
Concept Gradient: Concept-based Interpretation Without Linear Assumption

Andrew Bai

Department of Computer Science
University of California, Los Angeles
andrewbai@cs.ucla.edu

Chih-Kuan Yeh

Department of Computer Science
Carnegie Mellon University
cjyeh@cs.cmu.edu

Pradeep Ravikumar

Department of Computer Science
Carnegie Mellon University
pradeepr@cs.cmu.edu

Neil Y. C. Lin

Department of Bioengineering
University of California, Los Angeles
neillin@g.ucla.edu

Cho-Jui Hsieh

Department of Computer Science
University of California, Los Angeles
chohsieh@cs.ucla.edu

Abstract

Concept-based interpretations of black-box models are often more intuitive for humans to understand. The most widely adopted approach for concept-based interpretation is Concept Activation Vector (CAV). CAV relies on learning a linear relation between some latent representation of a given model and concepts. The linear separability is usually implicitly assumed but does not hold true in general. In this work, we started from the original intent of concept-based interpretation and proposed Concept Gradient (CG), extending concept-based interpretation beyond linear concept functions. We showed that for a general (potentially non-linear) concept, we can mathematically evaluate how a small change of concept affecting the model's prediction, which leads to an extension of gradient-based interpretation to the concept space. We demonstrated empirically that CG outperforms CAV in both toy examples and real world datasets.

1 Introduction

Explaining the prediction mechanism of machine learning models is important, not only for debugging and gaining trust, but also for humans to learn and actively interact with them. Many feature attribution methods have been developed to attribute importance to input features for the prediction of a model [38, 47]. However, input feature attribution may not be ideal in the case where the input features themselves are not intuitive for humans to understand. It is then desirable to generate explanations with human-understandable concepts instead, motivating the need of **concept based explanation**. For instance, to understand a machine learning model that takes microscopy images as input and predicts the underlying gene expression, attributing importance to high-level concepts such as morphological features of cells explains the predictions better than to raw pixel values.

The most common approach for concept-based interpretation is Concept Activation Vector (CAV) [23]. CAV represents a concept with a vector in some layer of the target model and evaluates the sensitivity of the target model's gradient in the concept vector's direction. Many followup works are based on CAV and share the same fundamental assumption that concepts can be represented as a linear

function in some layer of the target model [14, 35]. However, this assumption generally does not hold, and such restriction limits the application of concept-based attribution to relative simple concepts.

In this paper, we rethink the problem of concept-based explanation and tackle the fundamental linearly separability issue. One perspective of model interpretation is to measure how small perturbations to a feature affect the prediction, which can be measured by the input gradients in typical feature-based explanations. A question naturally arises: is it possible to extend the notion of gradient to the concept space? If so, non-linear concepts can be easily handled without additional assumptions. In this paper, we answer this question in affirmative and provide a simple formulation of Concept Gradient (CG), which measures the how small changes of concept affects the model’s prediction mathematically. If a unique function from concept to prediction exists, CG exactly recovers the gradient. Otherwise, CG captures the gradient from prediction to concept through the minimal input subspace, sufficient for capturing concept information. Given any model f and (potentially non-linear) concept g , CG simply computes the gradient to linearize both functions at individual input points to capture local linear behavior of the joint function. We discovered that when the concept function is linear, CG recovers CAV (with a slightly different scaling factor), which explains why CAV works well in linearly separable cases. The derivation and formulation of CG also provides the insight for understanding how concept functions and models interact locally. Empirically, we verified the correctness of CG on toy examples when the gradient (from model prediction to concept) can be obtained in closed forms. Furthermore, we showed in real world datasets that the linear separability assumption of CAV does not always hold and CG outperforms CAV by a large margin. The best recall@30 of CG is higher than the best of CAV by 6%. CG is also less sensitive to layer selection and the average recall@30 over layers of CG outperforms CAV by 30%.

2 Preliminaries

Problem definition In this paper we use $f : \mathbb{R}^d \rightarrow \mathbb{R}^k$ to denote the machine learning model to be explained, $x \in \mathbb{R}^d$ to denote input, and $y \in \mathbb{R}^k$ to denote the label. For an input sample $\hat{x} \in \mathbb{R}^d$, concept-based explanation aims to explain the prediction $f(\hat{x})$ based on a set of m concepts $\{c_1, \dots, c_m\}$. In particular, the goal is to reveal how important is each concept to the prediction. Concepts can be given in different ways, but in the most general forms, we can consider concepts as functions mapping from the input space to the concept space, denoted as $g_i : \mathbb{R}^d \rightarrow \mathbb{R}$ for each concept i . The function can be explicit or implicit. For example, morphological features such as cell perimeter and circularity can be given as explicit functions designed by domain experts. On the other hand, many concept-based explanations in the literature consider concepts given as a set of examples, which are finite observations from the underlying concept function. We further assume f and g are differentiable which follows the standard assumption of gradient-based explanation methods.

For simplicity we represent all g_i together as a joint function $g : \mathbb{R}^d \rightarrow \mathbb{R}^m$ mapping input to an m -dimensional concept space. We denote a vector in the concept space as $c \in \mathbb{R}^m$ and $\hat{c} = g(\hat{x})$ is the concept vector for a given instance. For each concept function g_i , we can define a local concept relevance score $R_{\hat{x}}(i, j)$ to represent how concept $\hat{c}_i$ affects target class prediction $\hat{y}_j = f_j(\hat{x})$. Further, by aggregating over all the input instances $\{x_1, \dots, x_n\}$, we can define a global concept relevance score $R(i, j)$ to represent how a concept g_i affects the target class prediction y_j . The goal is to calculate concept relevance scores such that the scores reflect the true underlying concept importance, possibly aligned with human intuition.

Recap of Concept Activation Vector (CAV) Concept activation vector is a concept-based interpretation method proposed by Kim, et al. [23]. The idea is to represent a concept with a (concept activation) vector and evaluate the alignment between the input gradients of target model and the vector. In order for the concept to be represented well by a vector, the concept labels must be linearly separable in the space where the vector is lies. The authors implicitly assumed that there exists a layer in the target model where concept labels can be linearly separated.

Let v_c denote the concept activation vector associated with concept c . The authors defines the conceptual sensitivity score to measure concept importance

$$S_f(x) := \nabla f(x) \cdot (v_c / \|v_c\|). \quad (1)$$

The main caveat with the conceptual sensitivity scores is the underlying concept function is not guaranteed to lie in the linear subspace of some neural activation space. Attempting to fit the concept function with a linear model likely leads to poor results for non-trivial concepts, leading to inaccurate conceptual sensitivity scores.

3 Proposed method

3.1 Definition of Concept Gradient (CG)

We assume x is the input instance, $y = f(x)$ is the k -dimensional model output and $c = g(x)$ is the m -dimensional concepts. For standard feature-based explanation, which aims to measure the important of each input dimension of x , gradient-based methods have been widely used. The gradient map $\nabla f(x)$ measures how small changes on each input dimension affects the output and serves as the foundation of many existing explanation methods [1, 37, 38]. This paper is the first formally extending the notion of gradient to concept-based explanation, and by doing so we enable explanation of any general nonlinear concept. We define the **Concept Gradient** (CG) to measure how small perturbations on each concept affects the label prediction:

$$(\text{General version of CG}) \quad \text{CG}(x) := \nabla g(x)^\dagger \nabla f(x). \quad (2)$$

In practice, CG can be easily computed by fine-tuning the original target model on the concept labels to create a non-linear concept model (see more in Section 4.2). Note that when computing the gradient of a multivariate function such as f , we follow the convention that $\nabla f(x) \in \mathbb{R}^{d \times k}$ where the $(\nabla f(x))_{ij} = \frac{\partial f_j(x)}{\partial x_i}$. And $\nabla g(x)^\dagger \in \mathbb{R}^{m \times d}$ is the pseudo-inverse of $\nabla g(x)$. CG(x) will thus be an $m \times k$ matrix and its (i, j) element measures the contribution of concept i to label j . Pseudo-inverse is a generalized version of matrix inversion—when the inverse does not exist, it forms the (unique) inverse mapping from the column space to the row space of the original matrix, while leaving all the other spaces untouched. Assume the Singular Value Decomposition (SVD) of $\nabla g(x)$ is $U_g \Sigma_g V_g^T$, then $\nabla g(x)^\dagger$ is defined as $V_g \Sigma_g^\dagger U_g$ where $\Sigma_g^\dagger$ is formed from Σ_g by taking the inverse of all the non-zero elements and leaving all the zeros alone.

Similar to CAV, CG can be computed at any layer of a neural network by setting x as the neurons in a particular layer. We will discuss a guideline for layer selection in Section 3.5.

