Shapley Flow: A Graph-based Approach to Interpreting Model Predictions

Jiaxuan Wang
University of Michigan
jiaxuan@umich.edu

Jenna Wiens
University of Michigan
wiensj@umich.edu

Scott Lundberg
Microsoft Research
scott.lundberg@microsoft.com

Abstract

Many existing approaches for estimating feature importance are problematic because they ignore or hide dependencies among features. A causal graph, which encodes the relationships among input variables, can aid in assigning feature importance. However, current approaches that assign credit to nodes in the causal graph fail to explain the entire graph. In light of these limitations, we propose Shapley Flow, a novel approach to interpreting machine learning models. It considers the entire causal graph, and assigns credit to *edges* instead of treating nodes as the fundamental unit of credit assignment. Shapley Flow is the unique solution to a generalization of the Shapley value axioms for directed acyclic graphs. We demonstrate the benefit of using Shapley Flow to reason about the impact of a model's input on its output. In addition to maintaining insights from existing approaches, Shapley Flow extends the flat, set-based, view prevalent in game theory based explanation methods to a deeper, *graph-based*, view. This graph-based view enables users to understand the flow of importance through a system, and reason about potential interventions.

1 Introduction

Explaining a model's predictions by assigning importance to its inputs (*i.e.*, feature attribution) is critical to many applications in which a user interacts with a model to either make decisions or gain a better understanding of a system (Simonyan et al., 2013; Lundberg and Lee, 2017; Zhou et al., 2016; Shrikumar et al., 2017; Baehrens et al., 2010; Binder et al., 2016; Springenberg et al., 2014; Sundararajan et al., 2017; Fisher et al., 2018; Breiman, 2001). However, correlation among input features presents a challenge when estimating feature importance.

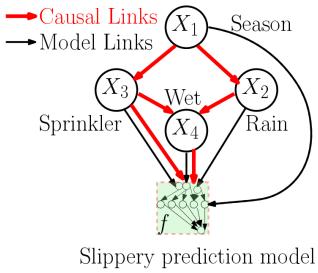


Figure 1: Causal graph for the sprinkler example from Chapter 1.2 of Pearl (2009). The model, f, can be expanded into its own graph. To simplify the exposition, although f takes 4 variables as input, we arbitrarily assumed that it only depends on X_3 and X_4 directly (*i.e.*, $f(X_1, X_2, X_3, X_4) = g(X_3, X_4)$ for some g).

Consider a motivating example adapted from Pearl (2009), in which we are given a model f that takes as input four features: the season of the year (X_1), whether or not it's raining (X_2), whether the sprinkler is on (X_3), and whether the pavement is wet (X_4) and outputs a prediction $f(\mathbf{x})$, representing the probability that the pavement is slippery (capital X denotes a random variable; lower case $\mathbf{x}$ denotes a particular sample). Assume, the inputs are related through the causal graph in **Figure 1**. When assigning feature importance, existing approaches that ignore this causal structure (Janzing et al., 2020; Sundararajan and Najmi, 2019; Datta et al., 2016) assign zero importance to the season, since it only indirectly affects the outcome through the other input variables. However, such a conclusion may lead a user astray - since changing X_1 would most definitely affect the outcome.

Recognizing this limitation, researchers have recently

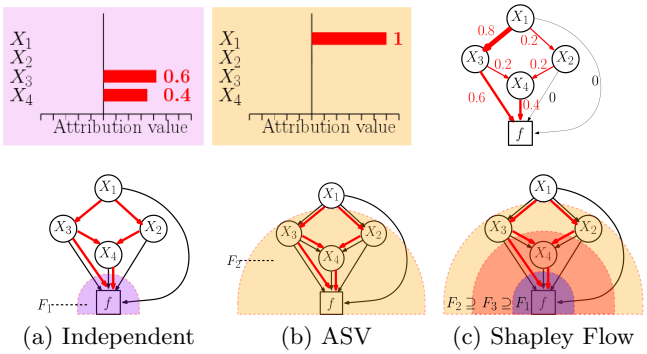
Proceedings of the 24th International Conference on Artificial Intelligence and Statistics (AISTATS) 2021, San Diego, California, USA. PMLR: Volume 130.

proposed approaches that leverage the causal structure among the input variables when assigning credit (Frye et al., 2019; Heskes et al., 2020). However, such approaches provide an incomplete picture of a system as they only assign credit to nodes in a graph. For example, the ASV method of Frye et al. (2019) solves the earlier problem of ignoring indirect or upstream effects, but it does so by ignoring direct or downstream effects. In our example, season would get all the credit despite the importance of the other variables. This again may lead a user astray - since intervening on X_3 or X_4 would affect the outcome, yet they are given no credit. The Causal Shapley values of Heskes et al. (2020) do assign credit to X_3 and X_4, but force this credit to be divided with X_1. This leads to the problem of features being given less importance simply because their downstream variables are also included in the graph.

Given that current approaches end up ignoring or dividing either downstream (*i.e.*, direct) or upstream (*i.e.*, indirect) effects, we develop Shapley Flow, a comprehensive approach to interpreting a model (or system) that incorporates the causal relationship among input variables, while accounting for both direct and indirect effects. In contrast to prior work, we accomplish this by reformulating the problem as one related to assigning credit to *edges* in a causal graph, instead of *nodes* (**Figure 2c**). Our key contributions are as follows.

- We propose the first (to the best of our knowledge) generalization of Shapley value feature attribution to graphs, providing a complete system-level view of a model.
- Our approach unifies three previous game theoretic approaches to estimating feature importance.
- Through examples on real data, we demonstrate how our approach facilitates understanding feature importance.

In this work, we take an axiomatic approach motivated by cooperative game theory, extending Shapley values to graphs. The resulting algorithm, Shapley Flow, generalizes past work on estimating feature importance (Lundberg and Lee, 2017; Frye et al., 2019; López and Saboya, 2009). The estimates produced by Shapley Flow represent the unique allocation of credit that conforms to several natural axioms. Applied to real-world systems, Shapley Flow can help a user understand both the direct and indirect impact of changing a variable, generating insights beyond current feature attribution methods.



