floral	fresh	fruity	green	herbal	meaty	mint
Raw	0.52	0.57	0.55	0.55	0.60	0.60	0.53	0.65	0.57	0.54	0.61	0.50	0.74	0.67	0.57	0.65	0.61
Morgan	0.55	0.64	0.50	0.52	0.52	0.60	0.51	0.59	0.66	0.55	0.61	0.53	0.76	0.70	0.55	0.55	0.60
GCN	0.91	0.88	0.90	0.93	0.92	0.85	0.78	0.91	0.88	0.91	0.83	0.74	0.82	0.79	0.72	0.89	0.89
GIN	0.80	0.88	0.85	0.77	0.79	0.83	0.63	0.90	0.74	0.87	0.76	0.65	0.71	0.71	0.65	0.88	0.75
Graphormer	0.88	0.87	0.85	0.84	0.90	0.81	0.67	0.93	0.87	0.93	0.76	0.70	0.81	0.79	0.71	0.92	0.84
GRPE	0.88	0.91	0.91	0.88	0.88	0.87	0.69	0.91	0.86	0.92	0.83	0.72	0.83	0.80	0.73	0.94	0.87
GraphGPS	0.89	0.87	0.86	0.92	0.89	0.82	0.75	0.93	0.90	0.89	0.81	0.74	0.82	0.79	0.76	0.92	0.91
TokenGT	0.90	0.94	0.90	0.88	0.87	0.84	0.73	0.93	0.89	0.90	0.79	0.74	0.81	0.79	0.78	0.90	0.90

	nutty	oily	onion	pineapple	pungent	roasted	rose	spicy	sulfurous	sweet	tropical	vegetable	waxy	winey	woody	AVG
Raw	0.66	0.62	0.71	0.53	0.52	0.62	0.57	0.57	0.71	0.56	0.61	0.55	0.56	0.51	0.58	0.59
Morgan	0.55	0.66	0.66	0.57	0.57	0.52	0.65	0.57	0.57	0.57	0.52	0.51	0.65	0.57	0.50	0.57
GCN	0.78	0.88	0.90	0.93	0.83	0.89	0.88	0.80	0.95	0.69	0.81	0.81	0.92	0.84	0.81	0.85
GIN	0.78	0.91	0.92	0.84	0.89	0.89	0.77	0.78	0.90	0.63	0.65	0.73	0.82	0.79	0.74	0.79
Graphormer	0.76	0.80	0.91	0.90	0.85	0.90	0.90	0.81	0.96	0.64	0.84	0.84	0.83	0.81	0.80	0.83
GRPE	0.82	0.83	0.94	0.90	0.87	0.91	0.92	0.79	0.96	0.67	0.86	0.84	0.88	0.82	0.81	0.85
GraphGPS	0.81	0.91	0.94	0.94	0.91	0.93	0.84	0.83	0.97	0.69	0.79	0.84	0.93	0.81	0.81	0.86
TokenGT	0.79	0.88	0.96	0.93	0.85	0.92	0.89	0.78	0.98	0.70	0.83	0.83	0.87	0.86	0.81	0.85

A.4 Additional Probing Performance

High-Level Molecular Properties. In the main paper (Table 3) we presented a subset of probing results for the high-level molecular properties. In Table 7 we show for completeness the detailed analysis for all high-level molecular properties from the MoleculeNet benchmark.

Table 7: Linear probing performance (AUC and R2 scores) for high-level molecular properties from MoleculeNet.

Dataset	BBBP	Tox21	ToxCast	Sider	ClinTox	MUV	HIV	BACE	AVG
Molecules	2,039	7,831	8,575	1,427	1,478	93,087	41,127	1,5113	
Tasks	1	12	617	27	2	17	1	1	
Raw	66.8	68.1	60.5	56.2	60.6	69.5	73.2	75.1	66.3
Morgan	63.2	64.0	54.1	60.5	57.4	67.2	65.8	75.0	63.4
GCN	56.5	60.2	57.0	55.2	64.9	72.9	66.3	71.1	63.0
GIN	58.8	58.1	55.7	53.2	57.0	64.6	72.4	77.1	62.1
Graphormer	59.6	69.0	59.2	57.5	78.5	75.7	75.8	63.0	67.2
GRPE	64.9	72.1	61.1	55.7	82.8	75.7	73.0	77.9	70.4
GraphGPS	63.9	70.3	60.0	60.7	72.7	74.4	72.2	74.5	68.5
TokenGT	57.1	70.8	59.3	58.4	88.5	80.4	75.8	75.2	70.6

