Markov Blanket-Driven Local Explanations for Fully Connected Neural Network Predictions

Tumelo Mkwambe TUMELOMKWAMBE1@STUDENTS.WITS.AC.ZA
School of Computer Science and Applied Mathematics
University of the Witwatersrand
Johannesburg, South Africa

Editor: Ritesh Ajoodha

Abstract

Deep neural networks achieve remarkable predictive performance across a variety of domains, yet their complex architectures render their decision-making processes largely opaque. In this work, we investigate the extent to which fully connected neural networks exploit statistical dependencies among input variables during inference and propose a method for identifying the subset of variables most influential to a given prediction. Our approach frames variable relevance as a dependency-aware, non-additive feature attribution problem, leveraging structure learning techniques to extract Markov blankets that explain the network's output without assuming linear contributions. We evaluate our method on both synthetic datasets generated from known Bayesian networks and real-world datasets from the UCI repository. Results demonstrate that our approach provides faithful, local explanations that capture the true influential variables, matching established post-hoc methods such as LIME and SHAP, even when the underlying data-generating structure is unknown.

Keywords: Explainable AI, Neural Networks, Markov Blanket, Interpretability, Post-hoc Explainability, Bayesian Networks

1 Introduction

In recent years, deep neural networks have gained prominence in classification, regression, computer vision, and natural language tasks due to their exceptional ability to learn intricate patterns in complex data, establishing dominance over traditional statistical and machine learning models. The depth and complexity of these networks enable them to approximate highly nonlinear functions and deliver accurate predictions, nonetheless, this often comes at the cost of model transparency, as shown in Figure 1. Their intrinsic black-box nature limits interpretability, reducing their utility in high-risk domains where intelligibility is essential for quality assurance, trust, and bias mitigation (Quinn et al., 2022).

The opacity in deep neural networks can be distinguished into two types: inference opacity and training opacity. Inference opacity refers to the inability, upon inspection, to explain why a machine learning model predicts an output y given an input x, whereas training opacity refers to the inability to explain the parameters of the model as a product of the training data (Søgaard, 2023). In deep neural networks, this opacity can be attributed to several characteristics that are inherent to their architecture, the most notable being their layered structure, in which nodes are connected across adjacent layers