(a) Independent (b) ASV (c) Shapley Flow

Figure 2: Top: Output of attribution methods for the example in **Figure 1**. Bottom: Causal structure (black edges) and explanation boundaries used by each method. As a reference, we copied the true causal links (red) from **Figure 1**. An explanation boundary $\mathcal{B} := (D, F)$ is a cut in the graph that defines a "model" F (nodes in the shaded area in each figure) to be explained. Refer to **Section 2.2** for a detailed discussion.

2 Problem Setup & Background

Given a model, or more generally a system, that takes a set of inputs and produces an output, we focus on the problem of quantifying the effect of each input on the output. Here, building off previous work, we formalize the problem setting.

2.1 Problem Setup

Quantifying the effect of each input on a model's output can be formulated as a credit assignment problem. Formally, given a target sample input $\boldsymbol{x}$, a background sample input $\boldsymbol{x'}$, and a model $f : \mathbb{R}^d \to \mathbb{R}$, we aim to explain the difference in output *i.e.*, $f(\boldsymbol{x}) - f(\boldsymbol{x'})$. We assume $\boldsymbol{x}$ and $\boldsymbol{x'}$ are of the same dimension d, and each entry can be either discrete or continuous.

We also assume access to a causal graph, as formally defined in Chapter 6 of Peters et al. (2017), over the d input variables. Given this graph, we seek an assignment function ϕ that assigns credit $\phi(e) \in \mathbb{R}$ to each edge e in the causal graph such that they collectively explain the difference $f(\boldsymbol{x}) - f(\boldsymbol{x'})$. In contrast with the classical setting (Lundberg and Lee, 2017; Sundararajan et al., 2017; Frye et al., 2020; Aas et al., 2019) in which credit is placed on features (*i.e.*, seeking a node assignment function $\psi(i) \in \mathbb{R}$ for $i \in [1 \cdots d]$), our edge-based approach is more flexible because we can recover node i's importance by defining $\psi(i) = \sum_{e \in \text{i's outgoing edges}} \phi(e)$. This exactly matches the classic Shapley axioms (Shapley, 1953) when the causal graph is degenerate with a single source node connected directly to all the input features.

Here, the effect of the input on the output is measured with respect to a background sample. For example, in a healthcare setting, we may set the features in the background sample to values that are deemed typical for a disease. We assume a single background value for notational convenience, but the formalism easily extends to the common scenario of multiple background values or a distribution of background values, P, by defining the explanation target to be $f(\boldsymbol{x}) - \mathbb{E}_{\boldsymbol{x}' \sim P} f(\boldsymbol{x}')$.

2.2 Feature Attribution with a Causal Graph

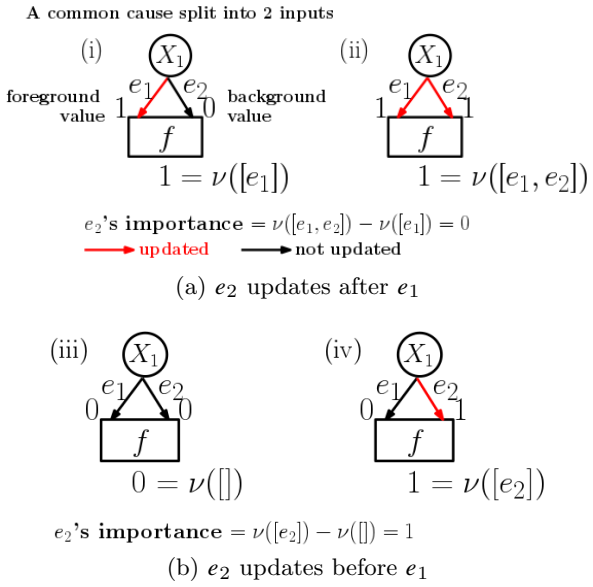
In our problem setup, we assume access to a causal graph, which can help in reasoning about the relationship among input variable. However, even with a causal graph, feature attribution remains challenging because it is unclear how to rightfully allocate credit for a prediction among the nodes and/or edges of the graph. Marrying interpretation with causality is an active field (see Moraffah et al. (2020) for a survey). A causal graph in and of itself does not solve feature attribution. While a causal graph can be used to answer a specific question with a specific counterfactual, summarizing many counterfactuals to give a comprehensive picture of the model is nontrivial. Furthermore, each node in a causal graph could be a blackbox model that needs to be explained. To address this challenge, we generalize game theoretic fairness principles to graphs.

Given a graph, $\mathcal{G}$, that consists of a causal graph over the the model of interest f and its inputs, we define the **boundary of explanation** as a cut $\mathcal{B} := (D, F)$ that partitions the input variables and the output of the model (*i.e.*, the nodes of the graph) into D and F where source nodes (nodes with no incoming edges) are in D and sink nodes (nodes with no outgoing edges) are in F. Note that $\mathcal{G}$ has a single sink, $f(\boldsymbol{x}) \in \mathbb{R}$. A cut set is the set of edges with one endpoint in D and another endpoint in F, denoted as $cut(\mathcal{B})$. It is helpful to think of F as an alternative model definition, where a boundary of explanation (*i.e.*, a model boundary) defines what part of the graph we consider to be the "model". If we collapse F into a single node that subsumes f, then $cut(\mathcal{B})$ represents the direct inputs to this new model.

