

---

# Threading the Needle of On and Off-Manifold Value Functions for Shapley Explanations

---

Chih-Kuan Yeh

Carnegie Mellon University

Kuan-Yun Lee

UC Berkeley

Frederick Liu

Google

Pradeep Ravikumar

Carnegie Mellon University

## Abstract

A popular explainable AI (XAI) approach to quantify feature importance of a given model is via Shapley values. These Shapley values arose in cooperative games, and hence a critical ingredient to compute these in an XAI context is a so-called value function, that computes the “value” of a subset of features, and which connects machine learning models to cooperative games. There are many possible choices for such value functions, which broadly fall into two categories: on-manifold and off-manifold value functions, which take an observational and an interventional viewpoint respectively. Both these classes however have their respective flaws, where on-manifold value functions violate key axiomatic properties and are computationally expensive while off-manifold value functions pay less heed to the data manifold and evaluate the model on regions for which it wasn’t trained. Thus, there is no consensus on which class of value functions to use. In this paper, we show that in addition to these existing issues, both classes of value functions are prone to adversarial manipulations on low density regions. We formalize the desiderata of value functions that respect both the model and the data manifold in a set of axioms and are robust to perturbation on off-manifold regions, and show that there exists a unique value function that satisfies these axioms, which we term the Joint Baseline value function, and the resulting Shapley value the Joint Baseline Shapley (JBshap), and validate the effectiveness of JBshap in experiments.

## 1 Introduction

Shapley values [Shapley, 1988] were originally proposed to measure the contributions of players in the context of cooperative games. They have since gained significant traction in the area of *explainable machine learning*, where they are used to measuring contributions of different features [Grömping, 2007, Lindeman, 1980, Owen, 2014, Owen and Prieur, 2017, Datta et al., 2016, Lundberg and Lee, 2017], data points [Ghorbani and Zou, 2019, Jia et al., 2019], neurons [Ghorbani and Zou, 2020], and concepts [Yeh et al., 2020], towards the output of a learned model.

Despite their popularity, using Shapley values in an explainable machine learning context is not straightforward. In a cooperative game setting, the game is characterized by a set value function that takes as input a subset (also called a coalition) of players, and outputs the *value* or utility of this subset. Given this set value function, Shapley values then provide real-valued player-wise contributions. In a machine learning context, however, there is no well-defined set value function that takes as input a subset of features and outputs the value of this subset. Instead, we have the model  $f$ , data distribution  $p$ , and the test data point  $\mathbf{x}$ . A critical ingredient in order to derive Shapley values is to define a “value function”  $v_{\mathbf{x}, f, p}(S)$  which takes a set of features  $S$  as input, along with the model  $f$ , the test data point  $\mathbf{x}$ , and data distribution  $p$ . The value function describes how  $f(\mathbf{x})$  would change if only a set of features in  $\mathbf{x}$  participates in a cooperative setting.

However, there is no consensus on how to specify the value function. Existing value functions can be roughly classified into two classes: so-called *off-manifold* value functions measure the utility of coalitions of features by perturbing features outside the coalition (and hence taking the inputs outside the data manifold), while so-called *on-manifold* value functions measure the utility of coalitions of features by marginalizing out features outside the coalition while respecting the data distribution/manifold. Both approaches come with their respective caveats [Kumar et al., 2020]. Off-manifold value functions by construc-

tion evaluate the model on data that is not from the training data distribution; and hence are sensitive to how the model handles off-manifold test data [Hooker and Menth, 2019, Slack et al., 2020], and where machine learning models are typically unstable [Frye et al., 2020]. Moreover, the resulting explanations are very susceptible to manipulation: one can manipulate the model function only on off-manifold regions while preserving its behavior on-manifold, just to get the desired explanation (e.g. to show that the model is unbiased, while maintaining biased behavior on-manifold) [Slack et al., 2020]. In other words, the off-manifold value functions respect the model function, but not the data distribution. This is problematic since these value functions evaluate the model outside the data manifold, where most ML training methods do not come with any guarantees on model performance.

On-manifold value functions on the other hand are in general computationally expensive, particularly if they involve conditional expectations due to marginalizing over out-of-coalition features while conditioning on the coalition features. Approximating these conditional expectations via empirical distributions could result in idiosyncratic issues, such as all features having the same attribution [Sundararajan and Najmi, 2019]. Lundberg and Lee [2017], Aas et al. [2019], Štrumbelj and Kononenko [2014] compute the conditional value function via distributional assumptions, such as independence and Gaussian mixtures, which may not be flexible enough for complex data. Lundberg et al. [2018] attempt to calculate the conditional expectation value function for tree ensembles, but which Sundararajan and Najmi [2019] criticize as having unclear assumptions on the features. Frye et al. [2020] propose approximating these conditional expectations via a supervised and an unsupervised approach. As we show however, the supervised approach converges to the empirical conditional value function, while the unsupervised approach generates samples whose coalition features need not be set to the conditioned values. Another principled approach of calculating the conditional expectation based value function is by using importance sampling on the joint distribution, but this is computationally expensive. Further, we show that conditional expectation based value functions, even when calculated accurately, will also be prone to adversarial manipulation off the data manifold. Another caveat with on-manifold value functions is that they result in Shapley values that violates natural axioms [Sundararajan and Najmi, 2019]. Moreover, they might respect the data distribution at the price of fidelity to the model function: a feature with no intervention effect can still receive non-zero feature importance, which was seen as a positive aspect by Adler

et al. [2018], but a negative aspect by Merrick and Taly [2020], Sundararajan and Najmi [2019].

So both on-manifold and off-manifold value functions, and their resulting Shapley value explanations, have their competing pros and cons, and Chen et al. [2020] among others have suggested to “pick one’s poison” when faced with a specific application. Kumar et al. [2020] have even suggested that these two competing approaches present a serious concern with using Shapley values itself. But what if we can design value functions that respect both the model and the data manifold, and are not susceptible to adversarial manipulations in low density data regions? Surprisingly, we show that we indeed can. Towards this, we first propose a set of axioms that aim to formalize the desiderata in the question above, and show that there exists a unique value function that satisfies these axioms, which we call the Joint Baseline value function, satisfies these axioms. While the Shapley value by Joint Baseline value function does not satisfy the original set of axioms proposed in Sundararajan and Najmi [2019], it does satisfy a new set of axioms which takes both data density and function model into account. Moreover, we show that this unique value function is robust to low density adversarial manipulations. We also show that the resulting Shapley value, Joint Baseline Shapley (JBshap), can be computed efficiently and scaled to high dimensional data such as images.

## 2 Related Work

Our work focuses on feature-based explanations, where we assign real-valued attributions to features with respect to their contributions towards the prediction of the model at a given test input. Feature-based explanations include perturbation-based attributions [Zeiler and Fergus, 2014, Ribeiro et al., 2016], and gradient-based explanations [Simonyan et al., 2013, Selvaraju et al., 2017, Shrikumar et al., 2017, Sundararajan et al., 2017]. Shapley value belongs to perturbation-based attributions [Ancona et al., 2018, Yeh et al., 2019]. Other forms of explanations include sample-based explanation methods which explains the decision of the model via attributions to training samples [Koh and Liang, 2017, Yeh et al., 2018], and concept-based explanation methods which assign attributions to high-level human concepts [Kim et al., 2018, Ghorbani et al., 2019, Bau et al., 2017].

Our work specifies a new set of axioms on the set value functions to be used with the Shapley value, which is novel to the best of our knowledge. To position our development, we briefly discuss the axiomatic treatment of Shapley values and extensions, as an approach for real-valued player-wise attributions. Shapley values originated in a cooperative game context,

where we are given a set function that specifies the utility of any subset of players. Here, Shapley values were shown to uniquely satisfy a set of natural axioms [Shapley, 1988]; and were discussed in an explainable AI context by [Lundberg and Lee, 2017]. In a more general so-called cost-sharing game context, we are given a cost-sharing function that goes from (non-negative) real-valued demands associated with players to the overall cost, and a number of extensions of Shapley values such as Aumann-Shapley and Shapley-Shubik have been developed to uniquely satisfy their respective sets of axioms [Friedman and Moulin, 1999, Aumann and Shapley, 2015]; and were discussed in an explainable AI context [Sundararajan and Najmi, 2019]. Our axiomatic developments on the other hand are complementary to these classical developments: we focus specifically on *how to reduce the explainable AI problem to a cooperative game setting*, upon which we could then use Shapley values that were actually developed in the context of such games.

### 3 Different Value Functions for Shapley Value Explanations

#### 3.1 Problem Definition

Given a machine learning model  $f(\cdot) : \mathbb{R}^d \rightarrow \mathbb{R}$  trained on  $d$ -dimensional data, data distribution  $p$ , and an *explicand* (i.e. test input)  $\mathbf{x} \in \mathbb{R}^d$ , we aim to attribute the model output  $f(\mathbf{x})$  to the individual features. We use  $\phi_{f,p,i}(\mathbf{x})$  to denote this *attribution* to the  $i$ -th feature. In certain cases, there is also a given baseline value  $\mathbf{x}' \in \mathbb{R}^d$ , which is often set to 0 or the data mean. When applying Shapley values (which originate in cooperative game theory) to this attribution problem, a critical first step is to capture the key elements of  $f, p, \mathbf{x}$  into a set value function  $v_{\mathbf{x}, f, p}(S) : 2^d \rightarrow \mathbb{R}$ , where  $S \subseteq [d]$  is a subset of features. In this context, subsets of features can be thought of as subsets of players; in the sequel, we will often refer to the  $i$ -th feature as the  $i$ -th *player* to emphasize the connection to game theory at play with Shapley explanations. Given such a set value function  $v(\cdot)$ , Shapley value attributions to the  $i$ -th player can be computed as  $\phi_i(v) = \sum_{S \subseteq [d] / i} \frac{s!(n-s-1)!}{n!} [v(S \cup i) - v(S)]$ . In other words, the Shapley value of player  $i$  is the weighted average of its marginal contribution  $v(S \cup i) - v(S)$  averaged over all possible subsets  $S \subseteq [d]$  that do not include player  $i$ . The Shapley value has been shown to be the unique solution satisfying a set of reasonable axioms; see [Shapley, 1988] for a further discussion and history. We will provide a more detailed discussion of such axioms in Sec. 5 and Appendix C.

In the explainable AI context, using the set value function  $v_{\mathbf{x}, f, p}(\cdot)$ , we can then compute the Shapley importance values  $\phi_{f,p,i}(\mathbf{x}) = \phi_i(v_{\mathbf{x}, f, p}(\cdot))$ , where  $\phi_i$  is the

Shapley value attribution to the  $i$ -th player as detailed above. The critical question however is: how to specify the set value function  $v_{\mathbf{x}, f, p}(\cdot)$ ?

#### 3.2 Notation for Existing On-Manifold and Off-Manifold Value Functions

Different choices of the set value function  $v$  would lead to different Shapley values  $\phi_i(v)$ , and indeed the proper selection of  $v$  has been of great interest within the explainable AI community. We first set up some notation. We let  $x_S$  denote the sub-vector of  $\mathbf{x}$  corresponding to the feature set  $S \subseteq [d]$ , and  $x_{\bar{S}}$  as the sub-vector corresponding to the complementary subset  $\bar{S} \subseteq [d]$ . Given a *baseline*  $\mathbf{x}' \in \mathbb{R}^d$ , we will be using the notation  $f(x_S; x'_{\bar{S}})$  to refer to the function  $f$  with input sub-vectors  $x_S$  and  $x'_{\bar{S}}$  corresponding to features in  $S$  and  $\bar{S}$  respectively.

##### 3.2.1 On-Manifold Value Functions

In this paper, we consider the prototypical on-manifold value function of Conditional Expectation, which in turn results in the Conditional Expectation Shapley value (*CES*) [Lundberg and Lee, 2017, Sundararajan and Najmi, 2019]. We will be overloading notation and term the *value function* itself as CES.

**CES:**  $v_{\mathbf{x}, f, p}^C(S) = \mathbb{E}_{\mathbf{x}'_{\bar{S}} \sim p(\mathbf{x}'_{\bar{S}} | x_S)}[f(x_S; x'_{\bar{S}})] = \int_{\mathbf{x}'_{\bar{S}}} f(x_S; \mathbf{x}'_{\bar{S}}) p(\mathbf{x}'_{\bar{S}} | x_S) d\mathbf{x}'_{\bar{S}}$ . In other words, CES takes the expectation of  $f(x_S; \mathbf{x}'_{\bar{S}})$  over the conditional density of  $\mathbf{x}'_{\bar{S}}$  given  $x_S$  (with respect to the joint density  $p(\cdot)$ ).

##### 3.2.2 Off-Manifold Value Functions

As discussed earlier, off-manifold value functions quantify the contribution of a coalition of features by perturbing or intervening on the off-coalition features. In this paper, we consider the prototypical off-manifold value functions of Baseline Shapley (*Bshap*) [Sun and Sundararajan, 2011, Lundberg and Lee, 2017]) and its randomized variant.

**Bshap:**  $v_{\mathbf{x}, f, p}^B(S) := f(x_S; \mathbf{x}'_{\bar{S}})$ . Bshap directly takes the value of  $f$  corresponding to  $\mathbf{x}$ , and ignores the data density model  $p$ .