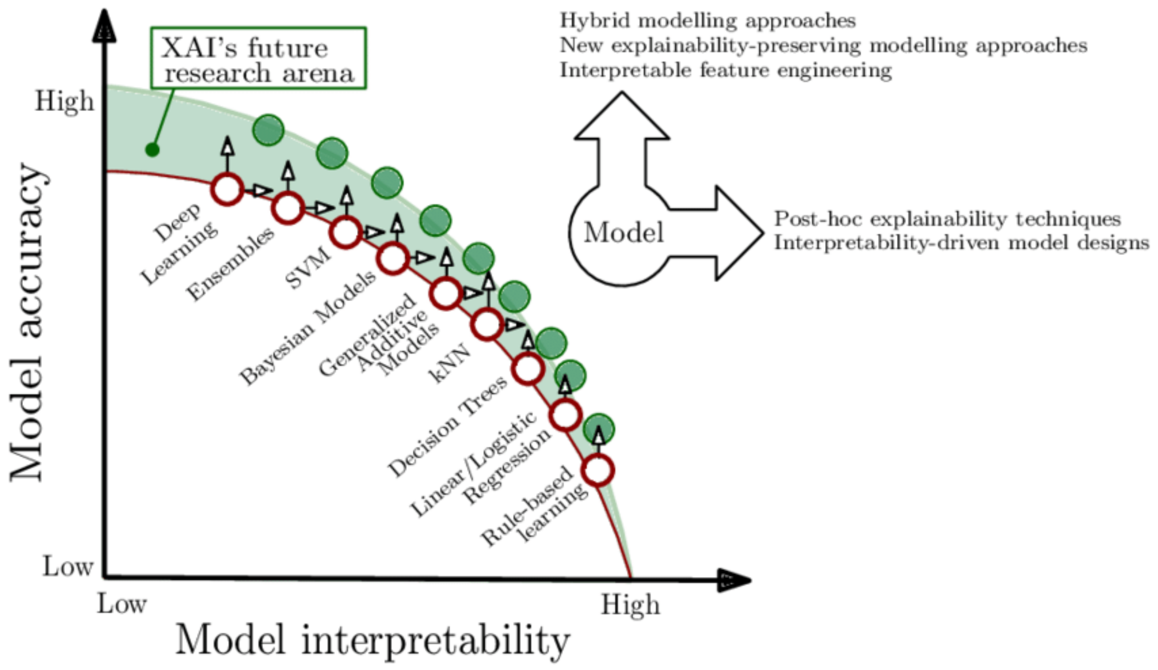

XAI's future research arena
High
Low
Model accuracy
Low
High
Model interpretability
Deep Learning
Ensembles
SVM
Bayesian Models
Generalized Additive Models
kNN
Decision Trees
Linear/Logistic Regression
Rule-based learning
Hybrid modelling approaches
New explainability-preserving modelling approaches
Interpretable feature engineering
Model
Post-hoc explainability techniques
Interpretability-driven model designs

Figure 1: Trade off between model interpretability and performance (Arrieta et al., 2020)

through weighted edges to form a complex input-output function. These layers transform inputs into increasingly abstract latent representations that lack grounded meaning. This absence of interpretable intermediate variables, combined with the massive scale of modern models, which often contain over a million parameters, makes it unrealistic for humans to fully understand how such systems make decisions or how their parameters are derived from data.

Recent work aimed at explaining the inference of these black-box models has been dominated by feature attribution methods, which attempt to interpret a model's prediction by identifying the features most influential in producing it. However, the most faithful of these methods tend to ignore interactions between variables during inference. In this paper, we examine the extent to which a model leverages dependencies among variables in its decision-making process and propose a dependency-aware approach for identifying the features the model considers statistically relevant for a prediction.

2 Related Work

For interpretable machine learning models with a graph structure, such as Bayesian networks and decision trees, all nodes and edges have meaningful representations. In Bayesian networks, nodes are random variables connected by edges if they exhibit probabilistic influence (Koller et al., 2007). In decision trees, nodes are features used to segregate data into their respective classes, the order of the nodes is determined by their capability to classify data and reduce impurity or uncertainty. Nodes in consecutive layers in neural networks are densely connected (fully connected for multilayer perceptrons) and have no intrinsic meaning, with the exception of those in the input and output layers. The challenge in making neural networks inference-transparent lies in finding how the input variables relate to the output variable given the dense web of connections and latent variables in between. In this chapter we provide a detailed overview of post-hoc explainability techniques developed to provide an intelligible input-output mapping in pre-trained black box models. Many such techniques exist and can be categorized by the type of model and data they address (Arrieta et al., 2020). Here, we focus specifically on feature attribution methods for complex, non-interpretable models such as neural networks. These methods estimate the contribution of input variables by assigning importance scores to each feature.

2.1 Perturbation-Based Methods

Perturbation-based methods (LIME, SHAP) make use of a baseline as a point of reference to assess how alterations to features affect the model's output. Alterations can vary, from perturbing a single feature while keeping the others fixed to assessing the effect the inclusion and exclusion of a feature has on the performance of the model.

LIME applies perturbations to the features of the data point being explained to generate samples which are weighted by their proximity to the original data point, feeds samples to the black-box model, and observes the output changes. Then, it trains a simpler, interpretable model (like linear regression) on these perturbed samples and compares the outputs with those of the black-box model (Ribeiro et al., 2016). The coefficients of features in the interpretable local model reveals which features most influenced that specific prediction. The framework uses the Submodular Pick algorithm to find features that are capable of explanations beyond the local scope. DeepSHAP is derived from SHAP which is a framework that uses approximations to Shapley values which descend from cooperative game theory (Lundberg and Lee, 2017). Shapley values consider all possible subsets of features, effectively perturbing the input by including or excluding features. DeepSHAP extends this approach to deep learning by leveraging DeepLIFT to efficiently approximate Shapley values. DeepLIFT functions by comparing the model's output on a given input to its output on a reference dataset, capturing how variations in the input influence the output. By examining the change in output when specific features of the input are altered or omitted, it estimates the contribution of each input feature to the model's prediction.

