

A Survey on Neural Network Interpretability

Yu Zhang , Peter Tiño , Aleš Leonardis , *Member, IEEE*, and Ke Tang , *Senior Member, IEEE*

Abstract—Along with the great success of deep neural networks, there is also growing concern about their black-box nature. The interpretability issue affects people’s trust on deep learning systems. It is also related to many ethical problems, e.g., algorithmic discrimination. Moreover, interpretability is a desired property for deep networks to become powerful tools in other research fields, e.g., drug discovery and genomics. In this survey, we conduct a comprehensive review of the neural network interpretability research. We first clarify the definition of interpretability as it has been used in many different contexts. Then we elaborate on the importance of interpretability and propose a novel taxonomy organized along three dimensions: type of engagement (passive vs. active interpretation approaches), the type of explanation, and the focus (from local to global interpretability). This taxonomy provides a meaningful 3D view of distribution of papers from the relevant literature as two of the dimensions are not simply categorical but allow ordinal subcategories. Finally, we summarize the existing interpretability evaluation methods and suggest possible research directions inspired by our new taxonomy.

Index Terms—Machine learning, neural networks, interpretability, survey.

I. INTRODUCTION

OVER the last few years, deep neural networks (DNNs) have achieved tremendous success [1] in computer vi-

sion [2], [3], speech recognition [4], natural language processing [5] and other fields [6], while the latest applications can be found in these surveys [7]–[9]. They have not only beaten many previous machine learning techniques (e.g., decision trees, support vector machines), but also achieved the state-of-the-art performance on certain real-world tasks [4], [10]. Products powered by DNNs are now used by billions of people,¹ e.g., in facial and voice recognition. DNNs have also become powerful tools for many scientific fields, such as medicine [11], bioinformatics [12], [13] and astronomy [14], which usually involve massive data volumes.

However, deep learning still has some significant disadvantages. As a really complicated model with millions of free parameters (e.g., AlexNet [2], 62 million), DNNs are often found to exhibit unexpected behaviours. For instance, even though a network could get the state-of-the-art performance and seemed to generalize well on the object recognition task, Szegedy *et al.* [15] found a way that could arbitrarily change the network’s prediction by applying a certain imperceptible change to the input image. This kind of modified input is called “adversarial example”. Nguyen *et al.* [16] showed another way to produce completely unrecognizable images (e.g., look like white noise), which are, however, recognized as certain objects by DNNs with 99.99% confidence. These observations suggest that even though DNNs can achieve superior performance on many tasks, their underlying mechanisms may be very different from those of humans’ and have not yet been well-understood.

A. An (Extended) Definition of Interpretability

To open the black-boxes of deep networks, many researchers started to focus on the model interpretability. Although this theme has been explored in various papers, no clear consensus on the definition of interpretability has been reached. Most previous works skimmed over the clarification issue and left it as “you will know it when you see it”. If we take a closer look, the suggested definitions and motivations for interpretability are often different or even discordant [17].

One previous definition of interpretability is *the ability to provide explanations in understandable terms to a human* [18], while the term *explanation* itself is still elusive. After reviewing previous literature, we make further clarifications of “explanations” and “understandable terms” on the basis of [18].

Interpretability is the ability to provide *explanations*¹ in *understandable terms*² to a human.

where

¹[Online]. Available: <https://www.acm.org/media-center/2019/march/turing-award-2018>

Manuscript received March 3, 2021; revised June 7, 2021; accepted July 9, 2021. Date of publication August 24, 2021; date of current version September 23, 2021. This work was supported in part by the Guangdong Provincial Key Laboratory under Grant 2020B121201001, in part by the Program for Guangdong Introducing Innovative and Entrepreneurial Teams under Grant 2017ZT07X386, in part by the Stable Support Plan Program of Shenzhen Natural Science Fund under Grant 20200925154942002, in part by the Science and Technology Commission of Shanghai Municipality under Grant 19511120602, in part by the National Leading Youth Talent Support Program of China, and in part by the MOE University Scientific-Technological Innovation Plan Program. The work of Peter Tino was supported by the European Commission Horizon 2020 Innovative Training Network SUNDIAL and also Alan Turing Institute, ATI Fellowship 1056900 (Machine Learning in the Space of Inferential Models) (Survey Network for Deep Imaging Analysis, and Learning), under Project ID: 721463. (*Corresponding author: Ke Tang.*)

Yu Zhang is with the Guangdong Key Laboratory of Brain-Inspired Intelligent Computation, Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen 518055, China, with the Research Institute of Trust-worthy Autonomous Systems, Southern University of Science and Technology, Shenzhen 518055, China, and also with the School of Computer Science, University of Birmingham, Edgbaston, Birmingham B15 2TT, U.K. (e-mail: zhangy3@mail.sustech.edu.cn).

Ke Tang is with the Guangdong Key Laboratory of Brain-Inspired Intelligent Computation, Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen 518055, P.R. China, and also with the Research Institute of Trust-worthy Autonomous Systems, Southern University of Science and Technology, Shenzhen 518055, China (e-mail: tangk3@sustech.edu.cn).

Peter Tiño and Aleš Leonardis are with the School of Computer Science, University of Birmingham, Edgbaston, Birmingham B15 2TT, U.K. (e-mail: P.Tino@cs.bham.ac.uk; a.leonardis@cs.bham.ac.uk).

Digital Object Identifier 10.1109/TETCI.2021.3100641

TABLE I
SOME INTERPRETABLE “TERMS” USED IN PRACTICE

Field	Raw input	Understandable terms
Computer vision	Images (pixels)	Super pixels (image patches) ^a Visual concepts ^b
NLP	Word embeddings	Words
Bioinformatics	Sequences	Motifs (position weight matrix) ^c

^a image patches are usually used in attribution methods [20].

^b colours, materials, textures, parts, objects and scenes [21].

^c proposed by [22] and became an essential tool for computational motif discovery.

- 1) **Explanations**, ideally, should be *logical decision rules* (if-then rules) or can be transformed to logical rules. However, people usually do not require explanations to be explicitly in a rule form (but only some key elements which can be used to construct explanations).
- 2) **Understandable terms** should be from the *domain knowledge* related to the task (or common knowledge according to the task).

Our definition enables new perspectives on the interpretability research: **1** We highlight *the form of explanations* rather than particular *explanators*. After all, explanations are expressed in a certain “language,” be it natural language, logic rules or something else. Recently, a strong preference has been expressed for the language of explanations to be as close as possible to logic [19]. In practice, people do not always require a full “sentence,” which allows various kinds of explanations (rules, saliency masks etc.). This is an important angle to categorize the approaches in the existing literature. **2** Domain knowledge is the basic unit in the construction of explanations. As deep learning has shown its ability to process data in the raw form, it becomes harder for people to interpret the model with its original input representation. With more domain knowledge, we can get more understandable representations that can be evaluated by domain experts. Table I lists several commonly used representations in different tasks.

We note that some studies distinguish between *interpretability* and *explainability* (or *understandability*, *comprehensibility*, *transparency*, *human-simulatability* etc. [17], [23]). In this paper we do not emphasize the subtle differences among those terms. As defined above, we see explanations as the core of interpretability and use interpretability, explainability and understandability interchangeably. Specifically, we focus on the interpretability of (deep) *neural networks* (rarely recurrent neural networks), which aims to provide explanations of their inner workings and input-output mappings. There are also some interpretability studies about the Generative Adversarial Networks (GANs). However, as a kind of generative models, it is slightly different from common neural networks used as discriminative models. For this topic, we would like to refer readers to the latest work [24]–[29], many of which share the similar ideas with the “hidden semantics” part of this paper (see Section II), trying to interpret the meaning of hidden neurons or the latent space.

Under our definition, the source code of Linux operating system is interpretable although it might be overwhelming for a developer. A deep decision tree or a high-dimensional linear model (on top of interpretable input representations) are also interpretable. One may argue that they are not simulatable [17]

(i.e. a human is able to simulate the model’s processing from input to output in his/her mind in a short time). We claim, however, they are still interpretable.

Besides above confined scope of interpretability (of a trained neural network), there is a much broader field of understanding the general neural network methodology, which cannot be covered by this paper. For example, the empirical success of DNNs raises many unsolved questions to theoreticians [30]. What are the merits (or inductive bias) of DNN architectures [31], [32]? What are the properties of DNNs’ loss surface/critical points [33]–[36]? Why DNNs generalize so well with just simple regularization [37]–[39]? What about DNNs’ robustness/stability [40]–[45]? There are also studies about how to generate adversarial examples [46], [47] and detect adversarial inputs [48].

B. The Importance of Interpretability

The need for interpretability has already been stressed by many papers [17], [18], [49], emphasizing cases where lack of interpretability may be harmful. However, a clearly organized exposition of such argumentation is missing. We summarize the arguments for the importance of interpretability into three groups.

1) *High Reliability Requirement*: Although deep networks have shown great performance on some relatively large test sets, the real world environment is still much more complex. As some unexpected failures are inevitable, we need some means of making sure we are still in control. Deep neural networks do not provide such an option. In practice, they have often been observed to have unexpected performance drop in certain situations, not to mention the potential attacks from the adversarial examples [50], [51].

Interpretability is not always needed but it is important for some prediction systems that are required to be highly reliable because an error may cause catastrophic results (e.g., human lives, heavy financial loss). Interpretability can make potential failures easier to detect (with the help of domain knowledge), avoiding severe consequences. Moreover, it can help engineers pinpoint the root cause and provide a fix accordingly. Interpretability does not make a model more reliable or its performance better, but it is an important part of formulation of a highly reliable system.

2) *Ethical and Legal Requirement*: A first requirement is to avoid algorithmic discrimination. Due to the nature of machine learning techniques, a trained deep neural network may inherit the bias in the training set, which is sometimes hard to notice. There is a concern of fairness when DNNs are used in our daily life, for instance, mortgage qualification, credit and insurance risk assessments.

Deep neural networks have also been used for new drug discovery and design [52]. The computational drug design field was dominated by conventional machine learning methods such as random forests and generalized additive models, partially because of their efficient learning algorithms at that time, and also because a domain chemical interpretation is possible. Interpretability is also needed for a new drug to get approved by the regulator, such as the Food and Drug Administration (FDA).

Besides the clinical test results, the biological mechanism underpinning the results is usually required. The same goes for medical devices.

Another legal requirement of interpretability is the “right to explanation” [53]. According to the EU General Data Protection Regulation (GDPR) [54], Article 22, people have the right not to be subject to an automated decision which would produce legal effects or similar significant effects concerning him or her. The data controller shall safeguard the data owner’s right to obtain human intervention, to express his or her point of view and to contest the decision. If we have no idea how the network makes a decision, there is no way to ensure these rights.

3) *Scientific Usage*: Deep neural networks are becoming powerful tools in scientific research fields where the data may have complex intrinsic patterns (e.g., genomics [55], astronomy [14], physics [56] and even social science [57]). The word “science” is derived from the Latin word “scientia,” which means “knowledge”. When deep networks reach a better performance than the old models, they must have found some unknown “knowledge”. Interpretability is a way to reveal it.

C. Related Work and Contributions

There have already been attempts to summarize the techniques for neural network interpretability. However, most of them only provide basic categorization or enumeration, without a clear taxonomy. Lipton [17] points out that the term interpretability is not well-defined and often has different meanings in different studies. He then provides simple categorization of both the need (e.g., trust, causality, fair decision-making etc.) and methods (post-hoc explanations) in interpretability study. Doshi-Velez and Kim [18] provide a discussion on the definition and evaluation of interpretability, which inspired us to formulate a stricter definition and to categorize the existing methods based on it. Montavon *et al.* [58] confine the definition of explanation to feature importance (also called explanation vectors elsewhere) and review the techniques to interpret learned concepts and individual predictions by networks. They do not aim to give a comprehensive overview and only include some representative approaches. Gilpin *et al.* [59] divide the approaches into three categories: explaining data processing, explaining data representation and explanation-producing networks. Under this categorization, the linear proxy model method and the rule-extraction method are equally viewed as proxy methods, without noticing many differences between them (the former is a local method while the latter is usually global and their produced explanations are different, we will see it in our taxonomy). Guidotti *et al.* [49] consider all black-box models (including tree ensembles, SVMs etc.) and give a fine-grained classification based on four dimensions (the type of interpretability problem, the type of explainer, the type of black-box model, and the type of data). However, they treat decision trees, decision rules, saliency masks, sensitivity analysis, activation maximization etc. equally, as explainers. In our view, some of them are certain types of explanations while some of them are methods used to produce explanations. Zhang and Zhu [60] review the methods to understand network’s mid-layer representations or to learn networks with interpretable representations in computer vision field.

This survey has the following contributions:

- We make a further step towards the definition of interpretability on the basis of reference [18]. In this definition, we emphasize the *type (or format) of explanations* (e.g., rule forms, including both decision trees and decision rule sets). This acts as an important dimension in our proposed taxonomy. Previous papers usually organize existing methods into various isolated (to a large extent) *explainers* (e.g., decision trees, decision rules, feature importance, saliency maps etc.).
- We analyse the real needs for interpretability and summarize them into 3 groups: interpretability as an important component in systems that should be highly-reliable, ethical or legal requirements, and interpretability providing tools to enhance knowledge in the relevant science fields. In contrast, a previous survey [49] only shows the importance of interpretability by providing several cases where black-box models can be dangerous.
- We propose a new taxonomy comprising three dimensions (passive vs. active approaches, the format of explanations, and local-semilocal-global interpretability). Note that although many ingredients of the taxonomy have been discussed in the previous literature, they were either mentioned in totally different context, or entangled with each other. To the best of our knowledge, our taxonomy provides the most comprehensive and clear categorization of the existing approaches.

The three degrees of freedom along which our taxonomy is organized allow for a schematic 3D view illustrating how diverse attempts at interpretability of deep networks are related. It also provides suggestions for possible future work by filling some of the gaps in the interpretability research (see Fig. 2).

D. Organization of the Survey

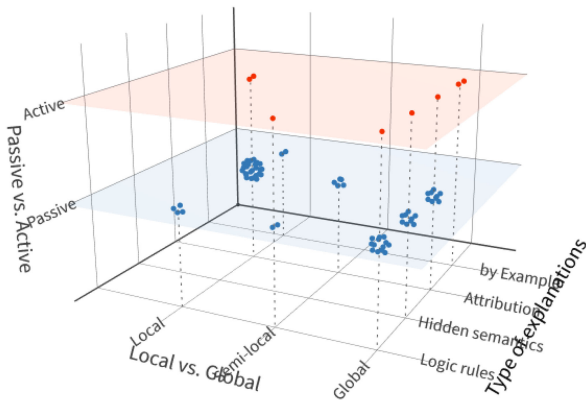
The rest of the survey is organized as follows. In Section II, we introduce our proposed taxonomy for network interpretation methods. The taxonomy consists of three dimensions, passive vs. active methods, type of explanations and global vs. local interpretability. Along the first dimension, we divide the methods into two groups, passive methods (Section III) and active methods (Section IV). Under each section, we traverse the remaining two dimensions (different kinds of explanations, and whether they are local, semi-local or global). Section V gives a brief summary of the evaluation of interpretability. Finally, we conclude this survey in Section VII.

II. TAXONOMY

We propose a novel taxonomy with three dimensions (see Fig. 1): 1) the passive vs. active approaches dimension, 2) the type/format of produced explanations, and 3) from local to global interpretability dimension respectively. **The first dimension** is categorical and has two possible values, *passive interpretation* and *active interpretability intervention*. It divides the existing approaches according to whether they require to change the network architecture or the optimization process. The passive interpretation process starts from a trained network, with all the weights already learned from the training set. Thereafter, the methods try to extract logic rules or extract some understandable

Dimension 1 — Passive vs. Active Approaches	
Passive	Post hoc explain trained neural networks
Active	Actively change the network architecture or training process for better interpretability
Dimension 2 — Type of Explanations (in the order of increasing explanatory power)	
To explain a prediction/class by	
Examples	Provide example(s) which may be considered similar or as prototype(s)
Attribution	Assign credit (or blame) to the input features (e.g. feature importance, saliency masks)
Hidden semantics	Make sense of certain hidden neurons/layers
Rules	Extract logic rules (e.g. decision trees, rule sets and other rule formats)
Dimension 3 — Local vs. Global Interpretability (in terms of the input space)	
Local	Explain network's <i>predictions on individual samples</i> (e.g. a saliency mask for an input image)
Semi-local	In between, for example, explain a group of similar inputs together
Global	Explain the network <i>as a whole</i> (e.g. a set of rules/a decision tree)

Fig. 1. The 3 dimensions of our taxonomy.

Fig. 2. The distribution of the interpretability papers in the 3D space of our taxonomy. We can rotate and observe the density of work in certain areas/planes and find the missing parts of interpretability research. (See <https://lyzhang-gh.github.io/tmp-data/index.html>).

patterns. In contrast, active methods require some changes before the training, such as introducing extra network structures or modifying the training process. These modifications encourage the network to become more interpretable (e.g., more like a decision tree). Most commonly such active interventions come in the form of regularization terms.

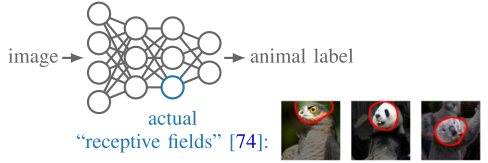




In contrast to previous surveys, the other two dimensions allow ordinal values. For example, the previously proposed dimension *type of explainer* [49] produces subcategories like *decision trees*, *decision rules*, *feature importance*, *sensitivity analysis* etc. However, there is no clear connection among these pre-recognised explainers (what is the relation between decision trees and feature importance). Instead, our **second dimension** is *type/format of explanation*. By inspecting various kinds of explanations produced by different approaches, we can observe differences in how explicit they are. Logic rules provide

the most clear and explicit explanations while other kinds of explanations may be implicit. For example, a saliency map itself is just a mask on top of a certain input. By looking at the saliency map, people construct an explanation “the model made this prediction because it focused on this highly influential part and that part (of the input)”. Hopefully, these parts correspond to some domain understandable concepts. Strictly speaking, implicit explanations by themselves are not complete explanations and need further human interpretation, which is usually automatically done when people see them. We recognize four major types of explanations here, *logic rules*, *hidden semantics*, *attribution* and *explanations by examples*, listed in order of decreasing explanatory power. Similar discussions can be found in the previous literature, e.g., Samek et al. [61] provide a short subsection about “type of explanations” (including explaining learned representations, explaining *individual predictions* etc.). However, it is mixed up with another independent dimension of the interpretability research which we will introduce in the following paragraph. A recent survey [62] follows the same philosophy and treats saliency maps and concept attribution [63] as different types of explanations, while we view them as being of the same kind, but differing in the dimension below.

The last dimension, from local to global interpretability (w.r.t. the input space), has become very common in recent papers (e.g., [18], [49], [58], [64]), where global interpretability means being able to understand the overall decision logic of a model and local interpretability focuses on the explanations of individual predictions. However, in our proposed dimension, there exists a transition rather than a hard division between global and local interpretability (i.e. semi-local interpretability). Local explanations usually make use of the information at the target input (e.g., its feature values, its gradient). But global explanations try to generalize to as wide ranges of inputs as possible (e.g., sequential *covering* in rule learning, *marginal contribution* for feature importance ranking). This view is also

TABLE II

EXAMPLE EXPLANATIONS OF NETWORKS. PLEASE SEE SECTION II FOR DETAILS. DUE TO LACK OF SPACE, WE DO NOT PROVIDE EXAMPLES FOR SEMI-LOCAL INTERPRETABILITY HERE. (WE THANK THE ANONYMOUS REVIEWER FOR THE IDEA TO IMPROVE THE CLARITY OF THIS TABLE.)

	Local (and semi-local) interpretability applies to a certain input $\mathbf{x}^{(i)}$ (and its associated output $\hat{y}^{(i)}$), or a small range of inputs-outputs	Global interpretability w.r.t. the whole input space
Rule as explanation	<p>Explain a certain $(\mathbf{x}^{(i)}, y^{(i)})$ with a decision rule:</p> <ul style="list-style-type: none"> The result “$\mathbf{x}^{(i)}$ is classified as $\hat{y}^{(i)}$” is because $\mathbf{x}_1, \mathbf{x}_4, \dots$ are present and $\mathbf{x}_3, \mathbf{x}_5, \dots$ are absent [69]. (Semi-local) For \mathbf{x} in the neighbourhood of $\mathbf{x}^{(i)}$, if $(\mathbf{x}_1 > \alpha) \wedge (\mathbf{x}_3 < \beta) \wedge \dots$, then $y = \hat{y}^{(i)}$ [65]. 	<p>Explain the whole model $y(\mathbf{x})$ with a decision rule set:</p> <p>The neural network can be approximated by</p> $\begin{cases} \text{If } (\mathbf{x}_2 < \alpha) \wedge (\mathbf{x}_3 > \beta) \wedge \dots, & \text{then } y = 1, \\ \text{If } (\mathbf{x}_1 > \gamma) \wedge (\mathbf{x}_5 < \delta) \wedge \dots, & \text{then } y = 2, \\ \dots & \\ \text{If } (\mathbf{x}_4 \dots) \wedge (\mathbf{x}_7 \dots) \wedge \dots, & \text{then } y = M \end{cases}$
Explaining hidden semantics (make sense of certain hidden neurons/layers)	<p>Explain a hidden neuron/layer $h(\mathbf{x}^{(i)})$:</p> <p>(*No explicit methods but many local attribution methods (see below) can be easily modified to “explain” a hidden neuron $h(\mathbf{x})$ rather than the final output y).</p>	<p>Explain a hidden neuron/layer $h(\mathbf{x})$ instead of $y(\mathbf{x})$:</p> <ul style="list-style-type: none"> An example active method [70] adds a special loss term that encourages filters to learn consistent and exclusive patterns (e.g. head patterns of animals) 
Attribution as explanation	<p>Explain a certain $(\mathbf{x}^{(i)}, y^{(i)})$ with an attribution $\mathbf{a}^{(i)}$:</p> <p>For $\mathbf{x}^{(i)}$:  → neural net → $\hat{y}^{(i)}$: junco bird</p> <p>The “contribution”¹ of each pixel:  [75]</p> <p>a.k.a. saliency map, which can be computed by different methods like gradients [71], sensitivity analysis² [72] etc.</p>	<p>Explain $y(\mathbf{x})$ with attribution to certain features in general:</p> <p>(Note that for a linear model, the coefficients is the global attribution to its input features.)</p> <ul style="list-style-type: none"> Kim et al. [63] calculate attribution to a target “concept” rather than the input pixels of a certain input. For example, “how sensitive is the output (a prediction of zebra) to a concept (the presence of stripes)?”
Explanation by showing examples	<p>Explain a certain $(\mathbf{x}^{(i)}, y^{(i)})$ with another $\mathbf{x}^{(i)'}:$</p> <p>For $\mathbf{x}^{(i)}$:  → neural net → $\hat{y}^{(i)}$: fish</p> <p>By asking how much the network will change $\hat{y}^{(i)}$ if removing a certain training image, we can find:</p> <p>most helpful³ training images:  [73]</p>	<p>Explain $y(\mathbf{x})$ collectively with a few prototypes:</p> <ul style="list-style-type: none"> Adds a (learnable) prototype layer to the network. Every prototype should be similar to at least an encoded input. Every input should be similar to at least a prototype. The trained network explains itself by its prototypes. [76]

¹ the contribution to the network prediction of $\mathbf{x}^{(i)}$.

² how sensitive is the classification result to the change of pixels.

³ without the training image, the network prediction of $\mathbf{x}^{(i)}$ will change a lot. In other words, these images help the network make a decision on $\mathbf{x}^{(i)}$.

supported by the existence of several semi-local explanation methods [65], [66]. There have also been attempts to fuse local explanations into global ones in a bottom-up fashion [19], [67], [68].

To help understand the latter two dimensions, Table II lists examples of typical explanations produced by different subcategories under our taxonomy. **(Row 1)** When considering *rule as explanation for local interpretability*, an example is to provide rule explanations which only apply to a given input $\mathbf{x}^{(i)}$ (and its associated output $\hat{y}^{(i)}$). One of the solutions is to find out (by perturbing the input features and seeing how the output changes) the minimal set of features $\mathbf{x}_k \dots \mathbf{x}_l$ whose presence supports the prediction $\hat{y}^{(i)}$. Analogously, features $\mathbf{x}_m \dots \mathbf{x}_n$ can be found which should not be present (larger values), otherwise $\hat{y}^{(i)}$ will change. Then an explanation rule for $\mathbf{x}^{(i)}$ can be constructed as “it is because $\mathbf{x}_k \dots \mathbf{x}_l$ are present and $\mathbf{x}_m \dots \mathbf{x}_n$ are absent that $\mathbf{x}^{(i)}$ is classified as $\hat{y}^{(i)}$ ” [69]. If a rule is valid not only for

the input $\mathbf{x}^{(i)}$, but also for its “neighbourhood” [65], we obtain a *semi-local interpretability*. And if a rule set or decision tree is extracted from the original network, it explains the general function of the whole network and thus provides *global interpretability*. **(Row 2)** When it comes to *explaining the hidden semantics*, a typical example (*global*) is to visualize what pattern a hidden neuron is mostly sensitive to. This can then provide clues about the inner workings of the network. We can also take a more pro-active approach to make hidden neurons more interpretable. As a high-layer hidden neuron may learn a mixture of patterns that can be hard to interpret, Zhang *et al.* [70] introduced a loss term that makes high-layer filters either produce consistent activation maps (among different inputs) or keep inactive (when not seeing a certain pattern). Experiments show that those filters are more interpretable (e.g., a filter may be found to be activated by the head parts of animals). **(Row 3)** *Attribution as explanation* usually provides *local interpretability*. Thinking about an animal

TABLE III
AN OVERVIEW OF THE INTERPRETABILITY PAPERS

		Local	Semi-local	Global
Passive	Rule	CEM ^[69] , CDRPs ^[77] , CVE ² ^[78] , DACE ^[79]	Anchors ^[65] , Interpretable partial substitution ^[80]	KT ^[81] , MoFN ^[82] , NeuralRule ^[83] , NeuroLinear ^[84] , GRG ^[85] , Gyan ^{FO} ^[86] , FZ ^[87] , [88], Trepan ^[89] , • ^[90] , DecText ^[91] , Global model on CEM ^[92]
	Hidden semantics	(*No explicit methods but many in the below cell can be applied here.)	—	Visualization ^[71] , [93]–[98], Network dissection ^[21] , Net2Vec ^[99] , Linguistic correlation analysis ^[100]
	Attribution ¹	LIME ^[20] , MAPLE ^[101] , Partial derivatives ^[71] , DeconvNet ^[72] , Guided backprop ^[102] , Guided Grad-CAM ^[103] , Shapley values ^{[104]–[107]} , Sensitivity analysis ^[72] , [108], [109], Feature selector ^[110] , Bias attribution ^[111]	DeepLIFT ^[112] , LRP ^[113] , Integrated gradients ^[114] , Feature selector ^[110] , MAME ^[68]	Feature selector ^[110] , TCAV ^[63] , ACE ^[115] , SpRAY ³ ^[67] , MAME ^[68] , DeepConsensus ^[116]
	By example	Influence functions ^[73] , Representer point selection ^[117]	—	—
Active	Rule	—	Regional tree regularization ^[118]	Tree regularization ^[119]
	Hidden semantics	—	—	“One filter, one concept” ^[70]
	Attribution	ExpO ^[120] , DAPr ^[121]	—	Dual-net (feature importance) ^[122]
	By example	—	—	Network with a prototype layer ^[76] , ProtoPNet ^[123]

FO First-order rule

FZ Fuzzy rule

¹ Some attribution methods (e.g., DeconvNet, Guided Backprop) arguably have certain non-locality because of the rectification operation.

² Short for counterfactual visual explanations

³ SpRAY is flexible to provide semi-local or global explanations by clustering local (individual) attributions.

classification task, input features are all the pixels of the input image. Attribution allows people to see which regions (pixels) of the image contribute the most to the classification result. The attribution can be computed e.g., by sensitivity analysis in terms of the input features (i.e. all pixels) or some variants [71], [72]. For *attribution for global interpretability*, deep neural networks usually cannot have as straightforward attribution as e.g., coefficients w in linear models $y = w^T x + b$, which directly show the importance of features globally. Instead of concentrating on input features (pixels), Kim *et al.* [63] were interested in attribution to a “concept” (e.g., how sensitive is a prediction of zebra to the presence of stripes). The concept (stripes) is represented by the normal vector to the plane which separates having-stripes and non-stripes training examples in the space of network’s hidden layer. It is therefore possible to compute how sensitive the prediction (of zebra) is to the concept (presence of stripes) and thus have some form of global interpretability. **(Row 4)** Sometimes researchers explain network prediction by *showing other known examples* providing similar network functionality. To explain a single input $x^{(i)}$ (*local interpretability*), we can find an example which is most similar to $x^{(i)}$ in the network’s hidden layer level. This selection of explanation examples can also be done by testing how much the prediction of $x^{(i)}$ will be affected if a certain example is removed from the training set [73]. To provide *global interpretability by showing examples*, a method is adding a (learnable) prototype layer to a network. The prototype layer forces the network to make predictions according to the proximity between input and the

learned prototypes. Those learned and interpretable prototypes can help to explain the network’s overall function.

With the three dimensions introduced above, we can visualize the distribution of the existing interpretability papers in a 3D view (Fig. 2 only provides a 2D snapshot, we encourage readers to visit the online interactive version for better presentation). Table III is another representation of all the reviewed interpretability approaches which is good for quick navigation.

In the following the sections, we will scan through Table III along each dimension. The first dimension results in two sections, passive methods (Section III) and active methods (Section IV). We then expand each section to several subsections according to the second dimension (type of explanation). Under each subsection, we introduce (semi-)local vs. global interpretability methods respectively.

III. PASSIVE INTERPRETATION OF TRAINED NETWORKS

Most of the existing network interpreting methods are passive methods. They try to understand the already trained networks. We now introduce these methods according to their types of produced explanations (i.e. the second dimension).

A. Passive, Rule as Explanation

Logic rules are commonly acknowledged to be interpretable and have a long history of research. Thus rule extraction is an appealing approach to interpret neural networks. In most cases,

rule extraction methods provide *global* explanations as they only extract a single rule set or decision tree from the target model. There are only a few methods producing (*semi*-)*local* rule-form explanations which we will introduce below (Section III-A1), followed are global methods (Section III-A2). Another thing to note is that although the rules and decision trees (and their extraction methods) can be quite different, we do not explicitly differentiate them here as they provide similar explanations (a decision tree can be flattened to a decision rule set). A basic form of a rule is

If P , then Q .

where P is called the antecedent, and Q is called the consequent, which in our context is the prediction (e.g., class label) of a network. P is usually a combination of conditions on several input features. For complex models, the explanation rules can be of other forms such as the propositional rule, first-order rule or fuzzy rule.

1) *Passive, Rule as Explanation, (Semi-)local*: According to our taxonomy, methods in this category focus on a trained neural network and a certain input (or a small group of inputs), and produce a logic rule as an explanation. Dhurandhar *et al.* [69] construct local rule explanations by finding out features that should be minimally and sufficiently *present* and features that should be minimally and necessarily *absent*. In short, the explanation takes this form “If an input x is classified as class y , it is because features f_i, \dots, f_k are present and features f_m, \dots, f_p are absent”. This is done by finding small sparse perturbations that are sufficient to ensure the same prediction by its own (or will change the prediction if applied to a target input)². A similar kind of methods is counterfactual explanations [124]. Usually, we are asking based on what features (x ’s values) the neural network makes the prediction of class c . However, Goyal *et al.* [78] try to find the minimum-edit on an input image which can result in a different predicted class c' . In other words, they ask: “What region in the input image makes the prediction to be class c , rather than c' ”. Kanamori *et al.* [79] introduced distribution-aware counterfactual explanations, which require above “edit” to follow the empirical data distribution instead of being arbitrary.

Wang *et al.* [77] came up with another local interpretability method, which identifies *critical data routing paths* (CDRPs) of the network for each input. In convolutional neural networks, each kernel produces a feature map that will be fed into the next layer as a channel. Wang *et al.* [77] associated every output channel on each layer with a gate (non-negative weight), which indicates how critical that channel is. These gate weights are then optimized such that when they are multiplied with the corresponding output channels, the network can still make the same prediction as the original network (on a given input). Importantly, the weights are encouraged to be sparse (most are close to zero). CDRPs can then be identified for each input by first identifying the *critical nodes*, i.e. the intermediate kernels associated with positive gates. We can explore and assign meanings to the critical nodes so that the critical paths

become local explanations. However, as the original paper did not go further on the CDRPs representation which may not be human-understandable, it is still more of an activation pattern than a real explanation.

We can also extract rules that cover a group of inputs rather than a single one. Ribeiro *et al.* [65] propose *anchors* which are if-then rules that are sufficiently precise (semi-)locally. In other words, if a rule applies to a group of similar examples, their predictions are (almost) always the same. It is similar to (actually, on the basis of) an attribution method LIME, which we will introduce in Section III-C. However, they are different in terms of the produced explanations (LIME produces attribution for individual examples). Wang *et al.* [80] attempted to find an interpretable partial substitution (a rule set) to the network that covers a certain subset of the input space. This substitution can be done with no or low cost on the model accuracy according to the size of the subset.

2) *Passive, Rule as Explanation, Global*: Most of the time, we would like to have some forms of an overall interpretation of the network, rather than its local behaviour at a single point. We again divide these approaches into two groups. Some rule extraction methods make use of the network-specific information such as the network structure, or the learned weights. These methods are called *decompositional* approaches in previous literature [125]. The other methods instead view the network as a black-box and only use it to generate training examples for classic rule learning algorithms. They are called *pedagogical* approaches.

a) *Decompositional approaches*: Decompositional approaches generate rules by observing the connections in a network. As many of these approaches were developed before the deep learning era, they are mostly designed for classic fully-connected feedforward networks. Considering a single-layer setting of a fully-connected network (only one output neuron),

$$y = \sigma \left(\sum_i w_i x_i + b \right)$$

where σ is an activation function (usually sigmoid, $\sigma(x) = 1/(1 + e^{-x})$), w are the trainable weights, x is the input vector, and b is the bias term (often referred as a threshold θ is the early time, and b here can be interpreted as the negation of θ). Lying at the heart of rule extraction is to search for combinations of certain values (or ranges) of attributes x_i that make y near 1 [82]. This is tractable only when we are dealing with small networks because the size of the search space will soon grow to an astronomical number as the number of attributes and the possible values for each attribute increase. Assuming we have n Boolean attributes x_i as an input, and each attribute can be *true* or *false* or *absent* in the antecedent, there are $O(3^n)$ combinations to search. We therefore need some search strategies.

One of the earliest methods is the KT algorithm [81]. KT algorithm first divides the input attributes into two groups, *pos-atts* (short for positive attributes) and *neg-atts*, according to the signs of their corresponding weights. Assuming activation function is sigmoid, all the neurons are booleanized to *true* (if close enough to 1) or *false* (close to 0). Then, all combinations of *pos-atts* are selected if the combination can on its own

²The authors also extended this method to learn a *global* interpretable model, e.g., a decision tree, based on custom features created from above *local* explanations [92].

make y be `true` (larger than a pre-defined threshold β without considering the neg-atts), for instance, a combination $\{x_1, x_3\}$ and $\sigma(\sum_{i \in \{1,3\}} w_i x_i + b) > \beta$. Finally, it takes into account the neg-atts. For each above pos-atts combination, it finds combinations of neg-atts (e.g., $\{x_2, x_5\}$) that when absent the output calculated from the selected pos-atts and unselected neg-atts is still `true`. In other words, $\sigma(\sum_{i \in \mathcal{I}} w_i x_i + b) > \beta$, where $\mathcal{I} = \{x_1, x_3\} \cup \{\text{neg-atts}\} \setminus \{x_2, x_5\}$. The extracted rule can then be formed from the combination \mathcal{I} and has the output class 1. In our example, the translated rule is

If $x_1(\text{is true}) \wedge x_3 \wedge \neg x_2 \wedge \neg x_5$, then $y = 1$.

Similarly, this algorithm can generate rules for class 0 (searching neg-atts first and then adding pos-atts). To apply to the multi-layer network situation, it first does layer-by-layer rule generation and then rewrites them to omit the hidden neurons. In terms of the complexity, KT algorithm reduces the search space to $\mathcal{O}(2^n)$ by distinguishing pos-atts and neg-atts (pos-atts will be either `true` or `absent`, and neg-atts will be either `false` or `absent`). It also limits the number of attributes in the antecedent, which can further decrease the algorithm complexity (with the risk of missing some rules).

Towell and Shavlik [82] focus on another kind of rules of “ M -of- N ” style. This kind of rule de-emphasizes the individual importance of input attributes, which has the form

If M of these N expressions are `true`, then Q .

This algorithm has two salient characteristics. The first one is link (weight) clustering and reassigning them the average weight within the cluster. Another characteristic is network simplifying (eliminating unimportant clusters) and re-training. Comparing with the exponential complexity of subset searching algorithm, M -of- N method is approximately cubic because of its special rule form.

NeuroRule [83] introduced a three-step procedure of extracting rules: 1) train the network and prune, 2) discretize (cluster) the activation values of the hidden neurons, 3) extract the rules layer-wise and rewrite (similar as previous methods). NeuroLinear [84] made a little change to the NeuroRule method, allowing neural networks to have continuous input. Andrews *et al.* [126] and Tickle *et al.* [127] provide a good summary of the rule extraction techniques before 1998.

b) Pedagogical approaches. By treating the neural network as a black-box, *pedagogical* methods (or hybrids of both) directly learn rules from the examples generated by the network. It is essentially reduced to a traditional rule learning or decision tree learning problem. For rule set learning, we have sequential covering framework (i.e. to learn rules one by one). And for decision tree, there are many classic algorithms like CART [128] and C4.5 [129]. Example work of decision tree extraction (from neural networks) can be found in references [89]–[91].

Odajima *et al.* [85] followed the framework of NeuroLinear but use a greedy form of sequential covering algorithm to extract rules. It is reported to be able to extract more concise and precise rules. Gyan method [86] goes further than extracting propositional rules. After obtaining the propositional rules by the above methods, Gyan uses the Least General Generalization (LGG [130]) method to generate first-order logic rules from them. There are also some approaches attempting to extract

fuzzy logic from trained neural networks [87], [88], [131]. The major difference is the introduction of the membership function of linguistic terms. An example rule is

If $(x_1 = \text{high}) \wedge \dots$, then $y = \text{class1}$.

where *high* is a fuzzy term expressed as a fuzzy set over the numbers.

Most of the above “Rule as Explanation, Global” methods were developed in the early stage of neural network research, and usually were only applied to relatively small datasets (e.g., the Iris dataset, the Wine dataset from the UCI Machine Learning Repository). However, as neural networks get deeper and deeper in recent applications, it is unlikely for a single decision tree to faithfully approximate the behaviour of deep networks. We can see that more recent “Rule as Explanation” methods turn to local or semi-local interpretability [80], [118].

B. Passive, Hidden Semantics as Explanation

The second typical kind of explanations is the meaning of hidden neurons or layers. Similar to the grandmother cell hypothesis³ in neuroscience, it is driven by a desire to associate abstract concepts with the activation of some hidden neurons. Taking animal classification as an example, some neurons may have high response to the head of an animal while others neurons may look for bodies, feet or other parts. This kind of explanations by definition provides *global* interpretability.

1) *Passive, Hidden Semantics as Explanation, Global:* Existing hidden semantics interpretation methods mainly focus on the computer vision field. The most direct way is to show what the neuron is “looking for,” i.e. visualization. The key to visualization is to find a representative input that can maximize the activation of a certain neuron, channel or layer, which is usually called *activation maximization* [93]. This is an optimization problem, whose search space is the potentially huge input (sample) space. Assuming we have a network taking as input a 28×28 pixels black and white image (as in the MNIST handwritten digit dataset), there will be $2^{28 \times 28}$ possible input images, although most of them are probably nonsensical. In practice, although we can find a maximum activation input image with optimization, it will likely be unrealistic and uninterpretable. This situation can be helped with some regularization techniques or priors.

We now give an overview over these techniques. The framework of activation maximization was introduced by Erhan *et al.* [93] (although it was used in the unsupervised deep models like Deep Belief Networks). In general, it can be formulated as

$$\mathbf{x}^* = \arg \max_{\mathbf{x}} (act(\mathbf{x}; \theta) - \lambda \Omega(\mathbf{x}))$$

where $act(\cdot)$ is the activation of the neuron of interest, θ is the network parameters (weights and biases) and Ω is an optional regularizer. (We use the bold upright \mathbf{x} to denote an input matrix in image related tasks, which allows row and column indices i and j .)

Simonyan *et al.* [71] for the first time applied activation maximization to a supervised deep convolutional network (for ImageNet classification). It finds representative images by maximizing the score of a class (before softmax). And the Ω is the L_2

³[Online]. Available: https://en.wikipedia.org/wiki/Grandmother_cell

norm of the image. Later, people realized high frequency noise is a major nuisance that makes the visualization unrecognizable [94], [98]. In order to get natural and useful visualizations, finding good priors or regularizers Ω becomes the core task in this kind of approaches.

Mahendran and Vedaldi [95] propose a regularizer *total variation*

$$\Omega(\mathbf{x}) = \sum_{i,j} \left((\mathbf{x}_{i,j+1} - \mathbf{x}_{i,j})^2 + (\mathbf{x}_{i+1,j} - \mathbf{x}_{i,j})^2 \right)^{\frac{\beta}{2}}$$

which encourages neighbouring pixels to have similar values. This can also be viewed as a low-pass filter that can reduce the high frequency noise in the image. This kind of methods is usually called image blurring. Besides suppressing high amplitude and high frequency information (with L_2 decay and Gaussian blurring respectively), Yosinski *et al.* [96] also include other terms to clip the pixels of small values or little importance to the activation. Instead of using many hand-crafted regularizers (image priors), Nguyen *et al.* [97] suggest using natural image prior learned by a generative model. As Generative Adversarial Networks (GANs) [132] showed recently great power to generate high-resolution realistic images [133], [134], making use of the generative model of a GAN appears to be a good choice. For a good summary and many impressive visualizations, we refer the readers to [98]. When applied to certain tasks, researchers can get some insights from these visual interpretations. For example, Minematsu *et al.* [135], [136] inspected the behaviour of the first and last layers of a DNN used for change detection (in video stream), which may suggest the possibility of a new background modelling strategy.

Besides visualization, there are also some work trying to find connections between kernels and visual concepts (e.g., materials, certain objects). Bau *et al.* [21] (Network Dissection) collected a new dataset Broden which provides a pixel-wise binary mask $L_c(\mathbf{x})$ for every concept c and each input image \mathbf{x} . The activation map of a kernel k is upsampled and converted (given a threshold) to a binary mask $M_k(\mathbf{x})$ which has the same size of \mathbf{x} . Then the alignment between a kernel k and a certain concept c (e.g., car) is computed as

$$IoU_{k,c} = \frac{\sum |M_k(\mathbf{x}) \cap L_c(\mathbf{x})|}{\sum |M_k(\mathbf{x}) \cup L_c(\mathbf{x})|}$$

where $|\cdot|$ is the cardinality of a set and the summation \sum is over all the inputs \mathbf{x} that contains the concept c . Along the same lines, Fong and Vedaldi [99] investigate the embeddings of concepts over multiple kernels by constructing M with a combination of several kernels. Their experiments show that multiple kernels are usually required to encode one concept and kernel embeddings are better representations of the concepts.

Dalvi *et al.* [100] also analysed the meaning of individual units/neurons in the networks for NLP tasks. They build a linear model between the network's hidden neurons and the output. The neurons are then ranked according to the significance of the weights of the linear model. For those top-ranking neurons, their linguistic meanings are investigated by visualizing their saliency maps on the inputs, or by finding the top words by which they get activated.

C. Passive, Attribution as Explanation

Attribution is to assign credit or blame to the input features in terms of their impact on the output (prediction). The explanation will be a real-valued vector which indicates feature importance with the sign and amplitude of the scores [58]. For simple models (e.g., linear models) with meaningful features, we might be able to assign each feature a score *globally*. When it comes to more complex networks and input, e.g., images, it is hard to say a certain pixel always has similar contribution to the output. Thus, many methods do attribution *locally*. We introduce them below and at the end of this section we mention a global attribution method on intermediate representation rather than the original input features.

1) *Passive, Attribution as Explanation, (Semi-)local*: Similarly to the compositional vs. pedagogical division of rule extraction methods, attribution methods can be also divided into two groups: gradient-related methods and model agnostic methods.

a) *Gradient-related and backpropagation methods*. Using gradients to explain the individual classification decisions is a natural idea as the gradient represents the “direction” and rate of the fastest increase on the loss function. The gradients can also be computed with respect to a certain output class, for example, along which “direction” a perturbation will make an input more/less likely predicted as a cat/dog. Baehrens *et al.* [137] use it to explain the predictions of Gaussian Process Classification (GPC), k -NN and SVM. For a special case, the coefficients of features in linear models (or general additive models) are already the partial derivatives, in other words, the (global) attribution. So people can directly know how the features affect the prediction and that is an important reason why linear models are commonly thought interpretable. While plain gradients, discrete gradients and path-integrated gradients have been used for attribution, some other methods do not calculate real gradients with the chain rule but only backpropagate attribution signals (e.g., do extra normalization on each layer upon backpropagation). We now introduce these methods in detail.

In computer vision, the attribution is usually represented as a saliency map, a mask of the same size of the input image. In reference [71], the saliency map is generated from the gradients (specifically, the maximum absolute values of the partial derivatives over all channels). This kind of saliency maps are obtained without effort as they only require a single backpropagation pass. They also showed that this method is equivalent to the previously proposed deconvolutional nets method [72] except for the difference on the calculation of ReLU layer's gradients. Guided backpropagation [102] combines above two methods. It only takes into account the gradients (the former method) that have positive error signal (the latter method) when backpropagating through a ReLU layer. There is also a variant Guided Grad-CAM (Gradient-weighted Class Activation Mapping) [103], which first calculates a coarse-grained attribution map (with respect to a certain class) on the last convolutional layer and then multiply it to the attribution map obtained from guided backpropagation. (Guided Grad-CAM is an extension of CAM [138] which requires a special global average pooling layer.)

TABLE IV

FORMULATION OF GRADIENT-RELATED ATTRIBUTION METHODS. S_c IS THE OUTPUT FOR CLASS c (AND IT CAN BE ANY NEURON OF INTEREST), σ IS THE NONLINEARITY IN THE NETWORK AND g IS A REPLACEMENT OF σ' (THE DERIVATIVE OF σ) IN $\frac{\partial S_c(\mathbf{x})}{\partial \mathbf{x}}$ IN ORDER TO REWRITE DEEPLIFT AND LRP WITH GRADIENT FORMULATION (SEE [139] FOR MORE DETAILS). \mathbf{x}_i IS THE i -TH FEATURE (PIXEL) OF \mathbf{x}

Method	Attribution
Gradient [71], [137]	$\frac{\partial S_c(\mathbf{x})}{\partial \mathbf{x}_i}$
Gradient \odot Input	$\mathbf{x}_i \cdot \frac{\partial S_c(\mathbf{x})}{\partial \mathbf{x}_i}$
LRP [113]	$\mathbf{x}_i \cdot \frac{\partial^g S_c(\mathbf{x})}{\partial \mathbf{x}_i}, g = \frac{\sigma(\mathbf{z})}{\mathbf{z}}$
DeepLIFT [112]	$(\mathbf{x}_i - \mathbf{x}_i^{\text{ref}}) \cdot \frac{\partial^g S_c(\mathbf{x})}{\partial \mathbf{x}_i}, g = \frac{\sigma(\mathbf{z}) - \sigma(\mathbf{z}^{\text{ref}})}{\mathbf{z} - \mathbf{z}^{\text{ref}}}$
Integrated Gradient [114]	$(\mathbf{x}_i - \mathbf{x}_i^{\text{ref}}) \cdot \int_0^1 \frac{\partial S_c(\tilde{\mathbf{x}})}{\partial \tilde{\mathbf{x}}_i} \Big _{\tilde{\mathbf{x}}=\mathbf{x}^{\text{ref}}+\alpha(\mathbf{x}-\mathbf{x}^{\text{ref}})} d\alpha$

However, gradients themselves can be misleading. Considering a piecewise continuous function,

$$y = \begin{cases} \mathbf{x}_1 + \mathbf{x}_2 & \text{if } \mathbf{x}_1 + \mathbf{x}_2 < 1; \\ 1 & \text{if } \mathbf{x}_1 + \mathbf{x}_2 \geq 1. \end{cases}$$

it is saturated when $\mathbf{x}_1 + \mathbf{x}_2 \geq 1$. At points where $\mathbf{x}_1 + \mathbf{x}_2 > 1$, their gradients are always zeros. DeepLIFT [112] points out this problem and highlights the importance of having a reference input besides the target input to be explained. The reference input is a kind of default or ‘neutral’ input and will be different in different tasks (e.g., blank images or zero vectors). Actually, as Sundararajan *et al.* [114] point out, DeepLIFT is trying to compute the “discrete gradient” instead of the (instantaneous) gradient. Another similar “discrete gradient” method is LRP [113] (choosing a zero vector as the reference point), differing in how to compute the discrete gradient. This view is also present in reference [139] that LRP and DeepLIFT are essentially computing backpropagation for modified gradient functions.

However, discrete gradients also have their drawbacks. As the chain rule does not hold for discrete gradients, DeepLIFT and LRP adopt modified forms of backpropagation. This makes their attributions specific to the network implementation, in other words, the attributions can be different even for two functionally equivalent networks (a concrete example can be seen in reference [114] appendix B). Integrated gradients [114] have been proposed to address this problem. It is defined as the path integral of all the gradients in the straight line between input \mathbf{x} and the reference input \mathbf{x}^{ref} . The i -th dimension of the integrated gradient (IG) is defined as follows,

$$\text{IG}_i(\mathbf{x}) \doteq (\mathbf{x}_i - \mathbf{x}_i^{\text{ref}}) \int_0^1 \frac{\partial f(\tilde{\mathbf{x}})}{\partial \tilde{\mathbf{x}}_i} \Big|_{\tilde{\mathbf{x}}=(\mathbf{x}^{\text{ref}}+\alpha(\mathbf{x}-\mathbf{x}^{\text{ref}}))} d\alpha$$

where $\frac{\partial f(\mathbf{x})}{\partial \mathbf{x}_i}$ is the i -th dimension of the gradient of $f(\mathbf{x})$. For those attribution methods requiring a reference point, semi-local interpretability is provided as users can select different reference points according to what to explain. Table IV summarizes the above gradient-related attribution methods (adapted from [139]). In addition to the “gradient” attribution discussed

above, Wang *et al.* [111] point out that bias terms can contain attribution information complementary to the gradients. They propose a method to recursively assign the bias attribution back to the input vector.

Those discrete gradient methods (e.g., LRP and DeepLIFT) provide semi-local explanations as they explain a target input w.r.t. another reference input. But methods such as DeconvNet and Guided Backprop, which are only proposed to explain individual inputs, arguably have certain non-locality because of the rectification operation during the process. Moreover, one can accumulate multiple local explanations to achieve a certain degree of global interpretability, which will be introduced in Section III-C2.

Although we have many approaches to produce plausible saliency maps, there is still a small gap between saliency maps and real explanations. There have even been adversarial examples for attribution methods, which can produce perceptively indistinguishable inputs, leading to the *same predicted labels*, but very *different attribution maps* [140]–[142]. Researchers came up with several properties a saliency map should have to be a valid explanation. Sundararajan *et al.* [114] (integrated gradients method) introduced two requirements, *sensitivity* and *implementation invariance*. Sensitivity requirement is proposed mainly because of the (local) gradient saturation problem (which results in zero gradient/attribution). Implementation invariance means two functionally equivalent networks (which can have different learned parameters given the over-parametrizing setting of DNNs) should have the same attribution. Kindermans *et al.* [143] introduced input invariance. It requires attribution methods to mirror the model’s invariance with respect to transformations of the input. For example, a model with a bias term can easily deal with a constant shift of the input (pixel values). Obviously, (plain) gradient attribution methods satisfy this kind of input invariance. For discrete gradient and other methods using reference points, they depend on the choices of reference. Adebayo *et al.* [75] took a different approach. They found that edge detectors can also produce masks which look similar to saliency masks and highlight some features of the input. But edge detectors have nothing to do with the network or training data. Thus, they proposed two tests to verify whether the attribution method will fail (1) if the network’s weights are replaced with random noise, or (2) if the labels of training data are shuffled. The attribution methods should fail otherwise it suggests that the method does not reflect the trained network or the training data (in other words, it is just something like an edge detector).

b) Model agnostic attribution: LIME [20] is a well-known approach which can provide local attribution explanations (if choosing linear models as the so-called interpretable components). Let $f: \mathbb{R}^d \rightarrow \{+1, -1\}$ be a (binary classification) model to be explained. Because the original input $\mathbf{x} \in \mathbb{R}^d$ might be uninterpretable (e.g., a tensor of all the pixels in an image, or a word embedding [144]), LIME introduces an intermediate representation $\mathbf{x}' \in \{0, 1\}^d$ (e.g., the existence of certain image patches or words). \mathbf{x}' can be recovered to the original input space \mathbb{R}^d . For a given \mathbf{x} , LIME tries to find a potentially interpretable model g (such as a linear model or decision tree) as a local explanation.

$$g_{\mathbf{x}} = \arg \min_{g \in G} L(f, g, \pi_{\mathbf{x}}) + \Omega(g)$$

where G is the explanation model family, L is the loss function that measures the fidelity between f and g . L is evaluated on a set of perturbed samples around x' (and their recovered input), which are weighted by a local kernel π_x . Ω is the complexity penalty of g , ensuring g to be interpretable. MAPLE [101] is a similar method using local linear models as explanations. The difference is it defines the locality as how frequently the data points fall into a same leaf node in a proxy random forest (fit on the trained network).

In game theory, there is a task to “fairly” assign each player a payoff from the total gain generated by a coalition of all players. Formally, let N be a set of n players, $v : 2^N \rightarrow \mathbb{R}$ is a characteristic function, which can be interpreted as the total gain of the coalition N . Obviously, $v(\emptyset) = 0$. A coalitional game can be denoted by the tuple $\langle N, v \rangle$. Given a coalitional game, Shapley value [145] is a solution to the payoff assignment problem. The payoff (attribution) for player i can be computed as follows,

$$\phi_i(v) = \frac{1}{|N|} \sum_{S \subseteq N \setminus \{i\}} \binom{|N| - 1}{|S|} (v(S \cup \{i\}) - v(S))$$

where $v(S \cup \{i\}) - v(S)$ is the marginal contribution of player i to coalition S . The rest of the formula can be viewed as a normalization factor. A well-known alternative form of Shapley value is

$$\phi_i(v) = \frac{1}{|N|!} \sum_{O \in \mathfrak{S}(N)} [v(P_i^O \cup \{i\}) - v(P_i^O)]$$

where $\mathfrak{S}(N)$ is the set of all ordered permutations of N , and P_i^O is the set of players in N which are predecessors of player i in the permutation O . Štrumbelj and Kononenko adopted this form so that v can be approximated in polynomial time [104] (also see [106] for another approximation method).

Back to the neural network (denoted by f), let N be all the input features (attributes), S is an arbitrary feature subset of interest ($S \subseteq N$). For an input x , the characteristic function $v(S)$ is the difference between the expected model output when we know all the features in S , and the expected output when no feature value is known (i.e. the expected output over all possible input), denoted by

$$v(S) = \frac{1}{|\mathcal{X}^{N \setminus S}|} \sum_{y \in \mathcal{X}^{N \setminus S}} f(\tau(x, y, S)) - \frac{1}{|\mathcal{X}^N|} \sum_{z \in \mathcal{X}^N} f(z)$$

\mathcal{X}^N and $\mathcal{X}^{N \setminus S}$ are respectively the input space containing feature sets N and $N \setminus S$. $\tau(x, y, S)$ is a vector composed by x and y according to whether the feature is in S .

However, a practical problem is the exponential computation complexity, let alone the cost of the feed-forward computing on each $v(S)$ call. Štrumbelj and Kononenko [104] approximate Shapley value by sampling from $\mathfrak{S}(N) \times \mathcal{X}$ (Cartesian product). There are other variants such as using different v . More can be found in reference [105] which proposes a unified view including not only the Shapley value methods but also LRP and DeepLIFT. There is also Shapley value through the lens of causal graph [107].

Sensitivity analysis can also be used to evaluate the importance of a feature. Specifically, the importance of a feature could

be measured as how much the model output will change upon the change of a feature (or features). There are different kinds of changes, e.g., perturbation, occlusion [72], [108] etc. [109].

Chen *et al.* [110] propose an instance-wise feature selector \mathcal{E} which maps an input x to a conditional distribution $P(S | x)$, where S is any subset (of certain size) of the original feature set. The selected features can be denoted by x_S . Then they aim to maximize the mutual information between the selected features and the output variable Y ,

$$\max_{\mathcal{E}} I(X_S; Y) \quad \text{subject to} \quad S \sim \mathcal{E}(X).$$

A variational approximation is used to obtain a tractable solution of the above problem.

2) *Passive, Attribution as Explanation, Global*: A natural way to get global attribution is to combine individual ones obtained from above local/semi-local methods. SpRAY [67] clusters on the individual attributions and then summarizes some groups of prediction strategies. MAME [68] is a similar method that can generate a multilevel (local to global) explanation tree. Salman *et al.* [116] provide a different way, which makes use of multiple neural networks. Each of the network can provide its own local attributions, on top of which a clustering is performed. Those clusters, intuitively the consensus of multiple models, can provide more robust interpretations.

The attribution does not necessarily attribute ‘credits’ or ‘blame’ to the *raw* input or features. Kim *et al.* [63] propose a method TCAV (quantitative Testing with Concept Activation Vectors) that can compute the model sensitivity of any user-interested concept. By first collecting some examples with and without a target concept (e.g., the presence of stripes in an animal), the concept can then be represented by a normal vector to the hyperplane separating those positive/negative examples (pictures of animals with/without stripes) in a hidden layer. The score of the concept can be computed as the (average) output sensitivity if the hidden layer representation (of an input x) moves an infinitesimally small step along the concept vector. This is a global interpretability method as it explains how a concept affects the output in general. Besides being manually picked by a human, these concepts can also be discovered automatically by clustering input segments [115].

D. Passive, Explanation by Example

The last kind of explanations we reviewed is explanation by example. When asked for an explanation for a new input, these approaches return other example(s) for supporting or counter example. One basic intuition is to find examples that the model considers to be most similar (in terms of latent representations) [146]. This is *local* interpretability but we can also seek a set of representative samples within a class or for more classes that provides *global* interpretability. A general approach is presented in [147]. There are other methods, such as measuring how much a training example affects the model prediction on a target input. Here we only focus on work related to deep neural networks.

1) *Passive, Explanation by Example, Local*: Koh and Liang [73] provide an interesting method to evaluate how much a training example affects the model prediction on an unseen test example. The change of model parameters upon a change of

training example is first calculated with approximation. Further, its influence on the loss at the test point can be computed. By checking the most influential (positively or negatively) training examples (for the test example), we can have some insights on the model predictions. Yeh *et al.* [117] show that the logit (the neuron before softmax) can be decomposed into a linear combination of training points' activations in the pre-logit layer. The coefficients of the training points indicate whether the similarity to those points is excitatory or inhibitory. The above two approaches both provide local explanations.

IV. ACTIVE INTERPRETABILITY INTERVENTION DURING TRAINING

Besides passive looking for human-understandable patterns from the trained network, researchers also tried to impose interpretability restrictions during the network training process, i.e. active interpretation methods in our taxonomy. A popular idea is to add a special regularization term $\Omega(\theta)$ to the loss function, also known as “interpretability loss” (θ collects all the weights of a network). We now discuss the related papers according to the forms of explanations they provide.

A. Active, *Rule as Explanation* (Semi-Local or Global)

Wu *et al.* [119] propose tree regularization which favours models that can be well approximated by shallow decision trees. It requires two steps (1) train a binary decision tree using data points $\{\mathbf{x}^{(i)}, \hat{y}^{(i)}\}_N$, where $\hat{y} = f_\theta(\mathbf{x})$ is the network prediction rather than the true label, (2) calculate the average path length (from root to leaf node) of this decision tree over all the data points. However, this tree regularization term $\Omega(\theta)$ is not differentiable. Therefore, a surrogate regularization term $\hat{\Omega}(\theta)$ was introduced. Given a dataset $\{\theta^{(j)}, \Omega(\theta^{(j)})\}_{j=1}^J$, $\hat{\Omega}$ can be trained as a multi-layer perceptron network which minimizes the squared error loss

$$\min_{\xi} \sum_{j=1}^J \left(\Omega(\theta^{(j)}) - \hat{\Omega}(\theta^{(j)}; \xi) \right)^2 + \epsilon \|\xi\|_2^2$$

$\{\theta^{(j)}, \Omega(\theta^{(j)})\}_{j=1}^J$ can be assembled during network training. Also, data augmentation techniques can be used to generate θ , especially in the early training phase. Tree regularization enables global interpretability as it forces a network to be easily approximable by a decision tree. Later, the authors also proposed regional tree regularization which did this in a semi-local way [118].

B. Active, *Hidden Semantics as Explanation* (global)

Another method aims to make a convolutional neural network learn better (disentangled) hidden semantics. Having seen feature visualization techniques mentioned above and some empirical studies [21], [148], CNNs are believed to have learned some low-level to high-level representations in the hierarchical structure. But even if higher-layers have learned some object-level concepts (e.g., head, foot), those concepts are usually entangled with each other. In other words, a high-layer filter may contain a mixture of different patterns. Zhang *et al.* [70]

propose a loss term which encourages high-layer filters to represent a single concept. Specifically, for a CNN, a feature map (output of a high-layer filter, after ReLU) is an $n \times n$ matrix. Zhang *et al.* predefined a set of n^2 ideal feature map templates (activation patterns) \mathbf{T} , each of which is like a Gaussian kernel only differing on the position of its peak. During the forward propagation, the feature map is masked (element-wise product) by a certain template $\mathbf{T} \in \mathbf{T}$ according to the position of the most activated “pixel” in the original feature map. During the back propagation, an extra loss is plugged in, which is the mutual information between \mathbf{M} (the feature maps of a filter calculated on all images) and $\mathbf{T} \cup \{\mathbf{T}^-\}$ (all the ideal activation patterns plus a negative pattern which is full of a negative constant). This loss term makes a filter to either have a consistent activation pattern or keep inactivated. Experiments show that filters in their designed architecture are more semantically meaningful (e.g., the “receptive field” [74] of a filter corresponds to the head of animals).

C. Active, *Attribution as Explanation*

Similar to tree regularization which helps to achieve better global interpretability (decision trees), ExpO [120] added an interpretability regularizer in order to improve the quality of local attribution. That regularization requires a model to have fidelitous (high fidelity) and stable local attribution. DAPr [121] (deep attribution prior) took into account additional information (e.g., a rough prior about the feature importance). The prior will be trained jointly with the main prediction model (as a regularizer) and biases the model to have similar attribution as the prior.

Besides performing attribution on individual input (*locally* in input space), Dual-net [122] was proposed to decide feature importance population-wise, i.e., finding an ‘optimal’ feature subset collectively for an input population. In this method, a *selector* network is used to generate an optimal feature subset, while an *operator* network makes predictions based on that feature set. These two networks are trained jointly. After training, the selector network can be used to rank feature importance.

D. Active, *Explanations by Prototypes* (global)

Li *et al.* [76] incorporated a prototype layer to a network (specifically, an autoencoder). The network acts like a prototype classifier, where predictions are made according to the proximity between (the encoded) inputs and the learned prototypes. Besides the cross-entropy loss and the (autoencoder) reconstruction error, they also included two interpretability regularization terms, encouraging every prototype to be similar to at least one encoded input, vice versa. After the network is trained, those prototypes can be naturally used as explanations. Chen *et al.* [123] add a prototype layer to a regular CNN rather than an autoencoder. This prototype layer contains prototypes that are encouraged to resemble parts of an input. When asked for explanations, the network can provide several prototypes for different parts of the input image respectively.

V. EVALUATION OF INTERPRETABILITY

In general, interpretability is hard to evaluate objectively as the end-tasks can be quite divergent and may require domain knowledge from experts [58]. Doshi-Velez and Kim [18] proposed three evaluation approaches: application-grounded, human-grounded, and functionally-grounded. The first one measures to what extent interpretability helps the end-task (e.g., better identification of errors or less discrimination). Human-grounded approaches are, for example, directly letting people evaluate the quality of explanations with human-subject experiments (e.g., let a user choose which explanation is of the highest quality among several explanations). Functionally-grounded methods find proxies for the explanation quality (e.g., sparsity). The last kind of approaches require no costly human experiments but how to properly determine the proxy is a challenge.

In our taxonomy, explanations are divided into different types. Although the interpretability can hardly be compared between different types of explanations, there are some measurements proposed for this purpose. For logic rules and decision trees, the size of the extracted rule model is often used as a criterion [85], [119], [127] (e.g., the number of rules, the number of antecedents per rule, the depth of the decision tree etc.). Strictly speaking, these criteria measure more about whether the explanations are efficiently interpretable. Hidden semantics approaches produce explanations on certain hidden units in the network. Network Dissection [21] quantifies the interpretability of hidden units by calculating their matchiness to certain concepts. As for the hidden unit visualization approaches, there is no a good measurement yet. For attribution approaches, their explanations are saliency maps/masks (or feature importance etc. according to the specific task). Samek *et al.* [149] evaluate saliency maps by the performance degradation if the input image is partially masked with noise in an order from salient to not salient patches. A similar evaluation method is proposed in [75] and Hooker *et al.* [150] suggest using a fixed uninformative value rather than noise as the mask and evaluating performance degradation on a retrained model. Samek *et al.* also use entropy as another measure in the belief that good saliency maps focus on relevant regions and do not contain much irrelevant information and noise. Montavon *et al.* [58] would like the explanation function (which maps an input to a saliency map) to be continuous/smooth, which means the explanations (saliency maps) should not vary too much when seeing similar inputs.

VI. DISCUSSION

In practice, different interpretation methods have their own advantages and disadvantages. Passive (post-hoc) methods have been widely studied because they can be applied in a relatively straightforward manner to most existing networks. One can choose methods that make use of a network's inner information (such as connection weights, gradients), which are usually more efficient (e.g., see Paragraph III-C1). Otherwise there are also model-agnostic methods that have no requirement of the model architecture, which usually compute the marginal effect of a certain input feature. But this generality is also a downside of passive methods, especially because there is no easy way to incorporate specific domain knowledge/priors. Active (interpretability intervention) methods put forward ideas about how

a network should be optimized to gain interpretability. The network can be optimized to be easily fit by a decision tree, or to have preferred feature attribution, better tailored to a target task. However, the other side of the coin is that such active intervention requires the compatibility between networks and interpretation methods.

As for the second dimension, the format of explanations, logical rules are the most clear (do not need further human interpretation). However, one should carefully control the complexity of the explanations (e.g., the depth of a decision tree), otherwise the explanations will not be useful in practice. Hidden semantics essentially explain a subpart of a network, with most work developed in the computer vision field. Attribution is very suitable for explaining individual inputs. But it is usually hard to get some overall understanding about the network from attribution (compared to, e.g., logical rules). Explaining by providing an example has the lowest (the most implicit) explanatory power.

As for the last dimension, local explanations are more useful when we care more about every single prediction (e.g., a credit or insurance risk assessment). For some research fields, such as genomics and astronomy, global explanations are more preferred as they may reveal some general "knowledge". Note, however, there is no hard line separating local and global interpretability. With the help of explanation fusing methods (e.g., MAME), we can obtain multilevel (from local to global) explanations.

VII. CONCLUSION

In this survey, we have provided a comprehensive review of neural network interpretability. First, we have discussed the definition of interpretability and stressed the importance of the format of explanations and domain knowledge/representations. Specifically, there are four commonly seen types of explanations: *logic rules*, *hidden semantics*, *attribution* and *explanations by examples*. Then, by reviewing the previous literature, we summarized 3 essential reasons why interpretability is important: the requirement of high reliability systems, ethical/legal requirements and knowledge finding for science. After that, we introduced a novel taxonomy for the existing network interpretation methods. It evolves along three dimensions: passive vs. active, types of explanations and global vs. local interpretability. The last two dimensions are not purely categorical but with ordinal values (e.g., semi-local). This is the first time we have a coherent overview of interpretability research rather than many isolated problems and approaches. We can even visualize the distribution of the existing approaches in the 3D space spanned by our taxonomy.

From the perspective of the new taxonomy, there are still several possible research directions in the interpretability research. First, the active interpretability intervention approaches are underexplored. Some analysis of the passive methods also suggests that the neural network does not necessarily learn representations which can be easily interpreted by human beings. Therefore, how to actively make a network interpretable without harming its performance is still an open problem. During the survey process, we have seen more and more recent work filling this blank.

Another important research direction may be how to better incorporate domain knowledge in the networks. As we have seen

in this paper, interpretability is about providing explanations. And explanations build on top of understandable terms (or concepts) which can be specific to the targeted tasks. We already have many approaches to construct explanations of different types, but the domain-related terms used in the explanations are still very simple (see Table I). If we can make use of terms that are more domain/task-related, we can get more informative explanations and better interpretability.

ACKNOWLEDGMENT

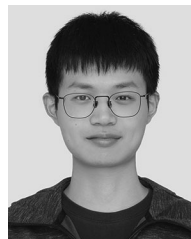
The Authors would like to thank MoD/Dstl, and EPSRC for providing the grant to support the U.K. academics involvement in the Department of Defense funded MURI project through EPSRC under Grant EP/N019415/1.

REFERENCES

- [1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, 2015, pp. 436–444.
- [2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.
- [3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.
- [4] G. Hinton *et al.*, "Deep neural networks for acoustic modeling in speech recognition," *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 82–97, Nov. 2012.
- [5] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 3104–3112.
- [6] D. Silver *et al.*, "Mastering the game of go without human knowledge," *Nature*, vol. 550, 2017, pp. 354–359.
- [7] S. Dong, P. Wang, and K. Abbas, "A survey on deep learning and its applications," *Comput. Sci. Rev.*, 2021.
- [8] P. Dixit and S. Silakari, "Deep learning algorithms for cybersecurity applications: A technological and status review," *Comput. Sci. Rev.*, 2021.
- [9] T. Bouwmans, S. Javed, M. Sultana, and S. K. Jung, "Deep neural network concepts for background subtraction: A systematic review and comparative evaluation," *Neural Netw.*, 2019.
- [10] Y. Wu *et al.*, "Google's neural machine translation system: Bridging the gap between human and machine translation," 2016, *arXiv:1609.08144*.
- [11] M. Wainberg, D. Merico, A. Delong, and B. J. Frey, "Deep learning in biomedicine," *Nature Biotechnol.*, vol. 36, 2018.
- [12] H. Y. Xiong *et al.*, "The human splicing code reveals new insights into the genetic determinants of disease," *Science*, vol. 347, 2015.
- [13] S. Zhang, H. Hu, T. Jiang, L. Zhang, and J. Zeng, "Titer: Predicting translation initiation sites by deep learning," *Bioinformatics*, vol. 33, 2017.
- [14] D. Parks, J. X. Prochaska, S. Dong, and Z. Cai, "Deep learning of quasar spectra to discover and characterize damped Ly α systems," *Monthly Notices Roy. Astronomical Soc.*, vol. 476, 2018.
- [15] C. Szegedy *et al.*, "Intriguing properties of neural networks," *International Conference on Learning Representations*, 2014.
- [16] A. Nguyen, J. Yosinski, and J. Clune, "Deep neural networks are easily fooled: High confidence predictions for unrecognizable images," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 427–436.
- [17] Z. C. Lipton, "The mythos of model interpretability," 2016, *arXiv*.
- [18] F. Doshi-Velez and B. Kim, "Towards a rigorous science of interpretable machine learning," 2017, *arXiv*.
- [19] D. Pedreschi, F. Giannotti, R. Guidotti, A. Monreale, S. Ruggieri, and F. Turini, "Meaningful explanations of black box ai decision systems," in *Proc. AAAI Conf. Artif. Intell.*, 2019, vol. 33, pp. 9780–9784.
- [20] M. T. Ribeiro, S. Singh, and C. Guestrin, "Why should i trust you?: Explaining the predictions of any classifier," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2016, pp. 1135–1144.
- [21] D. Bau, B. Zhou, A. Khosla, A. Oliva, and A. Torralba, "Network dissection: Quantifying interpretability of deep visual representations," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 6541–6549.
- [22] G. D. Stormo, T. D. Schneider, L. Gold, and A. Ehrenfeucht, "Use of the 'perceptron' algorithm to distinguish translational initiation sites in *e. coli*," *Nucleic Acids Res.*, vol. 10, 1982.
- [23] A. B. Arrieta *et al.*, "Explainable artificial intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI," *Inf. Fusion*, vol. 58, 2020.
- [24] D. Bau *et al.*, "Gan dissection: Visualizing and understanding generative adversarial networks," in *Proc. Int. Conf. Learn. Representations*, 2019.
- [25] C. Yang, Y. Shen, and B. Zhou, "Semantic hierarchy emerges in deep generative representations for scene synthesis," *Int. J. Comput. Vis.*, 2021.
- [26] A. Voynov and A. Babenko, "Unsupervised discovery of interpretable directions in the gan latent space," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 9786–9796.
- [27] A. Plummerault, H. L. Borgne, and C. Hudelot, "Controlling generative models with continuous factors of variations," in *Proc. Int. Conf. Learn. Representations*, 2020.
- [28] E. Härkönen, A. Hertzmann, J. Lehtinen, and S. Paris, "Ganspace: Discovering interpretable gan controls," in *Adv. Neural Inf. Process. Syst.*, 2020.
- [29] M. Kahng, N. Thorat, D. H. Chau, F. B. Viégas, and M. Wattenberg, "Gan Lab: Understanding complex deep generative models using interactive visual experimentation," *IEEE Trans. Visualization Comput. Graph.*, 2018.
- [30] R. Vidal, J. Bruna, R. Giryes, and S. Soatto, "Mathematics of deep learning," 2017, *arXiv:1712.04741*.
- [31] J. Bruna and S. Mallat, "Invariant scattering convolution networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1872–1886, Aug. 2013.
- [32] R. Eldan and O. Shamir, "The power of depth for feedforward neural networks," in *Proc. Conf. Learn. Theory*, 2016, pp. 907–940.
- [33] M. Nouiehed and M. Razaviyayn, "Learning deep models: Critical points and local openness," 2018, *arXiv:1803.02968*.
- [34] C. Yun, S. Sra, and A. Jadbabaie, "Global optimality conditions for deep neural networks," in *Proc. Int. Conf. Learn. Representations*, 2018.
- [35] A. Choromanska, M. Henaff, M. Mathieu, G. B. Arous, and Y. LeCun, "The loss surfaces of multilayer networks," in *Artif. Intell. Statist.*, 2015.
- [36] B. D. Haeffele and R. Vidal, "Global optimality in neural network training," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 7331–7339.
- [37] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, 2014.
- [38] P. Mianjy, R. Arora, and R. Vidal, "On the implicit bias of dropout," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 3540–3548.
- [39] H. Salehinejad and S. Valaee, "Ising-dropout: A regularization method for training and compression of deep neural networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2019, pp. 3602–3606.
- [40] B. Sengupta and K. J. Friston, "How robust are deep neural networks?," 2018, *arXiv:1804.11313*.
- [41] S. Zheng, Y. Song, T. Leung, and I. Goodfellow, "Improving the robustness of deep neural networks via stability training," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 4480–4488.
- [42] E. Haber and L. Ruthotto, "Stable architectures for deep neural networks," *Inverse Problems*, 2017.
- [43] B. Chang, L. Meng, E. Haber, L. Ruthotto, D. Begert, and E. Holtham, "Reversible architectures for arbitrarily deep residual neural networks," in *Proc. AAAI Conf. Artif. Intell.*, 2018.
- [44] K. K. Thekumparampil, A. Khetan, Z. Lin, and S. Oh, "Robustness of conditional gans to noisy labels," in *Adv. Neural Inf. Process. Syst.*, 2018.
- [45] A. Creswell and A. A. Bharath, "Dennoising adversarial autoencoders," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 4, pp. 968–984, Apr. 2019.
- [46] U. G. Konda Reddy Mopuri and V. B. Radhakrishnan, "Fast feature fool: A data independent approach to universal adversarial perturbations," in *Proc. Brit. Mach. Vis. Conf.*, 2017.
- [47] K. R. Mopuri, U. Ojha, U. Garg, and R. V. Babu, "Nag: Network for adversary generation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 742–751.
- [48] Z. Zheng and P. Hong, "Robust detection of adversarial attacks by modeling the intrinsic properties of deep neural networks," in *Proc. 32nd Int. Conf. Neural Inf. Process. Syst.*, 2018, pp. 7924–7933.
- [49] R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, F. Giannotti, and D. Pedreschi, "A survey of methods for explaining black box models," *ACM Comput. Surv.*, vol. 51, 2018.

- [50] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, "The limitations of deep learning in adversarial settings," in *Proc. IEEE Eur. Symp. Secur. Privacy*, 2016, pp. 372–387.
- [51] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard, "Universal adversarial perturbations," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 1765–1773.
- [52] E. Gawehn, J. A. Hiss, and G. Schneider, "Deep learning in drug discovery," *Mol. Inform.*, vol. 35, 2016.
- [53] B. Goodman and S. Flaxman, "European union regulations on algorithmic decision-making and a "right to explanation"," *AI Mag.*, vol. 38, 2017.
- [54] E. Parliament, Council of the European Union, "Regulation (eu) 2016/679 of the european parliament and of the council of 27 april 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing directive 95/46/ec (general data protection regulation)," *Official J. Eur. Union*, Apr. 2016, doi: <https://eur-lex.europa.eu/eli/reg/2016/679/oj>.
- [55] Y. Park and M. Kellis, "Deep learning for regulatory genomics," *Nature Biotechnol.*, vol. 33, 2015.
- [56] P. Baldi, P. Sadowski, and D. Whiteson, "Searching for exotic particles in high-energy physics with deep learning," *Nature Commun.*, vol. 5, 2014.
- [57] J. M. Hofman, A. Sharma, and D. J. Watts, "Prediction and explanation in social systems," *Sci.*, vol. 355, 2017.
- [58] G. Montavon, W. Samek, and K.-R. Müller, "Methods for interpreting and understanding deep neural networks," *Digit. Signal Process.*, vol. 73, 2018.
- [59] L. H. Gilpin, D. Bau, B. Z. Yuan, A. Bajwa, M. Specter, and L. Kagal, "Explaining explanations: An overview of interpretability of machine learning," in *Proc. IEEE 5th Int. Conf. Data Sci. Adv. Analytics*, 2018, pp. 80–89.
- [60] Q.-s. Zhang and S.-c. Zhu, "Visual interpretability for deep learning: A survey," *Front. Inf. Technol. Electron. Eng.*, vol. 19, 2018.
- [61] W. Samek and K.-R. Müller, "Towards explainable artificial intelligence," in *Explainable AI: Interpreting, Explaining Visualizing Deep Learn*. Springer, 2019.
- [62] F. Bodria, F. Giannotti, R. Guidotti, F. Naretto, D. Pedreschi, and S. Rinzivillo, "Benchmarking and survey of explanation methods for black box models," 2021, *arXiv:2102.13076*.
- [63] B. Kim *et al.*, "Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (TCAV)," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 2668–2677.
- [64] C. Molnar, G. Casalicchio, and B. Bischl, "Interpretable machine learning—a brief history, state-of-the-art and challenges," in *Proc. ECML PKDD Workshops*, 2020.
- [65] M. T. Ribeiro, S. Singh, and C. Guestrin, "Anchors: High-precision model-agnostic explanations," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018.
- [66] P. Adler *et al.*, "Auditing black-box models for indirect influence," *Knowl. Inf. Syst.*, vol. 54, 2018.
- [67] S. Lapuschkin, S. Wäldchen, A. Binder, G. Montavon, W. Samek, and K.-R. Müller, "Unmasking clever hans predictors and assessing what machines really learn," *Nature Commun.*, vol. 10, 2019.
- [68] K. Natesan Ramamurthy, B. Vinzamuri, Y. Zhang, and A. Dhurandhar, "Model agnostic multilevel explanations," *Adv. Neural Inf. Process. Syst.*, vol. 33, 2020.
- [69] A. Dhurandhar *et al.*, "Explanations based on the missing: Towards contrastive explanations with pertinent negatives," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018.
- [70] Q. Zhang, Y. Nian Wu, and S.-C. Zhu, "Interpretable convolutional neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 8827–8836.
- [71] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," 2013, *arXiv:1312.6034*.
- [72] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Proc. Eur. Conf. Comput. Vis.*, 2014.
- [73] P. W. Koh and P. Liang, "Understanding black-box predictions via influence functions," in *Proc. 34th Int. Conf. Mach. Learn.*, 2017, vol. 70, pp. 1885–1894.
- [74] B. Zhou, A. Khosla, À. Lapedriza, A. Oliva, and A. Torralba, "Object detectors emerge in deep scene cnns," in *Proc. 3rd Int. Conf. Learn. Representations*, San Diego, CA, USA, 7–9, 2015.
- [75] J. Adebayo, J. Gilmer, M. Muelly, I. Goodfellow, M. Hardt, and B. Kim, "Sanity checks for saliency maps," in *Proc. Adv. Neural Inf. Process. Syst.*, 31, 2018.
- [76] O. Li, H. Liu, C. Chen, and C. Rudin, "Deep learning for case-based reasoning through prototypes: A neural network that explains its predictions," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018.
- [77] Y. Wang, H. Su, B. Zhang, and X. Hu, "Interpret neural networks by identifying critical data routing paths," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 8906–8914.
- [78] Y. Goyal, Z. Wu, J. Ernst, D. Batra, D. Parikh, and S. Lee, "Counterfactual visual explanations," in *Proc. 36th Int. Conf. Mach. Learn.*, 2019, vol. 97, pp. 2376–2384.
- [79] K. Kanamori, T. Takagi, K. Kobayashi, and H. Arimura, "Dace: Distribution-aware counterfactual explanation by mixed-integer linear optimization," in *Proc. 29th Int. Joint Conf. Artif. Intell.*, 2020, pp. 2855–2862.
- [80] T. Wang, "Gaining free or low-cost interpretability with interpretable partial substitute," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 6505–6514.
- [81] L. Fu, "Rule Learning by Searching on Adapted Nets," AAAI, 1991.
- [82] G. G. Towell and J. W. Shavlik, "Extracting refined rules from knowledge-based neural networks," *Mach. Learn.*, vol. 13, 1993.
- [83] R. Setiono and H. Liu, "Understanding neural networks via rule extraction," *IJCAI*, 1995.
- [84] —, "NeuroLinear: From neural networks to oblique decision rules," *Neurocomputing*, vol. 17, 1997.
- [85] K. Odajima, Y. Hayashi, G. Tianxia, and R. Setiono, "Greedy rule generation from discrete data and its use in neural network rule extraction," *Neural Netw.*, vol. 21, 2008.
- [86] R. Nayak, "Generating rules with predicates, terms and variables from the pruned neural networks," *Neural Netw.*, vol. 22, 2009.
- [87] J. M. Benitez, J. L. Castro, and I. Requena, "Are artificial neural networks black boxes?," *IEEE Trans. Neural Netw.*, vol. 8, no. 5, pp. 1156–1164, Sep. 1997.
- [88] J. L. Castro, C. J. Mantas, and J. M. Benitez, "Interpretation of artificial neural networks by means of fuzzy rules," *IEEE Trans. Neural Netw.*, vol. 13, no. 1, pp. 101–116, Jan. 2002.
- [89] M. Craven and J. W. Shavlik, "Extracting tree-structured representations of trained networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 1996.
- [90] R. Krishnan, G. Sivakumar, and P. Bhattacharya, "Extracting decision trees from trained neural networks," *Pattern Recognit.*, vol. 32, 1999.
- [91] O. Boz, "Extracting decision trees from trained neural networks," in *Proc. 8th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2002.
- [92] T. Pedapati, A. Balakrishnan, K. Shanmugam, and A. Dhurandhar, "Learning global transparent models consistent with local contrastive explanations," *Adv. Neural Inf. Process. Syst.*, vol. 33, 2020.
- [93] D. Erhan, Y. Bengio, A. Courville, and P. Vincent, "Visualizing higher-layer features of a deep network," *University of Montreal*, vol. 1341, 2009.
- [94] F. Wang, H. Liu, and J. Cheng, "Visualizing deep neural network by alternately image blurring and deblurring," *Neural Netw.*, vol. 97, 2018.
- [95] A. Mahendran and A. Vedaldi, "Understanding deep image representations by inverting them," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 5188–5196.
- [96] J. Yosinski, J. Clune, A. Nguyen, T. Fuchs, and H. Lipson, "Understanding neural networks through deep visualization," *ICML Deep Learn. Workshop*, 2015.
- [97] A. Nguyen, A. Dosovitskiy, J. Yosinski, T. Brox, and J. Clune, "Synthesizing the preferred inputs for neurons in neural networks via deep generator networks," *NIPS*, 2016.
- [98] C. Olah, A. Mordvintsev, and L. Schubert, "Feature Visualization," *Distill*, 2017.
- [99] R. Fong and A. Vedaldi, "Net2vec: Quantifying and explaining how concepts are encoded by filters in deep neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 8730–8738.
- [100] F. Dalvi, N. Durrani, H. Sajjad, Y. Belinkov, D. A. Bau, and J. Glass, "What is one grain of sand in the desert? analyzing individual neurons in deep nlp models," in *Proc. AAAI Conf. Artif. Intell.*, 2019, pp. 6309–6317.
- [101] G. Plumb, D. Molitor, and A. S. Talwalkar, "Model agnostic supervised local explanations," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018.
- [102] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller, "Striving for simplicity: The all convolutional net," 2014, *arXiv:1412.6806*.
- [103] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 618–626.
- [104] E. Štrumbelj and I. Kononenko, "An efficient explanation of individual classifications using game theory," *J. Mach. Learn. Res.*, vol. 11, 2010.

- [105] S. M. Lundberg and S.-I. Lee, “A unified approach to interpreting model predictions,” 2017.
- [106] M. Ancona, C. Oztireli, and M. Gross, “Explaining deep neural networks with a polynomial time algorithm for shapley value approximation,” in *Proc. 36th Int. Conf. Mach. Learn.*, 2019, vol. 97, pp. 272–281.
- [107] T. Heskes, E. Sijben, I. G. Bucur, and T. Claassen, “Causal shapley values: Exploiting causal knowledge to explain individual predictions of complex models,” *Adv. Neural Inf. Process. Syst.*, vol. 33, 2020.
- [108] R. C. Fong and A. Vedaldi, “Interpretable explanations of black boxes by meaningful perturbation,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 3429–3437.
- [109] L. M. Zintgraf, T. S. Cohen, T. Adel, and M. Welling, “Visualizing deep neural network decisions: Prediction difference analysis,” 2017, *arXiv:1702.04595*.
- [110] J. Chen, L. Song, M. Wainwright, and M. Jordan, “Learning to explain: An information-theoretic perspective on model interpretation,” in *Proc. 35th Int. Conf. Mach. Learn.*, 2018, vol. 80, pp. 883–892.
- [111] S. Wang, T. Zhou, and J. Bilmes, “Bias also matters: Bias attribution for deep neural network explanation,” in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 6659–6667.
- [112] A. Shrikumar, P. Greenside, and A. Kundaje, “Learning important features through propagating activation differences,” in *Proc. 34th Int. