Evaluating Attribution for Graph Neural Networks

Benjamin Sanchez-Lengeling[1,5], Jennifer Wei[1], Brian Lee[1], Emily Reif[1], Peter Y. Wang[2], Wesley Wei Qian[1,3], Kevin McCloskey[1], Lucy Colwell[1,4], and Alexander Wiltschko[1,5]

[1]Google Research
[2]Stanford University, work done while a resident at X.
[3]University of Illinois at Urbana-Champaign
[4]University of Cambridge
[5]Email: {bmsanchez, alexbw}@google.com

Abstract

Interpretability of machine learning models is critical to scientific understanding, AI safety, and debugging. *Attribution* is one approach to interpretability, which highlights input dimensions that are influential to a neural network's prediction. Evaluation of these methods is largely qualitative for image and text models, because acquiring ground truth attributions requires expensive and unreliable human judgment. Attribution has been comparatively understudied for graph neural networks (GNNs), a model class of growing importance that makes predictions on arbitrarily-sized graphs. Graph-valued data offer an opportunity to quantitatively benchmark attribution methods, because challenging synthetic graph problems have computable ground-truth attributions. In this work we adapt commonly-used attribution methods for GNNs and quantitatively evaluate them using the axes of attribution *accuracy*, *stability*, *faithfulness* and *consistency*. We make concrete recommendations for which attribution methods to use, and provide the data and code for our benchmarking suite. Rigorous and open source benchmarking of attribution methods in graphs could enable new methods development and broader use of attribution in real-world ML tasks.

1 Introduction

With the increasing use of automated decision making aided by machine learning models, the credibility of the models we produce takes on a heightened importance, particularly for fields such as drug discovery. Credibility describes the extent to which a user trusts a model's output, and this concept can be difficult to formalize [17]. In the absence of a concrete definition of credibility, we may provide a window into the model's "decision making process", or provide *interpretability*. In fact, a new wide-reaching European regulatory framework for the application of ML (GDPR; [1]), explicitly requires interpretability of some kind for deployed models. There are many ways to provide interpretations of a model (which we review below), but in this work we focus on perhaps the simplest — *attribution*.

An attribution is a credit assignment on each individual input feature x_i of a data input x (e.g. for images, each pixel; for text, each character or word) that measures how important the feature is to the model's prediction of a target property y, often presented to users as a heatmap overlaid on the original data. The attribution heatmap visually indicates what aspects of a particular data example has the greatest influence on the model's prediction of property y.

Attributions can expose the statistical regularities that the model leverages to make a prediction [47]. If these patterns match our intuition, they can bolster our confidence in the model's predictions. If attributions instead reveal that the model is exploiting spurious correlations, or plainly violates a

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

practitioner's common sense, we may use attributions as a debugging tool — we can highlight, and then subsequently correct, spurious correlations in a dataset [29], or apply regularization to encourage desired behavior in the model [33, 38].

Attribution methods have been most studied in the domains of image modeling [16] and text [6], areas where humans have strong intuition. A "ground truth" credit assignment in these domains ultimately rests with subjective human judgment. Unfortunately, obtaining ground truth for realistic image and text tasks is subjective, expensive, and time-consuming.

The introduction and refinement of graph-based neural network models [11, 39] has opened up new and powerful capabilities for modeling structured data. For instance, social networks [52], protein-protein interaction networks [54], and molecules [18, 20] are naturally represented as graphs. Graph-valued data offer an opportunity to inexpensively and quantitatively benchmark attribution methods, due to the fact that challenging synthetic graph problems have computable ground-truth attributions. This allows us to quantitatively measure the performance of popular attribution methods on several GNN model types, built for a variety of tasks.

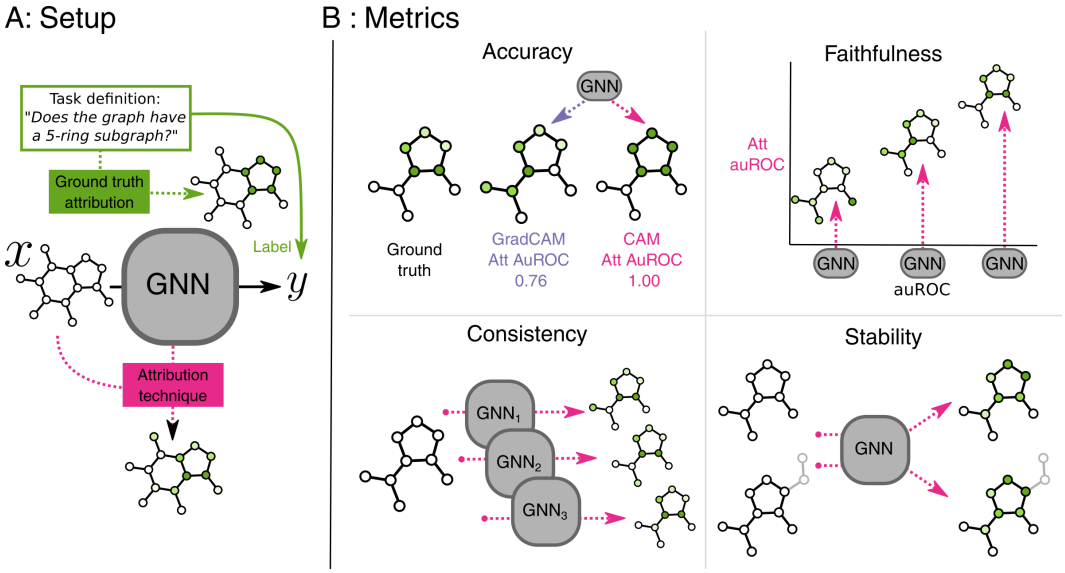

Figure 1: Schematic of attribution task setup and attribution metrics. **A**. We create classification and regression tasks for which we have a computable ground-truth. We train GNN models on these labels, and calculate attributions using the graph inputs and attribution methods we adapt to graphs. **B**. We quantify attribution performance with four metrics. *Accuracy* measures how well an attribution matches ground-truth. *Consistency* measures how accuracy varies across different hyperparameters of a model. *Faithfulness* measures how well the performance of an attribution method matches model performance. *Stability* measures how attributions change when the input is perturbed.