Depending on the causal graph, multiple boundaries of explanation may exist. Recognizing this multiplicity of choices helps shed light on an ongoing debate in the community regarding feature attribution and whether one should perturb features while staying on the data manifold or perturb them independently (Chen et al., 2020; Janzing et al., 2020; Sundararajan and Najmi, 2019). On one side, many argue that perturbing features independently reveals the functional dependence of the model, and is thus *true to the model* (Janzing et al., 2020; Sundararajan and Najmi, 2019; Datta et al., 2016). However, independent perturbation of the data can create unrealistic or invalid sets of model input values. Thus, on the other side, researchers argue that one should perturb features while staying on the data manifold, and so be *true to the data* (Aas et al., 2019; Frye et al., 2019). However, this can result in situations in which features not used by the model are given non-zero attribution. Explanation boundaries help us unify these two viewpoints. As illustrated in **Figure 2a**, when we independently perturb features, we assume the causal graph is flat and the explanation boundary lies between $\boldsymbol{x}$ and f (*i.e.*, $\boldsymbol{D}$ contains all of the input variables). In this example, since features are assumed independent all credit is assigned to the features that directly impact the model output, and indirect effects are ignored (no credit is assigned to X_1 and X_2). In contrast, when we perform on-manifold perturbations with a causal structure, as is the case in Asymmetric Shapley Values (ASV) (Frye et al., 2019), all the credit is assigned to the source node because the source node determines the value of all nodes in the graph (**Figure 2b**). This results in a different boundary of explanation, one between the source nodes and the remainder of the graph. Although giving X_1 credit does not reflect the true functional dependence of f, it does for the model defined by F_2 (**Figure 2c**). Perturbations that were previously faithful to the data are faithful to a "model", just one that corresponds to a different boundary. See **Section 6** in the Appendix for how on-manifold perturbation (without a causal graph) can be unified using explanation boundaries.

Beyond the boundary directly adjacent to the model of interest, f, and the boundary directly adjacent to the source nodes, there are other potential boundaries (**Figure 2c**) a user may want to consider. However, simply generating explanations for each possible boundary can quickly overwhelm the user (**Figures 2a, 2b** in the main text, and **8a** in the Appendix). Our approach sidesteps the issue of selecting a single explanation boundary by considering all explanation boundaries simultaneously. This is made possible by assigning credit to the edges in a causal graph (**Figure 2c**). Edge attribution is strictly more powerful than feature attribution because we can simultaneously capture the direct and indirect impact of edges. We note that concurrent work by Heskes et al. (2020) also recognized that existing methods have difficulty capturing the direct and indirect effects simultaneously. Their solution however is node based, so it is forced to split credit between parents and children in the graph.

While other approaches to assign credit on a graph exist, (*e.g.*, Conductance from Dhamdhere et al. (2018) and DeepLift from Shrikumar et al. (2016)), they



(a) e_2 updates after e_1

(b) e_2 updates before e_1

Figure 3: Edge importance is measured by the change in output when an edge is added. When a model is non-linear, say $f = OR$, we need to average over all scenarios in which e_2 can be added to gauge its importance. **Section 3.1** has a detailed discussion.

were proposed in the context of understanding internal nodes of a neural network, and depend on implicit linearity and continuity assumptions about the model. We aim to understand the causal structure among the input nodes in a fully model agnostic manner, where discrete variables are allowed, and no differentiability assumption is made. To do this we generalize the widely used Shapley value (Adadi and Berrada, 2018; Mittelstadt et al., 2019; Lundberg et al., 2018; Sundararajan and Najmi, 2019; Frye et al., 2019; Janzing et al., 2020; Chen et al., 2020) to graphs.

3 Proposed Approach: Shapley Flow

Our proposed approach, Shapley Flow, attributes credit to edges of the causal graph. In this section, we present the intuition behind our approach and then formally show that it uniquely satisfies a generalization of the classic Shapley value axioms, while unifying previously proposed approaches.

3.1 Assigning Credit to Edges: Intuition

Given a causal graph defining the relationship among input variables, we re-frame the problem of feature attribution to focus on the edges of a graph rather than nodes. Our approach results in edge credit assignments as shown in **Figure 2c**. As mentioned above, this eliminates the need for multiple explanations (*i.e.*, bar charts) pertaining to each explanation boundary. Moreover, it allows a user to better understand the nuances of a system by providing information regarding what would happen if a single causal link breaks.

Shapley Flow is the unique assignment of credit to edges such that a relaxation of the classic Shapley value axioms are satisfied for all possible boundaries of explanation. Specifically, we extend the efficiency, dummy, and linearity axioms from Shapley (1953) and add a new axiom related to boundary consistency. Efficiency states that the attribution of edges on any boundary must add up to $f(\mathbf{x}) - f(\mathbf{x}')$. Linearity states that explaining a linear combination of models is the same as explaining each model, and linearly combining the resulting attributions. Dummy states that if adding an edge does not change the output in any scenarios, the edge should be assigned 0 credit. Boundary consistency states that edges shared by different boundaries need to have the same attribution when explained using either boundary. These concepts are illustrated in **Figure 4** and formalized in **Section 3.3**.

An edge is important if removing it causes a large change in the model's prediction. However, what does it mean to remove an edge? If we imagine every edge in the graph as a channel that sends its source node's current value to its target node, then removing an edge e simply means messages sent through e fail. In the context of feature attribution, in which we aim to measure the difference between $f(\mathbf{x}) - f(\mathbf{x}')$, this means that e's target node still relies on the source's background value in $\mathbf{x}'$ to update its current value, as opposed to the source node's foreground value in $\mathbf{x}$, as illustrated in **Figure 3a**. Note that treating edge removal as replacing the parent node with the background value is equivalent to the approach advocated by Janzing et al. (2020), and matches the default behavior of SHAP and related methods. However, we cannot simply toggle edges one at a time. Consider a simple OR function $g(X_1, X_2) = X_1 \vee X_2$, with $x_1 = 1$, $x_2 = 1$, $x_1' = 0$, $x_2' = 0$. Removing either of the edges alone, would not affect the output and both x_1 and x_2 would be (erroneously) assigned 0 credit.

To account for this, we consider all scenarios (or partial histories) in which the edge we care about can be added (see **Figure 3b**). Here, ν is a function that takes a list of edges and evaluates the network with edges updated in the order specified by the list. For example, $\nu([e_1])$ corresponds to the evaluation of f when only e_1 is updated. Similarly $\nu([e_1, e_2])$ is the evaluation of f when e_1 is updated followed by e_2. The list $[e_1, e_2]$ is also referred to as a (complete) *history* as it specifies how $\mathbf{x}'$ changes to $\mathbf{x}$.