**RBshap:**  $v_{\mathbf{x}, f, p}^{RB}(S) = \mathbb{E}_{\mathbf{x}'_{\bar{S}}}[f(x_S; \mathbf{x}'_{\bar{S}})] = \int_{\mathbf{x}'_{\bar{S}}} f(x_S; \mathbf{x}'_{\bar{S}}) p_b(\mathbf{x}'_{\bar{S}}) d\mathbf{x}'_{\bar{S}}$ , where  $p_b$  is the marginal density of  $\mathbf{x}'_{\bar{S}}$ . RBshap is a natural extension of Bshap, where the baseline  $\mathbf{x}'$  in Bshap is generalized to a distribution of baselines, which is represented by  $p_b(\mathbf{x}')$ .

## 4 Issues of Existing Value Functions

### 4.1 Off-Manifold Value function: Not “Respecting” the Data Manifold

The main difference between on-manifold value functions and off-manifold value functions is that off-manifold value functions do not take the data manifold (density) into account. The machine learning model  $f$  may not be trained to have meaningful prediction values when the input is  $[\mathbf{x}_S; \mathbf{x}'_{\bar{S}}]$ . For instance, for image inputs,  $[\mathbf{x}_S; \mathbf{x}'_{\bar{S}}]$  would represent an image where certain random pixels are replaced by some baseline value (such as setting to black). [Frye et al., 2020] thus raise the concern that Shapley values calculated by off-manifold value functions could be over-reliant on function values in off-manifold regions, which is not where the model is trained on. Issues could arise when the model behavior is different on the data manifold and off the data manifold. For instance, if the image classifier tends to predict an image with many random black pixels as “dog”, the Shapley explanation for an image of a dog may be strongly affected by how the model handles such artifacts. This problem is emphasized by Slack et al. [2020] as they show that they can train a new model that only differs to the original model in off-manifold regions, and they are able to manipulate the Shapley explanations for off-manifold value functions. We further discuss this issue in Sec. 4.3.

### 4.2 On-Manifold Value function: Difficulty to Calculate Conditional Value

In contrast to the robustness challenges of the previous section, the main difficulty with conditional expectation (CES) based value functions is computational. Many variants have been proposed to simplify the computational burden of calculating the CES value, however, they may come with the cost of its new issues. We discuss some key variants and their respective issues.

#### 4.2.1 CES-Empirical

One variant is called CES-Empirical, which uses the empirical probability to calculate the conditioned value. To formally define CES-Empirical, let  $p^E(\mathbf{x}) = \frac{1}{m} \mathbb{1}[\mathbf{x} \text{ in dataset}]$  denote the empirical distribution, given  $m$  points in the dataset. In this case,  $v_{\mathbf{x}, f, p^E}^{CE}(S) = \mathbb{E}_{\mathbf{x}'_{\bar{S}} \sim p^E(\mathbf{x}'_{\bar{S}} | \mathbf{x}_S)}[f(\mathbf{x}_S; \mathbf{x}'_{\bar{S}})] = \sum_{i=1}^m \frac{1}{m} \mathbb{1}[\mathbf{x}_S^i = \mathbf{x}_S] f(\mathbf{x}_S^i)$  will be the average prediction of the points in the dataset where its feature set  $S$  is equal to  $\mathbf{x}_S$ .

Although CES-Empirical can be easy to calculate, the

caveats of CES-Empirical have been discussed in a line of recent work; we summarize the key ones below. The main criticism of CES-Empirical is that when an expli-cand has feature values that are unique in the dataset (which is likely to be the case with a continuous data distribution), each feature will get the same Shapley value importance even if the model is not symmetric in all features [Sundararajan and Najmi, 2019]. We showcase the issues of CES-Empirical with a concrete example in Appendix A.1.

#### 4.2.2 CES-Supervised

Another popular CES version is called CES-Supervised [Frye et al., 2020]. The method is motivated by the observation that the conditional expectation  $v_{\mathbf{x}, f, p}(S) = \mathbb{E}_{\mathbf{x}'_{\bar{S}} \sim p(\mathbf{x}'_{\bar{S}} | \mathbf{x}_S)}[f(\mathbf{x}_S; \mathbf{x}'_{\bar{S}})]$  can be seen to minimize the MSE loss  $\text{mse}(v(\mathbf{x}_S)) := \mathbb{E}_{\mathbf{x}'_{\bar{S}} \sim p(\mathbf{x}'_{\bar{S}} | \mathbf{x}_S)}[f(\mathbf{x}_S; \mathbf{x}'_{\bar{S}}) - v(\mathbf{x}_S)]^2$ . Thus, Frye et al. [2020] propose to learn the value function by a surrogate model  $g$  by minimizing the MSE loss:

$v_{\mathbf{x}, f, p^E}^{CS}(S) = \arg \min_g \mathbb{E}_{\mathbf{x} \sim p^E} \mathbb{E}_{S \sim \text{shapley}}[(f(\mathbf{x}) - g(\mathbf{x}_S))^2]$ , and the conditioned value  $v_{\mathbf{x}, f, p^E}^{CS}(S)$  can be obtained by the surrogate model  $g$  by  $g(\mathbf{x}_S)$ . Frye et al. [2020] propose to use a neural network with masked inputs as the surrogate model.

In order to calculate CES-supervised, Frye et al. [2020] follow the standard approach of empirical risk minimization, and optimize the empirical MSE loss:

$$\mathbb{E}_{\mathbf{x} \sim p^E(\mathbf{x})} \mathbb{E}_{S \sim \text{shapley}}[(f(\mathbf{x}) - v_{\mathbf{x}_S, f, p}^{CS})^2]. \quad (1)$$

One issue of CES-Supervised is that it requires retraining the model completely, where the sample space is all possible subsets for every input. The training of the model  $g$  is even more difficult than the training of the original model  $f$ .

Another issue is that the resulting optimal ERM CES-supervised estimator is exactly the CES-Empirical estimate, as we show in the following theorem.

**Proposition 1.** *The global minimizer to (1), where  $p^E(\mathbf{x}) = \frac{1}{m} \mathbb{1}[\mathbf{x} \text{ in dataset}]$  is exactly equal to  $f_{\mathbf{x}, f, p^E}^{CE}(S)$ . Thus,  $v_{\mathbf{x}_S, f, p}^{CS} = f_{\mathbf{x}, f, p^E}^{CE}(S)$  when the empirical distribution is used.*

The proof is in Appendix, which follows from a simple analysis of the stationary conditions of the objectives above. As a consequence of this surprising theorem, all pitfalls of CES-Empirical still carry over to CES-supervised when the empirical distribution is used. Even although Frye et al. [2020] use a masked encoder model to learn  $v_{\mathbf{x}_S, f, p}^{CS}$ , when the encoder is sufficiently flexible (e.g. large neural models) this still converges to CES-Empirical, and not the true conditioned value. One evidence is that CES-supervised is

invariant to the behavior of  $f$  off the data points in the dataset. However, the true conditioned expectation should be dependent on the behavior of  $f$  off the data manifold. We note that when the learning of the surrogate model has strong regularization, CES-SUP may not necessarily converge to CES-Empirical, but the fact that CES-SUP is invariant to the behavior of  $f$  off the data points validates that it does not capture the real conditioned value. Another issue of CES-Supervised, which is demonstrated in the experiment section, is that the empirical MSE loss can still be low even if the training objective is altered, and thus leaving CES-supervised to be prone to model-based attack.

#### 4.2.3 CES-Sample

Suppose we have access to the conditioned probability  $p(\mathbf{x}'_{\bar{S}}|\mathbf{x}_S)$ , we can use Monte-Carlo sampling to estimate  $v_{\mathbf{x},f,p}^{CESam}(S) = \int_{\mathbf{x}'_{\bar{S}}} f(\mathbf{x}_S; \mathbf{x}'_{\bar{S}}) p(\mathbf{x}'_{\bar{S}}|\mathbf{x}_S) d\mathbf{x}'_{\bar{S}}$ . To obtain the conditioned probability, Frye et al. [2020] proposed a Masked variational auto-encoder approach, which results in samples that does not respect the conditioned value, as we further discuss in the appendix. An alternative is to sample conditioned probability that satisfies the conditioned constraint is to use importance-sampling, by first sampling uniformly from all  $\mathbf{x}'_{\bar{S}}$ , and average the value of  $f(\mathbf{x}_S; \mathbf{x}'_{\bar{S}}) p(\mathbf{x}'_{\bar{S}}|\mathbf{x}_S)$  for each sampled point. The main downside of CES-Sample is that it is computationally expensive to calculate  $v_{\mathbf{x},f,p}^{CESam}(S)$ , each of which may take around 10 to 100 accesses to the model  $f$ ; in contrast, the computation of most other value functions require single access to the model  $f$ . Thus, CES-sample may be 10 $\times$  to 100 $\times$  more expensive to calculate compared to other value functions.

### 4.3 Off-Manifold and On-Manifold Value function: Sensitivity to Perturbation In Low-Density Regions

For off-manifold value functions, it is shown that one can manipulate the Shapley value explanations by perturbing the functions on off-manifold regions [Slack et al., 2020]. The methodology of Slack et al. [2020] is that given an original black-box model  $f$  which has explanation  $\phi(f)$ , one may perturb  $f$  to  $f'$  by only changing the behavior in low-density regions. In practice, we usually only check the test accuracy and test prediction, and the difference of  $f$  and  $f'$  may not be detected in several cases. However, the explanation  $\phi(f)$  and  $\phi(f')$  may differ drastically. This undermines the original explanation  $\phi(f)$ , as there are now ground truth for how the model  $f$  should behave off-manifold. While previous works [Frye et al., 2020] claim that on-manifold value function is robust to such manipulation, we show that one can also manipulate (the Shapley values corresponding to) on-manifold value func-

tions by perturbing the functions on low density regions (while fixing the model behavior on high density regions).

We first formalize a definition of robustness to off-manifold manipulations, and discuss the sensitivity (robustness) of existing value functions to off-manifold manipulations.

**Definition 1.** *Given any two models  $f_1(\mathbf{x})$ ,  $f_2(\mathbf{x})$  and any probability measure  $p(\mathbf{x})$ , if  $\max_{\mathbf{x}} |f_1(\mathbf{x}) - f_2(\mathbf{x})|p(\mathbf{x}) \leq \epsilon$  always entails  $|v_{\mathbf{x},f_1,p}(S) - v_{\mathbf{x},f_2,p}(S)| \leq T\epsilon$  for any  $S$ , we term the value function  $v_{\mathbf{x},f,p}(S)$  as strong  $T$ -robust to off-manifold perturbations.*

The premise  $\max_{\mathbf{x}} |f_1(\mathbf{x}) - f_2(\mathbf{x})|p(\mathbf{x}) \leq \epsilon$  bounds the maximum perturbation on low density regions. For instance, if we require that  $\max_{\mathbf{x}} |f_1(\mathbf{x}) - f_2(\mathbf{x})|p(\mathbf{x}) \leq 0.01$ , then in regions with density less than 0.01, we can perturb the function values by at most 1.0. However, we show that both Bshap and CES are not strong  $T$ -robust.

**Proposition 2.** *Bshap and CES are not strong  $T$ -robust to off-manifold perturbations for  $|T| < \infty$ .*

The key reason that Bshap and CES does not satisfy strong  $T$ -robustness is that the value function  $v_{\mathbf{x},f,p}(S)$  for Bshap and CES can be determined by the behavior of  $f$  on low density regions. To address this issue, we show that the strong-  $T$ -robustness can be obtained by taking the data density into account in the calculation of value functions.

We empirically evaluate the sensitivity of Bshap and CES when the function is perturbed in low-density regions in practice in Sec. 6.

## 5 Axioms for Value Functions & A New Value Function

In this section, we argue that it is possible to get the best of, and avoiding the pitfalls of, both worlds — of on-manifold and off-manifold value functions — by taking both the function and the data distribution into account.

We first propose a set of axioms on set value functions that takes these desiderata into account.

- 1. Linearity (over functions and distributions) (Lin.):** For any functions  $f, f_1, f_2, p, p_1, p_2$ ,  $\alpha_1 v_{\mathbf{x},f_1,p}(S) + \alpha_2 v_{\mathbf{x},f_2,p}(S) = v_{\mathbf{x},\alpha_1 f_1 + \alpha_2 f_2,p}(S)$ , for  $\alpha_1, \alpha_2 \in \mathbb{R}$  and  $\alpha_1 v_{\mathbf{x},f,\alpha_1 p_1}(S) + \alpha_2 v_{\mathbf{x},f,p_1}(S) = v_{\mathbf{x},f,\alpha_1 p_1 + \alpha_2 p_2}(S)$  for  $\alpha_1, \alpha_2 \geq 0$ .
- 2. Symmetry (over functions and distributions) (Sym.):** for all  $i, j \notin S$ ,  $f, p, \mathbf{x}, \mathbf{x}'$  all being symmetric in the  $i$ -th and  $j$ -th dimensions implies  $v_{\mathbf{x},f,p}(i \cup S) = v_{\mathbf{x},f,p}(j \cup S)$ . That is, if both

the function  $f$  and the probability  $p$  are symmetric with respect to two features, then so should the value function  $v$ .