Measuring Performance of Attribution Methods We use tasks with graph-valued data and computable ground truths (Figure 1, left) to examine qualities of an attribution method that are necessary for credibility: *accuracy*, *faithfulness*, *consistency* and *stability* [37] (Figure 1, right). We consider an attribution method to have high attribution performance if it scores well on all four properties. We focus on these qualities from [37] because they are quantitative, do not require soliciting human judgment, and are specific to attribution, as opposed to interpretability more broadly.

Accuracy. We assess attribution accuracy by quantifying how well attributions match ground-truth credit assignments. If the model is "right for the right reasons" [17], we expect the attribution method to highlight the correct, ground truth nodes in the input graph (Figure 1B, upper left).

Consistency. The accuracy of an attribution technique should be consistent across high-performing model architectures. To test attribution consistency, we quantify the variability in attribution accuracy using the top 10% of models through a hyperparameter scan over model architectures (Figure 1B, lower left).

Faithfulness. The performance of a faithful attribution method should reflect the performance of the model. To quantify faithfulness, we run two experiments where we intentionally damage the training dataset to degrade a model's predictive performance, and systematically measure how each attribution

method responds (Figure 1B, upper right). What we term faithfulness in this work is the concept of *fidelity* from Robnik-Šikonja and Bohanec [37].

Stability. An attribution method should be invariant to small changes in input features that do not affect an example's class label or the model's prediction. To assess stability, we make small graph perturbations on test set examples that leave the ground-truth attribution and predicted class label unchanged, and assess the degree of change in attribution accuracy (Figure 1B, lower right).

We perform these experiments across four popular GNN architectures (GCN, MPNN, GAT, Graph Nets), and six common attribution methods (GradInput, SmoothGrad, two variants of GradCAM, Integrated Gradients, CAM and Attention weights) which are further explained in section 3.2.

Contributions

We offer three main contributions:

- We build a comprehensive and open-source benchmarking suite for attribution methods on GNNs and graph-valued data[1].
- We evaluate the performance of commonly-used attribution methods in GNNs using the axes of *accuracy*, *faithfulness*, *stability* and *consistency* using targeted experiments.
- We find that, when applicable, CAM applied to GCNs is the best performing attribution method for GNNs. CAM is generally the best attribution method across all architectures, and all attribution methods tend to perform best when applied to GCNs. If CAM cannot be used, IG is a good substitute. For graph datasets containing only adjacency information, GradCAM is the best performance attribution method.

2 Related Work

This paper draws from two main areas of prior work: 1) analysis of attribution methods in interpretability and explainability, and 2) graph neural networks and interpretability for graphs.

There are several approaches to model interpretability that rely on simplifications or local approximations of models, feature subselection or attention [36, 13, 38, 26] , although recent analysis shows the use of attention for interpretability can be problematic [23, 12]. We focus on attribution, which has received significant research attention [40, 41, 42, 44, 24, 30, 41, 53]. Efforts to quantify the utility of attribution methods or apply sanity checks have been undertaken in input domains where human intuition is usually used to evaluate attribution quality [4, 50, 7, 32, 21, 5], like images and text. We are inspired by these approaches, and use them to quantify attribution performance against computable ground truths in the domain of graphs.

Attribution for GNNs have been studied and evaluated before, in specific cases. Duvenaud et al. [18] examined the activations of each message-passing layer in a GCN and identified relevant subgraphs. Other models used attention mechanisms in order to highlight important subgraphs [25, 28]. *GNNExplainer* [51] developed graph explanations by identifying single subgraphs that are relevant to a prediction. The works of [29, 34, 35] use saliency-based methods to identify important nodes and subgraphs in graph-level predictions. Several other works have applied or tested individual attribution methods for GNNs on a narrow set of tasks or applications [34, 10, 8]. Other relevant work that develops methods for GNN interpreability are [49] and *GraphLIME* [22].

3 Methodology

3.1 Graph neural networks

We define a graph and GNN following Battaglia et al. [11]. A graph G is a 3-tuple $G = (V, E, u)$ containing vertices V and edges E and possibly global context information u. We are primarily interested in graph labeling problems of the form $f : G(V, E, u) \rightarrow \mathbb{R}$. Briefly, a GNN is a neural network that takes a graph as input and outputs a graph with the same topology, but with updated node, edge and/or graph-level information. A readout layer is applied to the output graph to produce a real-valued prediction. One key feature of the GNNs we study is the message passing function, which allows nodes to update their states by aggregating feature information from neighboring nodes and edges. Depending on the message passing strategy, the message can contain information about the node, edge, or the global context.

[1] Code and data for this paper will be available at `github.com/google-research/graph-attribution`

Our experiments use four existing GNN architectures distinguished by their message-passing strategies. The first is a graph convolutional neural network (GCN) [18] in which the messages are calculated using only node states. The second model is the MPNN [20] in which the message function is based on a learned representation of the edge states and is used to update the node states. The third model is Graph Nets [11] in which the message function uses a global state vector in addition to node and edge states. The fourth is the Graph Attention Network (GAT) [46] which aggregates node information via an attention mechanism. Details about the implementations of these models may be found in Section 5.1.