For the same edge, attributions derived from different explanation boundaries should agree, otherwise simply

including more details of a model in the causal graph would change upstream credit allocation, even though the model implementation was unchanged. We refer to this property as *boundary consistency*. The Shapley Flow value for an edge is the difference in model output when removing the edge averaged over all histories that are boundary consistent (as defined below).

3.2 Model explanation as value assignments in games

The concept of Shapley value stems from game theory, and has been extensively applied in model interpretability (Štrumbelj and Kononenko, 2014; Datta et al., 2016; Lundberg and Lee, 2017; Frye et al., 2019; Janzing et al., 2020). Before we formally extend it to the context of graphs, we define the credit assignment problem from a game theoretic perspective.

Given the message passing system in **Section 3.1**, we formulate the credit assignment problem as a game specific to an explanation boundary $\mathcal{B} := (D, F)$. The game consists of a set of players $\mathcal{P}_\mathcal{B}$, and a payoff function $v_\mathcal{B}$. We model each edge external to F as a player. A *history* is a list of edges detailing the event from $t = 0$ (values being $\mathbf{x}'$) to $t = T$ (values being $\mathbf{x}$). For example, the history $[i, j, i]$ means that the edge i finishes transmitting a message containing its source node's most recent value to its target node, followed by the edge j, and followed by the edge i again. A *coalition* is a partial history from $t = 0$ to any $t \in [0 \cdots T]$. The *payoff function*, v, associates each coalition with a real number, and is defined in our case as the evaluation of F following the coalition.

This setup is a generalization of a typical cooperative game in which the ordering of players does not matter (only the set of players matters). However, given our message passing system, history is important. In the following sections, we denote '+' as list concatenation, '[]' as an empty coalition, and $\mathcal{H}_\mathcal{B}$ as the set of all possible histories. We denote $\tilde{\mathcal{H}}_\mathcal{B} \subseteq \mathcal{H}_\mathcal{B}$ as the set of boundary consistent histories. The corresponding coalitions for $\mathcal{H}_\mathcal{B}$ and $\tilde{\mathcal{H}}_\mathcal{B}$ are denoted as $\mathcal{C}_\mathcal{B}$ and $\tilde{\mathcal{C}}_\mathcal{B}$ respectively. A sample game setup is illustrated in **Figure 3**.

3.3 Axioms

We formally extend the classic Shapley value axioms (efficiency, linearity, and dummy) and include one additional axiom, the boundary consistency axiom, that connects all boundaries together.

- Boundary consistency: for any two boundaries $\mathcal{B}_1 = (D_1, F_1)$ and $\mathcal{B}_2 = (D_2, F_2)$, $\phi_{v_{\mathcal{B}_1}}(i) = \phi_{v_{\mathcal{B}_2}}(i)$ for $i \in cut(\mathcal{B}_1) \cap cut(\mathcal{B}_2)$

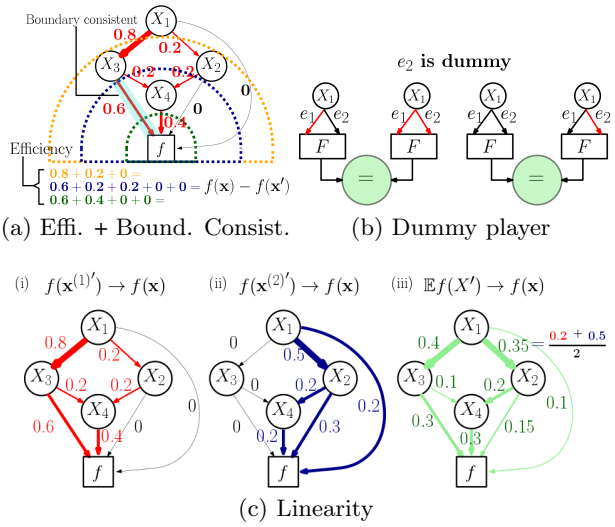


Figure 4: Illustration for axioms for Shapley Flow. Except for boundary consistency, all axioms stem from Shapley value's axioms (Shapley, 1953). Detailed explanations are included in **Section 3.3**.

> For edges that are shared between boundaries, their attributions must agree. In **Figure 4a**, the edge wrapped by a teal band is shared by both the blue and green boundaries, forcing them to give the same attribution to the edge.

In the general setting, not all credit assignments are boundary consistent; different boundaries could result in different attributions for the same edge[1]. This occurs when histories associated with different boundaries are inconsistent (**Figure 5**). Moving the boundary from $\mathcal{B}$ to $\mathcal{B}^*$ (where $\mathcal{B}^*$ is the boundary with D containing f's inputs), results in a more detailed set of histories. This expansion has 2 constraints. First, any history in the expanded set follows the message passing system in **Section 3.1**. Second, when a message passes through the boundary, it immediately reaches the end of computation as F is assumed to be a blackbox.

Denoting the history expansion function into $\mathcal{B}^*$ as HE (*i.e.*, HE takes a history h as input and expand it into a set of histories in $\mathcal{B}^*$ as output) and denoting the set of all boundaries as $\mathcal{M}$, a history h is *boundary consistent* if $\exists h_\mathcal{B} \in \mathcal{H}_\mathcal{B}$ for all $\mathcal{B} \in \mathcal{M}$ such that

$$\left(\bigcap_{\mathcal{B}\in\mathcal{M}} HE(h_\mathcal{B})\right) \cap HE(h) \neq \emptyset$$

That is h needs to have at least one fully detailed history in which all boundaries can agree on. $\tilde{\mathcal{H}}$ is all

[1] We include an example in the Appendix **Section 11** to demonstrate why considering all histories $\mathcal{H}$ can violate boundary consistency, thus motivating the need to only focus on boundary consistent histories.

