i-Algebra: Towards Interactive Interpretability of Deep Neural Networks

Xinyang Zhang,[1] Ren Pang,[1] Shouling Ji,[2] Fenglong Ma,[1] Ting Wang[1]
[1]Pennsylvania State University,
[2]Zhejiang University
{xqz5366, rbp5354, fenglong, ting}@psu.edu, sji@zju.edu.cn

Abstract

Providing explanations for deep neural networks (DNNs) is essential for their use in domains wherein the interpretability of decisions is a critical prerequisite. Despite the plethora of work on interpreting DNNs, most existing solutions offer interpretability in an ad hoc, one-shot, and static manner, without accounting for the perception, understanding, or response of end-users, resulting in their poor usability in practice.

In this paper, we argue that DNN interpretability should be implemented as the interactions between users and models. We present i-Algebra, a first-of-its-kind interactive framework for interpreting DNNs. At its core is a library of atomic, composable operators, which explain model behaviors at varying input granularity, during different inference stages, and from distinct interpretation perspectives. Leveraging a declarative query language, users are enabled to build various analysis tools (e.g., "drill-down", "comparative", "what-if" analysis) via flexibly composing such operators. We prototype i-Algebra and conduct user studies in a set of representative analysis tasks, including inspecting adversarial inputs, resolving model inconsistency, and cleansing contaminated data, all demonstrating its promising usability.

Introduction

The recent advances in deep learning have led to breakthroughs in a number of long-standing artificial intelligence tasks, enabling use cases previously considered strictly experimental. Yet, the state-of-the-art performance of deep neural networks (DNNs) is often achieved at the cost of their *interpretability*: it is challenging to understand how a DNN arrives at a particular decision, due to its high non-linearity and nested structure (Goodfellow, Bengio, and Courville 2016). This is a major drawback for domains wherein the interpretability of decisions is a critical prerequisite.

A flurry of interpretation methods (Sundararajan, Taly, and Yan 2017; Dabkowski and Gal 2017; Fong and Vedaldi 2017; Zhang, Nian Wu, and Zhu 2018) have since been proposed to help understand the inner workings of DNNs. Typically, a DNN (classifier), coupled with an interpretation model (interpreter), forms an interpretable deep learning system (IDLS), which is believed to improve the model trustworthiness (Tao et al. 2018; Guo et al. 2018).

Yet, despite the enhanced interpretability, today's IDLSes are still far from being practically useful. In particular, most IDLSes provide interpretability in an ad hoc, single-shot, and static manner, without accounting for the perception, understanding, and response of the users, resulting in their poor usability in practice. For instance, most IDLSes generate a static saliency map to highlight the most informative features of a given input; however, in concrete analysis tasks, the users often desire to know more, for instance,

- How does the feature importance change if some other features are present/absent?
- How does the feature importance evolve over different stages of the DNN model?
- What are the common features of two inputs that lead to their similar predictions?
- What are the discriminative features of two inputs that result in their different predictions?

Moreover, the answer to one question may trigger followup questions from the user, giving rise to an interactive process. Unfortunately, the existing IDLSes, limited by their non-interactive designs, fail to provide interpretability tailored to the needs of individual users.

Our Work – To bridge the striking gap, we present i-Algebra, a first-of-its-kind interactive framework for interpreting DNN models, which allows non-expert users to easily explore a variety of interpretation operations and perform interactive analyses. The overall design goal of i-Algebra is to implement the DNN interpretability as the interactions between the user and the model.

Specifically, to accommodate a range of user preferences for interactive modes with respect to different DNN models and analysis tasks, we design an expressive algebraic framework, as shown in Figure 1. Its fundamental building blocks are a library of atomic operators, which essentially produce DNN interpretability at varying input granularity, during different model stages, and from complementary inference perspectives. On top of this library, we define a SQL-like declarative query language, which allows users to flexibly compose the atomic operators and construct a variety of analysis tasks (e.g., "drill-down," "what-if," "comparative" analyses). As a concrete example, given two inputs x and x', the query below compares their interpretation at the l-th

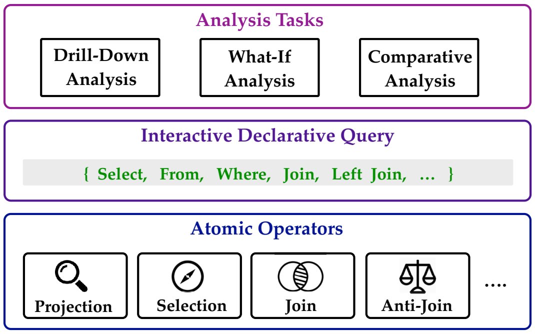

Figure 1: An interactive interpretation framework.

layer of the DNN f and finds the most discriminative features of x and x' with respect to their predictions.

```
select l from f(x) left join (select l from f(x'))
```

We prototype i-Algebra and evaluate its usability in three representative tasks: resolving model inconsistency, inspecting adversarial inputs, and cleansing contaminated data. The studies conducted on Amazon MTurk show that compared with the conventional interpretability paradigm, i-Algebra significantly improves the analysts' performance. For example, in the task of resolving model inconsistency, we observe over 30% increase in the analysts' accuracy of identifying correct predictions and over 29% decrease in their task execution time; in the task of identifying adversarial inputs, i-Algebra improves the analysts' overall accuracy by 26%; while in the task of cleansing poisoning data, i-Algebra helps the analysts' detecting over 60% of the data points misclassified by the automated tool.