2.2 Gradient-Based Methods

At the highest level of abstraction, gradient-based techniques (Layer-wise Relevance Propagation, Integrated Gradients) utilize the partial derivative of the output with respect to the input variable to quantify the variable's contributions. The simplest method, Gradients, assigns the product of an input feature's gradient and its value as the contribution to the model's prediction. More complex methods, like integrated gradients and layer-wise relevance propagation, leverage the structure of the neural network by taking into consideration activations and weights.

Integrated Gradients considers the straight line path between the input being explained and a reference input. It then computes the gradients of the network's output with respect to the input features at various points along this path. Finally, a path integral is utilized to compute the integrated gradients along the path for each feature (Sundararajan et al., 2017).

Layer-wise relevance propagation propagates relevance scores back through the network layers, considering both activations and weights to distribute the output attribution across features (Montavon et al., 2019). The primitive layer-wise relevance propagation basic rule has been shown to be equivalent to the product of a node's gradient and its activation thus making it equivalent to the Gradients method (Shrikumar et al., 2016).

2.3 Additive Feature Attribution Methods

Additive feature attribution methods (LIME, DeepSHAP, Integrated Gradients) decompose the prediction into a sum of the effects of each feature. These operate under the assumption that the features contribute to the output independently of each other. A linear interpretable approximation

$$g(x_1, ..., x_n) = \sum_{i=0}^{n} \phi_i x_i,$$

where ϕ_i denotes the attribution for feature i, is provided as an explanation for a non-interpretable model's predictions (Ribeiro et al., 2016; Lundberg and Lee, 2017; Sundararajan et al., 2017). Additive feature attribution methods are identified by their capability to provide a unique solution satisfying three main properties: consistency, missingness and local accuracy. The consistency property suggests that the generated explainable model's output must equal the complex model's output for the same input. Missingness suggests that absent features should have no effect on the output, hence their attributed score should be a reflection of that absence. Local accuracy suggests that the method must be capable of generating a model that approximates and explains the behavior of the complex model (in our case neural networks) with high fidelity.

2.4 Non-additive Feature Attribution Methods

Unlike additive feature attribution, non-additive feature attribution (Integrated Hessians, Layer-wise relevance propagation) disregards the assumption that features make independent contributions to the output. Non-additive feature attribution methods take into consideration interactions of features within non-interpretable models. A second characteristic distinguishing additive and non-additive feature attribution is one concerning methodology. Additive methods have a similar approach of using a linear representation or approximation to address inference-transparency whereas methods in non-additive feature attribution leverage the structure of the non-interpretable model and consider the connections among latent variables.

Our proposed method falls under non-additive feature attribution, as it does not rely on the linear attribution assumption, and also belongs to the class of perturbation-based methods, since it uses the instance under investigation as the baseline for assessing how changes to its feature values affect the model's output.

3 Methodology

3.1 Problem Formulation and Research Purpose

Consider a pretrained fully connected neural network $f : \mathbb{R}^{n_0} \to \mathbb{R}^{n_L}$ with n_0 input nodes, n_L output nodes and L layers which takes an input $x \in \mathbb{R}^{n_0}$ and produces an output $y \in \mathbb{R}^{n_L}$. Let n_l be the number of nodes in layer l, $a^{(l)} \in \mathbb{R}^{n_l}$ the vector of activation values of the nodes at layer l, $b^{(l)} \in \mathbb{R}^{n_l}$ the vector of bias values at layer l, $W^{(l)} \in \mathbb{R}^{n_l \times n_{l-1}}$ the weight matrix of the edges connecting nodes in layer $l-1$ to those in layer l and $g^{(l)} : \mathbb{R}^{n_l} \to \mathbb{R}^{n_l}$ the activation function at layer l. The feedforward step of the neural network progresses from the input layer $a^{(0)} = x$ to the output layer $a^{(L)} = y$ iteratively:

$$a^{(l)} = g^{(l)}\left(W^{(l)} a^{(l-1)} + b^{(l)}\right), \quad \forall l \in \{1, \dots, L\}.$$