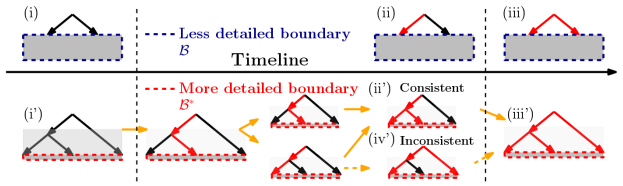


Figure 5: Boundary Consistency. For the blue boundary (upper), we show one potential history h. When we expand h to the red boundary (lower), h corresponds to multiple histories as long as each history contains states that match (i) (ii) and (iii). (i') matches (i), no messages are received in both states. (ii') matches (ii), the full impact of message transmitted through the left edge is received at the end of computation. (iii') matches (iii), all messages are received. In contrast, the history containing (iv') has no state matching (ii), and thus is inconsistent with h.

histories in $\mathcal{H}$ that are boundary consistent. We rely on this notion of boundary consistency in generalizing the Shapley axioms to any explanation boundary, $\mathcal{B}$:

- Efficiency: $\sum_{i \in cut(\mathcal{B})} \phi_{v_\mathcal{B}}(i) = f(\mathbf{x}) - f(\mathbf{x}')$.

  In the general case where $v_\mathcal{B}$ can depend on the ordering of h, the sum is $\sum_{h \in \tilde{\mathcal{H}}_\mathcal{B}} \frac{v_\mathcal{B}(h)}{|\tilde{\mathcal{H}}_\mathcal{B}|} - v_\mathcal{B}([])$. But when the game is defined by a model function f, $\sum_{h \in \tilde{\mathcal{H}}_\mathcal{B}} v_\mathcal{B}(h)/|\tilde{\mathcal{H}}_\mathcal{B}| = f(\boldsymbol{x})$ and $v_\mathcal{B}([]) = f(\boldsymbol{x'})$. An illustration with 3 boundaries is shown in **Figure 4a**.

- Linearity: $\phi_{\alpha u + \beta v} = \alpha\phi_u + \beta\phi_v$ for any payoff functions u and v and scalars α and β.

  Linearity enables us to compute a linear ensemble of models by independently explaining each model and then linearly weighting the attributions. Similarly, we can explain $f(\mathbf{x}) - \mathbb{E}(f(X'))$ by independently computing attributions for each background sample $\mathbf{x}^{(i)'}$ and then taking the average of the attributions, without recomputing from scratch whenever the background sample's distribution changes. An illustration with 2 background samples is shown in **Figure 4c**.

- Dummy player: $\phi_{v_\mathcal{B}}(i) = 0$ if $v_\mathcal{B}(S + [i]) = v_\mathcal{B}(S)$ for all $S, S + [i] \in \tilde{C}_\mathcal{B}$ for $i \in cut(\mathcal{B})$.

  Dummy player states that if an edge does not change the model's output when added to in all possible coalitions, it should be given 0 attribution. In **Figure 4b**, e_2 is a dummy edge because starting from any coalition, adding e_2 wouldn't change the output.

These last three axioms are extensions of Shapley's axioms. Note that Shapley value also requires the symmetry axiom because the game is defined on a set of players. For Shapley Flow values this symmetry assumption is encoded through our choice of an ordered history formulation. (Appendix **Section 8**).

3.4 Shapley Flow is the unique solution

Shapley Flow uniquely satisfies all axioms from the previous section. Here, we describe the algorithm, show its formulae, and state its properties. Please refer to **Appendix 7** and **8** for the pseudo code[2] and proof.

Description: Define a configuration of a graph as an arbitrary ordering of outgoing edges of a node when it is traversed by depth first search. For each configuration, we run depth first search starting from the source node, processing edges in the order of the configuration. When processing an edge, we update the value of the edge's target node by making the edge's source node value visible to its function. If the edge's target node is the sink node, the difference in the sink node's output is credited to every edge along the search path from source to sink. The final result averages over attributions for all configurations.

Formulae: Denote the attribution of Shapley Flow to a path as $\tilde{\phi}_v$, and the set of all possible orderings of source nodes to a sink path generated by depth first search (DFS) as Π_{dfs}. For each ordering $\pi \in \Pi_{\text{dfs}}$, the inequality of $\pi(j) < \pi(i)$ denotes that path j precedes path i under π. Since v's input is a list of edges, we define $\tilde{v}$ to work on a list of paths. The evaluation of $\tilde{v}$ on a list of paths is the value of v evaluated on the corresponding edge traversal ordering. Then

$$\tilde{\phi}_v(i) = \sum_{\pi \in \Pi_{\text{dfs}}} \frac{\tilde{v}([j : \pi(j) \le \pi(i)]) - \tilde{v}([j : \pi(j) < \pi(i)])}{|\Pi_{\text{dfs}}|} \tag{1}$$

To obtain an edge e's attribution $\phi_v(e)$, we sum the path attributions for all paths that contains e.

$$\phi_v(e) = \sum_{p \in \text{paths in } \mathcal{G}} \mathbb{1}_{p \text{ contains}}(e)\tilde{\phi}_v(p) \tag{2}$$

Additional properties: Shapley Flow has the following beneficial properties beyond the axioms.

Generalization of SHAP: if the graph is flat, the edge attribution is equal to feature attribution from SHAP because each input node is paired with a single edge leading to the model.

Generalization of ASV: the attribution to the source nodes is the same as in ASV if all the dependencies among features are modeled by the causal graph.

Generalization of Owen value: if the graph is a tree, the edge attribution for incoming edges to the leaf

[2] code can be found in `https://github.com/nathanwang000/Shapley-Flow`

nodes is the Owen value (López and Saboya, 2009) with a coalition structure defined by the tree.

Implementation invariance: implementation invariance means that no matter how the function is implemented, so long as the input and output remain unchanged, so does the attribution (Sundararajan et al., 2017), which directly follows boundary consistency (*i.e.*, knowing f's computational graph or not wouldn't change the upstream attribution).

Conservation of flow: efficiency and boundary consistency imply that the sum of attributions on a node's incoming edges equals the sum of its outgoing edges.

Model agnostic: Shapley Flow can explain arbitrary (non-differentiable) machine learning pipelines.

4 Practical Application

Shapley Flow highlights both the direct and indirect impact of features. In this section, we consider several applications of Shapley Flow. First, in the context of a linear model, we verify that the attributions match our intuition. Second, we show how current feature attribution approaches lead to an incomplete understanding of a system compared to Shapley Flow.