3.2 Attribution methods

An attribution method A takes a model M and a graph G to generate an attribution map, $G_A = (v_A, e_A); v_A, e_A \in \mathbb{R}$, where v_A, e_A are node and edge weightings relevant for predicting y. These weightings can be visualized as a heatmap superimposed on a graph. Our ground-truths for attributions are node-level, so we redistribute edge attributions equally onto their endpoint nodes' attributions. Global features, which can be interpreted as a bias term for the entire graph, have been studied elsewhere [27] and are not considered for attribution in this paper. We utilize the following methods for graphs:

Class Activation Map (CAM) [53]. CAM uses a global average pooling (GAP) layer prior to class outputs to obtain attributions. For CNNs, CAM attributions are derived by multiplying the final convolutional layer's feature map activations (act) with the output weights of the GAP layer, w. CAM can be adapted to graphs using graph model architectures that allow the addition of a GAP-like layer. For example, CAM on GCNs expresses the GAP layer as a summation over node and edge features at the last message passing layer.

$$G_A = (v_j = w^T act_{v_j}, e_k = w^T act_{e_k})$$

Grad $\odot$ Input [41]. GradInput attributions correspond to the element-wise product of the input graph with the gradient of $\hat{y}$ with respect to the input node and edge features, with an optional reduction step over the feature dimension to arrive at a node- or edge-level attribution.

$$G_A = w^T G, w = \frac{d\hat{y}}{dG}$$

GradCAM [40] extends GradInput by using intermediate activations. It corresponds to the element-wise product of the activation of an intermediate message-passing layer with the gradient of $\hat{y}$ with respect to the node and edge features of that intermediate layer. We analyze the attributions of two GradCAM variants. GradCAM(last) is the GradCAM attribution of the last message-passing layer (the nth layer G_n), and GradCAM(all) averages across all message-passing layers.

$$G_A(\text{last}) = w_n^T G_n(G), \quad G_A(\text{all}) = \frac{\sum_i^n w_i^T G_i(G)}{n}, \quad \text{with } w_n = \frac{d\hat{y}}{dG_n(G)}$$

SmoothGrad [42] averages attributions evaluated on noise-perturbed versions of an input. This was initially used to sharpen saliency maps in images, but we extend this technique to graphs by adding Gaussian noise to node and edge features. The variance of the noise (σ=0.15), and number of samples (n=100) is optimized for attribution AUROC on the Benzene task. We apply SmoothGrad to the GradInput attribution method detailed above.

$$G_A(m) = \sum_i^n m(G + \text{noise}_i), m = \text{GradInput}$$

Integrated Gradients (IG) [44] integrates the element-wise product of an interpolated input, G_{interp}, with the gradient of $\hat{y}$ with respect to G_{interp}, between the actual input G and a counterfactual input G'. We follow [29] and build counterfactual G' using a null graph, a graph with the same topology but all nodes and edges use an "unspecified" categorical feature.

Since null graphs function as a baseline, models should produce equal output probabilities to null graph inputs. Therefore, the input dataset is augmented with the null graphs of 20% of positively-labeled examples, which have a 50% probability of associating with a positive or negative label.

$$G_A = (G - G') \int_{\alpha=0}^{1} \frac{dy(G' + \alpha(G - G'))}{dG} d\alpha$$

Attention This method is specific to the GAT model. For a given node, the attention mechanism will produce attention scores on edges to adjacent nodes. We can use these attention scores as a measure of importance for propagating information relevant to the predictive task, and therefore as an attribution weighting. A GAT GNN might have several blocks and attention heads, so for each component we take their average to combine them into a scalar value assigned to each edge. We assign node importance by adding half of each edge score to each attached node. In our experiments we use a single attention head and a softmax as a normalization function.

Random. As a control, we also included an attribution method which produces random attributions drawn from a uniform distribution: $G_A = (v_j = \mathcal{U}, e_k = \mathcal{U})$.

3.3 Attribution tasks

Each classification or regression task involves identifying a particular subgraph, or the conjunction of two or more subgraphs. The graph datasets use molecular graphs (Figure 2). Each node and edge represents a one-hot encoded categorical variable indicating atom or bond type.

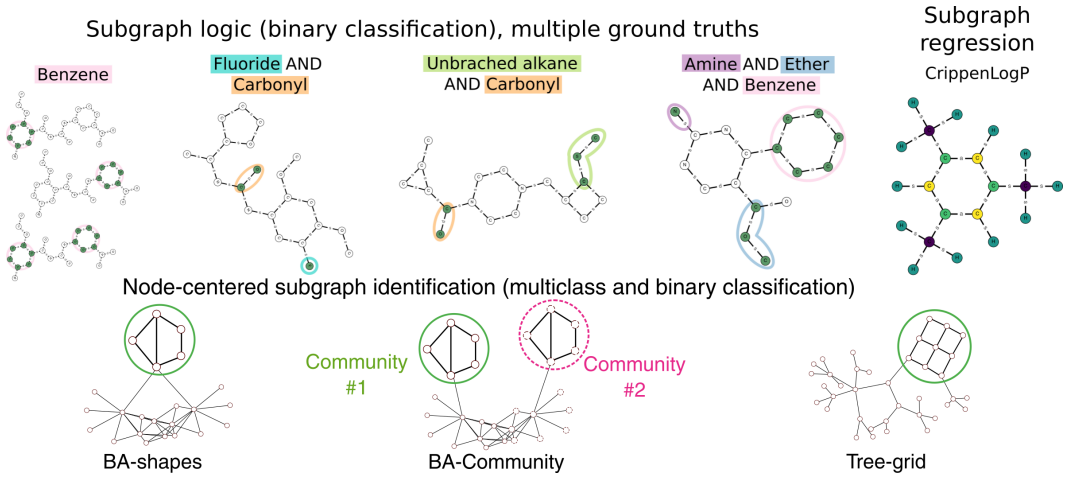

Figure 2: Example ground truth attributions for each task. The first four graph-classification tasks require a model to identify all nodes (green) in one or more subgraphs (colored lasso) in molecular graphs. Each graph may have multiple positive ground-truths, shown in the *Benzene* task. Ground truth attributions for the *CrippenLogP* regression task take on continuous values. Lower row has node-classification tasks. Relevant subgraphs are circled. Only one neighborhood of the graph is shown.