Hence, the output y is an amalgamation or transformation of the input features x over the neural network f with parameters $\{W^{(1)}, ..., W^{(L)}, b^{(1)}, ..., b^{(L)}\}$:

$$y = f(x) = g^{(L)}\left(W^{(L)} g^{(L-1)}\left(\cdots g^{(1)}\left(W^{(1)} x + b^{(1)}\right)\cdots\right) + b^{(L)}\right).$$

Note that the number of parameters $|W|$ scales with the width (number of nodes in a layer) and depth (number of layers in the network),

$$\begin{aligned} |W| &= \sum_{l=1}^{L} |W^{(l)}| + \sum_{l=1}^{L} |b^{(l)}|, \\ |W| &= \sum_{l=1}^{L} (n_{l-1} \times n_l) + \sum_{l=1}^{L} n_l. \end{aligned}$$

Given the dense web of connections, non-linear transformations, and substantial number of parameters within the feedforward neural network f, how can we obtain an intelligible and faithful input-output mapping that renders it inference-transparent?

The specific purpose of this study is to determine the extent to which feedforward neural networks rely on associations among input variables during inference. Methods like CASTLE Regularization explore neural networks trained to learn a causal structure in the training data (Kyono et al., 2020), nonetheless, they do not evaluate how standard fully connected networks use variable dependencies prior to such modifications. Clarifying this would illuminate the role these associations play and provide a more interpretable view of a model's inference. In our endeavor to understand the internal workings of these models, we propose a technique that represents the variable relationships learned by a feedforward network as a direct acyclic graph (DAG), highlighting the statistical dependencies that influence the model's predictions.

3.2 Experimental Setup

Research Question 1: To what extent does a feedforward neural network capture and utilize variable dependencies during inference?

We begin by defining a ground-truth Bayesian network $\mathcal{B}$ over a set of random variables $\mathcal{X} = \{X_0, X_1, \dots, X_n\}$. Using forward sampling, we draw a dataset $\mathcal{D}$ that reflects the dependencies encoded in the ground-truth network. Given $\mathcal{D}$, we specify a target variable $Y = X_n$ among the variables in $\mathcal{X}$ and train a multilayer perceptron (MLP) f to predict Y. For a prediction $Y = y$ by the MLP f given input $X = \{X_0 = x_0, \ldots, X_{n-1} = x_{n-1}\} = x$, we construct a directed acyclic graph $\mathcal{G}$ using our proposed method. After constructing $\mathcal{G}$, we compute the Markov blanket precision, recall, and f1 score by comparing the set of variables in Y's Markov blanket in $\mathcal{G}$ against that in the ground-truth network $\mathcal{B}$. These metrics quantify how effectively $\mathcal{G}$ recovers the true set of global influential variables, providing an evaluation of its ability to capture the dependencies influencing the neural network. To assess faithfulness, we evaluate how well each graph's Markov blanket explains model prediction, $f(x) = y$. We fix the features identified as relevant by a given blanket and randomize all remaining features, generating 10 random samples sharing values for the fixed features. For each sample, we compute the Jensen-Shannon divergence between the sample's predicted class distribution and the original input's predicted distribution. The divergences across the 10 samples are aggregated to obtain a divergence score for each Markov blanket for that prediction. To characterise overall explanatory faithfulness across the dataset, we repeat this procedure for 100 different predictions, each time generating samples and computing the corresponding divergence. The average Jensen-Shannon divergence across these 100 predictions should provide a stable estimate of how strongly the model's predictions depend on the features specified by either the estimated or ground-truth Markov blankets.

Research Question 2: How do the Markov blanket explanations compare to established post-hoc explainability techniques?

For real-world datasets where the underlying dependency structure is unknown, we perform the faithfulness evaluations of our proposed method given models trained on the real-world data to assess how well the inferred Markov blanket by our method capture the network's predictive behavior. To contextualize the performance of our approach, we benchmark its fidelity against popular feature attribution methods, notably LIME and SHAP, comparing how effectively each method identifies features that are truly relevant for the neural network's predictions. We exclude the less attributed features by LIME and SHAP to match the number of features in our estimated Markov blanket, then we employ the Jensen-Shannon divergence here as well to evaluate each method's fidelity to the model. This evaluation allows us to quantify the reliability of our method in highlighting influential variables.