4.1 Experimental Setup

We illustrate the application of Shapley Flow to a synthetic and a real dataset. In addition, we include results for a third dataset in the Appendix. Note that our algorithm assumes a causal graph is provided as input. In recent years there has been significant progress in causal graph estimation (Glymour et al., 2019; Peters et al., 2017). However, since our focus is not on causal inference, we make simplifying assumptions in estimating the causal graphs (see **Section 9.2** of the Appendix for details).

Datasets. *Synthetic*: As a sanity check, we first experiment with synthetic data. We create a random graph dataset with 10 nodes. A node i is randomly connected to node j (with j pointing to i) with 0.5 probability if $i > j$, otherwise 0. The function at each node is linear with weights generated from a standard normal distribution. Sources follow a $N(0, 1)$ distribution. This results in a graph with a single sink node associated with function f (*i.e.*, the 'model' of interest). The remainder of the graph corresponds to the causal structure among the input variables.

National Health and Nutrition Examination Survey: This dataset consists of $9,932$ individuals with 18 demographic and laboratory measurements (Cox, 1998). We used the same preprocessing as described by Lundberg et al. (2020). Given these inputs, the model, f, aims to predict survival.

Model training. We train f using an 80/20 random train/test split. For experiments with linear models, f is trained with linear regression. For experiments with non-linear models, f is fitted by 100 XGBoost trees with a max depth of 3 for up to 1000 epochs, using the Cox loss.

Causal Graph. For the nutrition dataset, we constructed a causal graph (**Figure 9a**) based on our limited understanding of the causal relationship among input variables. This graph represents an oversimplification of the true underlying causal relationships and is for illustration purposes only. We assigned attributes predetermined at birth (age, race, and sex) as source nodes because they temporally precede all other features. Poverty index depends on age, race, and sex (among other variables captured by the poverty index noise variable) and impacts one's health. Other features pertaining to health depend on age, race, sex, and poverty index. Note that the relationship among some features is deterministic. For example, pulse pressure is the difference between systolic and diastolic blood pressure. We include causal edges to account for such facts. We also account for when features have natural groupings. For example, transferrin saturation (TS), total iron binding capacity (TIBC), and serum iron are all related to blood iron. Serum albumin and serum protein are both blood protein measures. Systolic and diastolic blood pressure can be grouped into blood pressure. Sedimentation rate and white blood cell counts both measure inflammation. We add these higher level grouping concepts as new latent variables in the graph. To account for noise in modeling the outcome (*i.e.*, the effect of exogenous variables that are not used as input to the model), we add an independent noise node to each node (detailed in **Section 9.2** in the Appendix). **The resulting causal structure is an oversimplification of the true causal structure; the relationship between source nodes (e.g., race) and biomarkers is far more complex (Robinson et al., 2020). Nonetheless, it can help in understanding the in/direct effects of input variables on the outcome**.

4.2 Baselines

We compare Shapley Flow with other game theoretic feature attribution methods: independent SHAP (Lundberg and Lee, 2017), on-manifold SHAP (Aas et al., 2019), and ASV (Frye et al., 2019), covering both independent and on-manifold feature attribution.

Since Shapley value based methods are expensive to compute exactly, we use a Monte Carlo approximation

of **Equation 1**. In particular, we sample orderings from Π_{dfs} and average across those orderings.We randomly selected a background sample from each dataset and share it across methods so that each uses the same background. A single background sample allows us to ignore differences in methods due to variations in background sampling and is easier to explain the behavior of baselines (Merrick and Taly, 2020). To show that our result is not dependent on the particular choice of background sample, we include an example averaged over 100 background samples in **Section 10.4** in the Appendix (the qualitative results shown with a single background still holds). We sample $10,000$ orderings from each approach to generate the results. Since there's no publicly available implementation for ASV, we show the attribution for source nodes (the noise node associated with each feature) obtained from Shapley Flow (summing attributions of outgoing edges), as they are equivalent given the same causal graph. Since noise node's credit is used, intermediate nodes can report non zero credit in ASV.

For convenience of visual inspection, we show top 10 links used by Shapley Flow (credit measured in absolute value) on the nutrition dataset.

4.3 Sanity checks with linear models

To build intuition, we first examine linear models (*i.e.*, $f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + b$ where $\mathbf{w} \in \mathbb{R}^d$ and $b \in \mathbb{R}$; the causal dependence inside the graph is also linear). When using a linear model the ground truth direct impact of changing feature X_i is $w_i(x_i - x_i')$ (that is the change in output due to X_i directly), and the ground truth indirect impact is defined as the change in output when an intervention changes x_i' to x_i. Note that when the model is linear, only 1 Monte Carlo sample is sufficient to recover the exact attribution because feature ordering doesn't matter (the output function is linear in any boundary edges, thus only the background and foreground value of a feature matters). This allows us to bypass sampling errors and focus on analyzing the algorithms.

Results for explaining the datasets are included in **Table 1**. We report the mean absolute error (and its variance) associated with the estimated attribution (compared against the ground truth attribution), averaged across $1,000$ randomly selected test examples and all graph nodes for both datasets. Note that only Shapley flow results in no error for both direct and indirect effects.

4.4 Examples with non-linear models

We demonstrate the benefits of Shapley Flow with non-linear models containing both discrete and continuous variables. As a reminder, the baseline methods are not competing with Shapley Flow as the latter can recover all the baselines given the corresponding causal structure (**Figure 2**). Instead, we highlight why a holistic understanding of the system is better.

Methods	Nutrition (**D**)	Synthetic (**D**)	Nutrition (**I**)	Synthetic (**I**)
Independent	**0.0** (± 0.0)	**0.0** (± 0.0)	0.8 (± 2.7)	1.1 (± 1.4)
On-manifold	1.3 (± 2.5)	0.8 (± 0.7)	0.9 (± 1.6)	1.5 (± 1.5)
ASV	1.5 (± 3.3)	1.2 (± 1.4)	0.6 (± 1.9)	1.1 (± 1.5)
Shapley Flow	**0.0** (± 0.0)	**0.0** (± 0.0)	**0.0** (± 0.0)	**0.0** (± 0.0)

Table 1: Mean absolute error (std) for all methods on direct (**D**) and indirect (**I**) effect for linear models. Shapley Flow makes no mistake across the board.