Impact of the Pretraining Dataset. Similarly, in Table 5 we presented a subset of the results for the impact of the pretraining dataset and here we show all properties. Since the number of training samples in the *ogbg-molpcba-subset* and *ogbg-molhiv* datasets are the same, their performance can be interpreted as the impact of multitask learning vs. single-task pretraining. We bank on the

[2]Each dataset and the corresponding reference can be found on `https://github.com/pyrfume/pyrfume-data`.

assumption that equalising the training samples size would help us study the effect of the number of tasks. Table 8 suggest that probing performance increases significantly when we use the *ogbg-molpcba-subset* dataset instead of *ogbg-molhiv* dataset. This implies that pretraining on multitask learning increases the information-related to topological properties of the graph, which in this case is the existence of functional groups.

Table 8: Effect of the pretraining dataset on probing performance (AUC score).

Architecture	Dataset	Arom. Carb.	Arom. Ring.	Satur. Ring.	Aniline	Benzene	Bicyclic	Ketone	Methoxy	ParaHydrox	Pyridine	AVG
GCN	Molhiv	98.6	99.5	95.3	83.8	98.5	85.6	87.2	97.7	89.1	87.9	93.3
	Molpcba-sub	99.7	99.8	98.6	95.3	99.7	94.2	97.2	99.6	94.2	96.0	97.4
	Molpcba	99.6	99.7	98.1	96.3	99.6	95.5	96.8	99.0	94.4	95.6	97.5
GIN	Molhiv	82.0	90.3	83.3	65.2	81.9	66.5	63.5	71.1	73.5	71.2	74.9
	Molpcba-sub	97.6	99.7	97.5	91.1	97.6	94.7	91.4	95.2	90.5	91.4	94.7
	Molpcba	99.2	99.8	98.1	95.6	99.2	95.5	94.3	95.5	91.5	94.5	96.3
V-GCN	Molhiv	94.5	97.3	90.8	76.2	94.5	73.7	76.8	88.4	78.1	76.7	84.7
	Molpcba-sub	98.3	99.5	95.1	90.2	98.3	87.2	91.5	93.0	83.3	92.1	92.6
	Molpcba	98.7	99.5	95.4	93.3	98.7	89.4	95.0	92.9	87.6	93.0	94.4
V-GIN	Molhiv	85.4	92.5	87.4	71.3	85.4	68.5	69.5	70.5	74.7	73.0	77.8
	Molpcba-sub	94.5	99.7	95.1	81.8	94.6	88.3	84.4	84.7	81.6	85.8	89.0
	Molpcba	98.3	99.8	96.2	90.5	98.3	91.2	90.9	90.3	86.9	92.1	93.5

Impact of Design Choices. To complement the results we show in Fig. 4 on the impact of various architectural design choices we report additional results in Table 9. As before, adding residual connections (models with "Res") improves the probing performance. For GIN using a virtual node improves performance, while for GCN it has the opposite effect. Using jumping knowledge (models with "JK") also tends to improve the performance.

Table 9: Impact of using residual connections (models with "Res"), pooling with or without virtual nodes, and jumping knowledge (models with "JK") on the linear probing performance (AUC score).

	AromaticCarbocycle	AromaticRing	SaturatedRing	Aniline	Benzene	Bicyclic	Ketone	Methoxy	ParaHydroxylation	Pyridine
GCN	0.98	0.99	0.94	0.79	0.98	0.83	0.82	0.94	0.87	0.87
GCN-JK	0.98	0.99	0.98	0.88	0.98	0.88	0.92	0.97	0.88	0.91
GCN-Res	0.98	0.99	0.96	0.86	0.98	0.87	0.91	0.97	0.88	0.89
GCN-Res-JK	0.99	0.99	0.97	0.90	0.99	0.89	0.93	0.98	0.90	0.90
Virtual GCN	0.94	0.97	0.91	0.73	0.94	0.73	0.74	0.83	0.77	0.76
Virtual GCN-JK	0.98	0.99	0.97	0.89	0.98	0.85	0.88	0.96	0.85	0.87
Virtual GCN-Res	0.98	0.99	0.96	0.88	0.98	0.84	0.86	0.93	0.86	0.84
Virtual GCN-Res-JK	0.98	0.99	0.97	0.90	0.98	0.85	0.91	0.95	0.87	0.87
GIN	0.79	0.86	0.82	0.63	0.79	0.64	0.62	0.68	0.71	0.68
GIN-JK	0.98	0.99	0.97	0.90	0.98	0.90	0.91	0.97	0.88	0.89
GIN-Res	0.99	1.00	0.97	0.92	0.99	0.90	0.92	0.98	0.91	0.90
GIN-Res-JK	0.99	1.00	0.97	0.91	0.99	0.90	0.91	0.97	0.90	0.89
Virtual GIN	0.84	0.91	0.87	0.70	0.84	0.66	0.67	0.66	0.72	0.72
Virtual GIN-JK	0.98	0.99	0.96	0.87	0.98	0.84	0.85	0.91	0.85	0.83
Virtual GIN-Res	0.98	0.99	0.96	0.89	0.98	0.86	0.87	0.93	0.87	0.85
Virtual GIN-Res-JK	0.98	1.00	0.96	0.89	0.98	0.86	0.86	0.92	0.86	0.84