Our Contributions – To our best knowledge, i-Algebra represents the first framework for interactive interpretation of DNNs. Our contributions are summarized as follows.

- We promote a new paradigm for interactive interpretation of DNN behaviors, which accounts for the perception, understanding, and responses of end-users.
- We realize this paradigm with an expressive algebraic framework built upon a library of atomic interpretation operators, which can be flexibly composed to construct various analysis tasks.
- We prototype i-Algebra and empirically evaluate it in three representative analysis tasks, all showing its promising usability in practice.

Background and Overview

DNN Interpretation

We primarily focus on predictive tasks (e.g., image classification): a DNN f represents a function $f : \mathcal{X} \to \mathcal{C}$, which assigns a given input $x \in \mathcal{X}$ to one of a set of predefined classes $\mathcal{C}$. We mainly consider post-hoc, instance-level interpretation, which explains the causal relationship between input x and model prediction $f(x)$. Such interpretations are commonly given in the form of *attribution maps*. Typically, the interpreter g generates an attribution map $m = g(x; f)$, with its i-th element $m[i]$ quantifying the importance of x's i-th feature $x[i]$ with respect to $f(x)$.

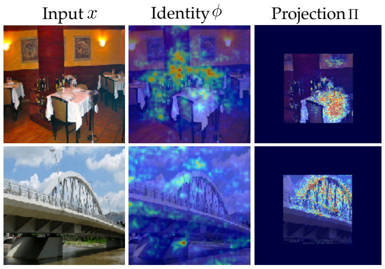

Figure 2: Sample inputs and their interpretation under the identity and projection operators (ImageNet and ResNet50).

Overview of i-Algebra

Despite the rich collection of interpretation models, they are used in an ad hoc and static manner within most existing IDLSes, resulting in their poor usability in practice (Zhang et al. 2020). To address this, it is essential to account for the perception, understanding, and response of end-users. We achieve this by developing i-Algebra, an interactive framework that allows users to easily analyze DNN's behavior through the lens of interpretation.

Mechanisms – i-Algebra is built upon a library of composable atomic operators, which provides interpretability at different input granularities (e.g., within a user-selected window), during different model stages (e.g., at a specific DNN layer), and from different inference perspectives (e.g., finding discriminative features). Note that we only define the functionality of these operators, while their implementation can be flexibly based on concrete interpretation models.

Interfaces – On top of this library, we define an SQL-like declarative query language to allow users to flexibly compose the operators to construct various analysis tasks (e.g., "drill-down," "what-if," "comparative" analysis), which accommodates the diversity of analytical needs from different users and circumvents the "one-size-fits-all" challenge.

An Interpretation Algebra

We begin by describing the library of atomic operators. Note that the operators can inherently be extended and all the operators are defined in a declarative manner, independent of their concrete implementation.

Atomic Operators

At the core of i-Algebra is a library of atomic operators, including *identity*, *projection*, *selection*, *join*, and *anti-join*. We exemplify with the Shapley value framework (Ancona, Öztireli, and Gross 2019; Chen et al. 2019) to illustrate one possible implementation of i-Algebra, which can also be implemented based on other interpretation models.

Identity – The *identity* operator represents the basic interpretation $\phi(x; \bar{x}, f)$, which generates the interpretation of a given input x with respect to the DNN f and a baseline input

$\bar{x}$ (e.g., an all-zero vector).[1] Conceptually, the identity operator computes the expected contribution of x's each feature to the prediction $f(x)$. Within the Shapley framework, with x as a d-dimensional vector ($x \in \mathbb{R}^d$) and I_k as a k-sized subset of $I = \{1, 2, \ldots, d\}$, we define a d-dimensional vector x_{I_k}, with its i-th dimension defined as:

$$[x_{I_k}]_i = \left\{ \begin{array}{ll} x_i & (i \in I_k) \\ \bar{x}_i & (i \notin I_k) \end{array} \right. \tag{1}$$

Intuitively, x_{I_k} substitutes $\bar{x}$ with x along the dimensions of I_k. Then the attribution map is calculated as:

$$[\phi(x)]_i = \frac{1}{d} \sum_{k=0}^{d-1} \mathbb{E}_{I_k}[f(x_{I_k \cup \{i\}}) - f(x_{I_k})] \tag{2}$$

where I_k is randomly sampled from $I \setminus \{i\}$.

Projection – While the basic interpretation describes the global importance of x's features with respect to its prediction $f(x)$, the user may wish to estimate the local importance of a subset of features. Intuitively, the global importance approximates the decision boundary in a high dimensional space, while the local importance focuses on a lower-dimensional space, thereby being able to describe the local boundary more precisely.