Before formally deriving CG, we use a few special cases to illustrate the method. First, for the case when x, y and c are all scalars ($k = m = d = 1$), there exists a function mapping c to y locally around a particular $\hat{c} = g(\hat{x})$ as long as $g'(\hat{x}) \neq 0$. Basic on calculus, under this condition in 1D case we have $\frac{dc}{dx} = 1/(\frac{dx}{dc})$ at $\hat{x}$, so concept gradient exactly recovers $\frac{dy}{dc}$:

$$\left. \frac{dy}{dc} \right|_{c=\hat{c}} = \left. \frac{dy}{dx} \right|_{x=\hat{x}} \cdot \left. \frac{dx}{dc} \right|_{c=\hat{c}} = \left. \frac{dy}{dx} \right|_{x=\hat{x}} \left(\left. \frac{dc}{dx} \right|_{x=\hat{x}} \right)^{-1} = f'(\hat{x}) \cdot \frac{1}{g'(\hat{x})} = \text{CG}(\hat{x}). \quad (3)$$

This can be easily extended to the case when $m = d$ and $\nabla g(\hat{x})$ is invertible. In this case, there exists an unique function $g^{-1}(c)$ mapping c to x locally around $\hat{c}$, and by the chain rule we show CG($\hat{x}$) is equivalent to the derivative of y with respect to c :

$$\left. \frac{\partial y}{\partial c} \right|_{c=\hat{c}} = \left. \frac{\partial f(g^{-1}(c))}{\partial c} \right|_{c=\hat{c}} = \nabla g^{-1}(\hat{c}) \nabla f(g^{-1}(\hat{c})) = (\nabla g(\hat{c}))^{-1} \nabla f(\hat{x}) = \text{CG}(\hat{x}).$$

The pseudo-inverse in (2) extends this derivation to the general case when m, d are in arbitrary dimensions, and we will formally derive it in Section 3.4.

3.2 CG for a single concept

In previous concept attribution methods such as CAV, a canonical setting is to measure the importance of a single concept. Here we try to simplify CG in this setting when concepts are scalars and discuss the relationship with CAV.

When $m = 1$, $\nabla g(x)$ is a $d \times 1$ vector and by the definition of pseudo-inverse, $\nabla g(x)^\dagger = \frac{\nabla g(x)^T}{\|\nabla g(x)\|^2}$, which leads to the following simplified concept gradient when explaining a single concept:

$$(\text{Single-concept CG}) \quad \text{CG}(x) = \left(\frac{1}{\|\nabla g(x)\|^2} \nabla g(x)^T \right) \nabla f(x) = \frac{1}{\|\nabla g(x)\|^2} \nabla g(x)^T \nabla f(x). \quad (4)$$

In this case, CG is simply a normalized version of the inner product between the gradient of concept ($\nabla g(x)$) and the gradient of prediction model ($\nabla f(x)$). This is the main formulation we are going to use in practice, since we found applying CG to each concept individually will match better with human perception, as will be discussed in Section 3.3.

In the special case when $g(x) = v_C$ is a linear function (the assumption of CAV), we have $\text{CG}(x) = v_C^T \nabla f(x) / \|v_C\|^2$, which is almost identical to conceptual sensitivity score in Eq 1 except a slightly different normalization term where CAV normalizes the inner product by $1/\|v_C\|$. Furthermore, the sign of CG and CAV will be identical which explains why CAV is capable of retrieving important concepts under the linearly separable case.

Here we use a simple example to demonstrate that the normalization term could be important in some special cases. Consider f as the following network with two-dimensional input $[x_0, x_1]$:

$$y = 0.1z_0 + z_1, \begin{bmatrix} z_0 \\ z_1 \end{bmatrix} = \begin{bmatrix} 100 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} h_0 \\ h_1 \end{bmatrix}, \begin{bmatrix} h_0 \\ h_1 \end{bmatrix} = \begin{bmatrix} 0.01 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \end{bmatrix},$$

and $c_0 = x_0, c_1 = x_1$. Then we know since $y = 0.1z_0 + z_1 = 0.1x_0 + x_1$, the contribution of c_1 should be 10 times larger than c_0 . In fact, $\frac{dy}{dc_0} = 0.1, \frac{dy}{dc_1} = 1$ and it's easy to verify that CG will correctly obtain the gradient no matter which layer is chosen for computing (4). However, the results will be wrong when a different normalization term is used when computing concept explanation on the hidden layer h . Since $c_0 = 100h_0, c_1 = h_1, y = 10h_0 + h_1$, we have

$$\text{For concept } c_0: v = \frac{dc_0}{dh} = [100, 0]^T, u = \frac{dy}{dh} = [10, 1], v^T u / \|v\| = \textcolor{red}{10}, v^T u / \|v\|^2 = \textcolor{blue}{0.1}$$

$$\text{For concept } c_1: v = \frac{dc_1}{dh} = [0, 1]^T, \frac{dy}{dh} = [10, 1], v^T u / \|v\| = \textcolor{red}{1}, v^T u / \|v\|^2 = \textcolor{blue}{1}.$$

Therefore, the normalization term used in CAV (in red color) will lead to a conclusion that $c_0 > c_1$, while CG (in blue color) will correctly get the actual gradient and conclude $c_1 > c_0$. This is mainly because CG is formally derived from the actual gradient, as we will discuss below. In contrast, CAV is based on the intuition of ‘‘gradient sensitivity’’ and not the exact chain-rule gradient, so is subject to per-dimension scaling.

Although the normalization term can be important in some special cases, in practice we do not find the attribution results to be much different with different normalization terms, probably because such extreme per-dimensional scaling rarely happens in well-trained neural networks. We compared different methods of calculating CG (including different normalization schemes) empirically in Section 4.2. Thus, in practice if the concept can be accurately modeled by a linear function, CAV is capable of retrieving the concept gradient. However, in general the linear separability assumption does not hold and CG can handle any nonlinear concept.

3.3 Should we explain multiple concepts jointly or individually?

When attributing the prediction to multiple concepts, our flexible framework enables two options: 1) treating each concept independently using single-concept attribution formulation (4) 2) Combining all the concepts together into $c \in \mathbb{R}^m$ and run CG. Intuitively, option 2 takes the correlations between concepts into account while option 1 does not. When both concept A and B are important for prediction but concept A is slightly more important than concept B, option 1 will identify both of them to be important while option 2 may attribute the prediction to concept B only. For example, when $y = x_0 + x_1, c_0 = x_0, c_1 = x_0 + 0.1x_1$, option 1 will identify both concepts to be important, while option 2 will produce negative score for c_0 . Details can be found in the appendix. We further compared these two options on real datasets. The results are presented in Section 4.2. We verified that empirically applying pseudo-inverse individually for each concept is better aligned with the attributes labeled by human.

3.4 Deriving CG

Here we derive CG in the general case when $\nabla g(x)$ is not invertible. For simplicity, we assume $\nabla g(x)$ has full column rank, which implies g is surjective (this is always true when $m = 1$). Otherwise we can directly constraint c within the row space of $\nabla g(x)$ and the arguments below will still go through.

Our goal is to construct a mapping from c to x and analyze the gradient. However, there are infinite many functions from c to x that can locally inverse $g(x)$ since the dimension of concept (m) could be much smaller than input dimension d . Despite infinite number of choices, we show the gradient of such function always follows a particular form:

Theorem 1. Consider a particular point $\hat{x}$ with $\hat{c} = g(\hat{x})$. Let $h : \mathbb{R}^m \rightarrow \mathbb{R}^d$ be a smooth and differentiable function mapping c to x and satisfy $g(h(c)) = c$ locally within the ϵ -ball around $\hat{c}$, then

the gradient of h will take the form of

$$\nabla h(\hat{c}) = \nabla g(\hat{x})^\dagger + G_\perp, \quad (5)$$

where any row vector of $G_\perp$ belongs to $\text{null}(\nabla g(x_0)^T)$ (null space of $\nabla g(x_0)^T$).

The proof of the theorem is deferred to the appendix. Intuitively, this implies the gradient of h will take a particular form in the space of $\nabla g(x)^T$ while being arbitrary in its null space since any change in the null space cannot locally affect c . We can verify that $g(x)$ is locally unchanged in the null space of $\nabla g(x)^T$, as

$$g(x + G_\perp) \approx g(x) + \nabla g(x)^T G_\perp = g(x).$$

This is saying there are multiple choices in Δx to achieve the same affect on c , since any additional perturbation in $\text{null}(\nabla g(x)^T)$ won't make any change to c locally. For example, when we want to change the concept of "color" in an image by Δx , we can have Δx only including the minimal change (e.g., only changing the color), or have Δx including change of color and any arbitrary change to another orthogonal factor (e.g., shape). For concept-based attribution, it is natural to consider the minimal space of x that can cover c , which corresponds to setting $G_\perp$ as 0 in (6).

If we pick such h then $\nabla h(\hat{c}) = \nabla g(\hat{x})^\dagger$, so we can represent y as a function of c locally by $y = p(c) := f(h(c))$ near $\hat{c}$. By chain rule we then have

$$\nabla p(\hat{c}) = \nabla h(\hat{c}) \nabla f(\hat{x}) = \nabla g(\hat{x})^\dagger \nabla f(\hat{x}) = \text{CG}(\hat{x}).$$

In summary, although there are infinitely many functions from c to y since the input space has larger dimension than the concept space, if we consider a subspace in x such that change of x will affect c (locally) and ignore the orthogonal space that is irrelevant to c , then any function mapping from c to y through this space will have gradient equal to Concept Gradient defined in (2).