A.5 Datasets Molecules Statistics

We use ogbg-molhiv and ogbg-molpcba to compare the effect of the pretraining dataset on the performance of the probing task since both contain small molecules with similar statistics. Table 10 shows the average number of nodes and edges per graph which indicate that both datasets contain small molecules with similar sparsity.

Table 10: The molecules statistic for Molhiv and Molpcba dataset

Name	#Graphs	#Nodes per graph	#Edges per graph
ogbg-molhiv	41,127	25.5	27.5
ogbg-molpcba	437,929	26.0	28.1

A.6 Oversmoothing Analysis

We compare the learned feature embedding of various models to see whether the oversmoothing phenomenon is responsible for degradation in probing performance. In Fig. 6, we compute the average pairwise distance between the (learned) node features and average it across all the molecules. For models like GCN-virtual and GIN-virtual, we observe that as the number of layers (hops) increases, the average distance shrinks. The transformer-based models are not as susceptible to this phenomenon. In any case, we cannot conclude that the decrease in probing performance is due to oversmoothing.

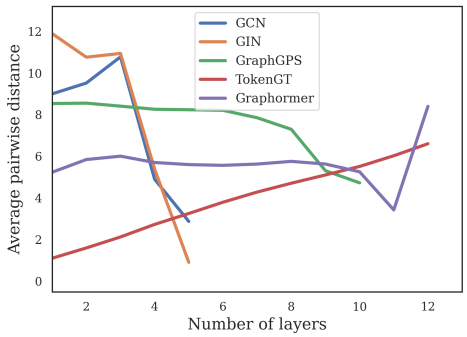

Figure 6: Effect of oversmoothing across several models.

A.7 Pairwise Probing

We provide additional results for pairwise probing on all the models for Nitro (Fig. 7) and Amide (Fig. 8) functional groups. The results and observations are similar as in the main paper. The top two principal components are enough to reliably identify the property of interest for most models. This is also reflected in the ratio of the captured variance.

A.8 Causal Effect on Odor

We also extend our pairwise probing for studying the causal effect of the functional group on high-level properties. In this case, we study the effect of a functional group on the smell of a molecule. In this regard, we select molecules that have specific smells and create a pair with and without specific functional groups. Next, we train a probe on the representation of the pre-trained model to predict specific smells. Then, we use our probe models and molecule pairs (with and without specific functional groups) to measure the effect of a functional group on predicting specific smells. Based on SCM we assumed in the main paper, since removing the functional group is an intervention and removes the edge from E to F, the change in the prediction probability is the causal effect of the functional group on the smell. We further remove the average effect of a functional group for all smells to prune the general effect of removing a substructure from the molecule.

Fig. 9 show results for 16 different odors. Although the result is noisy, there are several chemically meaningful signals. For example, the compounds that contain sulfide are sweet, or there is a relation between the ketone and the apple smell, which seems to be interesting for further study.

A.9 Overlap Between MoleculeNet and MOLPCBA datasets

Table 11 show the overlap between the Molecule Net datasets and ogbg-molpcba dataset.