Graph classification. We consider classification tasks with computable ground-truth attributions. These tasks are analogs of real-world problems, such as identifying biologically active chemical groups in a molecule. The goal of the task is to identify if a graph contains particular subgraphs of interest, expressed as a combination using the AND logical operator (such as Fluoride AND Carbonyl groups, Figure 2), following from [29]. The goal of the attribution task is to identify the ground-truth nodes in the Boolean query. We test the following subgraph logics: Benzene; Fluoride AND Carbonyl; Unbranched Alkane AND Carbonyl; Amine AND Ether AND Benzene. There may exist multiple valid attributions for a single input, since a subgraph may be present multiple times. If a graph has two subgraphs of type P, we allow the space of possible attribution solutions to be the first instance, the second, or both.

We use the dataset constructed by McCloskey et al. [29]. In order to reduce possible biases, the authors constructed each logic dataset to be balanced across the set of all binary combinations. For example, given a logic $P\&Q$, the dataset contains an equal number of examples for every logic combination: $P\&Q$, $P\&\neg Q$, $\neg P\&Q$ and $\neg P\&\neg Q$. Graphs are sampled from a database of drug-like molecule graphs [43]. 1,200 graphs are selected for each logic combination, and 10% of these graphs are reserved for the test set.

To study *faithfulness* in Section 4.2, we create datasets on five new two-subgraph logic tasks with molecules taken from Sterling and Irwin [43] using a diverse subsampling approach. These tasks are also balanced for every logic combination. Further information about the dataset construction and statistics can be found in section 5.5.

Graph regression. The Crippen LogP model [48] is an empirical chemistry model that predicts the water-octanol partition coefficient (a specific measure of solubility) of a molecule by assigning a

weight to each node type. For this task, a correct attribution should assign weights to nodes in the graph that corresponds to the weights assigned to each node type by the Crippen model. We utilize the Delaney solubility dataset [15] which has 1127 graphs, with a randomized 80%/20% train/test split. It is possible that models consistently learn to predict CLogP using an approach which does not align with the Crippen subgraph-based rules, but the achieved accuracy on this task (see Section 3) suggests otherwise.

Node classification. We utilize three synthetic tasks introduced in Ying et al. [51]: BA-shapes, BA-community, and Tree-grid. All graphs are generated by either the Barabasi-Albert (BA) algorithm or as a balanced binary tree (Tree), and augmented with several graph motifs that are randomly attached to the graph. Nodes are assigned labels based on their role in each motif. The predictive task is to identify each node's role in the motif, and the attribution task is then to highlight if a node participates in a motif. BA-shapes uses a 5-node motif, with four possible node labels and only graph structure information. Tree-grid uses a 9-node motif constructed from a 3 by 3 node grid with two possible labels only graph structure information. BA-community is the fusion of two BA-shape graphs, with eight possible node labels, four for each component graph. Nodes have randomly generated features that are correlated to each component.

4 Experiments

4.1 Attribution Accuracy and Consistency

To measure attribution accuracy we ran a hyperparameter scan over 195 model configurations (see Supplemental Section 5.4) per combination of *model architecture*, *attribution method*, and *task*. The models include GCN, MPNN and Graph Nets architectures. The attribution methods, when applicable, include SmoothGrad, GradCAM (all), GradCAM (last), GradInput, IG, and CAM, Attention.

We measure attribution accuracy by comparing predicted attribution to the ground truth attribution in both classification and regression tasks. We use AUROC [29] to evaluate attribution accuracy on classification tasks, and use Kendall's tau to evaluate attribution performance on regression tasks. If multiple attributions are valid (e.g., a subgraph is present twice in a graph), we take the maximum attribution value of all possible solutions.

We compute the mean and variance of attribution accuracy for the top 10% of trained models (as measured by average held-out test prediction accuracy, across four random seeds) per combination of model, attribution method, and task. The top 10% of models all achieve >95% predictive performance. Attribution accuracy is summarized for all task, model, and attribution method combinations in Figure 3, and visually in Figure S1. As a negative control, we found that random attributions yield poor attribution accuracy (dotted line in Figure S1), and that randomly initialized models show random mean attribution (Figure S4).

We observe that attribution accuracy was highest for CAM and IG across tasks and model architectures ($p<0.05$ for 90% of all pairwise comparisons with Holm-Bonferroni step-down correction[2]). CAM and IG were also the least variable across multiple hyperparameter scans of model architectures ($p<0.05$ for 37.5% of pairwise comparisons, excluding Graph Nets due to its high variability), suggesting that they are also the most consistent attribution methods (Figure S1). Of the three models, GCN achieves the highest mean attribution accuracy ($p<0.05$ for 31.7% of pairwise comparisons). The attribution accuracy of Graph Nets was highly variable for all attribution methods and tasks (Figure 3, Figure S1)($p<0.05$ for 20% of pairwise comparisons). This is not entirely surprising, given that it is a highly parameterized architecture, and deep nets are known to encode substantively different prediction functions across random initializations [19]. Furthermore, Graph Nets have a global context vector that may entangle information needed for proper node-level attribution between the global context vector and the node-level vectors. Care should be taken when choosing an attribution method with Graph Nets, and perhaps for more complicated GNN architectures in general.

Other factors can influence the accuracy of an attribution method. While we do not investigate this in detail here, we offer a few anecdotes which might aid practitioners in tuning or developing attribution methods. We observe that attribution accuracy declines in later training epochs (Figure S2), with a time course that is independent of generalization performance. This occurs even when there is no overfitting during prolonged training. We also observe that regularization affects attribution accuracy on some tasks independent of predictive performance (Figure S3).

[2]See 5.3 in Appendix for details on the tests for statistical significance.