3. **Dummy Player (over functions and distributions) (Dum.)**:  $f, p$  being invariant in the  $i$ -th dimension implies  $v_{\mathbf{x}, f, p}(S) = v_{\mathbf{x}, f, p}(i \cup S)$ . That is, a completely irrelevant feature in both the function value and data density should also be irrelevant in the value function.
4. **Null Player (over functions and distributions) (Null)**: if  $f', p'$  satisfies  $f(\mathbf{x}') = f'(\mathbf{x}'), p(\mathbf{x}') = p'(\mathbf{x}')$ , then  $v_{f, p, \mathbf{x}}(\emptyset) = v_{f', p', \mathbf{x}}(\emptyset)$ . That is, the value function when no players are in the set function solely depends on the function value and data density of the baseline.
5. **Efficiency (over functions and distributions) (Eff.)**:  $v_{f, p, \mathbf{x}}[d] - v_{f, p, \mathbf{x}}[\emptyset] = f(x)p(\mathbf{x}) - f(x')p(\mathbf{x}')$ . The difference between the value function of all players and no players is equal to the total joint density. Note that this axiom is not needed in the uniqueness of JBshap.
6. **Set Relevance (over functions and distributions) (Set.)**: If  $f_1(\mathbf{x}_S, \bar{\mathbf{x}}_{\bar{S}}) = f_2(\mathbf{x}_S, \bar{\mathbf{x}}_{\bar{S}})$  and  $p_1(\mathbf{x}_S, \bar{\mathbf{x}}_{\bar{S}}) = p_2(\mathbf{x}_S, \bar{\mathbf{x}}_{\bar{S}})$  for any  $\bar{x}$ ,  $v_{f_1, p_1, \mathbf{x}}(S) = v_{f_2, p_2, \mathbf{x}}(S)$ . The set relevance axiom states that the value function is determined by how  $f, p$  behaves when the feature set  $S$  is fixed to be equal to  $\mathbf{x}_S$ .
7. **Strong-T-Robustness (over functions and distributions) (Rob.)**: For any functions  $f_1, f_2, p$ , if  $\max_{\mathbf{x}} |f_1(\mathbf{x}) - f_2(\mathbf{x})|p(\mathbf{x}) \leq \epsilon$ , then  $|v_{\mathbf{x}, f_1, p}(S) - v_{\mathbf{x}, f_2, p}(S)| \leq T\epsilon$  for some constant  $T$ . This axiom has been discussed in Sec. 4.3.

As discussed earlier, Shapley values originated in a cooperative game context, as the unique *attribution function* that assigns attributions to individual players, given a set value function that specifies utilities of sets/coalitions of players. Our set of axioms above on the other hand are for the set value function itself: on what properties such set-value functions should satisfy. Our linearity axiom states that for a linear combination of models should have a corresponding linear combination of set-value functions. Similarly a convex combination (i.e. mixture) of distributions should have a corresponding convex combination of value functions.

Note that Shapley values themselves satisfy a linearity axiom, that states that a linear combination of set-value functions should have a linear combination of corresponding Shapley values. By combining the linearity axioms for set-value functions we specify here,

with the linearity axiom of Shapley value feature attributions, it can be seen that we get that a combined linearity axiom: that for linear (convex) combinations  $f(p)$ , the Shapley attributions should in turn be linear (convex) combinations of the corresponding Shapley values. We formalize the complete set of transitive axioms in Appendix Sec. C.

The set relevance axiom states that the value function is determined by the behavior of  $f$  and  $p$  when the feature set of  $S$  is set to the value of  $\mathbf{x}$ . The rationale is that the value function identifies the contribution of  $\mathbf{x}_S$  in  $f(\mathbf{x})$  and  $p(\mathbf{x})$ , and thus should be only relevant to  $f(\bar{\mathbf{x}})$  and  $p(\bar{\mathbf{x}})$  when  $\bar{\mathbf{x}}_S = \mathbf{x}_S$ . In other words, how  $f(\bar{\mathbf{x}}), p(\bar{\mathbf{x}})$  behaves when  $\bar{\mathbf{x}}_S$  is not equal to  $\mathbf{x}_S$  should not affect the value function  $v_{f, p, \mathbf{x}}(S)$ . Common value functions also satisfy set relevance axioms, for instance, CES satisfies the (Set.) axiom, and Bshap satisfies the (Set.) axiom with only  $f$  (see (Set-IS.) in Sec. C).

We next show the following uniqueness result:

**Theorem 1.**  $v_{\mathbf{x}, f, p}(S)$  satisfies (Lin.), (Sym.), (Dum.), (Null), (Eff.), (Set.), (Rob.) axioms if and only  $v_{\mathbf{x}, f, p}(S) = f(\mathbf{x}_S, \mathbf{x}'_{\bar{S}})p(\mathbf{x}_S, \mathbf{x}'_{\bar{S}})$ .

This result specifies the unique value function that satisfies natural axioms which take both the data and model into account. We term this unique value function the *Joint Baseline Shapley value* (JBshap):

**Definition 2.** (JBshap):

$$v_{\mathbf{x}, f, p}^J(S) = f(\mathbf{x}_S; \mathbf{x}'_{\bar{S}})p(\mathbf{x}_S; \mathbf{x}'_{\bar{S}}) \quad (2)$$

An interesting fact is that for the uniqueness result, the efficiency axiom is not needed, which can be seen in the proof in Sec. E. Suppose the model  $f(\mathbf{x}_S; \mathbf{x}'_{\bar{S}})$  converges to the conditional distribution  $p(y|\mathbf{x}_S, \mathbf{x}'_{\bar{S}})$ , then from (2), it can be seen that  $v_{\mathbf{x}, f, p}^J(S) \rightarrow p(y|\mathbf{x}_S, \mathbf{x}'_{\bar{S}})p(\mathbf{x}_S; \mathbf{x}'_{\bar{S}}) = p(y, \mathbf{x}_S, \mathbf{x}'_{\bar{S}})$ . In other words, as the model itself converges to the Bayes optimal classifier, JBshap assigns a value to  $\mathbf{x}_S$  that is consistent with the joint density of the observation  $y$  and  $(\mathbf{x}_S, \mathbf{x}'_{\bar{S}})$ . We highlight that JBshap resembles on-manifold value function since it respects the data manifold, but can also be considered as off-manifold since it takes an interventional approach similar to off-manifold value functions.

A natural extension of JBshap is to average over all possible baseline values  $\mathbf{x}'$ , and we obtain RJBshap:

**Definition 3.** (RJBshap):

$$v_{\mathbf{x}, f, p}^{RJ}(S) = \int_{\mathbf{x}'_{\bar{S}}} [f(\mathbf{x}_S; \mathbf{x}'_{\bar{S}})p(\mathbf{x}_S; \mathbf{x}'_{\bar{S}})p_b(\mathbf{x}')] d\mathbf{x}'_{\bar{S}},$$

where  $p_b(\cdot)$  is the prior probability for a set of baseline values. We point out that RJBshap still satisfies the

axioms (Lin.), (Sym.), (Dum.), (Null), (Eff.), (Rob), with the slight modification of (Sym.), (Null), (Eff.) by replacing the baseline  $\mathbf{x}'$  in JBshap by the random baseline  $\mathbf{x}'$  with distribution  $p_b(\mathbf{x}')$

It is easy to observe that JBshap is equal to Bshap when the function  $f(\mathbf{x})$  is replaced by  $f(\mathbf{x})p(\mathbf{x})$  since  $v_{\mathbf{x},f,p}^J(S) = v_{\mathbf{x},f,p,p}^B(S)$ . The clear difference is that JBshap takes account of the data density  $p(\cdot)$  and thus will not be effected by off-manifold regions  $\mathbf{x}$  since  $p(\mathbf{x})$  will be small. We can also relate RJBshap and CES, by noting that CES value function can be written as  $v_{\mathbf{x},f,p}^C(S) = \mathbb{E}_{\mathbf{x}'_S \sim p(\mathbf{x}'_S | \mathbf{x}_S)}[f(\mathbf{x}_S; \mathbf{x}'_S)] = \int_{\mathbf{x}'_S} f(\mathbf{x}_S; \mathbf{x}'_S)p(\mathbf{x}'_S; \mathbf{x}_S)/p(\mathbf{x}_S)d\mathbf{x}'_S$ . When the baseline follows an uniform solution for all possible values (such that  $p_b$  is a constant), CES corresponds to RJBshap divided by  $p(\mathbf{x}_S)/C_0$ , where  $C_0$  is the constant value of  $p_b$ . However, we point out that the calculation of  $p(\mathbf{x}_S)$  requires marginalizing over  $|\bar{S}|$  variables to obtain  $p(\mathbf{x}_S)$ , which is computationally difficult. Thus, RJBshap is much more computational tractable compared to CES. By not dividing over  $p(\mathbf{x}_S)$ , both JBshap and RJBshap are strong T-robust to off-manifold manipulations, in contrast to CES.

### 5.1 Estimating $p(\mathbf{x})$

In our discussion of off and on-manifold value functions, we discussed that the key caveat of on-manifold value function of CES in particular is that it requires expectation (or sampling) with respect to the conditional distributions  $p(\mathbf{x}'_{\bar{S}} | \mathbf{x}_S)$ . But if JBshap also requires computing the joint distribution, is it more computational efficient compared to CES at all?

We emphasize that JBshap is much more tractable computationally, since instead of the difficult conditional expectations, we only need to compute the much more amenable *joint density*  $p(\mathbf{x}'_{\bar{S}}; \mathbf{x}_S)$ . Moreover, we only need estimates of the data density on the space  $\mathbf{x}'_{\bar{S}}; \mathbf{x}_S$  instead of the entire domain  $\mathbb{R}^d$ . Towards this, we could use noise-contrastive estimation [Gutmann and Hyvärinen, 2010] to estimate the data density on space in the form  $\mathbf{x}'_{\bar{S}}; \mathbf{x}_S$ . The idea of noise-contrastive estimation is to estimate the data density of an unknown distribution by comparing it to a self-specified noise distribution. In this case, the event space where we need to query the data density  $p(\mathbf{x}'_{\bar{S}}; \mathbf{x}_S)$  is the subspace  $[\mathbf{x}'_{\bar{S}}; \mathbf{x}_S]$  for all possible  $\mathbf{x}$  and  $S$ . We thus specify a self specified noise  $Q$  to have an constant probability for every possible  $[\mathbf{x}'_{\bar{S}}; \mathbf{x}_S]$ .

To estimate  $p(\mathbf{x}'_{\bar{S}}; \mathbf{x}_S)$ , we can then train an Out-of-Distribution (OOD) classifier model  $\text{OOD}(\mathbf{x})$  that outputs 1 when the input  $\mathbf{x} \in D$  comes from the true data and outputs 0 when the input  $\mathbf{x} \in Q$  is self-generated noise, and control the input to the training

of the OOD classifier to have balanced true and noise data. When the OOD classifier converges,  $\text{OOD}(\mathbf{x}) = p(\mathbf{x} \in D | \mathbf{x}) = \frac{p(\mathbf{x} | \mathbf{x} \in D)}{p(\mathbf{x} | \mathbf{x} \in D) + p(\mathbf{x} | \mathbf{x} \in Q)}$ . By simple algebra, we can then get  $p(\mathbf{x} | \mathbf{x} \in D) = p(\mathbf{x} | \mathbf{x} \in Q) \times \frac{\text{OOD}(\mathbf{x})}{1 - \text{OOD}(\mathbf{x})}$ . We emphasize that since we do not require sampling from  $p(\mathbf{x}'_{\bar{S}}; \mathbf{x}_S)$ , and the subspace of  $\mathbf{x}'_{\bar{S}}; \mathbf{x}_S$  is much more tractable compared to the real space  $\mathbb{R}^d$ , and the estimation of  $p(\mathbf{x}'_{\bar{S}}; \mathbf{x}_S)$  is a much easier problem than estimating  $p(\mathbf{x})$  for general  $\mathbf{x}$ . And estimating even the full joint  $p(\mathbf{x})$ , is much easier than computing the conditional expectations  $p(\mathbf{x}'_{\bar{S}} | \mathbf{x}_S)$  (to provide a sense of the difficulty of the latter: even for simpler probabilistic graphical models, it is always tractable to compute the unnormalized joint density, while intractable in general to compute conditional marginals.)

While the calculation of JBshap only requires learning  $p(\mathbf{x})$  by training an OOD classifier, the calculation of CES either requires learning  $p(\mathbf{x})$  and sampling from it for each query (CES-Sample), or retraining a new model that estimates the conditioned value (CES-Supervised). Thus, the computation for calculating JBshap is much more affordable compared to the calculation of CES value functions, as we verify in the experiments.

## 6 Experiments

### 6.1 Robustness to off-manifold manipulation.

Recall that in Sec. 4.3, we have shown that both Bshap and CES are not strong  $T$ -robust, and thus are prone to manipulations in low-density regions. To evaluate the robustness to off-manifold manipulations in practice, we perform an empirical study on the UCI Census Income data dataset [Dua and Graff, 2017] for value functions Bshap, RBshap, JBshap, RJBshap, CES-Supervised, CES-Sample. Given a pretrained biased model, we want to test whether we are able to hide the dependency of a biased feature in the Shapley value by fine-tuning the model only on off-manifold regions.