4 Detailed Methods

4.1 Markov Blanket Explanations

Suppose we have a data instance x with prediction $f(x) = y$ from a neural network f. We begin by sampling random datapoints from the training dataset. For each sampled datapoint, we construct a perturbed version x' by replacing each feature's value with the corresponding value from x with replacement probability p. We then obtain the prediction $f(x') = y'$. Next, we apply a binary masking operation to each perturbed datapoint x', assigning a mask value of 0 to a feature if its value matches that of x, and 1 otherwise. We similarly mask the prediction, assigning 0 if the perturbed prediction remains unchanged and 1 if it differs from y. The resulting dataset $\mathcal{D}$) from this perturbation scheme is therefore generated relative to the instance of interest x, whose prediction we aim to explain, and summarizes the types of local perturbations to which the model is sensitive in the neighbourhood of x.

The subsequent step is to determine which individual features or feature subsets were statistically relevant to the model when making the prediction $f(x) = y$. Whereas a non-additive method like SHAP would attempt to discern the relevance of each feature under the assumption of independent contribution to model prediction, we employ structure learning using the greedy hill climbing search algorithm and the Bayesian information criterion (BIC) as our score function to acquire a less constrained and more interaction-aware explanation.

Greedy Hill Climbing is a local search heuristic algorithm used to optimize an objective function by iteratively selecting the best neighboring solution. For learning Bayesian network structures, the algorithm begins with an initial candidate structure, typically an empty graph. It proceeds by iteratively exploring the space of neighboring structures, generated by making local modifications such as adding, deleting, or reversing an edge. Only modifications that preserve the acyclic property of the graph are considered valid. At each iteration, given the current structure $\mathcal{G}$, the algorithm evaluates the BIC score for all valid neighbors $\mathcal{G}' \in Neighbors(\mathcal{G})$. It then selects the neighbor that yields the highest BIC score given the dataset $\mathcal{D}$. If the best neighbor $\mathcal{G}'$ improves upon the current score $(BIC(\mathcal{G}', \mathcal{D}) > BIC(\mathcal{G}, \mathcal{D}))$, the algorithm updates the current structure to $\mathcal{G}'$ and repeats the search. This process continues until no neighboring graph yields a higher BIC score, at which point the algorithm terminates, having reached a local maximum in the score landscape. The BIC score for structure learning,

$$BIC(\mathcal{G}, \mathcal{D}) = l(\hat{\theta}_{\mathcal{G}} : \mathcal{D}) - \frac{\log M}{2} Dim[\mathcal{G}],$$

evaluates how well a Bayesian network structure $\mathcal{G}$ explains the data $\mathcal{D}$ while penalizing overly dense graphs (Koller and Friedman, 2009). The first term, $l(\hat{\theta}_{\mathcal{G}} : \mathcal{D})$, is the log-likelihood of the data under the model with maximum-likelihood parameters, $\hat{\theta}_{\mathcal{G}}$, summed over all samples and all nodes. If a proposed parent set explains a variable well, then the variable's observed values in the data will frequently align with specific configurations of those parents. This produces a more certain conditional probability distribution, one in which the probabilities concentrate near 0 or 1 for each parent configuration resulting in a higher log-likelihood contribution. Conversely, if the variable's values do not exhibit any consistent pattern with respect to the proposed parents, then its conditional probability distribution becomes close to uniform over all parent configurations. This lack of predictive structure lowers the log-likelihood. The second term, $\frac{\log M}{2} Dim[\mathcal{G}]$, introduces a complexity penalty proportional to the number of parameters added by the proposed parent set. When a parent set fails to improve predictive power, this penalty outweighs any negligible gain in likelihood, thereby discouraging unnecessary edges.