The *projection* operator Π allows the user to zoom in a given input. For an input x and a window w (on x) selected by the user, $\Pi_w(x)$ generates the local importance of x's features within w. To implement it within the Shapley framework, we marginalize x's part outside the window w with the baseline input $\bar{x}$ and compute the marginalized interpretation. Let w corresponds to the set of indices $\{w_1, w_2, \ldots, w_{|w|}\}$ in I. To support projection, we define the coalition I_k as a k-sized subset of $\{w_1, w_2, \ldots, w_{|w|}\}$, and redefine the attribution map as: $[\Pi_w(x)]_i =$

$$\left\{ \begin{array}{ll} \frac{1}{|w|} \sum_{k=0}^{|w|-1} \mathbb{E}_{I_k}[f(x_{I_k \cup \{i\}}) - f(x_{I_k})] & i \in w \\ 0 & i \notin w \end{array} \right. \tag{3}$$

Figure 2 illustrates a set of sample images and their interpretation under the identity and projection operators (within user-selected windows). Observe that the projection operator highlights how the model's attention shifts if the features out of the window are nullified, which is essential for performing the "what-if" analysis.

Selection – While the basic interpretation shows the static importance of x's features, the user may also be interested in understanding how the feature importance varies dynamically throughout different inference stages. Intuitively, this dynamic importance interpretation captures the shifting of the "attention" of DNNs during different stages, which helps the user conduct an in-depth analysis of the inference process (Karpathy, Johnson, and Fei-Fei 2016).

The selection operator σ allows the user to navigate through different stages of DNNs and investigate the dynamic feature importance. Given an input x, a DNN f which consists of n layers $f_{[1:n]}$, and the layer index i selected by the user, $\sigma_i(x)$ generates the interpretation at the i-th layer.

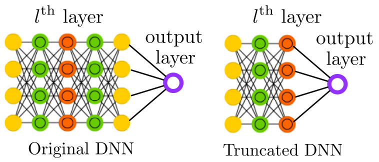

Figure 3: Implementation of the selection operator.

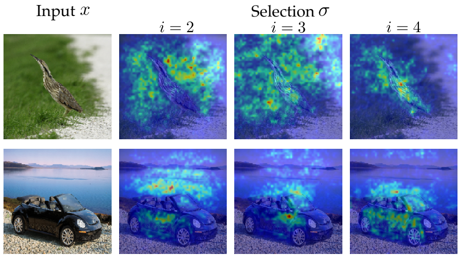

Figure 4: Sample inputs and their interpretation under the selection operator (ImageNet and ResNet50).

One possible implementation of $\sigma_l(x)$ is as follows. We truncate the DNN f at the l-th layer, concatenate it with the output layer, and re-train the linear connections between the l-th layer and the output layer, as illustrated in Figure 3. Let f_l denote the truncated DNN. Then the selection operator generates a d-dimensional map $\sigma_l(x)$ defined as:

$$[\sigma_l(x)]_i = [\phi(x; \bar{x}, f_l)]_i \tag{4}$$

which substitutes f in Eqn (2) with the truncated DNN f_l.

Figure 4 illustrates a set of sample inputs (from ImageNet) and their attribution maps under the selection operator. Specifically, we select $i = 2, 3, 4$ of the DNN (ResNet50). It is observed that the model's attention gradually focuses on the key objects within each image as i increases.

Join – There are also scenarios in which the user desires to compare two inputs x and x' from the same class and find the most informative features shared by x and x', from the perspective of the DNN model. The *join* of two inputs x and x', denoted by $x \bowtie x'$, compares two inputs and generates the interpretation highlighting the most informative features shared by x and x'. Note that the extension of this definition to the case of multiple inputs is straightforward.

Within the Shapley framework, $x \bowtie x'$ can be implemented as the weighted sum of the Shapley values of x and x' (given the weight of x's map as ϵ):

$$[x \bowtie x']_i = \epsilon \cdot [\phi(x; \bar{x}, f)]_i + (1 - \epsilon) \cdot [\phi(x'; \bar{x}, f)]_i \tag{5}$$

Intuitively, a large value of $[x \bowtie x']_i$ tends to indicate that the i-th feature is important for the predictions on both x and x' (with respect to the baseline $\bar{x}$).

Figure 5 illustrates two pairs of sample inputs and their attribution maps under the join operator as $\epsilon = 0.5$, which highlight the most important features with respect to their predictions ("horse" and "plane") shared by both inputs.

Anti-Join – Related to the join operator, the *anti-join* of two inputs x and x', denoted by $x \diamond x'$, compares two inputs x and x' from different classes and highlights their most informative and discriminative features. For instance, in image

[1] When the context is clear, we omit $\bar{x}$ and f in the notation.

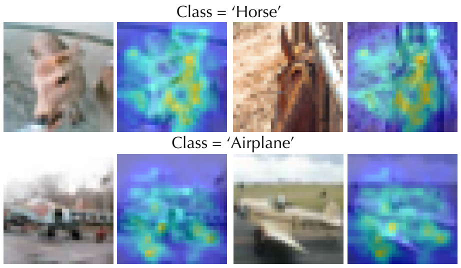

Figure 5: Sample inputs and their interpretation under the join operator (CIFAR10 and VGG19).

classification, the user may be interested in finding the most contrastive features of two images that result in their different classifications.