Independent SHAP ignores the indirect impact of features. Take an example from the nutrition dataset (**Figure 6**). Independent SHAP gives lower attribution to age compared to ASV. This happens because age, in addition to its direct impact, indirectly affects the output through blood pressure, as shown by Shapley Flow (**Figure 6a**). Independent SHAP fails to account for the indirect impact of age, leaving the user with a potentially misleading impression that age is less important than it actually is.

On-manifold SHAP provides a misleading interpretation. With the same example (**Figure 6**), we observe that on-manifold SHAP strongly disagrees with independent SHAP, ASV, and Shapley Flow on the importance of age. Not only does it assign more credit to age, it also flips the sign, suggesting that age is protective. However, **Figure 7a** shows that age and earlier mortality are positively correlated; then how could age be protective? **Figure 7b** provides an explanation. Since SHAP considers all partial histories regardless of the causal structure, when we focus on serum magnesium and age, there are two cases: serum magnesium updates before or after age. We focus on the first case because it is where on-manifold SHAP differs from other baselines (all baselines already consider the second case as it satisfies the causal ordering). When serum magnesium updates before age, the expected age given serum magnesium is higher than the foreground age (yellow line above the black marker). Therefore when age updates to its foreground value, we observe a decrease in age, leading to a decrease in the output (so age appears to be protective). From both an in/direct impact perspective, on-manifold perturbation can be misleading since it is based not on causal but on observational relationships.

ASV ignores the direct impact of features. As shown in **Figure 6**, serum protein appears to be more important in independent SHAP compared to ASV. From Shapley Flow (**Figure 6a**), we know serum protein is not given attribution in ASV because its up-

stream node, blood protein, gets all the credit. However, looking at ASV alone, one fails to understand that intervening on serum protein could have a larger impact on the output.

Shapley Flow shows both direct and indirect impacts of features. Focusing on the attribution given by Shapley Flow (**Figure 6a**). We not only observe similar direct impacts in variables compared to independent SHAP, but also can trace those impacts to their source nodes, similar to ASV. Furthermore, Shapley Flow provides more detail compared to other approaches. For example, using Shapley Flow we gain a better understanding of the ways in which age impacts survival. The same goes for all other features. This is useful because causal links can change (or break) over time. Our method provides a way to reason through the impact of such a change.

More case studies with an additional dataset are included in the Appendix.

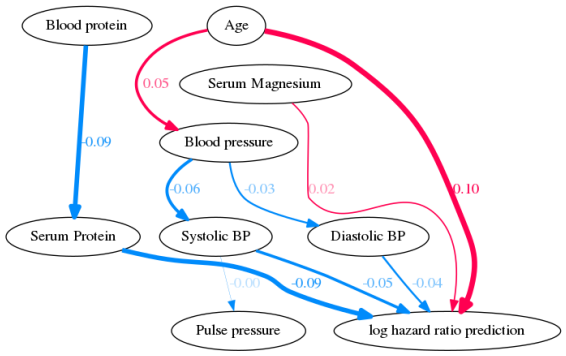
5 Discussion and Conclusion

We extend the classic Shapley value axioms to causal graphs, resulting in a unique edge attribution method: Shapley Flow. It unifies three previous Shapley value based feature attribution methods, and enables the joint understanding of both the direct and indirect impact of features. This more comprehensive understanding is useful when interpreting any machine learning model, both 'black box' methods, and 'interpretable' methods (such as linear models).

The key message of the paper is that model interpretation methods should include the whole machine learning pipeline in order to understand when a model can be applied. While our approach relies on access to a complete causal graph, Shapley Flow is still valuable because a) there are well-established causal relationships in domains such as healthcare, and ignoring such relationships can produce confusing explanations; b) recent advancements in causal estimation are complementary to our work and make defining these graphs easier; c) finally and most importantly, existing methods already implicitly make causal assumptions, Shapley Flow just makes these assumptions explicit (**Figure 2**). However, this does open up new research opportunities. Can Shapley Flow work with partially defined causal graphs? How to explore Shapley Flow attribution when the causal graph is complex? Can Shapley Flow be useful for feature selection? We leave those questions for future work.

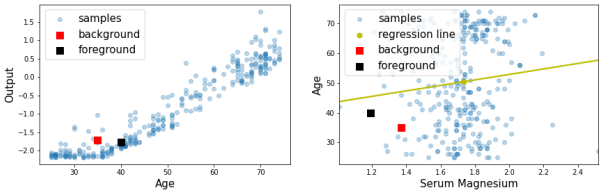
Top features	Age	Serum Magnesium	Serum Protein
Background sample	35	1.37	7.6
Foreground sample	40	1.19	6.5

Attributions	Independent	On-manifold	ASV
Age	0.1	-0.26	0.16
Serum Magnesium	0.02	0.2	0.02
Serum Protein	-0.09	0.07	0.0
Blood pressure	0.0	0.0	-0.14
Systolic BP	-0.05	-0.05	0.0
Diastolic BP	-0.04	-0.07	0.0
Serum Cholesterol	0.0	-0.15	0.0
Serum Albumin	0.0	-0.14	0.0
Blood protein	0.0	0.0	-0.08
White blood cells	0.0	0.11	0.0
Race	0.0	0.09	0.0
BMI	-0.0	0.08	-0.0
TIBC	0.0	0.06	0.0
Sex	0.0	-0.05	0.0
TS	0.0	0.05	0.0
Pulse pressure	0.0	-0.05	0.0
Poverty index	0.0	0.04	0.0
Red blood cells	0.0	0.03	0.0
Serum Iron	0.0	-0.02	0.0
Sedimentation rate	0.0	0.0	0.0
Iron	0.0	0.0	-0.0
Inflamation	0.0	0.0	0.0



(a) Shapley Flow

Figure 6: Comparison among baselines on a sample (top table) from the nutrition dataset, showing top 10 features/edges.