Table 11: The overlap between Molpcba molecule and MoleculeNet datasets

Dataset	#Sample	Overlap with MOLPCBA
bace	1513	0
bbbp	2039	88
clintox	1477	22
muv	93087	89748
sider	1427	14
tox21	7831	4687
toxcast	8576	570
molhiv	41127	1943

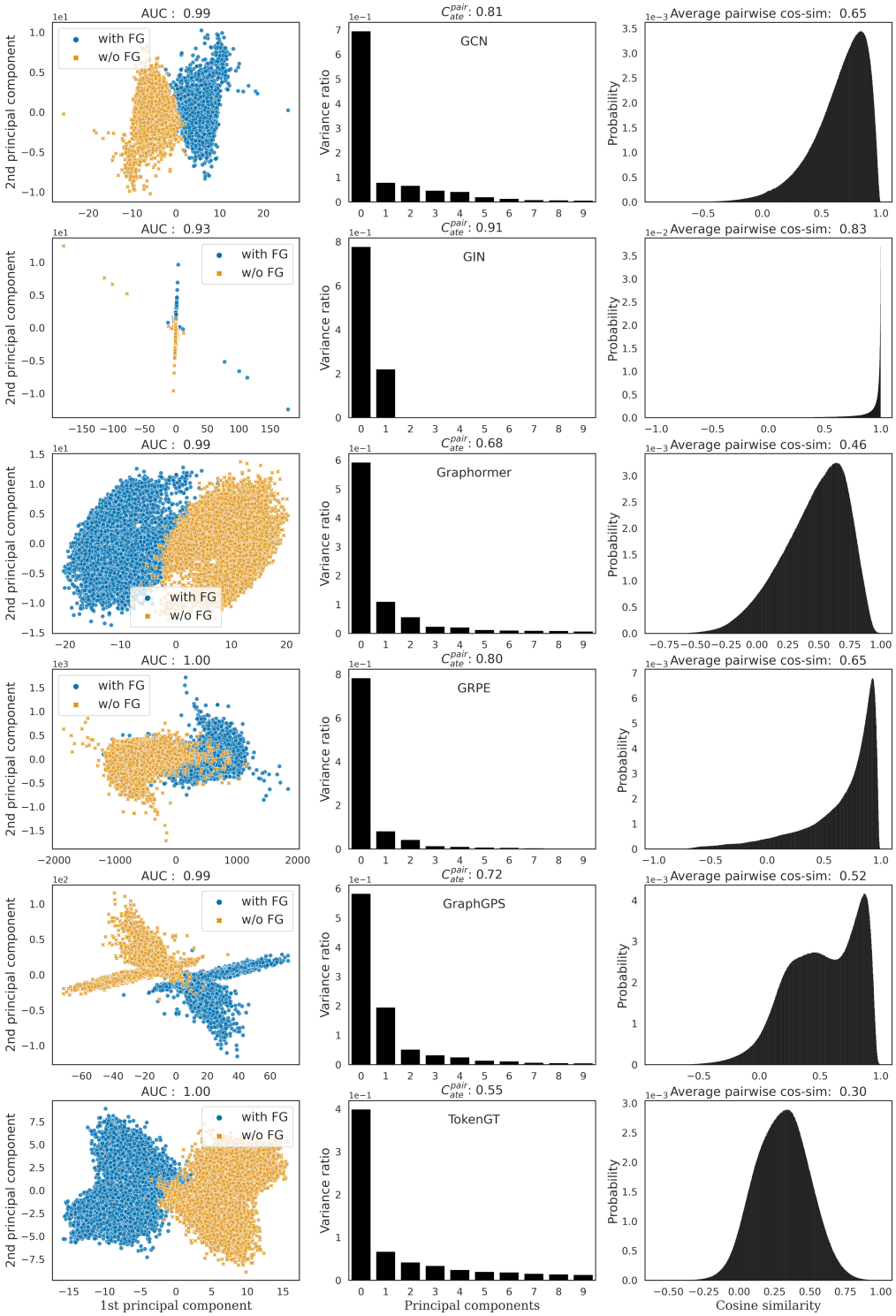

Figure 7: Pairwise probing for various architecture for Nitro functional group. We show the projection on the top two principal components (left), the ratio of explained variance per component (middle), and the pairwise cosine similarity (right).