Within the Shapley value framework, the anti-join operator $x \diamond x'$ can be implemented as the attribution map of x with respect to x' and that of x' with respect to x:

$$[x \diamond x']_i = ([\phi(x; x', f)]_i, [\phi(x'; x, f)]_i) \tag{6}$$

It is worth comparing Eqn (5) and Eqn (6): Eqn (5) compares x (and x') with the baseline $\bar{x}$, highlighting the contribution of each feature of x (and x') with respect to the difference $f(x) - f(\bar{x})$ (and $f(x') - f(\bar{x})$); meanwhile, Eqn (6) compares x and x', highlighting the contribution of each feature of x (and x') with respect to the difference $f(x) - f(x')$ (and $f(x') - f(x)$).

Figure 6 compares sample inputs and their attribution maps under the join operator. In each pair, one is a legitimate input and classified correctly (e.g., "ship" and "dog"); the other is an adversarial input (crafted by the PGD attack (Madry et al. 2018)) and misclassified (e.g., "frog" and "cat"). The anti-join operator highlights the most discriminative features that result in their different predictions.

Note that the anti-join operator is extensible to the case of the same input x but different models f and f'. Specifically, to compute x's features that discriminate $f(x)$ from $f'(x)$, we update the expectation in Eqn (2) as $\mathbb{E}_{I_k}[f(x_{I_k \cup \{i\}}) - f'(x_{I_k})]$. Intuitively, for i-th feature, we compute the difference of its contribution with respect to $f(x)$ and $f'(x)$.

Compositions

The library of atomic operators is naturally *composable*. For instance, one may combine the selection and projection operators, $\Pi_w(\sigma_l(x))$, which extracts the interpretation at the l-th layer of the DNN and magnifies the features within the window w; it is possible to compose the join and selection operators, $\sigma_l(x) \bowtie \sigma_l(x')$, which highlights the most discriminative features of x and x' resulting in their different predictions from the view of the l-th layer of the DNN f; further, it is possible to combine two anti-join operators, $x_1 \diamond x_2 \diamond x_3$, which generates the most discriminative features of each input with respect to the rest two.

To ensure their semantic correctness, one may specify that the compositions of different operators to satisfy certain properties (e.g., commutative). For instance, the composition of the selection (σ) and projection (Π) operators needs to satisfy the commutative property, that is, the order of applying the two operators should not affect the interpretation result, $\Pi_w \sigma_l(x) = \sigma_l \Pi_w(x)$å. Moreover, certain compositions are allowed only under certain conditions (conditional). For instance, the composition of two selection operators, $\sigma_l \sigma_{l'}(x)$, is only defined if $l \leq l'$, that is, it generates the interpretation of x with respect to the layer with the smaller index in l and l' of the DNN f. Further, the composition of the join and anti-join operators are undefined.

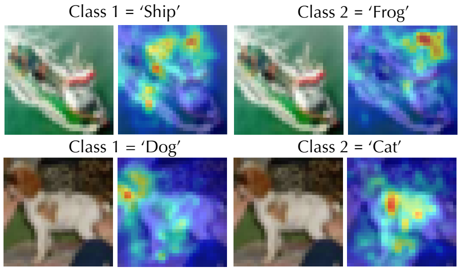

Figure 6: Sample inputs and their interpretation under the anti-join operator (CIFAR10 and VGG19).

Interactive Interpretation

Further, i-Algebra offers a declarative language that allows users to easily "query" the interpretation of DNN behaviors and build interactive analysis tasks as combinations of queries (cf. Figure 1).

A Declarative Query Language

Specifically, we define an SQL-like declarative query language for interpreting DNN behaviors. Next we first define the statements for each atomic operator and then discuss their compositions.

We use a set of keywords: "select" for the selection operator, "where" for the projection operator, "join" for the join operator, and "left join" for the anti-join operator. The atomic operators can be invoked using the following statements:

```
select * from f(x)
```

– the identity operator $\phi(x)$.

```
select * from f(x) where w
```

– the projection operator $\Pi_w(x)$.

```
select l from f(x)
```

– the selection operator $\sigma_l(x)$.

```
select * from f(x) join (select * from f(x'))
```

– the join operator $x \bowtie x'$.

```
select * from f(x) left join (select * from f(x'))
```

– the anti-join operator $x \diamond x'$.

Similar to the concept of "sub-queries" in SQL, more complicated operators can be built by composing the statements of atomic operators. Following are a few examples.

```
select l from f(x) where w
```

– the composition of selection and projection $\Pi_w \sigma_l(x)$.

```
select l from f(x) join (select l from f(x'))
```

– the composition of join and selection $\sigma_l(x) \bowtie \sigma_l(x')$.

Interactive Analysis

Through the declarative queries, users are able to conduct an in-depth analysis of DNN behaviors, including:

Drill-Down Analysis – Here the user applies a sequence of projection and/or selection to investigate how the DNN model f classifies a given input x at different granularities of x and at different stages of f. This analysis helps answer important questions such as: (i) how does the importance of x's features evolve through different stages of f? (ii) which parts of x are likely to be the cause of its misclassification? (iii) which stages of f do not function as expected?