We trained a four-layer neural network to predict whether an individual's income exceeds \$50k based on demographic features in the data based on the UCI income Data. To create a biased model, we add 0.1 times the feature value of "sex" to the individual's income following the setting of Slack et al. [2020]. We then train a "fine-tuned network" such that the dependency of the feature "sex" is set to the negative off the data manifold (while it is positive on the data manifold), while making the same prediction as the original biased model on over 99% of the testing points. To validate that the bias is not hidden in the 1% testing point with different predictions, we find that the average difference (L1 Loss) in the prediction probability for data where the fine-tuned model and the original model dis-

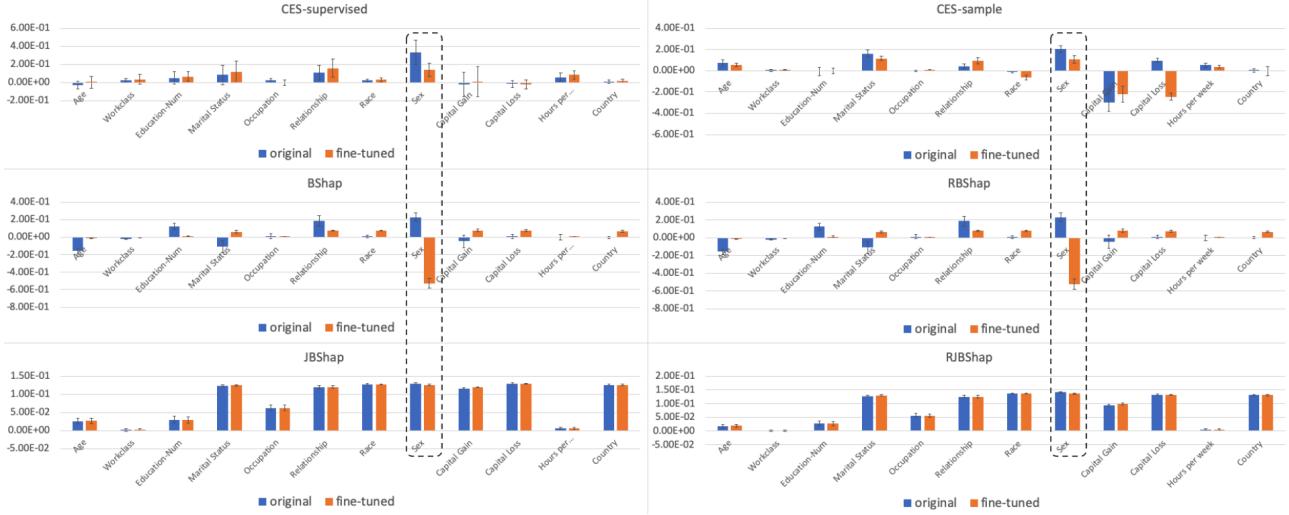


Figure 1: Global Shapley values for different value functions on the UCI Adult dataset; with two bars for each feature: on an original model (blue, left) and a fine-tuned model (orange, right). The importance for “sex” feature (which is boxed) of RBshap, Bshap, and CES is significantly reduced after the fine-tuning, while the importance for “sex” feature of JBshap is almost unchanged.

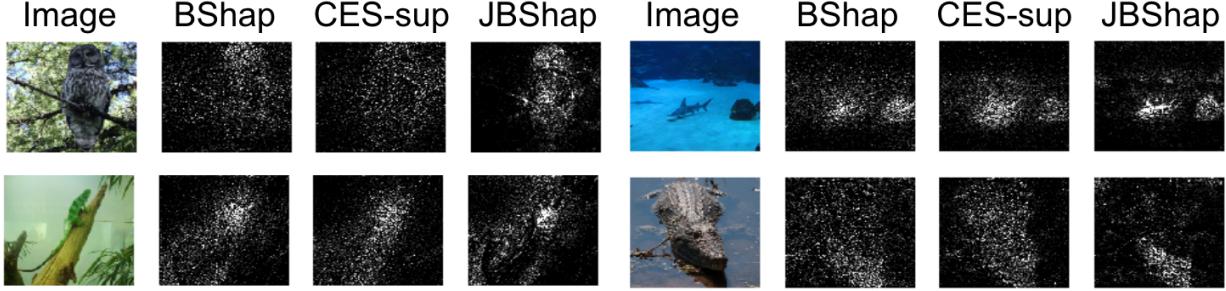


Figure 2: Visualization of Shapley values for JBShap, Bshap, CES-Supervised on Imagenet.

agrees (and agrees) is 0.07 (0.01), demonstrating that the model prediction for fine-tuned model and original model are close for all test data points.

For CES-Supervised, we train a “fine-tuned network” for the surrogate model  $g$ , with the constraint that the fine-tuned network behaves similar to the original surrogate model on the on-manifold points and has a similar empirical MSE loss as the original surrogate model (validation MSE loss for the fine-tuned model decreased from  $0.0278 \pm 0.001$  to  $0.0277 \pm 0.001$ , while training MSE Loss increased from  $0.0136 \pm 0.001$  to  $0.0139 \pm 0.001$ ). To determine off-manifold data, we train an OOD detector and use noise-contrastive estimation to obtain  $p(\mathbf{x})$  to calculate JBshap and CES-Sample. For numerical stability, we clip the output of the OOD classifier between the range  $[0.01, 0.99]$ . For CES-Sample, we sample 100 points per value function with  $p(\mathbf{x})$ . We then average the Shapley value based on different value functions on 100 testing points with feature of “sex” having value 1 to obtain the global Shapley value. We do 5 separate runs with different data splits (via random seeds), we report the global

Shapley value for different value functions in the top figure with error bars representing the standard deviation. We normalize the global Shapley value such that the sum of absolute value of Shapley value for all features sum up to one.

We present our results in Figure 1; each feature has two bars, each corresponding to the average normalized Shapley values, of the original model (left) and the fine-tuned model (right). We observe that the importance of the feature “sex” of the fine-tuned model is strongly reduced for existing value functions Bshap, RBshap and has a negative value for the fine-tuned model. For CES-Sample and CES-Supervised, the respective Shapley value of the feature “sex” is not perturbed as severely as Bshap and RBshap, but they are also reduced more than half compared to the original. On the other hand, the Shapley value for JBshap and RJBshap are only slightly reduced for the fine-tuned model. This verifies our theoretical analysis that JBshap is robust to off-manifold manipulations, in contrast to existing value functions. We also note that the Shapley value of JBshap for all features have stan-

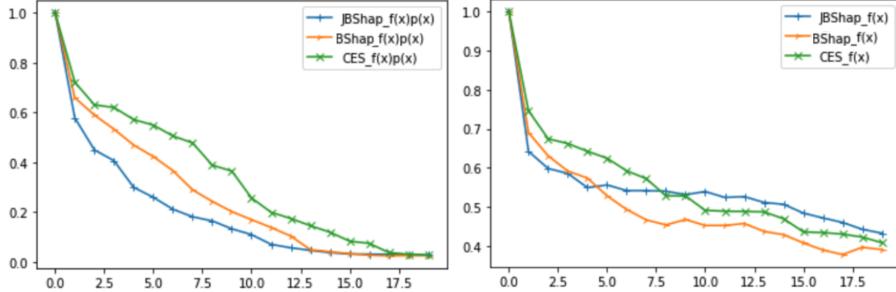


Figure 3: Deletion curve for JBShap, BShap, and CES-supervised on Imagenet for joint density (left) and model output (right).

dard deviation less than 0.01, demonstrating the robustness of JBshap over different OOD classifiers and data splits.

## 6.2 Visualization on High Dimensional Data.

In this section, we visualize the Shapley values for different value functions on image data. We perform experiments on Imagenet subset with the first 50 classes (the total number of classes is 1000). We first finetune a Resnet-18 model that reaches 80% top-1 accuracy as our base model. We then train a masked surrogate model with the same architecture to calculate CES-Supervised, and we train an OOD classifier to estimate  $p(\mathbf{x})$  by noise contrastive estimation to calculate JBshap. To calculate the Shapley value, we use the permutation sampling approach introduced in Castro et al. [2009] with 10 permutations per image. We compare the Shapley values from different value functions: Bshap, JBshap, CES-Supervised. Note that we omit other variants of value function since they require more than one function pass during the calculation of the value function, which will make the computation infeasible on high-dimension. The training of the surrogate model takes around 2976 seconds, while the training of the OOD classifier only take around 74 seconds. The cost of the training of the surrogate model will become even more costly if the model is trained on the full set of imangenet, not just a subset of imangenet. Thus, the additional training time for JBshap is much lower than the additional training time for CES-Supervised. The Shapley value computation for each image takes around 1800 seconds, which is equal for Bshap, JBshap, and CES-supervised. All computations are done on a single 1080-Ti GPU. The visualization results are shown in Figure 2, and we can observe that while the Shapley value for different value functions all focus on the object in the image, the Shapley value for JBshap concentrates on a smaller part of the image. For instance, in the bottom right image, Bshap and CES-supervised focuses on the whole alligator, while JBshap focuses more specifically on the head on the alligator. One possible reason is that Bshap and CES-supervised captures pixels that contributes to the

model prediction, while JBshap focuses on pixels that contributes to both the data density of the image and the model prediction.

**Quantitative evaluation on ImageNet** We perform an evaluation for the Shapley values based on different variants of value function, where we choose the well-known deletion evaluation criteria for our results in ImageNet over the average of 50 data points, where we remove the top pixels by Shapley value and evaluate how the function decreases (lower is better) [Samek et al., 2016, Arras et al., 2019] in Fig. 6.1, where we show the resulting joint density  $f(x)p(x)$  in the left and model value  $f(x)$  in the right after removing top pixels, where the x-axis is the percent of pixels. The area under curve (AUC) (where smaller is better) of  $f(x)p(x)$  for JBShap, Bshap, CES is 0.207, 0.27, 0.348 respectively, and the AUC for  $f(x)$  for JBShap, Bshap, CES is 0.549, 0.504, 0.556 respectively. Thus, we hypothesize that JBShap highlights features that affect both  $f(x)$  and  $p(x)$ , and Bshap highlights pixels that only affect  $f(x)$ . We also report the rank correlation version of sensitivity-n Anchona et al. [2018], which measures the correlation between the function drop and sum of Shapley values for random sets of pixels, where the random sets of pixels is set to 1 – 20 percent of pixels, and the average sensitivity-n rank correlation for JBShap, Bshap, CES, are  $0.182 \pm 0.007$ ,  $0.132 \pm 0.004$ ,  $0.119 \pm 0.01$  respectively.

## 7 Conclusion

In this work, we discussed the issues of interventional and conditional value functions, including different implementation variants of conditional value functions. To address these issues, we propose JBShap, which uniquely satisfies a set of axioms considering both the model and data manifold, while being robust to manipulation and computationally efficient.

**Acknowledgement** We acknowledge the support of ONR via N000141812861, and NSF via IIS-1909816.

## References

- Lloyd S. Shapley. *A value for n-person games*, page 31–40. 1988.
- Ulrike Grömping. Estimators of relative importance in linear regression based on variance decomposition. *The American Statistician*, 61(2):139–147, 2007.
- Richard Harold Lindeman. Introduction to bivariate and multivariate analysis. Technical report, 1980.
- Art B Owen. Sobol’indices and shapley value. *SIAM/ASA Journal on Uncertainty Quantification*, 2(1):245–251, 2014.
- Art B Owen and Clémentine Prieur. On shapley value for measuring importance of dependent inputs. *SIAM/ASA Journal on Uncertainty Quantification*, 5(1):986–1002, 2017.
- Anupam Datta, Shayak Sen, and Yair Zick. 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, 2016.
- Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *NIPS*, 2017.
- Amirata Ghorbani and James Zou. Data shapley: Equitable valuation of data for machine learning. In *International Conference on Machine Learning*, pages 2242–2251. PMLR, 2019.
- Ruoxi Jia, David Dao, Boxin Wang, Frances Ann Hubis, Nick Hynes, Nezihe Merve Gürel, Bo Li, Ce Zhang, Dawn Song, and Costas J Spanos. Towards efficient data valuation based on the shapley value. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 1167–1176. PMLR, 2019.
- Amirata Ghorbani and James Zou. Neuron shapley: Discovering the responsible neurons. *arXiv preprint arXiv:2002.09815*, 2020.
- Chih-Kuan Yeh, Been Kim, Sercan Arik, Chun-Liang Li, Tomas Pfister, and Pradeep Ravikumar. On completeness-aware concept-based explanations in deep neural networks. *Advances in Neural Information Processing Systems*, 33, 2020.
- I Elizabeth Kumar, Suresh Venkatasubramanian, Carlos Scheidegger, and Sorelle Friedler. Problems with shapley-value-based explanations as feature importance measures. *arXiv preprint arXiv:2002.11097*, 2020.
- Giles Hooker and Lucas Mentch. Please stop permuting features: An explanation and alternatives. *arXiv preprint arXiv:1905.03151*, 2019.
- Dylan Slack, Sophie Hilgard, Emily Jia, Sameer Singh, and Himabindu Lakkaraju. Fooling lime and shap: Adversarial attacks on post hoc explanation methods. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, pages 180–186, 2020.
- Christopher Frye, Damien de Mijolla, Laurence Cowton, Megan Stanley, and Ilya Feige. Shapley-based explainability on the data manifold. *arXiv preprint arXiv:2006.01272*, 2020.
- Mukund Sundararajan and Amir Najmi. The many shapley values for model explanation. *arXiv preprint arXiv:1908.08474*, 2019.
- Kjersti Aas, Martin Jullum, and Anders Løland. Explaining individual predictions when features are dependent: More accurate approximations to shapley values. *arXiv preprint arXiv:1903.10464*, 2019.
- Erik Štrumbelj and Igor Kononenko. Explaining prediction models and individual predictions with feature contributions. *Knowledge and information systems*, 41(3):647–665, 2014.
- Scott M Lundberg, Gabriel G Erion, and Su-In Lee. Consistent individualized feature attribution for tree ensembles. *arXiv preprint arXiv:1802.03888*, 2018.
- Philip Adler, Casey Falk, Sorelle A Friedler, Tionney Nix, Gabriel Rybeck, Carlos Scheidegger, Brandon Smith, and Suresh Venkatasubramanian. Auditing black-box models for indirect influence. *Knowledge and Information Systems*, 54(1):95–122, 2018.
- Luke Merrick and Ankur Taly. The explanation game: Explaining machine learning models using shapley values. In *International Cross-Domain Conference for Machine Learning and Knowledge Extraction*, pages 17–38. Springer, 2020.
- Hugh Chen, Joseph D Janizek, Scott Lundberg, and Su-In Lee. True to the model or true to the data? *arXiv preprint arXiv:2006.16234*, 2020.
- Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.
- Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Why should i trust you?: Explaining the predictions of any classifier. In *KDD*. ACM, 2016.
- Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.
- Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Parikh. Grad-cam: Visual explanations from deep networks via gradient-based localization. *International conference on computer vision*, 2017.

- Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. *International Conference on Machine Learning*, 2017.
- Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3319–3328. JMLR.org, 2017.
- Marco Ancona, Enea Ceolini, Cengiz Oztireli, and Markus Gross. A unified view of gradient-based attribution methods for deep neural networks. *International Conference on Learning Representations*, 2018.
- Chih-Kuan Yeh, Cheng-Yu Hsieh, Arun Sai Suggala, David I. Inouye, and Pradeep Ravikumar. On the (in)fidelity and sensitivity of explanations. In *NeurIPS*, 2019.
- Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *ICML*, 2017.
- Chih-Kuan Yeh, Joon Sik Kim, Ian En-Hsu Yen, and Pradeep Ravikumar. Representer point selection for explaining deep neural networks. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada.*, pages 9311–9321, 2018.
- Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas, et al. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). In *ICML*, 2018.
- Amirata Ghorbani, James Wexler, James Zou, and Been Kim. Towards automatic concept-based explanations. *NeurIPS*, 2019.
- David Bau, Bolei Zhou, Aditya Khosla, Aude Oliva, and Antonio Torralba. Network dissection: Quantifying interpretability of deep visual representations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6541–6549, 2017.
- Eric Friedman and Herve Moulin. Three methods to share joint costs or surplus. *Journal of economic Theory*, 87(2):275–312, 1999.
- Robert J Aumann and Lloyd S Shapley. *Values of non-atomic games*. Princeton University Press, 2015.
- Yi Sun and Mukund Sundararajan. Axiomatic attribution for multilinear functions. In *Proceedings of the 12th ACM conference on Electronic commerce*, pages 177–178, 2011.
- Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 297–304. JMLR Workshop and Conference Proceedings, 2010.
- Dheeru Dua and Casey Graff. UCI machine learning repository, 2017. URL <http://archive.ics.uci.edu/ml>.
- Javier Castro, Daniel Gómez, and Juan Tejada. Polynomial calculation of the shapley value based on sampling. *Computers & Operations Research*, 36(5):1726–1730, 2009.
- Wojciech Samek, Alexander Binder, Grégoire Montavon, Sebastian Lapuschkin, and Klaus-Robert Müller. Evaluating the visualization of what a deep neural network has learned. *IEEE transactions on neural networks and learning systems*, 28(11):2660–2673, 2016.
- Leila Arras, Ahmed Osman, Klaus-Robert Müller, and Wojciech Samek. Evaluating recurrent neural network explanations. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 113–126, 2019.

## A Issues for CES Versions

### A.1 Sparsity Issue For CES-Empirical and CES-Supervised

We discuss an issue for CES-Empirical and CES-Supervised in this section with a concrete example.

**Example 1.** Consider a model where  $f(\mathbf{x}) = 10\mathbf{x}_1 - 10\mathbf{x}_2$ , where there are three candidate features  $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3$ . The data distribution is generated from a multivariate Gaussian  $\mathcal{N}(0, I)$ . However, we do not know the true distribution of the data, and we can only model the distribution by a mixture Gaussian model where each data point is the center of a Gaussian ball. Suppose we are to explain the outcome of a data point  $(3, -3, 10)$ . Here,  $\mathbf{x}_3$  is clearly an outlier value, which is way larger than other data points. Therefore, the value function of  $S = \mathbf{x}_3$  for CES is equal to  $v_{\mathbf{x}, f, p}(\{\mathbf{x}_3\}) = \mathbb{E}_{\mathbf{x}'_S \sim p(\mathbf{x}'_S | \mathbf{x}_3=10)}[f(\mathbf{x}_3 = 10; \mathbf{x}'_S)]$  will be very close to 60. On the other hand, the value of feature 1 is not an outlier, the value function of  $S = \mathbf{x}_1$  for CES will be close to  $v_{\mathbf{x}, f, p}(\{\mathbf{x}_1\}) = \mathbb{E}_{\mathbf{x}'_S \sim p(\mathbf{x}'_S | \mathbf{x}_1=3)}[f(\mathbf{x}_1 = 3; \mathbf{x}'_S)] = 30$ . It is easy to see that for this data point, Shapley value for CES will assign a higher importance value to feature 3 than feature 1 due to sparsity of outlier values, even if feature 3 is not related to the model at all.

Since the true probability distribution is not known in this case, we use the smoothed empirical distribution  $P^{SE}(\mathbf{x}) = \sum_{i=1}^m \frac{1}{m} \mathcal{N}(\mathbf{x}_i, I)(\mathbf{x})$  to estimate the distribution for JBshap and CES-Sample. We consider CES-Emp, CES-Sample, CES-supervised, Bshap, RBshap, JBshap in this example. For Bshap and JBshap, we let the baseline be the average value of the dataset. For CES-supervised, we learn a KNN regressor with a Gaussian kernel as the predictor. We report the relative importance of feature 3 compared to the average importance of feature 1 and feature 2 for each Shapley value in Figure 4, where the ideal value should be  $\leq 0$ . The X-axis is the number of the data in the dataset, which effects CES-Sample, JBshap by the smoothed empirical distribution, effects CES-Empirical by the empirical distribution, and effects CES-Supervised by the training of the supervised value function.

We observe that CES-Empirical gives equal contribution to feature 3 compared to feature 1 and feature 2 due to the sparsity issue (see Remark 3.2 in Sundararajan and Najmi [2019]). For both CES-Sample and CES-Supervised, the relative importance of feature 3 is still higher than 1 with around 1000 data points. We further observe that the relative importance of feature 3 of CES-Sample decreases to near 0 when more than 10000 data points are in the dataset. However, even with 100,000 data points in the dataset, the relative importance of feature 3 is still larger than

1 for CES-supervised (but converging to 1). This is not surprising since in Proposition 1 we showed that CES-Supervised converges to CES-Empirical, which suffers from the sparsity issue. While CES-Sample may need more data points to obtain the correct Shapley value, CES-supervised do not converge to the correct value even with 100,000 data points due to the sparsity issue.

On the other hand, JBshap, Bshap, RBshap gives a low importance value for feature 3 even with only 10 data points. Thus, CES-Sample should be preferred over CES-Supervised and CES-Empirical when the data is continuous and the sparsity issue may become a problem.

### A.2 Issue for CES-Masked-VAE

In Sec. 4.2.3, we mentioned that one version of CES sample is proposed by Frye et al. [2020] to used a masked variational autoencoder, which we call CES-Masked-VAE. We briefly explain the proposed method and a major flaw of the method.

**CES-Masked-VAE** Suppose we have a density model that computes  $p(\mathbf{x}|\mathbf{x}_S)$  for all possible subsets  $S \subseteq [d]$ ; Frye et al. [2020] suggest using masked VAEs to do so. One can then sample  $\mathbf{x}'_{\bar{S}}$  from  $p(\mathbf{x}'_{\bar{S}}|\mathbf{x}_S)$ , and calculate  $\mathbb{E}_{\mathbf{x}'_{\bar{S}} \sim p(\mathbf{x}'_{\bar{S}}|\mathbf{x}_S)}[f(\mathbf{x}_S; \mathbf{x}'_{\bar{S}})]$  via Monte-Carlo sampling. We use masked VAEs as a density model in our experiments, following the suggestion of Frye et al. [2020], and hence term this approach CES-Masked-VAE.

**Unfaithfulness of Masked CES-Masked-VAE to coalition constraints.** The CES-Masked-VAE approach in Frye et al. [2020] trains a masked encoder  $r_\Psi(z|\mathbf{x}_S)$  that aims to agree with the full encoder  $q_\phi(z|\mathbf{x})$  by minimizing the KL-divergence between them. However, one issue is that this does not guarantee that the masked encoder  $r_\Psi(z|\mathbf{x}_S)$  will have support in latent spaces corresponding to  $\mathbf{x}'$  where  $\mathbf{x}'_S = \mathbf{x}_S$ . Thus, sampling from  $p(\mathbf{x}'|\mathbf{x}_S) = \int dz r_\Psi(z|\mathbf{x}_S) p_\theta(\mathbf{x}'|z)$  may obtain  $\mathbf{x}'$  which does not satisfy  $\mathbf{x}'_S = \mathbf{x}_S$ , which is unfaithful to the conditioned value. To verify this, we trained a CES-Masked-VAE on Cifar-10 with methods specified in Frye et al. [2020], and sample  $\mathbf{x}'$  from the Masked VAE estimate of  $p(\mathbf{x}'|\mathbf{x}_S)$ . We show some visualization results in Fig. 5. We can observe that each image, the sampled images disagree with the masked image on parts that are not masked. For instance, the color of the truck head in image 1, the color of the bird in image 2, the right ear of the dog in image 3, the caption in image 4, and the color of the background in image 5. This verifies that the masked VAE approach does not generate

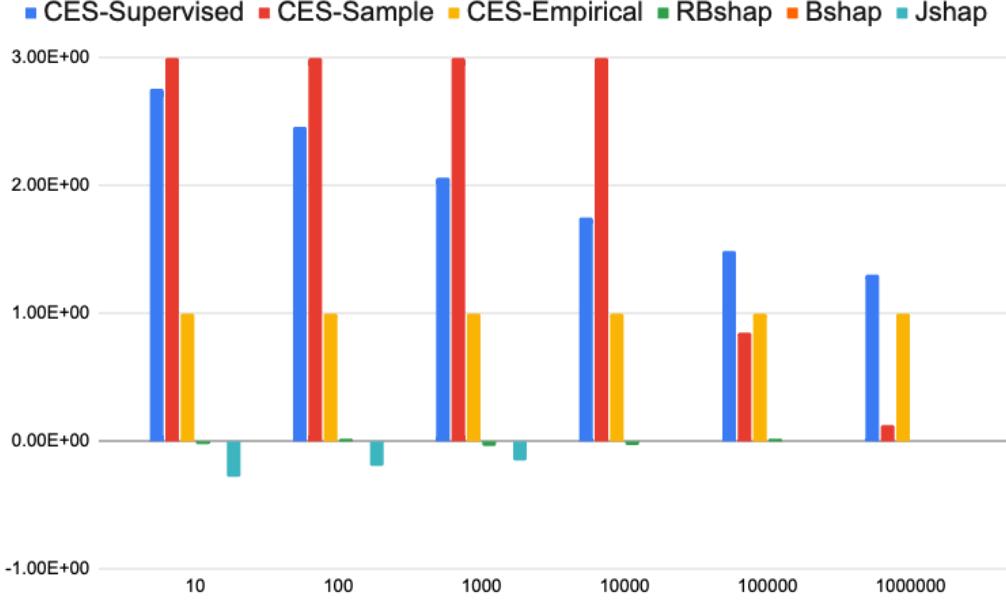


Figure 4: Relative importance ratio of feature 3 with respect to the average importance of feature 1 and feature 2, the ideal value should be  $\leq 0$ . We show the relative important to be 3 if the actual value is larger than 3 for better visualization.

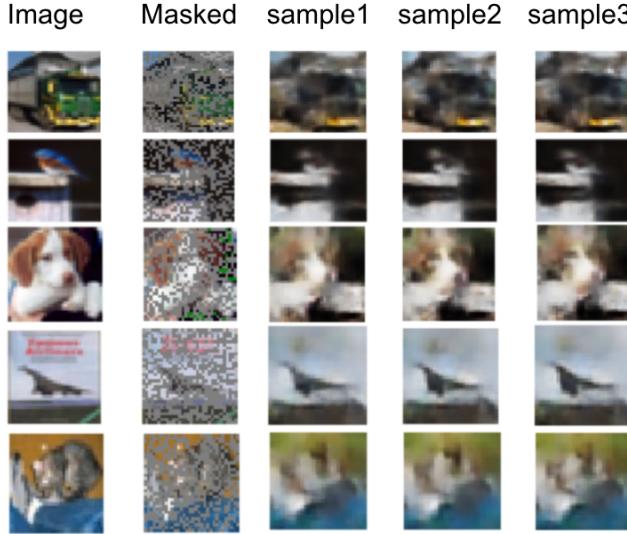


Figure 5: Sampling result from masked VAE.

samples from true  $p(\mathbf{x}'|\mathbf{x}_S)$ , as the samples  $\mathbf{x}'_S$  should be equal to the original image  $\mathbf{x}$  on pixels that are not masked.