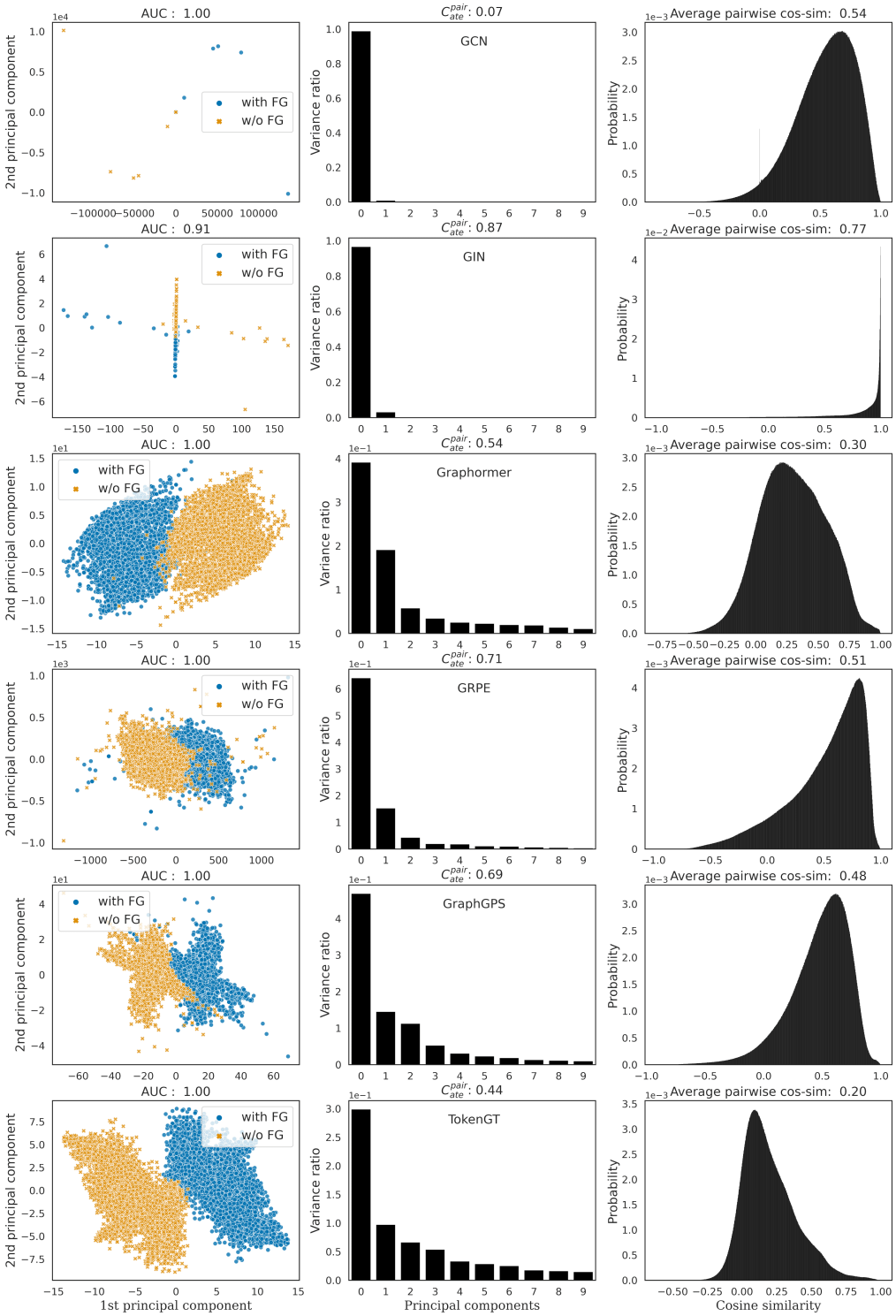

Figure 8: Pairwise probing for various architecture for Amide functional group. We show the projection on the top two principal components (left), the ratio of explained variance per component (middle), and the pairwise cosine similarity (right).

A.10 Discussion and Limitations

Here, we discuss some of the limitations of the proposed approach and the probing framework in general. For a more detailed discussion see Belinkov (2022). First, the probing results depend on