(a) Age vs. output (b) Age vs. magnesium

Figure 7: Age appears to be protective in on-manifold SHAP because it steals credit from other variables.

References

Aas, K., Jullum, M., and Løland, A. (2019). Explaining individual predictions when features are dependent: More accurate approximations to shapley values. *arXiv preprint arXiv:1903.10464*.

Adadi, A. and Berrada, M. (2018). Peeking inside the black-box: A survey on explainable artificial intelligence (xai). *IEEE Access*, 6:52138–52160.

Baehrens, D., Schroeter, T., Harmeling, S., Kawanabe, M., Hansen, K., and Müller, K.-R. (2010). How to explain individual classification decisions. *The Journal of Machine Learning Research*, 11:1803–1831.

Binder, A., Montavon, G., Lapuschkin, S., Müller, K.-R., and Samek, W. (2016). Layer-wise relevance propagation for neural networks with local renormalization layers. In *International Conference on Artificial Neural Networks*, pages 63–71. Springer.

Breiman, L. (2001). Random forests. *Machine learning*, 45(1):5–32.

Chen, H., Janizek, J. D., Lundberg, S., and Lee, S.-I. (2020). True to the model or true to the data? *arXiv preprint arXiv:2006.16234*.

Cox, C. S. (1998). *Plan and operation of the NHANES I Epidemiologic Followup Study, 1992*. Number 35. National Ctr for Health Statistics.

Datta, A., Sen, S., and Zick, Y. (2016). Algorithmic transparency via quantitative input influence: Theory and experiments with learning systems. In *2016 IEEE symposium on security and privacy (SP)*, pages 598–617. IEEE.

Dhamdhere, K., Sundararajan, M., and Yan, Q. (2018). How important is a neuron? *arXiv preprint arXiv:1805.12233*.

Fisher, A., Rudin, C., and Dominici, F. (2018). All models are wrong but many are useful: Variable importance for black-box, proprietary, or misspecified prediction models, using model class reliance. *arXiv preprint arXiv:1801.01489*, pages 237–246.

Frye, C., de Mijolla, D., Cowton, L., Stanley, M., and Feige, I. (2020). Shapley-based explainability on the data manifold. *arXiv preprint arXiv:2006.01272*.

Frye, C., Feige, I., and Rowat, C. (2019). Asymmetric shapley values: incorporating causal knowledge into model-agnostic explainability. *arXiv preprint arXiv:1910.06358*.

Glymour, C., Zhang, K., and Spirtes, P. (2019). Review of causal discovery methods based on graphical models. *Frontiers in genetics*, 10:524.

Heskes, T., Sijben, E., Bucur, I. G., and Claassen, T. (2020). Causal shapley values: Exploiting causal knowledge to explain individual predictions of complex models. *Advances in neural information processing systems*.

Janzing, D., Minorics, L., and Blöbaum, P. (2020). Feature relevance quantification in explainable ai: A causal problem. In *International Conference on Artificial Intelligence and Statistics*, pages 2907–2916. PMLR.

López, S. and Saboya, M. (2009). On the relationship between shapley and owen values. *Central European Journal of Operations Research*, 17(4):415.

Lundberg, S. M., Erion, G., Chen, H., DeGrave, A., Prutkin, J. M., Nair, B., Katz, R., Himmelfarb, J., Bansal, N., and Lee, S.-I. (2020). From local explanations to global understanding with explainable ai for trees. *Nature machine intelligence*, 2(1):2522–5839.

Lundberg, S. M., Erion, G. G., and Lee, S.-I. (2018). Consistent individualized feature attribution for tree ensembles. *arXiv preprint arXiv:1802.03888*.

Lundberg, S. M. and Lee, S.-I. (2017). A unified approach to interpreting model predictions. In *Advances in neural information processing systems*, pages 4765–4774.

Merrick, L. and Taly, A. (2020). The explanation game: Explaining machine learning models using shapley values. In Holzinger, A., Kieseberg, P., Tjoa, A. M., and Weippl, E., editors, *Machine Learning and Knowledge Extraction*, pages 17–38, Cham. Springer International Publishing.

Mittelstadt, B., Russell, C., and Wachter, S. (2019). Explaining explanations in ai. In *Proceedings of the conference on fairness, accountability, and transparency*, pages 279–288.

Moraffah, R., Karami, M., Guo, R., Raglin, A., and Liu, H. (2020). Causal interpretability for machine learning-problems, methods and evaluation. *ACM SIGKDD Explorations Newsletter*, 22(1):18–33.

Pearl, J. (2009). *Causality*. Cambridge university press.

Peters, J., Janzing, D., and Schölkopf, B. (2017). *Elements of causal inference*. The MIT Press.

Robinson, W. R., Renson, A., and Naimi, A. I. (2020). Teaching yourself about structural racism will improve your machine learning. *Biostatistics*, 21(2):339–344.

Shapley, L. S. (1953). A value for n-person games. *Contributions to the Theory of Games*, 2(28):307–317.

Shrikumar, A., Greenside, P., and Kundaje, A. (2017). Learning important features through prop-

agating activation differences. *arXiv preprint arXiv:1704.02685*.

Shrikumar, A., Greenside, P., Shcherbina, A., and Kundaje, A. (2016). Not just a black box: Learning important features through propagating activation differences. *arXiv preprint arXiv:1605.01713*.

Simonyan, K., Vedaldi, A., and Zisserman, A. (2013). Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*.

Springenberg, J. T., Dosovitskiy, A., Brox, T., and Riedmiller, M. (2014). Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*.

Štrumbelj, E. and Kononenko, I. (2014). Explaining prediction models and individual predictions with feature contributions. *Knowledge and information systems*, 41(3):647–665.

Sundararajan, M. and Najmi, A. (2019). The many shapley values for model explanation. *arXiv preprint arXiv:1908.08474*.

Sundararajan, M., Taly, A., and Yan, Q. (2017). Axiomatic attribution for deep networks. *International Conference on Machine Learning*.

Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., and Torralba, A. (2016). Learning deep features for discriminative localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2921–2929.