## B More Visualization Results

In this section, we include more visualization results for Shapley values of Imagenet images. We show the visualization results for Bshap, JBshap, CES-Masked-

VAE, CES, and RJBshap with different testing images in Fig. 6.

## C Translative Relation For Shapley value Axioms

Recall the set of axioms we introduced in Sec. 5, we rename their abbreviations here for simpler presentations.

Lin. in main text is renamed as (L-DS), Sym. in main text is renamed as (S-DS), Dum. in main text is renamed as (D-DS), Null in main text is renamed as (N-DS), Eff. in main text is renamed as (E-DS), Set. in main text is renamed as (Set-DS), and the renaming makes the axioms easier to distinguish to other versions of the respective axioms.

1. **Linearity (over functions and distributions) (L-DS):** For any functions  $f, f_1, f_2, p, p_1, p_2, \alpha_1 v_{\mathbf{x}, f_1, p}(S) + \alpha_2 v_{\mathbf{x}, f_2, p}(S) = v_{\mathbf{x}, \alpha_1 f_1 + \alpha_2 f_2, p}(S)$ , for  $\alpha_1, \alpha_2 \in \mathbb{R}$  and  $\alpha_1 v_{\mathbf{x}, f, \alpha_1 p_1}(S) + \alpha_2 v_{\mathbf{x}, f, \alpha_2 p_1}(S) = v_{\mathbf{x}, f, \alpha_1 p_1 + \alpha_2 p_2}(S)$  for  $\alpha_1, \alpha_2 \geq 0; \alpha_1 + \alpha_2 = 1$ .
2. **Symmetry (over functions and distributions) (S-DS):** for all  $i, j \notin S$ ,  $f, p, \mathbf{x}, \mathbf{x}'$  all being symmetric in the  $i$ -th and  $j$ -th dimensions implies  $v_{\mathbf{x}, f, p}(i \cup S) = v_{\mathbf{x}, f, p}(j \cup S)$ .
3. **Dummy Player (over functions and distributions) (D-DS):**  $f, p$  being invariant in the  $i$ -th

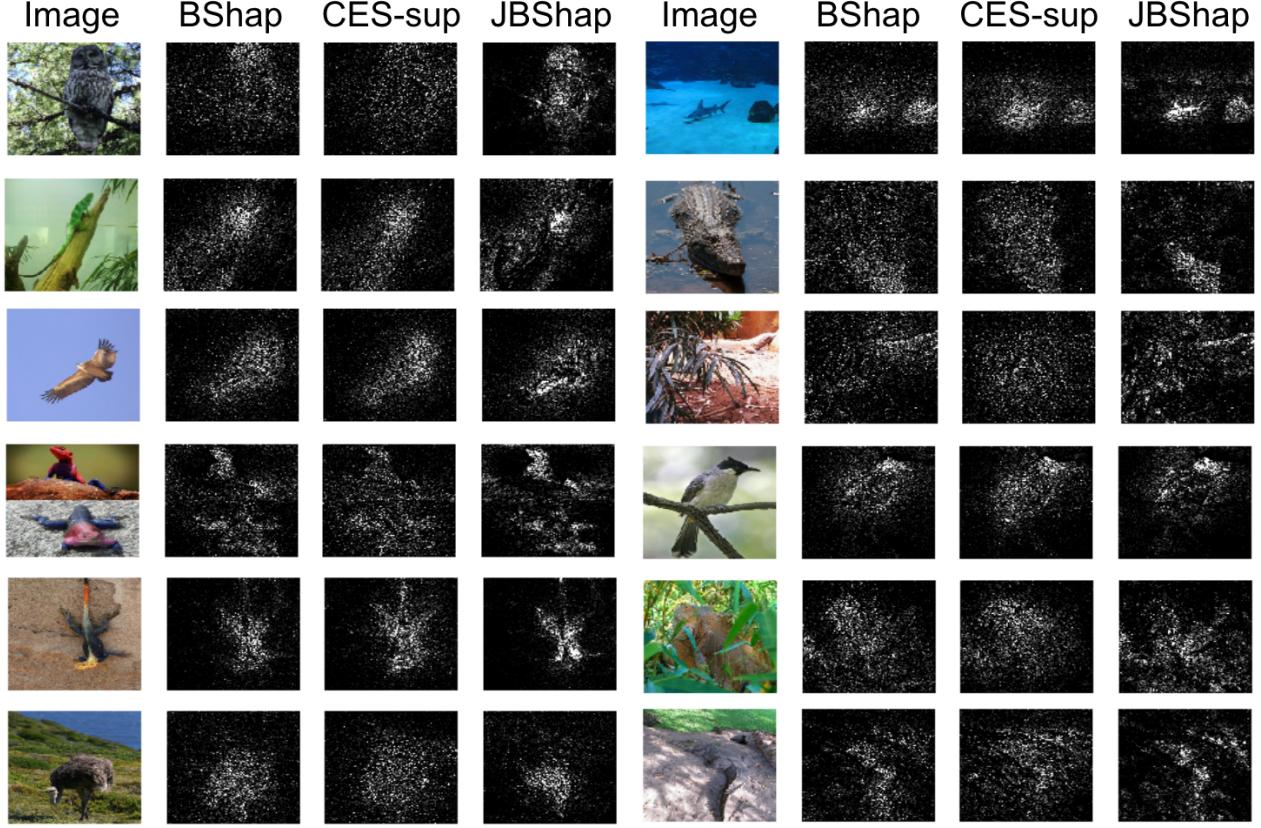


Figure 6: More Visualization result on Imagenet.

- dimension implies  $v_{\mathbf{x}, f, p}(S) = v_{\mathbf{x}, f, p}(i \cup S)$ .
4. **Null Player (over functions and distributions) (N-DS):** if  $f', p'$  satisfies  $f(\mathbf{x}') = f'(\mathbf{x}')$ ,  $p(\mathbf{x}') = p'(\mathbf{x}')$ , then  $v_{f, p, \mathbf{x}}(\emptyset) = v_{f', p', \mathbf{x}}(\emptyset)$ .
  5. **Efficiency (over functions and distributions) (A-DS):**  $v_{f, p, \mathbf{x}}[d] - v_{f, p, \mathbf{x}}[\emptyset] = f(x)p(\mathbf{x}) - f(x')p(\mathbf{x}')$ .
  6. **Set Relevance (over functions and distributions) (Set-DS):** If  $f_1(\mathbf{x}_S, \bar{x}_{\bar{S}}) = f_2(\mathbf{x}_S, \bar{x}_{\bar{S}})$  and  $p_1(\mathbf{x}_S, \bar{x}_{\bar{S}}) = p_2(\mathbf{x}_S, \bar{x}_{\bar{S}})$  for any  $\bar{x}$ ,  $v_{f_1, p_1, \mathbf{x}}(S) = v_{f_2, p_2, \mathbf{x}}(S)$ .
  7. **Strong-T-Robustness (over functions and distributions) (Rob-DS):** For any functions  $f_1, f_2, p$ , if  $\max_{\mathbf{x}} |f_1(\mathbf{x}) - f_2(\mathbf{x})|p(\mathbf{x}) \leq \epsilon$ , then  $|v_{\mathbf{x}, f_1, p}(S) - v_{\mathbf{x}, f_2, p}(S)| \leq T\epsilon$ .

We then introduce the following definitions for a set of axioms for a value function. Define the set  $[d] := \{1, 2, \dots, d\}$ . We first start from axioms with only a single input function  $f$ :

1. **Linearity (over functions) (L-IS):**  $v_{\mathbf{x}, f_1, p}(S) + v_{\mathbf{x}, f_2, p}(S) = v_{\mathbf{x}, f_1 + f_2, p}(S)$ .

2. **Symmetry (over functions) (S-IS):** for all  $i, j \notin S$ ,  $f$ ,  $\mathbf{x}$ , all being symmetric in the  $i$ -th and  $j$ -th dimensions implies  $v_{\mathbf{x}, f, p}(i \cup S) = v_{\mathbf{x}, f, p}(j \cup S)$ .

3. **Dummy Player (over functions) (D-IS):**  $f$  being invariant in the  $i$ -th dimension implies  $v_{\mathbf{x}, f, p}(S) = v_{\mathbf{x}, f, p}(i \cup S)$ .

4. **Null Player (over functions) (N-IS):** if  $f(\mathbf{x}) = f'(\mathbf{x})$ , then  $v_{f, p, \mathbf{x}}(\emptyset) = v_{f', p, \mathbf{x}}(\emptyset)$ .

5. **Efficiency (over functions) (E-IS):**  $v_{f, p, \mathbf{x}}[d] - v_{f, p, \mathbf{x}}[\emptyset] = f(x) - f(x')$ .

6. **Set Relevance (over functions) (Set-IS):** If  $f_1(\mathbf{x}_S, \bar{x}_{\bar{S}}) = f_2(\mathbf{x}_S, \bar{x}_{\bar{S}})$  for any  $\bar{x}$ ,  $v_{f_1, p_1, \mathbf{x}}(S) = v_{f_2, p_2, \mathbf{x}}(S)$ .

The difference is that these set of axioms disregard the data distribution  $p$ .

We then define the following axioms between the **Set** function  $v$  and **Explanation**  $\phi$  as Axioms-SE. This is a classic set of axioms in game theory and has been widely discussed. See, e.g., the work of Sundararajan and Najmi [2019], Shapley [1988], Lundberg and Lee [2017].

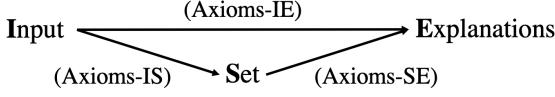


Figure 7: An illustration of the relation between the three sets of axioms. Axioms-IS and Axioms-SE satisfy a transfer property: broadly speaking, a pair  $v, \phi$  satisfies Axioms-IE if  $v$  satisfies Axioms-IS and  $\phi$  satisfies Axioms-SE.

1. **Linearity from set to explanation (L-SE):**  $\phi(v_1) + \phi(v_2) = \phi(v_1 + v_2)$  for any two value functions  $v_1$  and  $v_2$ .
2. **Symmetry from set to explanation (S-SE):**  $v(S \cup i) = v(S \cup j)$  for any  $S \subseteq [d] \setminus \{i, j\}$ ,  $\phi_i(v) = \phi_j(v)$ .
3. **Dummy Player from set to explanation (D-SE):**  $v(S \cup i) = v(S)$  for any  $S \subseteq [d] \setminus \{i\}$  implies  $\phi_i(v) = 0$ .
4. **Efficiency from set to explanation (E-SE):**  $\sum_i \phi_i(v) = v([d]) - v(\emptyset)$ .

The Shapley value is well-known to be the unique function that satisfies the above four axioms. However, this set of axioms assumes a set function. On the other hand, the machine learning model takes a real-valued input instead of a set.

Thus, we propose the following set of axioms between the Input functions ( $f$  and  $p$ ) and Explanation  $\phi$ , which we term Axioms-IE. Axioms-IE can be viewed as a parallel to the axioms proposed by Sundararajan and Najmi [2019]; the main difference is that our axioms take into account the in-data classifier  $p$ .

1. **Linearity from double input to explanation (L-DE):**  $\phi_{f_1,p}(\mathbf{x}) + \phi_{f_2,p}(\mathbf{x}) = \phi_{f_1+f_2,p}(\mathbf{x})$  and  $\phi_{f,p_1}(\mathbf{x}) + \phi_{f,p_2}(\mathbf{x}) = \phi_{f,p_1+p_2}(\mathbf{x})$ , for any functions  $f, f_1, f_2, p, p_1, p_2$ .
2. **Symmetry from double input to explanation (S-DE):** if  $f, p, \mathbf{x}', \mathbf{x}$  are symmetric in the  $i$ -th and  $j$ -th dimensions, then  $\phi_{f,p,i}(\mathbf{x}) = \phi_{f,p,j}(\mathbf{x})$ .
3. **Dummy Player from double input to explanation (D-DE):** both  $f, p$  being invariant in the  $i$ -th dimension implies  $\phi_{f,p,i}(\mathbf{x}) = 0$ . In other words, if feature  $i$  is not used in the model, then
4. **Efficiency from double input to explanation (E-DE):**  $\sum_i \phi_{f,p,i}(\mathbf{x}) = f(\mathbf{x})p(\mathbf{x}) - f(\mathbf{x}')p(\mathbf{x}')$ .

We remark that when ignoring  $p$ , this recovers a version of Axioms-IE proposed by Sundararajan and Najmi [2019]: when ignoring  $p$ , it should be quite evident that value functions that depend on  $p$  (such as

CES and JBshap) would easily fail to satisfy linearity and dummy defined by Sundararajan and Najmi [2019]. However, we argue that (CD-DE) is the form of dummy player axiom satisfied by CES and (D-DE) is the form of dummy player axiom satisfied by JBshap. This suggests that previous discussions on this topic may need restatements based on our proposed Axioms-IE, which takes into account the dependencies of  $p$ .

The intricate relation between Axioms-IS, Axioms-SE and Axioms-IE is captured in the following theorem.