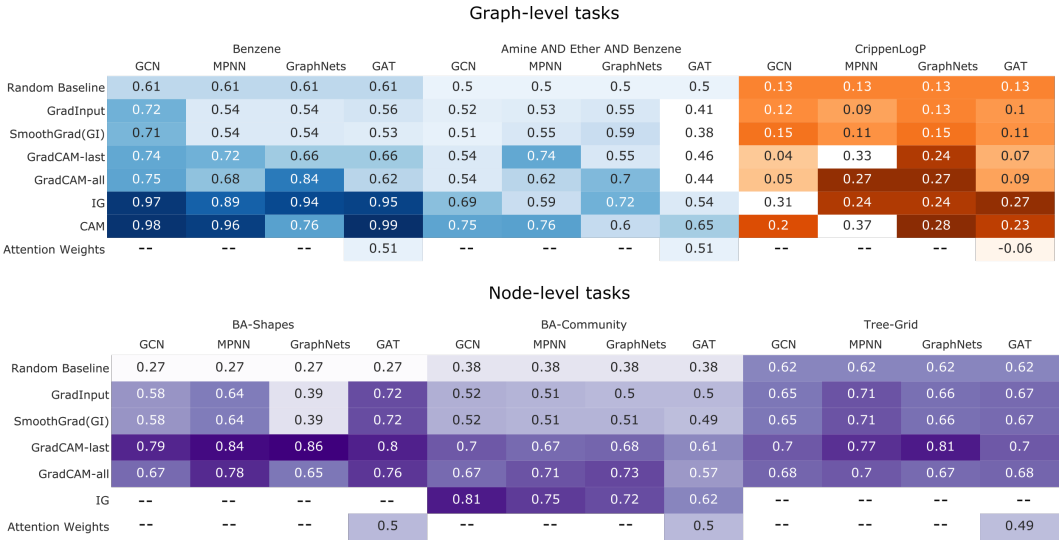

Figure 3: Attribution method accuracy across tasks and model architectures. Colors are used to distinguish two metric types — attribution AUROC for attribution on classification tasks, and attribution Kendall's tau on the regression task. CAM and IG perform consistently well across tasks and models. For error bars, please see Figure S1.

4.2 Attribution Faithfulness

Ideally, the accuracy of a faithful attribution method will be monotonically sensitive to the enforced degradation in the model's predictive accuracy. This property is important because we want to be able to use attribution to inspect and better understand functioning models, and also to debug malfunctioning ones. We perform two experiments in which we intentionally degrade model performance by damaging the training dataset and measuring how each attribution method responds. We quantify faithfulness with the *performance-attribution relative correlation (PARC)* score, calculated as Kendall's tau metric between the intentionally controlled predictive test set performance of the model and attribution accuracy (Figure 4A; we introduce the term *PARC* to distinguish this use of Kendall's tau, from its use in measuring regression attribution accuracy). A high *PARC* score in each of these experiments is indicative of faithfulness, whereas a low *PARC* score indicates the attribution method is unfaithful, as its accuracy is insensitive to model performance. The *PARC* measure is related to reliability plots and expected calibration error from the uncertainty quantification literature [14, 31, 45], but here, each observation is comprised of a single model fit at a particular strength of handicap, as opposed to a bin of predictive performance.

First, we train models with progressively noised training labels by selecting a subset of the training set, from 0% to 100%, and shuffling labels within this subset. We find that across all tasks in this experiment, CAM has the highest *PARC* score overall ($p<0.05$ for 94% of the comparisons, excluding Graph Nets due to its high variability), and scores were highest overall for MPNNs and GCNs (Figure 4B) ($p<0.05$ for 23.3% of the comparisons). In contrast, Graph Nets show a highly variable relationship between predictive performance and attribution accuracy across tasks ($p<0.05$ for 33.3% of the comparisons).

Second, in a more targeted experiment, we introduce spurious correlations of increasing strength into the training dataset. Specifically, we coerced models away from predicting an original target subgraph (P) to predicting the presence of two subgraphs ($P\&Q$) by introducing progressively stronger correlations between $P\&Q$ in the training data. As the co-occurrence of $P\&Q$ in training graphs strengthened, the model's performance inadvertently increased from random to highly accurate on a task it was never trained to perform — predicting the presence of $P\&Q$. All labels come from identifying P, while the test sets only contain examples of $P\&Q$ or $P\&\neg Q$ (Figure 5A, B). Thus, any change in model performance is only attributable to the introduction of a spurious correlation.

We expect a faithful attribution method to reveal why the model was "right for the wrong reasons" in predicting P when training examples contain increasing proportion of of $P\&Q$. For a range of training datasets containing varying proportions of $P\&Q$ and $P\&\neg Q$, we assessed whether an attribution method will faithfully highlight both subgraphs $P\&Q$ as the frequency of spurious

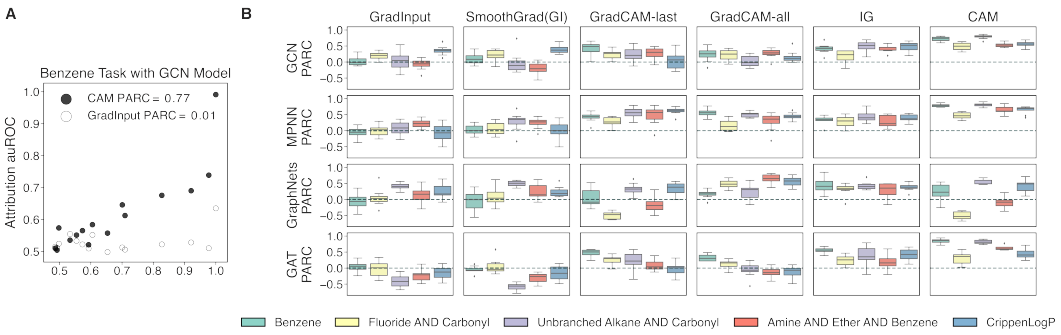

Figure 4: Noising labels to control model performance and measure attribution faithfulness. **A**. Attribution vs task performance for CAM (filled dots) and GradInput (unfilled dots) in the GCN / benzene task pair. The correlation between these two quantities is measured with *PARC*. A value of 1 indicates perfect faithfulness. Attribution and task performance is highly correlated for CAM (filled dots) but not for GradInput (unfilled dots). **B**. Box plots for the PARC score, measuring attribution faithfulness with noising labels, over 8 repeated runs.With GCN, MPNN and GAT, CAM has the highest faithfulness in all tasks. The best performing method for a Graph Nets is task-dependent. Dashed lines represent *PARC* from uniformly random attribution.