3.5 Layer selection

The representation of input x is relevant to CG, as the information contained differs between representations. Similar to CAV, CG faces the challenge of properly selecting a layer to perform calculation. For a feed-forward neural network model, information irrelevant to the target task is removed when propagating through the layers. Let us denote the representation of x in the l^{th} layer of the target model f as x_{f_l} . We hypothesized that the optimal layer l^* for performing CG is where the representation $x_{f_{l^*}}$ contains minimally necessary and sufficient information to predict concept labels. Let H denote information entropy and I denote mutual information. Here we overload $\mathcal{X}$ and $\mathcal{C}$ to denote input and concept random variables.

$$l^* = \arg \min_l H(\mathcal{X}_{f_l} | \mathcal{C}) \quad \text{subject to} \quad I(\mathcal{X}_{f_l}; \mathcal{C}) = I(\mathcal{X}; \mathcal{C}).$$

Since $H(\mathcal{X}_{f_l} | \mathcal{C}) = H(\mathcal{X}_{f_l}) - I(\mathcal{X}_{f_l}; \mathcal{C})$ and $H(\mathcal{X}_{f_l}) \geq H(\mathcal{X}_{f_{l+1}})$, l^* corresponds to the latest layer in f where $\mathcal{X}_{f_l}$ can predict $\mathcal{C}$. Intuitively, the representation of x_{f_l} needs to contain sufficient information to correctly predict concepts to ensure the concept gradients $\nabla g(x)$ are accurate. On the other hand, if there is redundant information in x_{f_l} that can be utilized to predict the target y and concept c , then g may not rely on the same information as f , which causes misalignment in gradients $\nabla g(x)$ and $\nabla f(x)$ leading to underestimation of concept importance.

The algorithm for selecting the optimal layer to perform CG is simple. The model g is initialized with weights from f and all the weights are initially frozen. Starting from the last layer, we unfreeze the layer weights and finetune g to predict concepts. We train until the model converges and evaluate the concept prediction accuracy on a holdout validation set. The next step is to unfreeze the previous layer and repeat the whole process until the concept prediction accuracy saturates and no longer improves as more layers are unfrozen. We have then found the optimal layer for CG as well as the concept model g .

4 Experimental Results

We started out with a synthetic example to demonstrate that the linear separability assumption does not hold even in simple cases and CG is superior to CAV. We then benchmarked CG on real world datasets with ground-truth concept labels to show that CG outperforms CAV in accurately attributing concept importance and show some qualitative results.

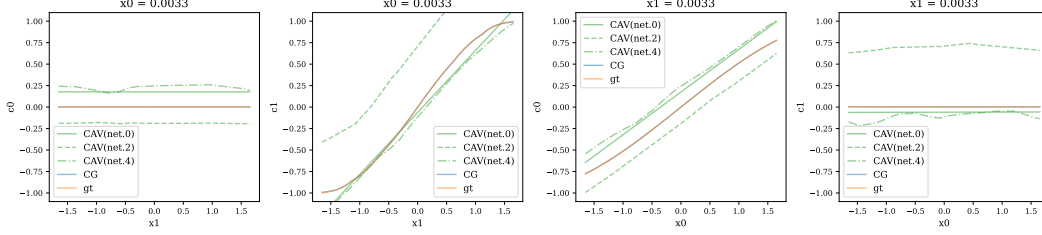


Figure 1: Visualization of concept predictions (from both CAV and CG). CG concept predictions are almost identical to the ground truth while CAV is unable to accurately capture the concept relation.

4.1 Synthetic example

The purpose of this example is to test whether TCAV and CG can recover the actual gradient $\frac{\partial y}{\partial c}$ in a synthetic scenario where the derivative of target y with respect to concept c uniquely exist and can be expressed in closed-form. Theoretically, we have shown CG is capable of exactly recovering the derivative via chain rule in Eq 3. On the other hand, CAV can only retrieve the gradient attribution if the concepts are linearly separable in some input latent representation. We find that the linear separability assumption doesn't hold even in such simple scenarios.

Dataset We define the inputs x_0, x_1 , concepts c_0, c_1 , and target y as follows

$$x_0, x_1 \in [-1.65, 1.65], c_0 = \sin(k_0 \cdot x_0), c_1 = \sin(k_1 \cdot x_1), y = \alpha_0 \cdot c_0 + \alpha_1 \cdot c_1.$$

The coefficients are randomly generated $k_0 = 0.5388, k_1 = 0.9198, \alpha_0 = 0.3633, \alpha_1 = 0.2271$. Clearly, $\frac{\partial y}{\partial c_0} = \alpha_0$ and $\frac{\partial y}{\partial c_1} = \alpha_1$. The derivative of concepts with respect to inputs can also be represented in closed-form expression

$$\frac{\partial c_0}{\partial x_0} = k_0 \cos(k_0 \cdot x_0), \quad \frac{\partial c_0}{\partial x_1} = 0, \quad \frac{\partial c_1}{\partial x_1} = k_1 \cos(k_1 \cdot x_1), \quad \frac{\partial c_1}{\partial x_0} = 0.$$

Training First, we trained a 4-hidden-layer, fully-connected neural network model f that maps (x_0, x_1) to y . This serves as the target model to be interpreted. For CAV, we calculate the CAVs corresponding to the two concepts (c_0, c_1) in the first 3 hidden layers. Note that the last hidden layer cannot be used to calculate CAV otherwise the concept saliency score S would degenerate to a constant for all input (x_0, x_1) . For CG, we trained two 4-hidden-layer, fully-connected neural network models g_0, g_1 that maps (x_0, x_1) to c_0 and c_1 , respectively.

Evaluation Fig 1 visualizes the concept prediction results with CAV and CG. Here we fixed one of the input variables (x_0 or x_1) to a constant value and show the relation between the remaining input variable and the predicted concepts. CG captures the concept relation significantly better than the linear functions of CAV (for all layers). CG concept predictions are almost identical to the ground truth. Next we compare the concept importance attribution between CAV and CG. For CAV, we calculate the concept saliency S as concept importance. For CG, we calculate the concept gradient via chain rule as shown in Eq 3. Recall the ground truth importance attribution for y is α_0, α_1 for c_0, c_1 , respectively, constant for every input sample. The mean square error for the predicted concept importance is $5.64 \times 10^{-2}, 3.47 \times 10^{-2}, 6.63 \times 10^{-2}$, and 3.6×10^{-3} for CAV (net.0), CAV (net.2), CAV (net.4), and CG, respectively. The error of CG is an order less than even the best of the CAVs. Thus, we have shown that CG is capable of capturing the concept relation better, which leads to more accurate gradient estimation and outperforming CAV in concept importance attribution.

4.2 Quantitative analysis

In this experiment, our goal is to quantitatively benchmark how well CG is capable of correctly retrieving relevant concepts in a setting where the ground truth concept importance is available.

Dataset We conducted the experiment on the CUB-200-2011 [40] dataset, a dataset for fine-grained bird image classification. It consists of 11k bird images, 200 bird classes, and 312 binary bird attributes. These attributes are descriptors of bird parts (e.g. bill shape, breast pattern, eye color) that can be used for classification. We followed experimental setting and preprocessing in [25] where class-wise attributed labels are derived from instance-wise attribute labels via majority vote for denoising and sparse attributes are filtered out, which leaves 112 attributes as concepts for interpretation.

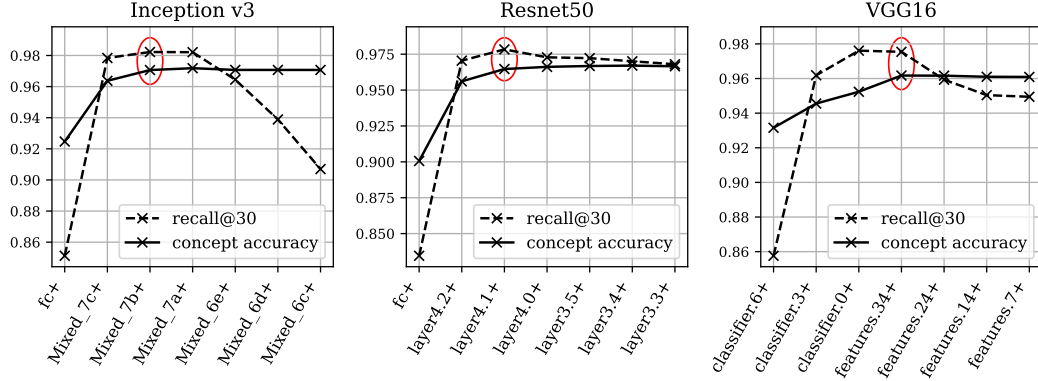


Figure 2: CG concept recall and concept prediction accuracy for different finetuned layers and model architectures on CUB (left to right, deep to shallow layers). The optimal recall generally occurs when concept accuracy plateaus (circled).

Training We first trained models to predict the class labels as the target model f . We then finetuned f to predict concepts labels, freezing some model weights, to serve as the concept model g . The study is conducted on three CNN architectures: Inception v3, Resnet50, and VGG16. We performed extensive study on how finetuning different portions of the model as well as which layer is used for evaluating concept gradient affect CG’s importance attribution. We also performed trained CAVs on different model layers as baselines for comparison. The layer for evaluating CG or CAV defaults to previous layer of finetuning, unless specified otherwise. More training details are in the Appendix.