**Theorem 2.** *If set function  $v_{\mathbf{x},f,p}(\cdot)$  satisfies Axioms-IS, and explanation  $\phi_{\mathbf{x},f,p}(\cdot)$  satisfies Axioms-SE, then  $(v_{\mathbf{x},f,p}(\cdot), \phi(v_{\mathbf{x},f,p}(\cdot)))$  satisfies Axioms-IE. Similarly, if set function  $v_{\mathbf{x},f,p}(\cdot)$  satisfies (L-DS), (S-DS), (D-DS), (M-DS), (N-DS), and (E-DS), and explanation  $\phi_{\mathbf{x},f,p}(\cdot)$  is the Shapley value, then  $(v_{\mathbf{x},f,p}(\cdot), \phi(v_{\mathbf{x},f,p}(\cdot)))$  satisfies (L-DE), (S-DE), (D-DE), (E-DE).*

Here, theorem states the translatable relationship between set of axioms from input to set, set to explanations, and input to explanations, as shown in Fig. 7.

This further motivates the design of axioms shown in Sec.5.

We remark that these sets of axioms can be useful to explanations not limited to the Shapley value. For instance, the Banzhaf value (SE version) shares a set of axioms with the Shapley value (SE version), together with our proposed (DS version of axioms), we may get resulting (DE) axioms for the Banzhaf value also. However, since this paper is focused to the Shapley value as an explanation, we do not further discuss explanations other than the Shapley value.

## D Experiment Details

### D.1 Hiding Unfairness with Noise contrastive estimation

For the hiding unfairness experiment, we use the noise contrastive estimation to estimate the data density  $p$ . We build the OOD classifier, we let the noise be  $\mathbf{x}_S \sim p(\mathbf{x} | \mathbf{x} \in Q)$  where  $\mathbf{x} \sim p^E(\mathbf{x})$ ,  $S \sim$  uniform over all set, where the noise is masked image which is equally likely for any available masking. Therefore, for any two  $\mathbf{x}_S$  in the noise distribution, they will have exactly the same pdf. We then train a OOD classifier on equal number of true data  $D$  and noise  $Q$ , using the binary cross entropy loss. The resulting  $\text{OOD}(\mathbf{x})$  will be equal to  $p(\mathbf{x} \in D | \mathbf{x}) = \frac{p(\mathbf{x} | \mathbf{x} \in D)}{p(\mathbf{x} | \mathbf{x} \in D) + p(\mathbf{x} | \mathbf{x} \in Q)}$  if the OOD classifier is trained on balanced noise and true data. By some calculation

we get  $p(\mathbf{x}|\mathbf{x} \in D) = p(\mathbf{x}|\mathbf{x} \in Q) \times \frac{\text{OOD}(\mathbf{x})}{1-\text{OOD}(\mathbf{x})}$ . Since  $p(\mathbf{x}|\mathbf{x} \in Q)$  is equal for any possible point in  $Q$ , we may treat it as a constant, and thus  $p(\mathbf{x}|\mathbf{x} \in D) \propto \frac{\text{OOD}(\mathbf{x})}{1-\text{OOD}(\mathbf{x})}$ .

By Thm.2 in Gutmann and Hyvärinen [2010], since  $p(\mathbf{x}|\mathbf{x} \in Q)$  is non-zero whenever  $p(\mathbf{x}|\mathbf{x} \in D)$  is non-zero, and if additionally  $\int g(\mathbf{x})g(\mathbf{x})P(\mathbf{x})p_d(\mathbf{x})d\mathbf{x}$  has full rank, where  $P(\mathbf{x}) = 1 - \text{OOD}(\mathbf{x})$  and  $g(\mathbf{x}) = \nabla_{\theta}p(\mathbf{x}|\mathbf{x} \in D, \theta)$ , then  $p(\mathbf{x}|\mathbf{x} \in D) \rightarrow p(\mathbf{x})$ , such that our probability estimate  $p(\mathbf{x}|\mathbf{x} \in D)$  converges to the true density  $p(\mathbf{x})$  in probability. Note that we only need a scaled version of  $p(\mathbf{x})$ , so we also scale  $p(\mathbf{x})$  to  $[0, 1]$  for simplicity.

We normalize all features to have 0 mean and unit variance, and we thus choose baseline value to be 0 for Bshap, JBshap, RBshap, RJBshap. We use two layer NNs for the original model and the fine-tuned model  $f'$ , and use a three layer NN to model OOD( $\mathbf{x}$ ). Recall that we have added 0.1 times the feature value of “sex” to the income value to make the model biased. To create a fine-tuned model such that the Shapley value is negative, we add an additional loss to enforce  $f'$  (the fine-tuned model) is negatively correlated to the feature value of “sex” on off-manifold regions. To train the fine-tuned model, we have two losses, the first is to enforce  $f'$  is close to the original model  $f$  on the data manifold (points in the original dataset). The second is to enforce the prediction of  $f'$  to be negatively correlated to the value of “sex” feature only on regions with the scaled  $p(\mathbf{x})$  is smaller than 0.1. We use squared error for both losses and train the model with a weight 1 : 100, while the fine-tuned model still predicts the same result with the original model on over 98 percent of the data.

To train the fine-tuned surrogate model for CES-Supervised, we also have two losses. the first is to enforce  $g'$  (the fune-tuned surrogate model) is close to the original surrogate model  $g$  on the data manifold (points in the original dataset). The second is to enforce the prediction of  $f'$  to be negatively correlated to the value of “sex” feature only on regions with the scaled  $p(\mathbf{x})$  is smaller than 0.1. We use squared error for both losses and train the model with a weight 1 : 1, while the fine-tuned model still predicts the same result with the original model on over 99 percent of the data.

We argue why the surrogate model learned for CES-Supervised can be fine-tuned in this setting. Note that CES-supervised learns a surrogate model based on the MSE loss. However, during the training of the surrogate model, some other losses (such as data augmentation and weight regularization) would have to be added to guarantee convergences on more compli-

cated dataset such as imangenet. Therefore, it would be difficult if not impossible to know whether the learned surrogate model is fine-tuned or not, and the only indicator is the MSE loss. As we show in the main text, the fine-tuned surrogate model has MSE loss very close to the original surrogate model (even lower), and thus it would be difficult to differentiate whether the surrogate model is fine-tuned or not. Our setting assumes that training of the model is a process that the user cannot completely control, and since CES-supervised requires a full retraining of the surrogate model, it is still prone to perturbations.

Since there are only 12 features, we calculate the exact Shapley value by the formula regression formula in Ancona et al. [2018]. For the randomized version RBshap, RJBshap, CES-Sample, we sample 10 points per value function. To report the global Shapley value, we sum the Shapley value over all first 100 data points whose sex feature is originally “0” (before normalization).

## D.2 Visualizing on Imagenet

We choose the first 50 class in imangenet to reduce the number of training examples (and thus the training cost is reduced) so that CES-supervised will be possible to compute.

For visualization, we mask out the bottom 30 percent of the pixels with the least L1-norm of Shapley value, and scale Shapley value for all pixels to  $[0, 1]$ . We use the softmax output of the target class of the CNN model as the function  $f$ . For RJBshap, we let the baseline distribution to be a constant vector with all elements in the vector to range uniformly between  $[0, 1]$ .

We now introduce the sampling method for calculating JBShap on high-dimensional image data. We observe that due to the multiplication of data density  $p(\cdot)$ , the value function of  $v_{\mathbf{x}, f, p}(S)$  is very small when  $|S|$  is much smaller than the total number of features. The sampling track proposed by Castro et al. [2009] calculates the Shapley value for feature  $i$  as  $\mathbb{E}_S[v(S \cup i) - v(S)]$ , where we first sample a permutation of all features, and  $S$  is the set of all features that occurred before  $i$  in the permutation (if  $i$  is sampled first  $S = \emptyset$ ). We slightly modify the sampling by setting  $v(S) = 0$  whenever  $|S| < 0.8 \times d$ , where  $d$  is the total features of the model. Thus, whenever the feature  $i$  occurs in the first 80% of the features in the sampled permutation, we use the value 0 instead of calculating  $v(S \cup i) - v(S)$ . This sampling trick is only applicable to JBShap since  $p(x)$  is often small when many pixels are set to the baseline value.

### D.3 CES-Masked-VAE

For the training of the masked VAE and VAE model, we used 4 convolutional layers with 256 filters followed by a dense layer each for the encoder, masked encoder, and the decoder. The latent dimension of  $z$  is 256, and the VAE modeled is trained 500 epochs and  $\beta$  is set to 1, following the structure of Frye et al. [2020]. For the reported difference  $\mathbb{E}_{\mathbf{x}' \sim p(\mathbf{x}'|\mathbf{x}_S)}[\frac{1}{|S|}\|\mathbf{x}_S - \mathbf{x}'_S\|_1]$  and  $\mathbb{E}_{\mathbf{x}' \sim p(\mathbf{x}'|\mathbf{x}_S)}[\frac{1}{|S|}\|\mathbf{x}_{\bar{S}} - \mathbf{x}'_{\bar{S}}\|_1]$ , we average over 10 runs, where each runs contain 10000 testing images each with one sampled mask  $S$ , and for each  $\mathbf{x}_S$  we samples  $\mathbf{x}' \sim p(\mathbf{x}'|\mathbf{x}_S)$  20 times, and report the average of  $[\frac{1}{|S|}\|\mathbf{x}_S - \mathbf{x}'_S\|_1]$  and  $[\frac{1}{|S|}\|\mathbf{x}_{\bar{S}} - \mathbf{x}'_{\bar{S}}\|_1]$ .

### D.4 CES-supervised

We trained the CES-supervised by modifying the training code of regular Imagenet by adding a noise layer in the data generation process to mask the image randomly, and change the training target to the output of a pre-trained model on the unmasked image, and adopt the MSE loss. We follow the normal training of the imangenet with a resnet-18 models, and we trained with 20 epochs with batch size 256, a single GPU, and learning rate 0.01.

To evaluate the True MSE loss and the empirical MSE loss, we randomly sample 100  $\mathbf{x}_S$ . For each  $\mathbf{x}_S$ , we calculate  $[(f(\mathbf{x}) - v_{\mathbf{x}_S, f, p}^{CS})^2]$  for the empirical MSE loss for  $v_{\mathbf{x}_S, f, p}^{CS}$ , and  $\mathbb{E}_{\mathbf{x}' \sim p(\mathbf{x}'|\mathbf{x}_S)}[(f(\mathbf{x}) - v_{\mathbf{x}_S, f, p}^{CS})^2]$  for the true MSE loss for  $v_{\mathbf{x}_S, f, p}^{CS}$ , where sampling from  $\mathbf{x}' \sim p(\mathbf{x}'|\mathbf{x}_S)$  is done by importance sampling with 100 samples.

## E Proof

### E.1 Proof of Proposition 1

*Proof.* We consider  $v_{\mathbf{x}_S, f, p^E}^{CS}$  with a fix  $\mathbf{x}_S, f$ , and plug in  $p^E(\mathbf{x}) = \frac{1}{m}\mathbb{1}[\mathbf{x} \text{ in dataset}]$ . Eq. (1) can be rewritten as

$$\mathbb{E}_{S \sim \text{shapley}}\left[\sum_{i=1}^m (f(\mathbf{x}^i) - v_{\mathbf{x}_S, f, p}^{CS})^2\right]. \quad (3)$$

Since we consider  $v_{\mathbf{x}_S, f, p^E}^{CS}$  with  $\mathbf{x}_S, f$  fixed, we can take the first order condition on  $v_{\mathbf{x}_S, f, p^E}^{CS}$  with a fix  $\mathbf{x}_S, f$  in Eq. (3) since it is convex, we then obtain the optimal  $v_{\mathbf{x}_S, f, p}^{CS*}$  should satisfy:

$$-p_{\text{shapley}}(S)\left[\sum_{i=1}^m 2\mathbb{1}[\mathbf{x}_S^i = \mathbf{x}_S](f(\mathbf{x}_i) - v_{\mathbf{x}_S, f, p}^{CS*})\right] = 0. \quad (4)$$

Thus, we obtain

$$v_{\mathbf{x}_S, f, p}^{CS*} = \frac{\sum_{i=1}^m \mathbb{1}[\mathbf{x}_S^i = \mathbf{x}_S]f(\mathbf{x}^i)}{\sum_{i=1}^m \mathbb{1}[\mathbf{x}_S^i = \mathbf{x}_S]} = f_{\mathbf{x}, f, p^E}(S). \quad (5)$$

□

### E.2 Proof of Proposition 2

*Proof.* Assume that Bshap and CES are strong  $T$ -robust with some  $T = T_1 > 0$ . Consider a model  $f_1(\mathbf{x}) = \mathbf{x}_1$ , with two candidate features  $\mathbf{x}_1, \mathbf{x}_2$ . We assume the features follow standard Gaussian  $\mathcal{N}(0, I)$  distributions. Suppose we are to explain a data point  $\mathbf{x} = (0, T_0)$ . We choose  $T_0$  such that  $T_0 > \text{pdf}^{-1}(\frac{1}{T_1})$ , and thus  $p(\mathbf{x} = (0, T_0)) < p(\mathbf{x}_2 = T_0) < \frac{1}{T_1}$ .