The graphical structure obtained in the previous step captures variable dependencies that are statistically meaningful for predicting each variable's values. However, our goal is to identify only the subset of variables whose configurations consistently influence the model's prediction. Instead of attempting to explain the entire data distribution, we focus on the variables to which the model is sensitive in the local neighborhood of the data instance x. To achieve this, we extract only the dependency structure that sufficiently explains the target variable, whose values are determined by the model. This approach allows us to disregard perturbations to features that, although present in a sufficiently large and variable dataset, are merely artifacts of the perturbation scheme and do not meaningfully affect the model's output, and thus give no insight into the significant set of variable dependencies. As a result, the output from our proposed method is the Markov blanket of the target variable from the learned graphical structure.

4.2 Datasets

To evaluate our first research question, we use three datasets generated via forward sampling from well-known Bayesian networks: the Asia network (8 nodes and 8 edges), the Alarm network (37 nodes and 46 edges), and the Insurance network (27 nodes and 52 edges). For each dataset, we train a fully connected feedforward neural network to predict a selected target variable. The target variable in each case is chosen based on its ability to yield a model that generalizes reliably. To evaluate our second research question, we use two datasets from the UCI Machine Learning Repository: the Nursery dataset introduced by Olave et al. (1989) and the TicTacToe dataset from Bache and Lichman (2013).

5 Results

As shown in Table 1, which reports the average Jensen-Shannon divergence over 100 predictions, our estimated Markov blankets provide a dependency structure that effectively explains the model's behaviour, despite that Table 2 confirms that the method does not perfectly recover the ground-truth blankets. The neural networks clearly exploit the dependencies within the true blankets, as evidenced by the substantially lower divergence relative to the baseline across all datasets. Nonetheless, the estimated blankets frequently yield explanations that are more faithful to the model's behaviour. When the estimated blankets achieve high F1-scores, the divergence induced by the ground-truth blankets and the divergence induced by the estimated blankets become nearly indistinguishable, as shown in Figure 2. This indicates that in regions where the underlying ground-truth graphical dependencies are most predictive, our estimated blankets closely approximate the ground-truth structure. Overall, while the global dependencies encoded in the ground-truth network are indeed explanatory of the model's behaviour, our results show that the most faithful dependency structure for individual predictions at the local scale may differ and can be reliably retrieved using our proposed method.

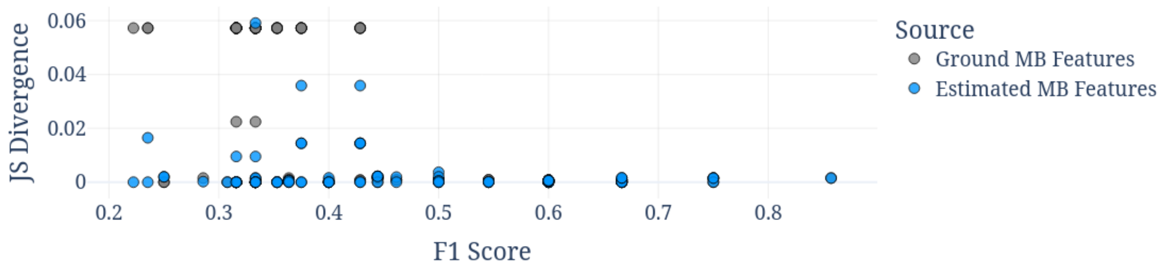
Table 1: Average Jensen-Shannon Divergence Given Fixed Features

Fixed Features	Asia	Alarm	Insurance
No Features	0.1433 ± 0.0706	0.1392 ± 0.0893	0.6049 ± 0.1694
Ground MB Features	0.0208 ± 0.0270	0.0045 ± 0.0104	0.0091 ± 0.0463
Estimated MB Features	$\mathbf{0.0028 \pm 0.0083}$	$\mathbf{0.0032 \pm 0.0068}$	$\mathbf{0.0089 \pm 0.0357}$
$\neg$ Ground MB Features	0.1359 ± 0.0616	0.0634 ± 0.0657	0.1622 ± 0.1815
$\neg$ Estimated MB Features	0.1408 ± 0.0690	0.0803 ± 0.0809	0.2348 ± 0.1849