Evaluation We evaluate the performance of concept importance attribution by measuring the concept recall. Specifically, we treat concept prediction as a multilabel classification problem. For an input instances, there are multiple concepts with positive labels. A good importance attribution method should assign highest concept importance to concepts associated with positive labels. We rank the concepts according to their attributed importance and take the top k to calculate recall@ k . Higher recall implies better alignment between predicted and ground truth concept importance. Table 1 compares the best result of TCAV and CG on all 3 model architectures. CG outperforms TCAV significantly on every model we have experimented with.

For CG, we experimented with finetuning with different portions of the model weights frozen. We plotted the CG concept recalls in Fig 2. The plus sign in x-axis implies all layers after the specified layer are also finetuned. For reference, we also plotted the concept prediction accuracy. The first thing we notice is that as the number of finetuned layers increases, the concept validation accuracy increases until some layer, then plateaus. This is consistent with our analysis in Section 3.5. The mutual information between the representation of $\mathcal{X}_i$ and the concept $\mathcal{C}$ gradually reduces in deeper layers. Therefore as more layers are unfrozen and finetuned, more information can be used to predict concepts. The optimal recall occurs when the concept accuracy plateaus, as predicted in our analysis. The same phenomenon is observed in all 3 architectures: Mixed_7b for Inception v3, layer4.1 for Resnet50, and features.34 for VGG16.

Table 1: Performance comparison on CUB

Model	Method	R@30	R@40	R@50.
Inception v3	TCAV	0.7348	0.8304	0.8950
	CG	0.9822	0.9983	0.9999
Resnet50	TCAV	0.6148	0.7053	0.7785
	CG	0.9783	0.9977	0.9998
VGG16	TCAV	0.7217	0.8149	0.8791
	CG	0.9761	0.9949	0.9984

For CAV, we experimented with different layers of the model. We plotted the CAV concept recalls in Fig 3. Similar to the CG results, the concept accuracy are provided for reference to show how well the CAVs capture the concepts. We observe that the CAVs in later layers perform better in both recall and concept accuracy. The trend is generally monotonic. The accuracy never saturates since linear functions are insufficient to predict the concepts, even for the final layer where the concept is closest to being linearly separable in the representation of x . CAV is equivalent to CG in the final layer since the final layer is a linear layer. Interestingly, the result is the best for CAV, but the *worst* for CG in the final layer. This is also verified in Fig 2 and Fig 3, where the worst result of CG matches the best result of CAV at the final layer. Therefore, the performance of CG dominates that of CAV’s.

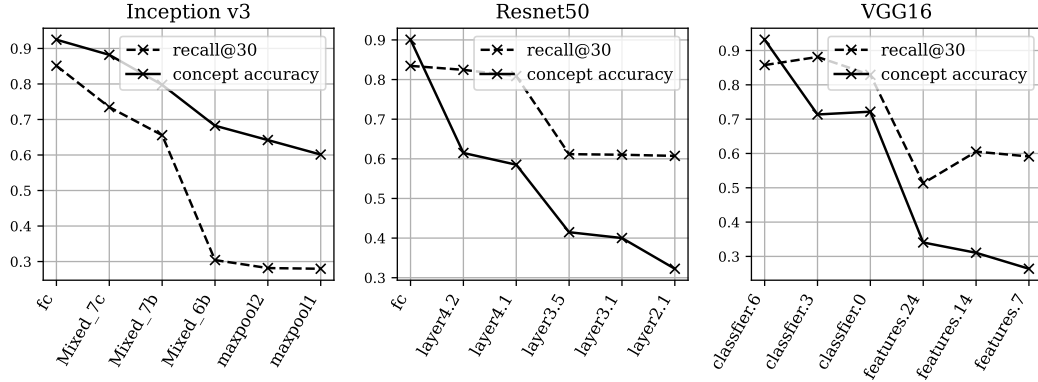


Figure 3: CAV concept recall and prediction accuracy for different selected layers and model architectures on CUB (left to right, deep to shallow layers). The optimal recall occurs when finetuning the final layer which coincides with the highest concept accuracy. Performing CAV in the final layer is equivalent to CG, so best CAV recall in this figure corresponds to the worst CG recall in Fig 2.

Finally, we compare different methods of calculating the concept gradient, including pseudo-inverse jointly with all concepts versus each concept separately as well as various gradient normalization schemes. We ran all experiments on Inception v3 with finetuned layers Mixed_7b+. The results are presented in Table 2. Joint represents the pseudo-inverse being performed on all the concepts jointly while individual represents the pseudo-inverse being performed on each individual concept separately. Un-normalized represents pure inner product between $\nabla f(x)$ and $\nabla g(x)$ without normalization, normed $\nabla g(x)$ with normalized $\nabla g(x)$, and cosine with both normalized $\nabla f(x)$ and $\nabla g(x)$. Recall that individual pseudo-inverse is exactly inner product normalized with squared concept norm. We observe that the concept attribution with joint pseudo-inverse is not aligned with human perception of concept importance while all other methods performs equally well. This supports the argument that the gradient norms in trained neural network are well-behaved and it is unlikely to encounter cases where normalization influences the attribution results significantly.

4.3 Qualitative analysis

The purpose of this experiment is to provide intuition and serve as a sanity check by visualizing instances and how CG works.

Dataset We conducted the experiment on the Animals with Attributes 2 (AwA2) dataset [42], an image classification dataset with 37k animal images, 50 animal classes, and 85 binary attributes for each class. These concepts cover a wide range of semantics, from low level colors and textures, to high level abstract descriptions (e.g. “smart”, “domestic”). We further filtered out 60 concepts that is visible in the input images to perform interpretation.

Evaluation The evaluation is performed on the validation set. Fig 4 visualizes the instances with the highest CG importance attribution for 6 selected concepts, filtering out samples from the same class (top 1 instance in the top 3 classes). The concepts are selected to represent different levels of semantics. The top row contains colors (low-level), the middle row contains textures (medium-level), and the bottom row contains body components (high-level). Observe that CG is capable of handling different levels of semantics simultaneously well, owing to the expressiveness of non-linear concept model g . Additionally, we presented random sampled instances from the validation set and listed top-10 most important concepts as attributed by CG (see the appendix). We intentionally avoided curating the visualization samples to demonstrate true importance attribution performance of CG. The most important concepts for each instance passed the sanity check. There are no contradictory concept-class pairings and importance is attributed to concepts existent in the images.

5 Related work

Our work belongs to post-hoc concept-based explanations. Post-hoc explanations aims to explain a given fixed machine learning model, which can be contrasted with self-interpretable models [6, 8, 27, 41] which train an inherently interpretable model from scratch. Other classes of post-hoc explanations include featured-based explanations [49, 31, 10, 37, 38], counterfactual explanations [11, 18, 39, 16, 20, 32, 19], and sample-based explanations [4, 22, 24, 43, 33, 21]. Our work considers

Table 2: Comparison of different CG calculations

Method	pseudo-inverse		inner product		
	joint	individual	un-normalized	normed $\nabla g(x)$	cosine
recall@30	0.3146	0.9822	0.9818	0.9822	0.9822
recall@40	0.4121	0.9983	0.9983	0.9983	0.9983
recall@50	0.5053	0.9999	0.9983	0.9999	0.9999

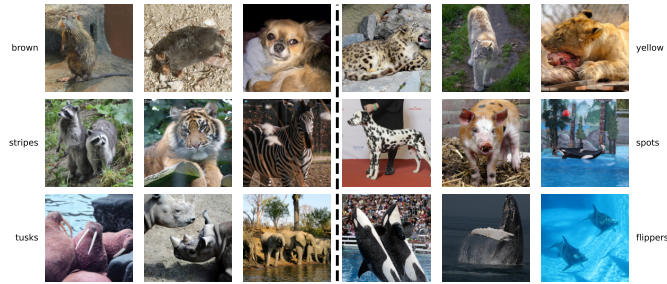


Figure 4: Visualization of instances with highest CG attributed importance (AwA2 validation set) for each concept (top 1 instance in the top 3 classes per concept). CG is capable of handling low level (colors), middle level (textures), and high level (body components) concepts simultaneously.

the gradient from prediction to concepts, which is in spirit connected to feature explanations which considers the gradient from prediction to feature inputs [47, 1].

Concept-based explanations aims to provide human-centered explanations which answer the question “does this human understandable concept relates to the model prediction?” [23, 48]. Some follow-ups for concept-based explanations include when are concept sufficient to explain a model [44], computing interventions on concepts for post-hoc models [15] and self-interpretable models [25], combining concept with other feature attributions [35], unsupervised discovery of concepts [14, 45, 13], and debiasing concepts [2]. The most similar work to ours is the work of [9], which is motivated by the issue of CAV that concept does not necessarily lie in the linear subspace of some activation layer. They address this problem in the self-explainable model setting by training a self-interpretable model and limits the concepts to be whitened. On the other hand, our work address the non-linear concept problem of CAV in the post-hoc setting by learning a non-linear concept component and connects to the activation space via chain rule. As a result, our method can be applied to the setting when the user has no control of the training process of the model. Another work that considers nonlinear modeling of concepts is that of [46]. They adopted a causal-based interpretation instead and consider the co-occurrence of predicted concept and target labels to capture the logical causal relation.