Comparative Analysis – In a comparative analysis, the user applies a combination of join and/or anti-join operators on the target input x and a set of reference inputs $\mathcal{X}$ to compare how the DNN f processes x and $x' \in \mathcal{X}$. This analysis helps answer important questions, including: (i) from f's view, why are x and $x' \in \mathcal{X}$ similar or different? (ii) does f indeed find the discriminative features of x and $x' \in \mathcal{X}$? (iii) if x is misclassified into the class of x', which parts of x are likely to be the cause?

What-If Analysis – In what-if analysis, the user modifies parts of the input x before applying the operators and compares the interpretation before and after the modification. The modification may include (i) nullification (e.g., replacing parts of x with baseline), (ii) substitution (e.g., substituting parts of x with another input), and (iii) transformation (e.g., scaling, rotating, shifting). This analysis allows the user to analyze f's sensitivity to each part of x and its robustness against perturbation.

Note that these tasks are not exclusive; rather, they may complement each other by providing different perspectives on the behaviors of DNN models.

Empirical Evaluation

We prototype i-Algebra and empirically evaluate its usability in a set of case studies. The evaluation is designed to answer the following key questions.

- RQ1: Versatility – Does i-Algebra effectively support a range of analysis tasks?
- RQ2: Effectiveness – Does it significantly improve the analysis efficacy in such tasks?
- RQ3: Usability – Does it provide intuitive, user-friendly interfaces for analysts?

We conduct user studies on the Amazon MTurk platform, in which each task involves 1,250 assignments conducted by 50 qualified workers. We apply the following quality control: (i) the users are instructed about the task goals and declarative queries, and (ii) the tasks are set as small batches to reduce bias and exhaustion.

Case A: Resolving Model Inconsistency

Two DNNs trained for the same task often differ slightly due to (i) different training datasets, (ii) different training regimes, and (iii) randomness inherent in training algorithms (e.g., random shuffle and dropout). It is thus critical to identify the correct one when multiple DNNs disagree on the prediction on a given input. In this case study, the user is requested to use i-Algebra to resolve cases that are inconsistent between two DNNs f and f'.

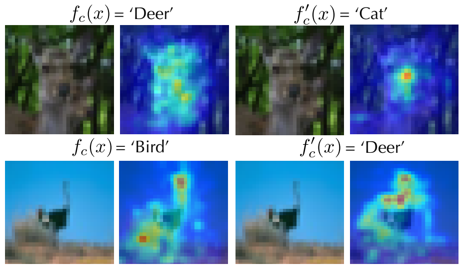

Figure 7: Sample inputs, their classification by f and f', and their interpretation by i-Algebra in Case A.

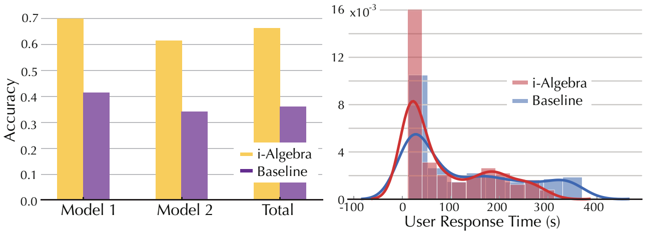

Figure 8: Users' accuracy and URT measures under baseline and i-Algebra in Case A.

Setting – On CIFAR10, we train two VGG19 models f and f'. In the testing set of CIFAR10, 946 samples are predicted differently by f and f', in which 261 samples are correctly predicted by f and 565 samples by f'. Within this set, 824 inputs are classified correctly by either f or f', which we collect as the testing set $\mathcal{T}$ for our study.

We randomly sample 50 inputs (half predicted correctly by f and the rest by f') from $\mathcal{T}$ to form the testing set. The baseline directly generates the interpretation $f(x)$ and $f'(x)$ for each input x; i-Algebra applies the *Anti-Join* operator to highlight x's most discriminative features (from the views of f and f') that result in its different predictions, with the declarative query given as:

```
select * from f(x) left join (select * from f'(x))
```

Figure 7 shows a set of sample inputs, their classification under f and f', and their interpretation by i-Algebra. Observe that the discriminative features on the correct model tend to agree with human perception better.

Evaluation – We evaluate i-Algebra in terms of (i) effectiveness – whether it helps users identify the correct predictions, and (ii) efficiency – whether it helps users conduct the analysis more efficiently. We measure the effectiveness using the metric of accuracy (the fraction of correctly distinguished inputs among total inputs) and assess the efficiency using the metric of user response time (URT), which is measured by the average time the users spend on each input.

Figure 8 compares the users' performance using the baseline and i-Algebra on the task of resolving model inconsistency. We have the following observations: (i) i-Algebra significantly improves the users' accuracy of identifying the correct predictions on both f and f', with the overall accuracy increasing by around 30%; (ii) Despite its slightly more complicated interfaces, the URT on i-Algebra does not observe a significant change from the baseline, highlighting its easy-to-use mechanisms and interfaces.