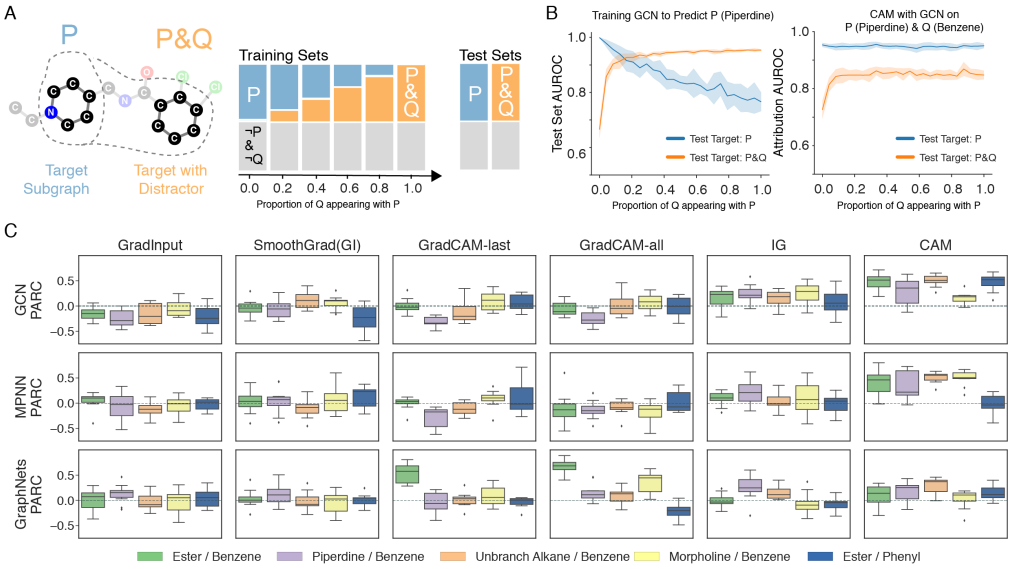

Figure 5: Evaluating faithfulness by handicapping a model with spurious correlations. **A**. Schematic of the make-up of the train and test datasets. The train set contains 50% examples that satisfy $\neg P \& \neg Q$, a shrinking proportion of $P \& \neg Q$, and a growing proportion of the distractor subgraph $P \& Q$. The two test sets include $\neg P \& \neg Q$, together with (i) $P \& \neg Q$, or (ii) $P \& Q$. **B**. Models were trained to predict the presence of subgraph P in the presence of an increasing proportion of $P \& Q$ examples in the training set. As an example, we show the performance of CAM applied to a GCN. Performance on the original target task of identifying subgraph P degraded, while performance on the *originally unspecified task* of predicting the presence of $P \& Q$ increased (left). Concomitantly, attribution performance increased for $P \& Q$ (right). **C**. Box plots for the *PARC* score $P \& Q$ dataset show the PARC score over 10 repeated runs on each (model, attribution, task) triplet. CAM and IG have the best faithfulness here using the GCN and MPNN model, and the best performance under Graph Nets is task dependent. Dashed lines indicate *PARC* for a random attribution control.

correlations increased. We measured the relationship between the model's AUROC on the task of predicting $P \& Q$ against its attribution AUROC on $P \& Q$, and summarized this with the *PARC* score. Using this metric, we found that CAM and IG consistently had high faithfulness across different identities of subgraphs P and Q, and also across model architectures (Figure 5) ($p<0.05$ for 48.8% of the comparisons, excluding Graph Nets due to its high variability).

4.3 Attribution Stability

Finally, we wished to evaluate the stability of attribution methods under input perturbations generated following [29]. Inputs are perturbed with 2-degree alterations which maintain positive ground-truth node attributions. Per task, we selected 40 original test examples and generated 10 perturbations per example, across 4 classification tasks, yielding 1600 perturbations. We measured the change in attribution performance, as Δ*Attribution AUROC*. A stable attribution method should have Δ*Attribution AUROC* values close to or equalling zero. (Figure 6, left).

We observed that while all attribution methods are on average unaffected by our graph perturbations, CAM exhibits the the smallest variance in Δ*Attribution AUROC* ($p<0.05$ for 71.7% of the comparisons). All attribution methods show high variance for Graph Nets, among which CAM has the lowest variance. We also include a negative control of random attributions (Figure 6, right).

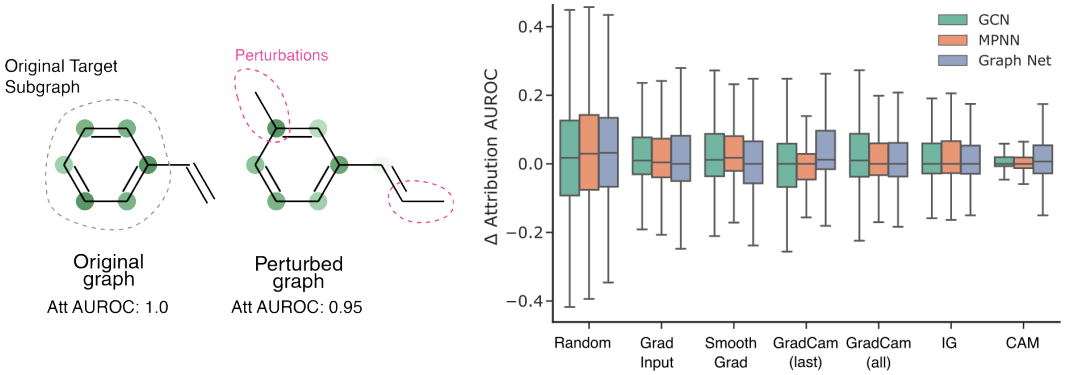

Figure 6: Attribution stability as measured by change in attribution accuracy on perturbed input graphs. Left. Illustrative example of a 2-degree perturbation of a molecule, along with the subgraphs defining the ground-truth, and resulting attributions. **Right.** Aggregate changes in attribution accuracy on target subgraphs for perturbations that leave the predicted label unchanged and true label unchanged, reported as Δ *Attribution AUROC*. Aggregating across test examples, perturbations and tasks, the Δ *Attribution AUROC* for all attribution methods and models center around zero, but CAM demonstrates the lowest variability, indicative of high stability.