6 Conclusion

We revisited the fundamental assumptions of CAV, one of the most popular concept-based explanation methods. We tackled the problem from a mathematical standpoint and proposed CG to generalize CAV beyond linear functions and corrected normalization term. Empirical experiments demonstrated that CG outperforms CAV in both synthetic and real datasets. Currently CG depends on the representation of input. Devising a input representation invariant method is an interesting future direction.

Acknowledgments and Disclosure of Funding

I would like to thank Karen who supported me and offered deep insight into the study.

References

- [1] Marco Ancona, Enea Ceolini, Cengiz Öztireli, and Markus Gross. Towards better understanding of gradient-based attribution methods for deep neural networks. *arXiv preprint arXiv:1711.06104*, 2017.
- [2] Mohammad Taha Bahadori and David Heckerman. Debiasing concept-based explanations with causal analysis. In *International Conference on Learning Representations*, 2020.
- [3] E. J. Benjamin, P. A. Wolf, R. B. D’Agostino, H. Silbershatz, W. B. Kannel, and D. Levy. Impact of atrial fibrillation on the risk of death: the Framingham Heart Study. *Circulation*, 98(10):946–952, Sep 1998.
- [4] Jacob Bien and Robert Tibshirani. Prototype selection for interpretable classification. *The Annals of Applied Statistics*, pages 2403–2424, 2011.
- [5] J. T. Bigger. Why patients with congestive heart failure die: arrhythmias and sudden cardiac death. *Circulation*, 75(5 Pt 2):28–35, May 1987.

- [6] Diane Bouchacourt and Ludovic Denoyer. Educe: Explaining model decisions through unsupervised concepts extraction. *arXiv preprint arXiv:1905.11852*, 2019.
- [7] W. Bougouin, E. Marijon, E. Puymirat, P. Defaye, D. S. Celermajer, J. Y. Le Heuzey, S. Boveda, S. Kacet, P. Mabo, C. Barnay, A. Da Costa, J. C. Deharo, J. C. Daubert, J. Ferrières, T. Simon, and N. Danchin. Incidence of sudden cardiac death after ventricular fibrillation complicating acute myocardial infarction: a 5-year cause-of-death analysis of the FAST-MI 2005 registry. *Eur Heart J*, 35(2):116–122, Jan 2014.
- [8] Chaofan Chen, Oscar Li, Daniel Tao, Alina Barnett, Cynthia Rudin, and Jonathan K Su. This looks like that: deep learning for interpretable image recognition. In *Advances in Neural Information Processing Systems*, pages 8928–8939, 2019.
- [9] Zhi Chen, Yijie Bei, and Cynthia Rudin. Concept whitening for interpretable image recognition. *Nature Machine Intelligence*, 2(12):772–782, 2020.
- [10] Piotr Dabkowski and Yarin Gal. Real time image saliency for black box classifiers. In *Advances in Neural Information Processing Systems*, pages 6967–6976. NeurIPS, 2017.
- [11] Amit Dhurandhar, Pin-Yu Chen, Ronny Luss, Chun-Chen Tu, Paishun Ting, Karthikeyan Shanmugam, and Payel Das. Explanations based on the missing: Towards contrastive explanations with pertinent negatives. In *Advances in Neural Information Processing Systems*, pages 592–603. NeurIPS, 2018.
- [12] A. Eisen, D. L. Bhatt, P. G. Steg, K. A. Eagle, S. Goto, J. Guo, S. C. Smith, E. M. Ohman, and B. M. Scirica. Angina and Future Cardiovascular Events in Stable Patients With Coronary Artery Disease: Insights From the Reduction of Atherothrombosis for Continued Health (REACH) Registry. *J Am Heart Assoc*, 5(10), 09 2016.
- [13] Asma Ghandeharioun, Been Kim, Chun-Liang Li, Brendan Jou, Brian Eoff, and Rosalind W Picard. Dissect: Disentangled simultaneous explanations via concept traversals. *arXiv preprint arXiv:2105.15164*, 2021.
- [14] Amirata Ghorbani, James Wexler, James Zou, and Been Kim. Towards automatic concept-based explanations. In *Advances in Neural Information Processing Systems*, 2019.
- [15] Yash Goyal, Amir Feder, Uri Shalit, and Been Kim. Explaining classifiers with causal concept effect (cace). *arXiv preprint arXiv:1907.07165*, 2019.
- [16] Yash Goyal, Ziyang Wu, Jan Ernst, Dhruv Batra, Devi Parikh, and Stefan Lee. Counterfactual visual explanations. In *International Conference on Machine Learning*, pages 2376–2384. ICML, 2019.
- [17] P. Harikrishnan, T. Gupta, C. Palaniswamy, D. Kolte, S. Khera, M. Mujib, W. S. Aronow, C. Ahn, S. Sule, D. Jain, A. Ahmed, H. A. Cooper, J. Jacobson, S. Iwai, W. H. Frishman, D. L. Bhatt, G. C. Fonarow, and J. A. Panza. Complete Heart Block Complicating ST-Segment Elevation Myocardial Infarction: Temporal Trends and Association With In-Hospital Outcomes. *JACC Clin Electrophysiol*, 1(6):529–538, Dec 2015.
- [18] Lisa Anne Hendricks, Ronghang Hu, Trevor Darrell, and Zeynep Akata. Grounding visual explanations. In *ECCV. ECCV*, 2018.
- [19] Cheng-Yu Hsieh, Chih-Kuan Yeh, Xuanqing Liu, Pradeep Kumar Ravikumar, Seungyeon Kim, Sanjiv Kumar, and Cho-Jui Hsieh. Evaluations and methods for explanation through robustness analysis. In *International Conference on Learning Representations. ICLR*, 2021. URL <https://openreview.net/forum?id=4dXmpCDGNp7>.
- [20] S. Joshi, O. Koyejo, Warut D. Vijitbenjaronk, Been Kim, and Joydeep Ghosh. Towards realistic individual recourse and actionable explanations in black-box decision making systems. *ArXiv*, abs/1907.09615, 2019.
- [21] Rajiv Khanna, Been Kim, Joydeep Ghosh, and Oluwasanmi Koyejo. Interpreting black box predictions using fisher kernels. *arXiv preprint arXiv:1810.10118*, pages 3382–3390, 2018.

- [22] Been Kim, Rajiv Khanna, and Oluwasanmi O Koyejo. Examples are not enough, learn to criticize! criticism for interpretability. In *Advances in Neural Information Processing Systems*, pages 2280–2288. NeurIPS, 2016.
- [23] 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 *International Conference on Machine Learning*, pages 2673–2682. ICML, 2018.
- [24] Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *International Conference on Machine Learning*, pages 1885–1894. ICML, 2017.
- [25] Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang. Concept bottleneck models. *arXiv preprint arXiv:2007.04612*, 2020.
- [26] B. A. Koplan and W. G. Stevenson. Ventricular tachycardia and sudden cardiac death. *Mayo Clin Proc*, 84(3):289–297, Mar 2009.
- [27] Guang-He Lee, Wengong Jin, David Alvarez-Melis, and Tommi Jaakkola. Functional transparency for structured data: a game-theoretic approach. In *International Conference on Machine Learning*, pages 3723–3733. PMLR, 2019.
- [28] Ari D Leib, Lisa A Foris, Tran Nguyen, and Karam Khaddour. Dressler syndrome. 2017.
- [29] C. Medi, J. M. Kalman, and S. B. Freedman. Supraventricular tachycardia. *Med J Aust*, 190(5): 255–260, Mar 2009.
- [30] S. Orn, J. G. Cleland, M. Romo, J. Kjekshus, and K. Dickstein. Recurrent infarction causes the most deaths following myocardial infarction with left ventricular dysfunction. *Am J Med*, 118 (7):752–758, Jul 2005.
- [31] Vitali Petsiuk, Abir Das, and Kate Saenko. Rise: Randomized input sampling for explanation of black-box models. *arXiv preprint arXiv:1806.07421*, 2018.
- [32] Rafael Poyiadzi, Kacper Sokol, Raúl Santos-Rodríguez, T. D. Bie, and Peter A. Flach. Face: Feasible and actionable counterfactual explanations. *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, 2020.
- [33] Garima Pruthi, Frederick Liu, Satyen Kale, and Mukund Sundararajan. Estimating training data influence by tracing gradient descent. *Advances in Neural Information Processing Systems*, 33, 2020.
- [34] A. Roguin, D. Behar, H. Ben Ami, S. A. Reisner, S. Edelstein, S. Linn, and Y. Edoute. Long-term prognosis of acute pulmonary oedema—an ominous outcome. *Eur J Heart Fail*, 2(2): 137–144, Jun 2000.
- [35] Jessica Schrouff, Sebastien Baur, Shaobo Hou, Diana Mincu, Eric Loreaux, Ralph Blanes, James Wexler, Alan Karthikesalingam, and Been Kim. Best of both worlds: local and global explanations with human-understandable concepts. *arXiv preprint arXiv:2106.08641*, 2021.
- [36] F. Shamshad, S. Kenchaiah, P. V. Finn, J. Soler-Soler, J. J. McMurray, E. J. Velazquez, A. P. Maggioni, R. M. Califf, K. Swedberg, L. Kober, Y. Belenkov, S. Varshavsky, M. A. Pfeffer, and S. D. Solomon. Fatal myocardial rupture after acute myocardial infarction complicated by heart failure, left ventricular dysfunction, or both: the VALsartan In Acute myocardial iNfarcTion Trial (VALIANT). *Am Heart J*, 160(1):145–151, Jul 2010.
- [37] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. *International Conference on Machine Learning*, 2017.
- [38] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *International Conference on Machine Learning*, pages 3319–3328. PMLR, 2017.
- [39] Jasper van der Waa, Marcel Robeer, Jurriaan van Diggelen, Matthieu Brinkhuis, and Mark Neerincx. Contrastive Explanations with Local Foil Trees. In *2018 Workshop on Human Interpretability in Machine Learning (WHI)*. WHI, 2018.

- [40] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The caltech-ucsd birds-200-2011 dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011.
- [41] Fulton Wang and Cynthia Rudin. Falling rule lists. In *Artificial Intelligence and Statistics*, pages 1013–1022, 2015.
- [42] Yongqin Xian, Christoph H Lampert, Bernt Schiele, and Zeynep Akata. Zero-shot learning—a comprehensive evaluation of the good, the bad and the ugly. *IEEE transactions on pattern analysis and machine intelligence*, 41(9):2251–2265, 2018.
- [43] Chih-Kuan Yeh, Joon Kim, Ian En-Hsu Yen, and Pradeep K Ravikumar. Representer point selection for explaining deep neural networks. In *Advances in Neural Information Processing Systems*, pages 9291–9301. NeurIPS, 2018.
- [44] 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*, volume abs/1901.09392, pages 10965–10976, 2019.
- [45] Chih-Kuan Yeh, Been Kim, Serkan 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.
- [46] Mohammad Nokhbeh Zaeem and Majid Komeili. Cause and effect: Concept-based explanation of neural networks. In *2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 2730–2736. IEEE, 2021.
- [47] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.
- [48] Bolei Zhou, Yiyou Sun, David Bau, and Antonio Torralba. Interpretable basis decomposition for visual explanation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 119–134. ECCV, 2018.
- [49] Luisa M Zintgraf, Taco S Cohen, Tameem Adel, and Max Welling. Visualizing deep neural network decisions: Prediction difference analysis. *arXiv preprint arXiv:1702.04595*, 2017.

A Limitations

CG is a gradient-based interpretation methods, which is can only be applied to differentiable white-box models. Gradient-based methods convey how small changes in the input affect the output via gradients. Larger changes in the input requires intervention-based causal analysis to predict how the output is affected. As a modification upon TCAV, CG also requires users to specify concepts with sufficient representative data samples or close-form concept functions. Sufficient amount of data is more important for CG to prevent the nonlinear function to overfit. Automatic discovery of novel concepts requires introducing other tools. Finally, CG requires fitting non-linear concept models if the concept is provided in the form of representative data samples. This might be computationally intensive for complex non-linear models (e.g. neural networks). The quality of interpretation highly depends on how accurately the concept is captured by the concept model.

B Example for demonstrating different CG calculation methods

We use a simple toy example to explain the difference between option 1 and 2. Assume $y = x_0 + x_1$ and two concepts $c_0 = x_0, c_1 = x_0 + 0.1x_1$. Since these relationships are all linear, concept gradient will be invariant to reference points. The results computed by option 1 and 2 are

$$\begin{aligned} \text{Option 1 (individually apply (4))}: R_{c_0,y} &= 1, R_{c_1,y} \approx 1.01 \\ \text{Option 2 (apply CG jointly by (2))}: R_{c_0,y} &= -9, R_{c_1,y} = 10 \end{aligned}$$

At the first glance, it is weird why the contribution of c_0 is negative to y with option 2. But in fact there exists a unique function $y = -9c_0 + 10c_1$ mapping c to y (when jointly considering two concepts), which leads to negative contribution of c_0 . This shows that when considering two concepts together as a joint function, the gradient is trying to capture the effect of one concept with respect to others, which may be non-intuitive to human.

C Proof

Theorem 2. Let $h : \mathbb{R}^m \rightarrow \mathbb{R}^d$ be a smooth and differentiable function mapping c to x and satisfy $g(h(c)) = c$ locally within an ϵ -ball around c_0 . Then the gradient of h will take the form of

$$\nabla h(c_0) = \nabla g(x_0)^\dagger + g_\perp, \quad (6)$$

where $g_\perp$ is in the null space of $\nabla g(x_0)^T$.

Proof. Assume $h : \mathbb{R}^m \rightarrow \mathbb{R}^d$ is a function mapping c to x . By Taylor expansion we have

$$g(x') = g(x) + \nabla g(x)^T(x' - x) + O(\|x' - x\|^2) \quad (7)$$

$$h(c') = h(c) + \nabla h(c)^T(c' - c) + O(\|c' - c\|^2), \quad (8)$$

where $x' = h(c')$ and $x = h(c)$. Therefore

$$\begin{aligned} g(x') &= g(x) + \nabla g(x)^T(h(c') - h(c)) + O(\|h(c') - h(c)\|^2) \\ &= g(x) + \nabla g(x)^T \nabla h(c)^T(c' - c) + O(\|\nabla g(x)\|^2 \|c' - c\|^2) + O(\|h(c') - h(c)\|^2). \end{aligned}$$

The last two terms are both $O(\|c - c'\|^2)$ since $\nabla g(x)$ is a constant and h is smooth (thus locally Lipschitz), so

$$c' - c = \nabla g(x)^T \nabla h(c)^T(c' - c) + O(\|c - c'\|^2). \quad (9)$$

Therefore, $\nabla g(x)^T \nabla h(c)^T = I$, which means $\nabla h(c) = \nabla g(x)^\dagger + g_\perp$. $\square$

C.1 Definition of Concept Gradient

Given input space $\mathcal{X}$ and two differentiable functions $f : \mathcal{X} \rightarrow \mathcal{Y}$ and $g : \mathcal{X} \rightarrow \mathcal{C}$, where $\mathcal{Y}$ is the target label space and $\mathcal{C}$ is the concept label space. We define the concept gradient of $x \in \mathcal{X}$ to attribute the prediction of the model to the concepts:

$$\text{CG}(x) := \nabla f(x) \cdot \nabla g(x)^\dagger, \quad (10)$$

where $\nabla g(x)^\dagger$ denotes the pseudo-inverse of $\nabla g(x)$. Let $y = f(x) \in \mathcal{Y}$ and $c = g(x) \in \mathcal{C}$. Essentially concept gradient approximates gradient-based saliency of y to c , $h'(c)$, via chain rule. For the case where $\mathcal{X}$, $\mathcal{Y}$, and $\mathcal{C}$ are all scalar fields, the concept gradient exactly recovers $h'(c)$ if $g'(x) \neq 0$ for all $x \in \mathcal{X}$,

Given input space $\mathcal{X}$ and two differentiable functions $f : \mathcal{X} \rightarrow \mathcal{Y}$ and $g : \mathcal{X} \rightarrow \mathcal{C}$, where $\mathcal{Y}$ is the target label space and $\mathcal{C}$ is the concept label space. Suppose there exist an unknown differentiable function $h : \mathcal{C} \rightarrow \mathcal{Y}$ s.t. $f = h \circ g$. Let f' , g' , and h' denote the first-order derivatives. We define the concept gradient of $x \in \mathcal{X}$ as

$$\text{CG}(x) := \nabla f(x) \cdot \nabla g(x)^\dagger$$

where $\nabla g(x)^\dagger$ denotes the pseudo-inverse of $\nabla g(x)$. Let $y = f(x) \in \mathcal{Y}$ and $c = g(x) \in \mathcal{C}$. Essentially concept gradient approximates gradient-based saliency of y to c , $h'(c)$, via chain rule. For the case where $\mathcal{X}$, $\mathcal{Y}$, and $\mathcal{C}$ are all scalar fields, the concept gradient exactly recovers $h'(c)$ if $g'(x) \neq 0$ for all $x \in \mathcal{X}$,

$$h'(c) = \frac{dy}{dc} = \frac{dy}{dx} \cdot \frac{dx}{dc} = \frac{dy}{dx} \cdot \left(\frac{dc}{dx}\right)^{-1} = f'(x) \cdot \frac{1}{g'(x)} = \text{CG}(x)$$