Case B: Detecting Adversarial Inputs

One intriguing property of DNNs is their vulnerability to adversarial inputs, which are maliciously crafted samples to deceive target DNNs (Madry et al. 2018; Carlini and Wagner 2017). Adversarial inputs are often generated by carefully perturbing benign inputs, with difference imperceptible to human perception. Recent work has proposed to leverage interpretation as a defense mechanism to detect adversarial inputs (Tao et al. 2018). Yet, it is shown that the interpretation model is often misaligned with the underlying DNN model, resulting in the possibility for the adversary to deceive both models simultaneously (Zhang et al. 2020). In this use case, the users are requested to leverage i-Algebra to inspect potential adversarial inputs from multiple different interpretation perspectives, making it challenging for the adversary to evade the detection across all such views.

Setting – We use ImageNet as the dataset and consider a pre-trained ResNet50 (77.15% top-1 accuracy) as the target DNN. We train a set of truncated models ($l = 2, 3, 4$) for the selection operator $\sigma_l(x)$. We apply ADV^2 (Zhang et al. 2020), an attack designed to generate adversarial inputs deceiving both the DNN and its coupled interpreter. Specifically, ADV^2 optimizes the objective function:

$$\begin{array}{ll} \min_x & \ell_{\text{prd}}\left(f(x), c_t\right) + \lambda \ell_{\text{int}}\left(g(x; f), m_t\right) \\ \text{s.t.} & \Delta\left(x, x_\circ\right) \leq \epsilon \end{array} \tag{7}$$

where ℓ_{prd} ensures that the adversarial input x is misclassified to a target class c_t by the DNN f, and ℓ_{int} ensures that x generates an attribution map similar to a target map m_t (the attribution map of the benign input $x_\circ$).

We randomly sample 50 inputs and generate their adversarial counterparts, which are combined with another 50 randomly sampled benign inputs to form the testing set $\mathcal{T}$. We request the users to identify the adversarial inputs through the lens of baseline and i-Algebra. By varying l and w, i-Algebra provides interpretation at various inference stages and input granularity, with the query template as:

```
select l from f(x) where w
```

Figure 9 shows sample adversarial inputs and their interpretation under i-Algebra. Observe that from complementary perspectives, the adversarial inputs show fairly distinguishable interpretation.

Evaluation – We assess the usability of i-Algebra in terms of (i) whether it helps users identify the adversarial inputs more accurately, and (ii) whether it helps users conduct the analysis more efficiently. Specifically, considering adversarial and benign inputs as positive and negative cases, we measure the effectiveness using precision and recall. We assess the efficiency using the metric of user response time (URT), which is measured by the average time the users spend on each task with the given tool.

Figure 10 compares the users' performance and URT using the baseline interpretation and i-Algebra on the task of identifying adversarial inputs. We have the following observations: (i) through the lens of interpretation from multiple complementary perspectives, i-Algebra improves the users' effectiveness of distinguishing adversarial and benign inputs with about 26% increase in the overall accuracy; (ii) compared with the baseline, the average URT on i-Algebra grows from 60.45s to 303.54s, which can be intuitively explained by its requirement for multiple rounds of interactive queries. Given the significant performance improvement, the cost of execution time is well justified.

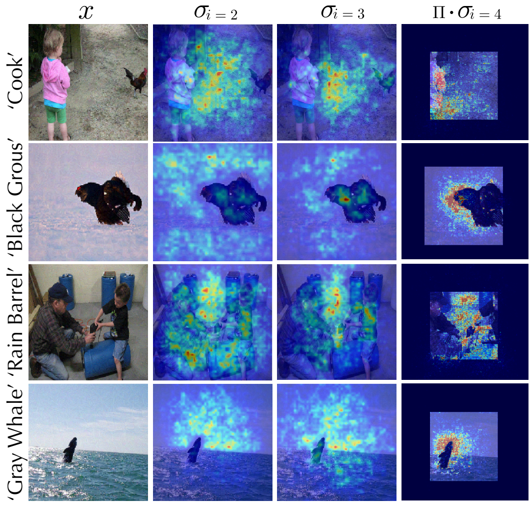

Figure 9: Sample adversarial inputs and interpretation.

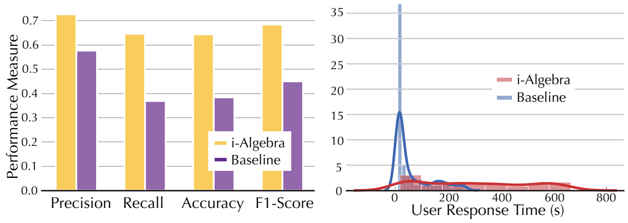

Figure 10: Users' performance and URT measures under baseline and i-Algebra in Case B.

Case C: Cleansing Poisoning Data

Orthogonal to adversarial examples, another concern for the safety of DNNs is their vulnerability to manipulations of their training data. In the backdoor attacks (e.g., (Gu, Dolan-Gavitt, and Garg 2017)), by injecting corrupted inputs in training a DNN, the adversary forces the resultant model to (i) misclassify the inputs embedded with particular patterns ("triggers") to a target class and (ii) behave normally on benign inputs. Figure 11 shows a set of trigger-embedded inputs which are misclassified from "truck" to "deer".