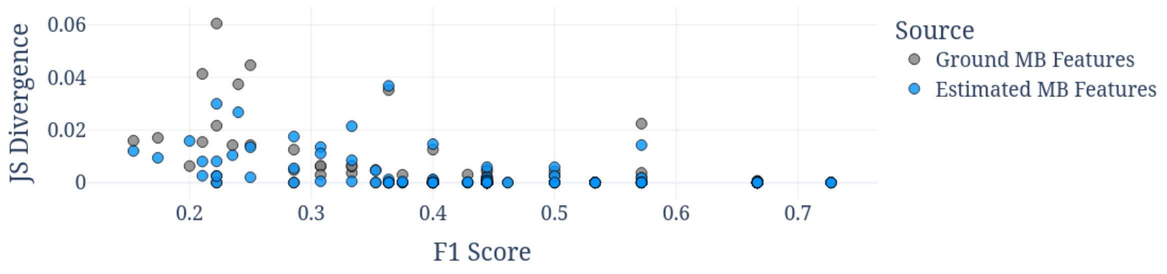
Table 2: Markov blanket retrieval metrics

	Asia	**Alarm**	**Insurance**
Precision	0.3146 ± 0.1313	0.3507 ± 0.1212	0.2821 ± 0.1134
Recall	0.8767 ± 0.1991	0.6375 ± 0.2313	0.3633 ± 0.0790
F1 Score	0.4429 ± 0.1387	0.4371 ± 0.1363	0.3003 ± 0.0487

[Asia]



[Alarm]



[Insurance]

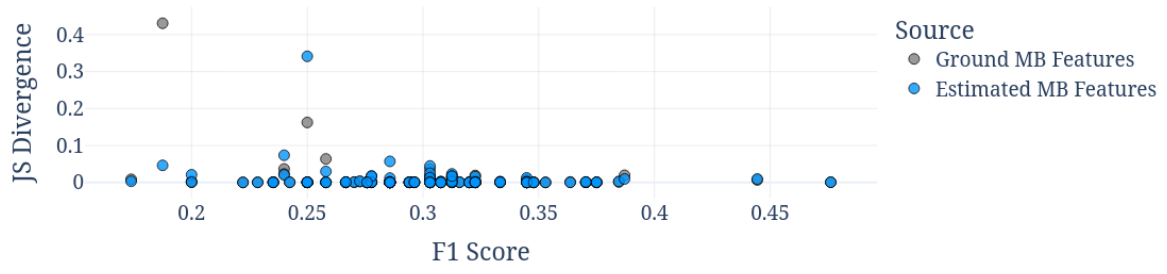


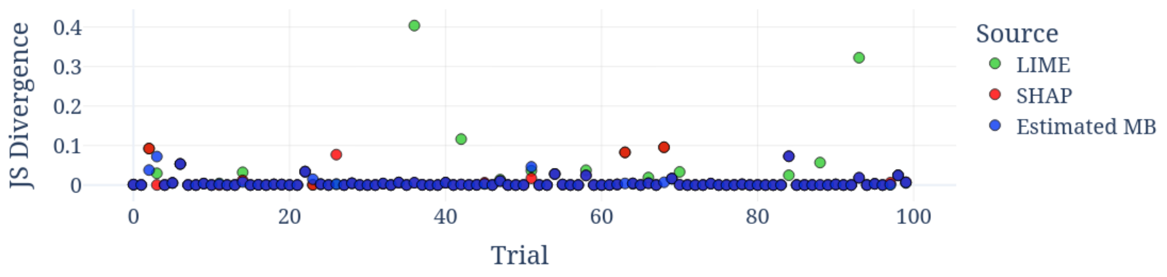
Figure 2: Change in Jensen-Shannon divergence from ground-truth and estimated Markov blankets over 100 predictions

Our approach also achieves a level of faithfulness comparable to SHAP, demonstrating its reliability even in the absence of prior structural assumptions or confirmed variable dependencies, this is shown Table 3 and Figure 3.