We now generalize $\mathcal{X}$ to a n-dimensional vector space. Since f , g , and h are differentiable, we can perform Taylor expansion around x and c

$$f(x') = f(x) + \nabla f(x)(x' - x) + O((x' - x)^2) \quad (11)$$

$$g(x') = g(x) + \nabla g(x)(x' - x) + O((x' - x)^2) \quad (12)$$

$$h(c') = h(c) + h'(c)(c' - c) + O((c' - c)^2) \quad (13)$$

Let us denote $\Delta x = x' - x$. We can plug Eq 12 into Eq 13

$$\begin{aligned} h(g(x')) - h(c) &= h'(c)(g(x') - c) + O((g(x') - c)^2) \\ &= h'(c) \cdot \left(g(x) + \nabla g(x)\Delta x + O(\Delta x^2) - c \right) + O((g(x') - c)^2) \\ &= h'(c) \cdot \left(\nabla g(x)\Delta x + O(\Delta x^2) \right) + O((g(x') - c)^2) \\ &\approx h'(c) \cdot \left(\nabla g(x)\Delta x \right) \end{aligned}$$

$$\begin{aligned} f(x') - f(x) &= \nabla f(x)\Delta x + O(\Delta x^2) \\ &\approx \nabla f(x)\Delta x \end{aligned}$$

$$\begin{aligned} h(g(x')) - h(c) &= f(x') - f(x) \\ h'(c) \cdot \left(\nabla g(x)\Delta x \right) &\approx \nabla f(x)\Delta x \end{aligned}$$

Let us denote the set of right inverses for $\nabla g(x)$ as $G_r^{-1}(x)$.

$$G_r^{-1}(x) = \{\nabla g(x)^\dagger + g_\perp^T, \forall g_\perp : \langle \nabla g(x), g_\perp \rangle = 0\}$$

By definition for all $g_r^{-1} \in G_r^{-1}(x)$,

$$h'(c) \cdot \Delta x \approx \nabla f(x) \cdot g_r^{-1} \cdot \Delta x$$

If $\nabla g(x)$ is invertible, $G_r^{-1}(x) = \{\nabla g(x)^\dagger\}$ and the equality exactly holds

$$h'(c) = \nabla f(x) \cdot \nabla g(x)^\dagger = \nabla f(x) \cdot \nabla g(x)^{-1}$$

If $\nabla g(x)$ is not invertible, g_r^{-1} is not unique and infinitely many right inverses exist for $\nabla g(x)$. In this case, how does the selection of right inverse relate to interpretation? For interpretation, the goal is to attribute a small change Δc via g_r^{-1} to a small change Δx . Non-invertibility of $\nabla g(x)$ implies there are many ways we could perform the attribution. Consider the alignment between some $g_r^{-1} = \nabla g(x)^\dagger + g_\perp^T$ and $\nabla g(x)$,

$$\frac{\nabla g(x) \cdot (\nabla g(x)^\dagger + g_\perp^T)}{\|\nabla g(x)\| \cdot \|\nabla g(x)^\dagger + g_\perp^T\|} = \frac{\nabla g(x) \cdot \nabla g(x)^\dagger}{\|\nabla g(x)\| \cdot \|\nabla g(x)^\dagger + g_\perp^T\|} \leq \frac{\nabla g(x) \cdot \nabla g(x)^\dagger}{\|\nabla g(x)\| \cdot \|\nabla g(x)^\dagger\|}$$

We argue that the best interpretation is when the attribution is faithful to the relation g between $\mathcal{X}$ and $\mathcal{C}$, i.e., when g_r^{-1} is best aligned with $\nabla g(x)$. Observe that $\nabla g(x)^\dagger$ is in the same direction as $\nabla g(x)$. On the other hand, any right inverse with $g_\perp \neq 0$ is attributing *some* proportion of dc to dx in a direction that is orthogonal to $\nabla g(x)$. Thus, the best choice for g_r^{-1} is $\nabla g(x)^\dagger$.

C.2 Connection with CAV

CAV is a special case of concept gradient with g restricted to linear functions. Let $\mathbf{v}_c$ denote the concept activation vector associated with concept c . CAV defines the conceptual sensitivity S as the inner product of the input gradient and concept activation vector,

$$S(x) := \nabla f(x) \cdot \frac{\mathbf{v}_c}{\|\mathbf{v}_c\|}$$

If g is restricted to linear functions,

$$g(x) = \mathbf{v}_c^T \cdot x + b_c$$

for some constant bias b_c . Concept gradient is equivalent to CAV conceptual sensitivity normalized by the norm of the concept activation vector,

$$\text{CG}(x) = \nabla f(x) \cdot \nabla g(x)^\dagger = \nabla f(x) \cdot (\mathbf{v}_c^T)^\dagger = \nabla f(x) \cdot \frac{\mathbf{v}_c}{\|\mathbf{v}_c\|^2} = \frac{S(x)}{\|\mathbf{v}_c\|}$$

Thus, if the concept can be accurately modeled by a linear function, CAV is capable of retrieving the concept gradient. However, in general the linear separability assumption does not hold. In contrast, concept gradient considers general function classes for g , which better captures the relation between $\mathcal{X}$ and $\mathcal{C}$. Given accurately modeling the concept with g is a necessary condition for correct interpretation, concept gradient is superior to CAV.

D Alternative perspective for layer selection

We can also view layer selection in a typical bias-variance tradeoff perspective. If we selected a later layer to evaluate CG, we are biased towards using a representation of x that is optimized for predicting the target y , not c . However, since the information consists in the representation is less, we also enjoy the benefit of less variance. On the other hand, if we selected an earlier layer to evaluate CG, then we suffer less from the bias (towards y) but is penalized with higher variance due to abundance of information. The optimal layer is where the representation of x is capable of predicting the concept (minimized bias) while no redundant information is available (minimized variance).

We verified the bias-variance tradeoff hypothesis with experiments. More bias (with respect to target labels) in later layers is confirmed with the observation that finetuning more layers yields higher concept prediction accuracy (see Fig 2). Less variance in later layers is confirmed with the experiment below. We repeated the CUB experiments on the Inception v3 model with 5 different random seeds and evaluated the variance of the gradient $\nabla g(x)$ over repeated trials, averaged over all data points. Specifically, the gradients for models finetuned starting from different layers are evaluated on the same layer (Mixed_6d) for fair comparison. The results are shown in Fig 5 and confirmed the variance hypothesis.

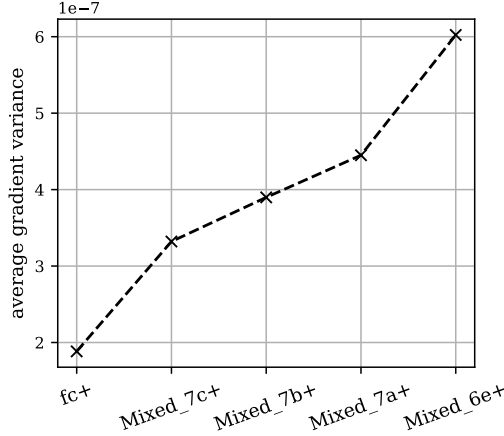


Figure 5: Variance of gradients finetuning starting from different layers. The variance is higher when finetuning starting from earlier layers.

E Experiments on the Myocardial Infarction Complications Database

To verify whether CG is effective for critical applications (e.g. medical domain) with non-natural-image data, we conducted experiment on the Myocardial infarction complications database. The database consists of 1,700 entries of patient data with 112 fields of input tabular features, 11 complication fields for prediction, and 1 field of lethal outcome also for prediction. The input features are measurements taken when the patients were admitted to the hospital due to myocardial infarction as well as past medical records. The target models predicts lethal outcome given the 112 input fields. The concept models predict the complications given the same 112 input fields. Our goal is to interpret the lethal outcome with the complications and compare our interpretation with existing literature regarding how each complication affects the risk of death. We expect good interpretations to assign high relevance to complications that poses the highest mortality risk.

Table 3 shows the CG interpretation scores as well as the excerpted description of the complication mortality risk in existing medical literature. The severity of descriptions in medical literature are largely aligned with the CG scores. Highest risk complications are attributed the most importance (e.g. relapse of myocardial infarction, chronic heart failure) while the lower risk complications are attributed the least importance (e.g. supraventricular tachycardia, post-infarction angina). This supports CG as an effective method for real-life practical use for interpreting models in critical domains.

F Comparison with other nonlinear concept modeling interpretation methods

[46] proposed a causal-based interpretation method which also considers nonlinear modeling of concepts. Different from our gradient-based perspective, they directly utilize the concept model prediction output and evaluate the co-occurrence of concept and target labels as a form of causal relation. Their interpretation evaluates whether concepts are necessary or sufficient conditions for the target function. Although their method is only loosely related to CG, we included the comparison of performance on the CUB experiment for curious readers that might wonder how different approaches of utilizing the nonlinear concept model might affect the interpretation quality.