5 Conclusion and Discussion

In this work we have created a framework to quantitatively evaluate attribution methods in GNNs, which we expect will aid in developing better methodology for interpretability in graphs, and perhaps in other data modalities as well. Overall, if attribution performance is critical to a machine learning task, we recommend the use of CAM paired with a GCN. However, CAM is not compatible with all GNN architectures, since it requires the last layer to be a global pooling operation. In these cases, IG is an appropriate second choice, and failing that, GradCAM. Care should be taken in choosing an attribution method for the more complicated Graph Nets architecture, because attribution performance is highly task dependent. Surprisingly, most attribution methods have lower performance with GAT models. Further, using attention as an attribution technique performed poorly. However, we only explored one approach for attribution using attention, and it is possible that further elaborations of GAT models or new techniques to translate attention into attribution would improve performance.

As a final caveat, we examined two types of graph data (molecular, synthetic) and three types of graph tasks (node classification, graph classification, graph regression). This portfolio of attribution challenges is not exhaustive, and we imagine that expanding our testing suite could yield new insights.

Overall, we find that simpler models give better attribution performance, across many attribution techniques. Attribution techniques that are directly related to predicted labels like CAM and IG also tend to have better attribution performance, across many models.

There remains much room for improvement in attribution performance for GNNs. We hope that the field finds our rubric for attribution performance (accuracy, fidelity, consistency, stability) and our quantitative benchmarking suite of attribution in GNNs useful for developing new attribution methods. With improved attributions, interpretability and credibility in ML on graphs, we hope to see applications of these techniques expand in social science, chemistry, biology and beyond.

Broader Impact

Better attribution methods for graph neural networks will help improve ML interpretability, and therefore ML credibility, in the domain of machine learning on graph-valued data. Specifically, we wish for machine learning models to not just to learn and exploit correlations in training data, but to help practitioners understand the correlations the model has learned and create new scientific knowledge. Applications of high-performing attribution methods on GNNs include pharmaceutical development, material design, social network analysis, and more. Improving the ability to inspect these models will hopefully improve their rate of adoption. GNNs have also been applied to social networks, and improving the field's capability to inspect and interrogate trained models will hopefully also improve the public discourse around the topic. However, our work reveals that no attribution method is perfect, and risk remains in placing perfect confidence in the output of existing attribution methods applied to graph neural networks.

Financial Disclosure

Funding in direct support of this work came from Google. No third party funding was involved.

Acknowledgments

We thank Alex D'Amour, Zelda Mariet, David Belanger, D Sculley, David Melis and Been Kim for their helpful comments, and Google Brain as well as the Cambridge office for creating a supportive research environment, even amidst a rapidly changing world.

References

[1] General data protection regulation (GDPR) – official legal text. `https://gdpr-info.eu/`. Accessed: 2020-4-29.

[2] RDKit: Open-source cheminformatics. `http://www.rdkit.org`.

[3] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL `https://www.tensorflow.org/`. Software available from tensorflow.org.

[4] Julius Adebayo, Justin Gilmer, Michael Muelly, Ian Goodfellow, Moritz Hardt, and Been Kim. Sanity checks for saliency maps. In S Bengio, H Wallach, H Larochelle, K Grauman, N Cesa-Bianchi, and R Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 9505–9515. Curran Associates, Inc., 2018.

[5] David Alvarez-Melis and Tommi S Jaakkola. Towards robust interpretability with Self-Explaining neural networks. June 2018.

[6] Leila Arras, Franziska Horn, Grégoire Montavon, Klaus-Robert Müller, and Wojciech Samek. "what is relevant in a text document?": An interpretable machine learning approach, 2017.

[7] Leila Arras, Ahmed Osman, Klaus-Robert Müller, and Wojciech Samek. Evaluating recurrent neural network explanations, 2019.

[8] Salim Arslan, Sofia Ira Ktena, Ben Glocker, and Daniel Rueckert. Graph saliency maps through spectral convolutional networks: Application to sex classification with brain connectivity. In *Graphs in Biomedical Image Analysis and Integrating Medical Imaging and Non-Imaging Modalities*, pages 3–13. Springer, 2018.

[9] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. July 2016.

[10] Federico Baldassarre and Hossein Azizpour. Explainability techniques for graph convolutional networks. *arXiv preprint arXiv:1905.13686*, 2019.

[11] Peter W Battaglia, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, Caglar Gulcehre, Francis Song, Andrew Ballard, Justin Gilmer, George Dahl, Ashish Vaswani, Kelsey Allen, Charles Nash, Victoria Langston, Chris Dyer, Nicolas Heess, Daan Wierstra, Pushmeet Kohli, Matt Botvinick, Oriol Vinyals, Yujia Li, and Razvan Pascanu. Relational inductive biases, deep learning, and graph networks. June 2018.

[12] Gino Brunner, Yang Liu, Damián Pascual, Oliver Richter, Massimiliano Ciaramita, and Roger Wattenhofer. On identifiability in transformers. August 2019.

[13] Brandon Carter, Jonas Mueller, Siddhartha Jain, and David Gifford. What made you do this? understanding black-box decisions with sufficient input subsets. October 2018.

[14] Morris H DeGroot and Stephen E Fienberg. The comparison and evaluation of forecasters. *Journal of the Royal Statistical Society: Series D (The Statistician)*, 32(1-2):12–22, 1983.

[15] John S Delaney. ESOL: estimating aqueous solubility directly from molecular structure. *J. Chem. Inf. Comput. Sci.*, 44(3):1000–1005, May 2004.

[16] Jérémie Despraz, Stéphane Gomez, Héctor F Satizábal, and Carlos Andrés Peña-Reyes. Towards a better understanding of deep neural networks representations using deep generative networks, 2017.