Table 3: Divergence given post-hoc technique fixed features

Fixed Features	Nursery	Tic-tac-toe
No Features	0.4166 ± 0.1369	0.2800 ± 0.1226
Estimated MB Features	**0.0055 ± 0.0136**	0.0302 ± 0.0935
LIME Features	0.0162 ± 0.0545	0.0564 ± 0.1339
Shap Features	0.0074 ± 0.0195	**0.0228 ± 0.0815**

[Nursery]



[TicTacToe]

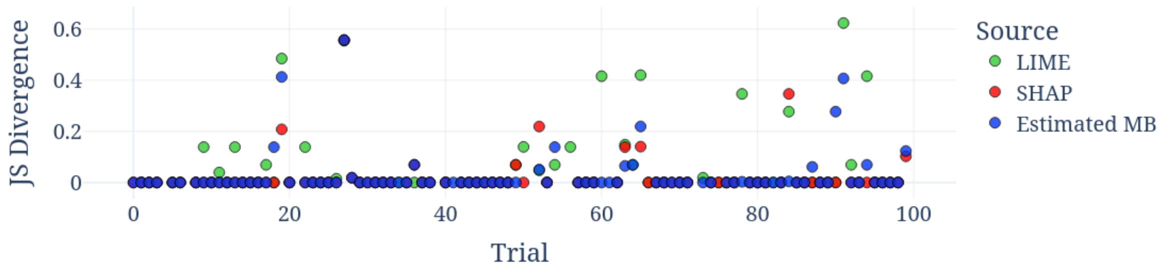


Figure 3: Jensen-Shannon divergence from post-hoc techniques for 100 predictions

6 Conclusion

In this paper, we demonstrated that neural networks do leverage variable dependencies during inference, and we introduced a mechanism capable of recovering the set of the most influential variables behind a prediction based on variable dependencies. Our experiments showed that when a dataset admits a directed acyclic graph representation, the predictions of fully connected neural networks trained on such data can be faithfully explained using the underlying graphical structure. Furthermore, our empirical results demonstrated that our method is also capable of producing faithful graphical explanations for individual predictions even in settings where no predominant or apparent structure is confirmed in advance.

Code: https://github.com/TumeloMkwambe/Honours-RP

References

Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador García, Sergio Gil-López, Daniel Molina, Richard Benjamins, et al. Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. *Information fusion*, 58:82–115, 2020.

Kevin Bache and Moshe Lichman. Tic-tac-toe endgame dataset. UCI Machine Learning Repository, University of California, Irvine. `https://archive.ics.uci.edu/dataset/101/tic+tac+toe+endgame`, 2013.

Daphne Koller and Nir Friedman. Structure learning in bayesian networks. In *Probabilistic Graphical Models: Principles and Techniques*, pages 790–804. MIT Press, 2009.

Daphne Koller, Nir Friedman, Lise Getoor, and Ben Taskar. Graphical models in a nutshell. *Introduction to statistical relational learning*, 43:1359–1366, 2007.

Trent Kyono, Yao Zhang, and Mihaela van der Schaar. Castle: Regularization via auxiliary causal graph discovery. *Advances in Neural Information Processing Systems*, 33:1501–1512, 2020.

Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30, 2017.

Grégoire Montavon, Alexander Binder, Sebastian Lapuschkin, Wojciech Samek, and Klaus-Robert Müller. Layer-wise relevance propagation: an overview. *Explainable AI: interpreting, explaining and visualizing deep learning*, pages 193–209, 2019.

M. Olave, V. Rajkovic, and M. Bohanec. An application for admission in public school systems. In *Expert Systems in Public Administration: Evolving Practices and Norms*. Elsevier Science Publishers, 1989.

Thomas P Quinn, Stephan Jacobs, Manisha Senadeera, Vuong Le, and Simon Coghlan. The three ghosts of medical ai: Can the black-box present deliver? *Artificial intelligence in medicine*, 124:102158, 2022.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144, 2016.

Avanti Shrikumar, Peyton Greenside, Anna Shcherbina, and Anshul Kundaje. Not just a black box: Learning important features through propagating activation differences. *arXiv preprint arXiv:1605.01713*, 2016.

Anders Søgaard. On the opacity of deep neural networks. *Canadian Journal of Philosophy*, 53(3):224–239, 2023.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *International conference on machine learning*, pages 3319–3328. PMLR, 2017.