We benchmarked on the CUB experiment setting in Section 4.2 focusing on the Inception v3 model. The experiment results presented in Table 4 suggest interpreting concepts as necessary conditions perform much better sufficient conditions. This is expected since no concept in the CUB dataset alone is sufficient to predict

Table 4: Performance comparison on CUB

Method	R@30	R@40	R@50.
TCAV	0.7348	0.8304	0.8950
CG	0.9822	0.9983	0.9999
Sufficient [46]	0.0994	0.1488	0.2083
Necessary [46]	0.5451	0.6436	0.7243

Table 3: Mortality risk attribution with respect to myocardial infarction complications and comparison with existing medical literature

Complication	CG	Excerpted mortality risk description
Relapse of myocardial infarction	3.466	Recurrent infarction causes the most deaths following myocardial infarction with left ventricular dysfunction. [30]
Chronic heart failure	3.265	Patients with congestive heart failure have a high incidence of sudden cardiac death that is attributed to ventricular arrhythmias. The mortality rate in a group of patients with class III and IV heart failure is about 40% per year, and half of the deaths are sudden. [5]
Atrial fibrillation	2.285	AF increases the risk of death by 1.5-fold in men and 1.9-fold in women. [3]
Myocardial rupture	1.617	Myocardial rupture is a relatively rare and usually fatal complication of myocardial infarction (MI). [36]
Pulmonary edema	1.505	Pulmonary oedema in patients with acute MI hospitalized in coronary care units was reported to be associated with a high mortality of 38–57%. [34]
Ventricular fibrillation	0.910	Patients developing VF in the setting of acute MI are at higher risk of in-hospital mortality. [7]
Third-degree AV block	0.691	In patients with CHB complicating STEMI, there was no change in risk-adjusted in-hospital mortality during the study period. [17]
Ventricular tachycardia	0.514	The risk and consequently the therapeutic approach are determined by the underlying heart disease. Ventricular tachycardia is most commonly associated with ischemic heart disease or other forms of structural heart disease that are associated with a risk of sudden death. [26]
Dressler syndrome	0.316	The prognosis for patients with DS is typically considered to be quite good. [28]
Supraventricular tachycardia	0.236	Although SVT is usually not life-threatening, many patients suffer recurrent symptoms that have a major impact on their quality of life. [29]
Post-infarction angina	-1.404	After adjustment, angina was only weakly associated with cardiovascular death, myocardial infarction, or stroke, but significantly associated with heart failure, cardiovascular hospitalization, and coronary revascularization. [12]

the target bird class. In general, [46]’s method performs poorly on retrieving relevant concepts relative to TCAV and even worse when compared with CG. Interestingly, they benchmarked on a synthetic dataset where captions are artificially added to natural images to serve as concepts and demonstrated clear causal relations with their proposed method in their original paper. We suspect causal relations may not be well captured by co-occurrence of concept and target labels alone in real world datasets.

G Ablation study on concept model configuration

Experiments in Section 4.2 and 4.3 shares the same model architecture for the target and concept model. The two design choices here are 1) using the same model architecture and 2) warm-starting the training of concept models with target model weights. The choices are rather straightforward since both models need to share the same input feature representation for the gradients with respect to the input layer to be meaningful. Nevertheless, we conducted an ablation study on the CUB experiment to verify how using different model architectures different weight initialization for the concept model affects interpretation. In this case, CG needs to be evaluated in the input layer, the only layer where the feature representation is shared between target and concept models.

The model architecture alablton study results are presented in Table 5. The CG scores are all evaluated in the input layer. Evidently, using the same model architecture for both the target and concept models is crucial for good interpretation quality with CG. The degradation of interpretation when using different architectures may be caused by mismatched usage of input feature representation between models. The concept model weight initialization results are presented in Table 6. Using target model weights as initialization outperforms using ImageNet pretrained weights significantly. The more similar the pretrained task is to the downstream task is, the better the finetuned performance. In this case, the concept prediction accuracy suggests the target model task of bird classification is a better pretraining task for predicting bird concepts, allowing the concepts to be better captured by the concept function (higher accuracy). This naturally leads to better interpretation results.

Table 5: Ablation study on concept model architecture

Target model	Concept model	R@30	R@40	R@50.
Inception v3	Inception v3	0.8716	0.9310	0.9570
Inception v3	Resnet50	0.5352	0.6201	0.6835
Inception v3	VGG16	0.4860	0.5609	0.6194
Resnet50	Resnet50	0.9329	0.9755	0.9881
Resnet50	Inception v3	0.6233	0.7051	0.7569
Resnet50	VGG16	0.5542	0.6312	0.6916
VGG16	VGG16	0.9549	0.9856	0.9941
VGG16	Inception v3	0.5254	0.6060	0.6635
VGG16	Resnet50	0.6094	0.6984	0.7582

Table 6: Ablation study on concept model weight initialization

Weight initialization	Concept accuracy	R@30	R@40	R@50.
ImageNet pretrained	0.9161	0.5771	0.6701	0.7394
target model pretrained	0.9724	0.8716	0.9310	0.9570

H Experiment details

H.1 Animal with Attributes 2 (AwA2)

Data preprocessing Since the original task is proposed for zero-shot classification, the class labels in the default training and validation set is disjoint. To construct a typical classification task, we combined all data together then performed a 80:20 split for the new training and validation set. During training, the input images are augmented by random color jittering, horizontal flipping, and resizing, then cropped to the default input resolution of the model architecture (299 for Inception v3, 224 for others). During evaluation, the input images are resized and center cropped to the input resolution.

The attribute labels provided in the dataset contains both concrete and abstract concepts. Some abstract concepts cannot be identified merely by looking at the input image (e.g. new world vs old world). We filtered out 25 attributes that are not identifiable via the input image and used the remaining 60 attributes for interpretation.

Training We trained the target model f with Inception v3 architecture with ImageNet pretrained weights (excluding the final fully connected layer). We optimized with Adam (learning rate of 0.001, beta1 of 0.9, and beta2 of 0.999) with weight decay 0.0004 and schedule the learning rate decay by 0.1 every 15 epochs until the learning rate reaches 0.00001. We trained the model for a maximum of 200 epochs and early stopped if the validation accuracy had not improved for 50 consecutive epochs. The validation accuracy of the trained target model is 0.947.

We trained the concept model g by finetuning different parts of f (freezing different layer model weights). We reweighted the loss for positive and negative class to balance class proportions. We optimized with Adam (learning rate of 0.01, beta1 of 0.9, and beta2 of 0.999) with weight decay 0.00004 and schedule the learning rate decay by 0.1 every 25 epochs until the learning rate reaches 0.00001. We trained the model for a maximum of 200 epochs and early stopped if the validation accuracy had not improved for 50 consecutive epochs.

Evaluation Visualization is conducted on the validation set. Finetuning from Mixed_7a is the latest layer that still predicted the concepts well. According to the layer selection guideline, we selected Mixed_7a to evaluate CG. We computed CG with the individual inverse method (4). Fig 6 shows random samples from the validation set and their top 10 rated concepts. These samples are intentionally randomly sampled as opposed to intentionally curated to provide an intuition of the true effectiveness of CG. In general, the retrieved highest ranked concepts are relevant with the input image. In terms of sanity check, there are no contradictions in the concepts (e.g. furry is never assigned to an whales).



Figure 6: Visualization of randomly sampled instances (AwA2 validation set) and the most important concepts associated with their respective CG attribution (top 10 concept, 60 in total).

Table 7: Finetune layers for architectures

Architecture	Finetuned layers
Inception v3	fc+, Mixed_7c+, Mixed_7b+, Mixed_7a+, Mixed_6e+, Mixed_6d+, Mixed_6c+
Resnet50	fc+, layer4.2+, layer4.1+, layer4.0+, layer3.5+, layer3.4+, layer3.3+
VGG16	classifier.6+, classifier.3+, classifier.0+, features.34+, features.24+, features.14+, features.7+

H.2 Caltech-UCSD Birds-200-2011 (CUB)

Data preprocessing Similar to the AWA2 experiments, the input images are augmented by random color jittering, horizontal flipping, and resizing, then cropped to the default input resolution of the model architecture (299 for Inception v3, 224 for others) during training. During evaluation, the input images are resized and center cropped to the input resolution. We followed [25] procedure of denoising the attribute labels by majority voting for the class attribute label and removing attributes with insufficient data samples. Denoising is necessary since the annotations are collected from Amazon Mechanical Turk and contradicting labels are not uncommon. A total of 112 attributes remains for conducting interpretation. The class attribute labels are assigned as instance labels, i.e., instances from the same class share the same attribute labels.

Training We trained the target model f with three different CNN architectures: Inception v3, Resnet50, and VGG16 (with batch normalization), each with ImageNet pretrained weights (excluding the final fully connected layer). We searched for hyperparameters over a range of learning rates ($[0.01, 0.001]$), learning rate schedules (decaying by 0.1 for every $[15, 20, 25]$ epochs until reaching 0.0001), and weight decay ($[0.0004, 0.00004]$). We optimized with the SGD optimizer. We trained the model for a maximum of 200 epochs and early stopped if the validation accuracy had not improved for 50 consecutive epochs. The validation accuracy of the trained target model is 0.797, 0.764, and 0.782 for the three models, respectively.

We trained the concept model g by finetuning different parts of f (freezing different layer model weights). The different layers we started to finetune from for each model architecture is listed in Table 7. The plus sign in the table represents all layers after the specified layer are all finetuned while all layers prior to the specified layer have their weights kept frozen. We reweighted the loss for positive and negative class to balance class proportions. We searched for the hyperparameters and trained the model same as the target model f .

Evaluation Evaluation is conducted on the testing set. CG and CAV are evaluated in the layer prior to finetuning. We evaluated the global recalls for $k = 30, 40, 50$ and reported for $k = 30$ as the recall trend is monotonic for other thresholds. These thresholds are chosen since the number of concepts with positive labels for each instance is in the range of 30 to 40.