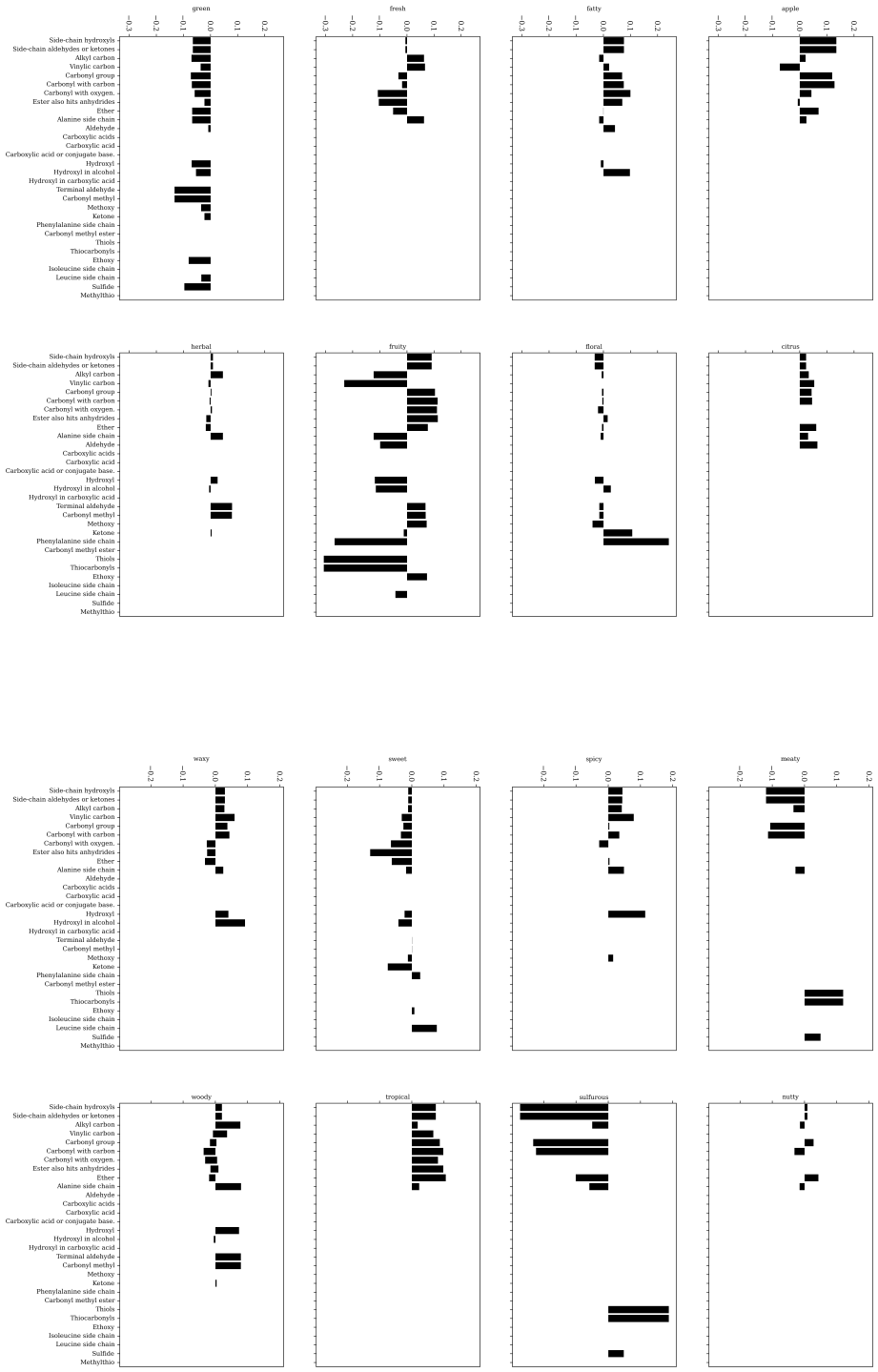

Figure 9: Causal Effect of functional group on smell

the dataset used for probing and its size. To mitigate this effect we employed incremental probing as suggested by Pimentel & Cotterell (2021). Moreover, by using the same probing dataset for all

models this issue becomes less important since our main goal is to compare different architectures. Second, it can be tempting to conclude that because a property p is encoded in the representation $\boldsymbol{z}$ it must be important for the prediction of the original model. This is not necessarily the case. For example, Ravichander et al. (2021) show that models can learn to encode linguistic properties even when they are not needed to solve the main task. To investigate whether the property is important for prediction some authors propose to intervene on $\boldsymbol{z}$ to *remove* information about p. Elazar et al. (2021) use iterative null space projection to remove information, while Feder et al. (2021) adversarially remove properties. We leave such studies for graph representations for future work. Third, all of the properties we considered were predefined – we either compute them without supervision (using RDKit) or use an annotated dataset. Alternatively, as proposed by Michael et al. (2020), we can look for clusters in the representation space and verify whether they correspond to known properties.