Since the backdoored DNNs correctly classify benign inputs, once trained, they are insidious to be detected. One mitigation is to detect corrupted instances in the training set, and then to use cleansed data to re-train the DNN (Tran, Li, and Madry 2018). Typically, the analyst applies statistical

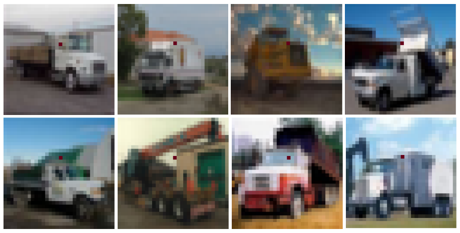

Figure 11: Sample trigger inputs misclassified as “deer”.

Prediction	Ground-truth	
	+	-
+	48	654
-	399	3574

Table 1: Statistics of samples in Case C.

analysis on the deep representations of the training data and detects poisoning inputs based on their statistical anomaly.

The users use i-Algebra to refine the results detected by the automated detection method. Through the lens of the interpretation, the users may inspect the inputs that fall in the uncertain regions (e.g., 1.25 $\sim$ 1.75 s.t.d.) and identify false positives and false negatives by the automated method.

Setting – We use CIFAR10 as the dataset and VGG19 as the DNN. We use the backdoor attack in (Gu, Dolan-Gavitt, and Garg 2017) to generate poisoning instances in one particular class “truck”. We apply spectral signature (Tran, Li, and Madry 2018) to identify potential poisoning instances. For each input x_i in a particular class, it examines x_i’s latent representation $\mathcal{R}(x_i)$ at the penultimate layer. After obtaining the top right singular vector of the centered representation $[\mathcal{R}(x_i) - \frac{1}{n}\sum_{i=1}^{n}\mathcal{R}(x_i)]_{i=1}^{n}$, the inputs beyond 1.5 standard deviation from the center are identified as poisonous. However, this automated tool is fairly inaccurate. Table 1 summarizes the predicted results and the ground truth. In this case, we request the users to identify positive and negative cases that are misclassified by the automated tool, with the query template given as:

```
select l from f(x)
```

Evaluation – We measure the users’ effectiveness of distinguishing positive and negative cases. The results are listed in Table 2. Observe that equipped with i-Algebra, the users successfully classify around 60% of the data point misclassified by the automated tool. Figure 12 shows the distribution of URT for this task. Note that a majority of users take less than 50s to complete the tasks.

Precision	Recall	Accuracy	F1-Score
0.609	0.6343	0.586	0.622

Table 2: Users’ performance under i-Algebra in Case C.

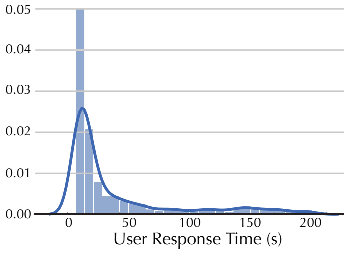

Figure 12: URT distribution in Case C.

Related Work

Interpretable Deep Learning – Typically, DNN interpretability can be obtained by either designing interpretable models (Zhang, Nian Wu, and Zhu 2018) or extracting post-hoc interpretations. The post-hoc interpretation methods can be categorized as backprop- (Sundararajan, Taly, and Yan 2017), representation- (Selvaraju et al. 2017), meta-model- (Dabkowski and Gal 2017), and perturbation-based (Fong and Vedaldi 2017). Instead of developing yet another interpretation method or enhancing existing ones, this work proposes the paradigm of interactive interpretability, which can be flexibly implemented upon existing methods.

Model Attacks and Defenses – DNNs are becoming the new targets of malicious attacks, including adversarial attacks (Carlini and Wagner 2017) and poisoning attacks (Shafahi et al. 2018). Although a line of work strives to improve DNN robustness (Tramèr et al. 2018; Tran, Li, and Madry 2018), existing defenses are often penetrated by even stronger attacks (Ling et al. 2019), resulting in a constant arms race. Our work involves the users in the process of model robustness improvement, which is conducive to enhancing model trustworthiness.

Interactive Learning – Interactive learning couples humans and machine learning models tightly within the learning process. The existing work can be roughly categorized as model understanding (Krause, Perer, and Bertini 2016; Krause, Perer, and Ng 2016), which allows users to interpret models’ input-output dependence, and model debugging (Wu et al. 2019; Nushi, Kamar, and Horvitz 2018), which allows users to detect and fix models’ mistakes. i-Algebra can be leveraged to not only understand DNNs’ behaviors but also facilitate diverse security tasks including model debugging, data cleansing, and attack inspection.

Conclusion

This work promotes a paradigm shift from *static* interpretation to *interactive* interpretation of DNNs, which significantly improves the usability of existing interpretation models in practice. We design and prototype i-Algebra, a first-of-its-kind interactive framework for DNN interpretation. The extensive studies in three representative analysis tasks all demonstrate the promising usability of i-Algebra.