[17] Finale Doshi-Velez and Been Kim. Towards a rigorous science of interpretable machine learning. 2017.

[18] David Duvenaud, Dougal Maclaurin, Jorge Aguilera-Iparraguirre, Rafael Gómez-Bombarelli, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P Adams. Convolutional networks on graphs for learning molecular fingerprints. September 2015.

[19] Stanislav Fort, Huiyi Hu, and Balaji Lakshminarayanan. Deep ensembles: A loss landscape perspective. *arXiv preprint arXiv:1912.02757*, 2019.

[20] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural Message Passing for Quantum Chemistry. April 2017.

[21] Sara Hooker, Dumitru Erhan, Pieter-Jan Kindermans, and Been Kim. A benchmark for interpretability methods in deep neural networks. In *Advances in Neural Information Processing Systems*, pages 9734–9745, 2019.

[22] Qiang Huang, Makoto Yamada, Yuan Tian, Dinesh Singh, Dawei Yin, and Yi Chang. Graphlime: Local interpretable model explanations for graph neural networks. *arXiv preprint arXiv:2001.06216*, 2020.

[23] Sarthak Jain and Byron C Wallace. Attention is not explanation. February 2019.

[24] Andrei Kapishnikov, Tolga Bolukbasi, Fernanda Viégas, and Michael Terry. XRAI: Better attributions through regions. June 2019.

[25] Mostafa Karimi, Di Wu, Zhangyang Wang, and Yang Shen. Explainable deep relational networks for predicting Compound-Protein affinities and contacts.

[26] Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas, and Rory Sayres. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). *arXiv preprint arXiv:1711.11279*, 2017.

[27] Pieter-Jan Kindermans, Sara Hooker, Julius Adebayo, Maximilian Alber, Kristof T Schütt, Sven Dähne, Dumitru Erhan, and Been Kim. The (un)reliability of saliency methods. November 2017.

[28] Boris Knyazev, Graham W Taylor, and Mohamed Amer. Understanding attention and generalization in graph neural networks. In *Advances in Neural Information Processing Systems*, pages 4202–4212, 2019.

[29] Kevin McCloskey, Ankur Taly, Federico Monti, Michael P Brenner, and Lucy J Colwell. Using attribution to decode binding mechanism in neural network models for chemistry. *Proc. Natl. Acad. Sci. U. S. A.*, 116(24):11624–11629, June 2019.

[30] Grégoire Montavon, Alexander Binder, Sebastian Lapuschkin, Wojciech Samek, and Klaus-Robert Müller. Layer-Wise relevance propagation: An overview, 2019.

[31] Alexandru Niculescu-Mizil and Rich Caruana. Predicting good probabilities with supervised learning. In *Proceedings of the 22nd international conference on Machine learning*, pages 625–632, 2005.

[32] Ahmed Osman, Leila Arras, and Wojciech Samek. Towards ground truth evaluation of visual explanations. March 2020.

[33] Gregory Plumb, Maruan Al-Shedivat, Angel Alexander Cabrera, Adam Perer, Eric Xing, and Ameet Talwalkar. Regularizing black-box models for improved interpretability. February 2019.

[34] Phillip Pope, Soheil Kolouri, Mohammad Rostrami, Charles Martin, and Heiko Hoffmann. Discovering molecular functional groups using graph convolutional neural networks. *arXiv preprint arXiv:1812.00265*, 2018.

[35] Kristina Preuer, Günter Klambauer, Friedrich Rippmann, Sepp Hochreiter, and Thomas Unterthiner. Interpretable deep learning in drug discovery. In *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, pages 331–345. Springer, 2019.

[36] Marco Ribeiro, Sameer Singh, and Carlos Guestrin. "why should I trust you?": Explaining the predictions of any classifier, 2016.

[37] Marko Robnik-Šikonja and Marko Bohanec. Perturbation-Based explanations of prediction models, 2018.

[38] Andrew Slavin Ross, Michael C Hughes, and Finale Doshi-Velez. Right for the right reasons: Training differentiable models by constraining their explanations. March 2017.

[39] F Scarselli, M Gori, Ah Chung Tsoi, M Hagenbuchner, and G Monfardini. The graph neural network model, 2009.

[40] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-CAM: Visual explanations from deep networks via gradient-based localization. October 2016.

[41] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. April 2017.

[42] Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. SmoothGrad: removing noise by adding noise. June 2017.

[43] Teague Sterling and John J Irwin. ZINC 15–ligand discovery for everyone. *J. Chem. Inf. Model.*, 55(11):2324–2337, November 2015.

[44] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, pages 3319–3328, Sydney, NSW, Australia, 2017. JMLR.org.

[45] Leonardo Teixeira, Brian Jalaian, and Bruno Ribeiro. Are graph neural networks miscalibrated? *arXiv preprint arXiv:1905.02296*, 2019.

[46] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. October 2017.

[47] James Wexler, Mahima Pushkarna, Tolga Bolukbasi, Martin Wattenberg, Fernanda Viegas, and Jimbo Wilson. The What-If tool: Interactive probing of machine learning models. July 2019.

[48] Scott A Wildman and Gordon M Crippen. Prediction of physicochemical parameters by atomic contributions. *J. Chem. Inf. Comput. Sci.*, 39(5):868–873, September 1999.

[49] Shangsheng Xie and Mingming Lu. Interpreting and understanding graph convolutional neural network using gradient-based attribution methods. *arXiv preprint arXiv:1903.03768*, 2019.

[50] Mengjiao Yang and Been Kim. Benchmarking attribution methods with relative feature importance. July 2019.

[51] Zhitao Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. Gnnexplainer: Generating explanations for graph neural networks. In *Advances in Neural Information Processing Systems*, pages 9240–9251, 2019.

[52] Wayne W Zachary. An information flow model for conflict and fission in small groups, 1977.

[53] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2921–2929, 2016.

[54] Marinka Zitnik and Jure Leskovec. Predicting multicellular function through multi-layer tissue networks. July 2017.