It is clear that  $v_{\mathbf{x}, f_1, p}^C(\{\mathbf{x}_2\}) = v_{\mathbf{x}, f_1, p}^B(\{\mathbf{x}_2\}) = 0$  since the model  $f_1$  is unrelated to  $\mathbf{x}_2$ . However, if we perturb the model by setting  $f_2(\mathbf{x}) = \mathbf{x}_1 + C\mathbb{1}[\mathbf{x}_2 = T_0]$ ,  $v_{\mathbf{x}, f_2, p}^C(\{\mathbf{x}_2\}) = v_{\mathbf{x}, f_2, p}^B(\{\mathbf{x}_2\}) = C$ .  $\max_{\mathbf{x}} |f_1(\mathbf{x}) - f_2(\mathbf{x})|/p(\mathbf{x}) < \frac{C}{T_1}$ , but  $|v_{\mathbf{x}, f_1, p}^B(S) - v_{\mathbf{x}, f_2, p}^B(S)| = C$  and  $|v_{\mathbf{x}, f_1, p}^C(S) - v_{\mathbf{x}, f_2, p}^C(S)| = C$ . Thus, Bshap and CES are not T-robust for any  $T_1 > 0$ .

□

### E.3 Proof of Theorem. 1

In this section, we provide a proof of Theorem 1. We first start with the first part: If  $v_{\mathbf{x}, f, p}(S)$  satisfies (Lin.), (Sym.), (Dum.), (Null), (Eff.), (Set.), (Rob.) axioms implies  $v_{\mathbf{x}, f, p}(S) = f(\mathbf{x}_S; \mathbf{x}'_{\bar{S}})p(\mathbf{x}_S; \mathbf{x}'_{\bar{S}})$ .

We begin with the following proposition, which characterizes the dependency of  $v_{\mathbf{x}, f, p}(S)$  on  $f, p, \mathbf{x}_S, S$ .

**Proposition 3.** *If (D-DS), (N-DS), (SET-DS) are all satisfied for input function to set function, then we have  $v_{\mathbf{x}, f, p}(S)$  only depends on  $f(\mathbf{x}_S; \mathbf{x}'_{\bar{S}}), p(\mathbf{x}_S; \mathbf{x}'_{\bar{S}})$ . That is, if for some  $f_1, f_2, p_1, p_2$  so that  $f_1(\mathbf{x}_S; \mathbf{x}'_{\bar{S}}) = f_2(\mathbf{x}_S; \mathbf{x}'_{\bar{S}})$  and  $p_1(\mathbf{x}_S; \mathbf{x}'_{\bar{S}}) = p_2(\mathbf{x}_S; \mathbf{x}'_{\bar{S}})$ , then  $v_{\mathbf{x}, f_1, p_1}(S) = v_{\mathbf{x}, f_2, p_2}(S)$ .*

*Proof.* By (Set-DS), we know that  $v_{\mathbf{x}, f, p}(S)$  depends only on  $f(\mathbf{x}_S; \cdot)$  and  $p(\mathbf{x}_S; \cdot)$ . We now want to show that for  $f_1, f_2, p_1, p_2$  such that  $f_1(\mathbf{x}_S; \mathbf{x}'_{\bar{S}}) = f_2(\mathbf{x}_S; \mathbf{x}'_{\bar{S}})$  and  $p_1(\mathbf{x}_S; \mathbf{x}'_{\bar{S}}) = p_2(\mathbf{x}_S; \mathbf{x}'_{\bar{S}})$ , but  $f_1(\mathbf{x}_S; \mathbf{x}''_{\bar{S}}) \neq f_2(\mathbf{x}_S; \mathbf{x}''_{\bar{S}})$  or  $p_1(\mathbf{x}_S; \mathbf{x}''_{\bar{S}}) \neq p_2(\mathbf{x}_S; \mathbf{x}''_{\bar{S}})$ , where  $\mathbf{x}''$  is some arbitrary input not equal to  $\mathbf{x}'$ ,  $v_{\mathbf{x}, f_1, p_1}(S) = v_{\mathbf{x}, f_2, p_2}(S)$ . Thus, proposition 3 is much stricter than (Set-DS).

Given any arbitrary  $f_1, f_2, p_1, p_2$ , we now construct  $f'_1, f'_2, p'_1, p'_2$  by setting  $f'_1(u) := f_1(u|\text{do } u_S = \mathbf{x}_S)$ ,  $p'_1(u) = p_1(u|\text{do } u_S = \mathbf{x}_S)$ ,  $f'_2(u) := f_2(u|\text{do } u_S = \mathbf{x}_S)$ ,  $p'_2(u) = p_2(u|\text{do } u_S = \mathbf{x}_S)$ . By (Set-DS), we know that  $v_{\mathbf{x}, f_1, p_1}(S) = v_{\mathbf{x}, f'_1, p'_1}(S)$  and  $v_{\mathbf{x}, f_2, p_2}(S) = v_{\mathbf{x}, f'_2, p'_2}(S)$  (\*). We also clearly see that  $f'_1, f'_2, p'_1, p'_2$

are dummy in all features in set  $S$ . Thus, by (D-DS), we have  $v_{\mathbf{x}, f'_1, p'_1}(S) = v_{\mathbf{x}, f'_1, p'_1}(\emptyset)$ ,  $v_{\mathbf{x}, f'_2, p'_2}(S) = v_{\mathbf{x}, f'_2, p'_2}(\emptyset)$  (\*\*). By (N-DS), we know  $v_{\mathbf{x}, f'_1, p'_1}(\emptyset)$  depends only on  $f'_1(\mathbf{x}')$  and  $p'_1(\mathbf{x}')$  which is  $f_1(\mathbf{x}_S; \mathbf{x}'_{\bar{S}})$  and  $p_1(\mathbf{x}_S; \mathbf{x}'_{\bar{S}})$ . Similarly,  $v_{\mathbf{x}, f'_2, p'_2}(\emptyset)$  depends only on  $f_2(\mathbf{x}_S; \mathbf{x}'_{\bar{S}})$  and  $p_2(\mathbf{x}_S; \mathbf{x}'_{\bar{S}})$ . Therefore, if  $f_1, f_2, p_1, p_2$  satisfies  $f_1(\mathbf{x}_S; \mathbf{x}'_{\bar{S}}) = f_2(\mathbf{x}_S; \mathbf{x}'_{\bar{S}})$  and  $p_1(\mathbf{x}_S; \mathbf{x}'_{\bar{S}}) = p_2(\mathbf{x}_S; \mathbf{x}'_{\bar{S}})$ , it follows that  $v_{\mathbf{x}, f'_1, p'_1}(\emptyset) = v_{\mathbf{x}, f'_2, p'_2}(\emptyset)$ . Combining this with the properties (\*) and (\*\*) allows us to conclude  $v_{\mathbf{x}, f_1, p_1}(S) = v_{\mathbf{x}, f_2, p_2}(S)$ , which completes our proof.

□

**Final Proof of Thm. 1** With the proposition in hand, we are ready to prove Theorem 1.

The linearity property of  $v$  (with respect to  $f, p$ ) implies  $v_{\mathbf{x}, k, f, p}(S) = kv_{\mathbf{x}, f, p}(S)$ ,  $v_{\mathbf{x}, f, kp}(S) = kv_{\mathbf{x}, f, p}(S)$  for any  $k \in \mathbb{R}$ . Now, by proposition 3, we can write  $v_{\mathbf{x}, f, p}(S) = G(f(\mathbf{x}_S; \mathbf{x}'_{\bar{S}}), p(\mathbf{x}_S; \mathbf{x}'_{\bar{S}}))$  for some functional  $G$ . Let us define  $A_f(\mathbf{x}, S) := f(\mathbf{x}_S; \mathbf{x}'_{\bar{S}})$  and  $B_p(\mathbf{x}, S) := p(\mathbf{x}_S; \mathbf{x}'_{\bar{S}})$ .

The linearity property yields a key observation: given any arbitrary pair  $(\mathbf{x}_0, S_0)$  and two functions  $f_1, f_2$ , we have

$$\begin{aligned} &G(A_{f_1}(\mathbf{x}_0, S_0)A_{f_2}(\mathbf{x}_0, S_0), B_p(\mathbf{x}_0, S_0)) \\ &= A_{f_1}(\mathbf{x}_0, S_0)G(A_{f_2}(\mathbf{x}_0, S_0), B_p(\mathbf{x}_0, S_0)). \end{aligned}$$

The above holds for any choice of  $f_1, f_2$  given a fixed pair  $(\mathbf{x}_0, S_0)$ . We may henceforth select  $f_2 \equiv 1$ ,  $\mathbf{x} = \mathbf{x}_0$ ,  $S = S_0$ , yielding

$$\begin{aligned} &G(A_{f_1}(\mathbf{x}_0, S_0), B_p(\mathbf{x}_0, S_0)) \\ &= A_{f_1}(\mathbf{x}_0, S_0)G(1, B_p(\mathbf{x}_0, S_0)). \end{aligned}$$

A similar reasoning gives for any choice  $f, p$

$$\begin{aligned} &G(A_f(\mathbf{x}_0, S_0), B_p(\mathbf{x}_0, S_0)) \\ &= B_p(\mathbf{x}_0, S_0)G(A_f(\mathbf{x}_0, S_0), 1), \end{aligned}$$

and consequently

$$\begin{aligned} &G(A_f(\mathbf{x}_0, S_0), B_p(\mathbf{x}_0, S_0)) \\ &= A_f(\mathbf{x}_0, S_0)B_p(\mathbf{x}_0, S_0)G(1, 1) \end{aligned}$$

holds for any pair  $(\mathbf{x}_0, S_0)$ . It then follows that

$$\begin{aligned} v_{\mathbf{x}, f, p}(S) &= G(A_f(\mathbf{x}, S), B_p(\mathbf{x}, S)) \\ &= Cf(\mathbf{x}_S; \mathbf{x}'_{\bar{S}})p(\mathbf{x}_S; \mathbf{x}'_{\bar{S}}), \end{aligned}$$

where  $C := G(1, 1)$ . Finally, with the efficiency axiom, we see that only  $C = 1$  satisfies the axiom. This concludes one direction of Theorem 1.

We remark that the uniqueness of the value function (up to a constant) does not rely on (Eff.), (Sym.),

(Rob.) as they are mostly not used in the proof of the first part, and thus can be rewritten as a stronger theoretical result. However, we include them in the Theorem to make the theoretical statement more concise and emphasize that JBShap satisfies these desired properties.

Moving onwards, it remains to prove that  $v_{\mathbf{x}, f, p}(S)$  satisfies the (Lin.), (Sym.), (Dum.), (Null), (Eff.), (Set.), (Rob.) axioms if  $v_{\mathbf{x}, f, p}(S) = f(\mathbf{x}_S; \mathbf{x}'_{\bar{S}})p(\mathbf{x}_S; \mathbf{x}'_{\bar{S}})$ . It is easy to see that the first six axioms hold. Robustness is shown by fixing any functions  $f_1, f_2, p$  and noting

$$\begin{aligned} &(f_1(\mathbf{x}_S; \mathbf{x}'_{\bar{S}})p(\mathbf{x}_S; \mathbf{x}'_{\bar{S}}) - f_2(\mathbf{x}_S; \mathbf{x}'_{\bar{S}})p(\mathbf{x}_S; \mathbf{x}'_{\bar{S}})) \\ &= (f_1(\mathbf{x}_S; \mathbf{x}'_{\bar{S}}) - f_2(\mathbf{x}_S; \mathbf{x}'_{\bar{S}}))p(\mathbf{x}_S; \mathbf{x}'_{\bar{S}}) \\ &\leq \max_{\mathbf{x}} |(f_1(\mathbf{x}) - f_2(\mathbf{x}))|p(\mathbf{x}) \\ &\leq \frac{1}{T} \max_{\mathbf{x}} |f_1(\mathbf{x}) - f_2(\mathbf{x})|p(\mathbf{x}). \end{aligned}$$

As expected, strong  $T$ -robustness is satisfied when  $T \leq 1$ .

#### E.4 Proof of Theorem 2

Theorem 2 is proved through the following six straightforward propositions, where the proofs simply follow by the definition.

**Proposition 4.** If two given input function and set function pair  $(f, p, v_{\mathbf{x}, f, p}(\cdot))$  satisfies (L-DS), and set function and explanation pair  $(v_{\mathbf{x}, f, p}(\cdot), \phi(v_{\mathbf{x}, f, p}(\cdot)))$  satisfies (L-SE), then  $(f, p, \phi(v_{\mathbf{x}, f, p}(\cdot)))$  satisfy (L-DE).

**Proposition 5.** If two given input function and set function pair  $(f, p, v_{\mathbf{x}, f, p}(\cdot))$  satisfies (S-DS), and set function and explanation pair  $(v_{\mathbf{x}, f, p}(\cdot), \phi(v_{\mathbf{x}, f, p}(\cdot)))$  satisfies (S-SE), then  $(f, p, \phi(v_{\mathbf{x}, f, p}(\cdot)))$  satisfy (S-DE).

**Proposition 6.** If two given input function and set function pair  $(f, p, v_{\mathbf{x}, f, p}(\cdot))$  satisfies (D-DS), and set function and explanation pair  $(v_{\mathbf{x}, f, p}(\cdot), \phi(v_{\mathbf{x}, f, p}(\cdot)))$  satisfies (D-SE), then  $(f, p, \phi(v_{\mathbf{x}, f, p}(\cdot)))$  satisfy (D-DE).

**Proposition 7.** If two given input function and set function pair  $(f, p, v_{\mathbf{x}, f, p}(\cdot))$  satisfies (E-DS), and set function and explanation pair  $(v_{\mathbf{x}, f, p}(\cdot), \phi(v_{\mathbf{x}, f, p}(\cdot)))$  satisfies (E-SE), then  $(f, p, \phi(v_{\mathbf{x}, f, p}(\cdot)))$  satisfy (E-DE).

By combining all propositions above, we conclude Theorem 2.