Acknowledgments

This work is supported by the National Science Foundation under Grant No. 1951729, 1953813, and 1953893. Any opinions, findings, and conclusions or recommendations are those of the authors and do not necessarily reflect the views of the National Science Foundation. Shouling Ji was partly supported by the National Key Research and Development Program of China under No. 2018YFB0804102 and No. 2020YFB2103802, NSFC under No. 61772466, U1936215, and U1836202, the Zhejiang Provincial Natural Science Foundation for Distinguished Young Scholars under No. LR19F020003, and the Fundamental Research Funds for the Central Universities (Zhejiang University NGICS Platform).

References

Ancona, M.; Öztireli, C.; and Gross, M. 2019. Explaining Deep Neural Networks with a Polynomial Time Algorithm for Shapley Values Approximation. In *Proceedings of IEEE Conference on Machine Learning (ICML)*.

Carlini, N.; and Wagner, D. A. 2017. Towards Evaluating the Robustness of Neural Networks. In *Proceedings of IEEE Symposium on Security and Privacy (S&P)*.

Chen, J.; Song, L.; Wainwright, M. J.; and Jordan, M. I. 2019. L-Shapley and C-Shapley: Efficient Model Interpretation for Structured Data. In *Proceedings of International Conference on Learning Representations (ICLR)*.

Dabkowski, P.; and Gal, Y. 2017. Real Time Image Saliency for Black Box Classifiers. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*.

Fong, R. C.; and Vedaldi, A. 2017. Interpretable Explanations of Black Boxes by Meaningful Perturbation. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*.

Goodfellow, I.; Bengio, Y.; and Courville, A. 2016. *Deep Learning*. MIT Press.

Gu, T.; Dolan-Gavitt, B.; and Garg, S. 2017. BadNets: Identifying Vulnerabilities in the Machine Learning Model Supply Chain. *ArXiv e-prints* .

Guo, W.; Mu, D.; Xu, J.; Su, P.; Wang, G.; and Xing, X. 2018. LEMNA: Explaining Deep Learning Based Security Applications. In *Proceedings of ACM SAC Conference on Computer and Communications (CCS)*.

Karpathy, A.; Johnson, J.; and Fei-Fei, L. 2016. Visualizing and Understanding Recurrent Networks. In *Proceedings of International Conference on Learning Representations (ICLR)*.

Krause, J.; Perer, A.; and Bertini, E. 2016. Using Visual Analytics to Interpret Predictive Machine Learning Models. In *Proceedings of IEEE Conference on Machine Learning (ICML)*.

Krause, J.; Perer, A.; and Ng, K. 2016. Interacting with Predictions: Visual Inspection of Black-Box Machine Learning Models. In *Proceedings of the CHI Conference on Human Factors in Computing Systems (CHI)*.

Ling, X.; Ji, S.; Zou, J.; Wang, J.; Wu, C.; Li, B.; and Wang, T. 2019. DEEPSEC: A Uniform Platform for Security Analysis of Deep Learning Model. In *Proceedings of IEEE Symposium on Security and Privacy (S&P)*.

Madry, A.; Makelov, A.; Schmidt, L.; Tsipras, D.; and Vladu, A. 2018. Towards Deep Learning Models Resistant to Adversarial Attacks. In *Proceedings of International Conference on Learning Representations (ICLR)*.

Nushi, B.; Kamar, E.; and Horvitz, E. 2018. Towards Accountable AI: Hybrid Human-Machine Analyses for Characterizing System Failure. In *Proceedings of AAAI Conference on Artificial Intelligence (AAAI)*.

Selvaraju, R. R.; Cogswell, M.; Das, A.; Vedantam, R.; Parikh, D.; and Batra, D. 2017. Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*.

Shafahi, A.; Ronny Huang, W.; Najibi, M.; Suciu, O.; Studer, C.; Dumitras, T.; and Goldstein, T. 2018. Poison Frogs! Targeted Clean-Label Poisoning Attacks on Neural Networks. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*.

Sundararajan, M.; Taly, A.; and Yan, Q. 2017. Axiomatic Attribution for Deep Networks. In *Proceedings of IEEE Conference on Machine Learning (ICML)*.

Tao, G.; Ma, S.; Liu, Y.; and Zhang, X. 2018. Attacks Meet Interpretability: Attribute-Steered Detection of Adversarial Samples. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*.

Tramèr, F.; Kurakin, A.; Papernot, N.; Goodfellow, I.; Boneh, D.; and McDaniel, P. 2018. Ensemble Adversarial Training: Attacks and Defenses. In *Proceedings of International Conference on Learning Representations (ICLR)*.

Tran, B.; Li, J.; and Madry, A. 2018. Spectral Signatures in Backdoor Attacks. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*.

Wu, T.; Ribeiro, M. T.; Heer, J.; and Weld, D. S. 2019. Errudite: Scalable, Reproducible, and Testable Error Analysis. In *Proceedings of Annual Meeting of the Association for Computational Linguistics (ACL)*.

Zhang, Q.; Nian Wu, Y.; and Zhu, S.-C. 2018. Interpretable Convolutional Neural Networks. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Zhang, X.; Wang, N.; Shen, H.; Ji, S.; Luo, X.; and Wang, T. 2020. Interpretable Deep Learning under Fire. In *Proceedings of USENIX Security Symposium (SEC)*.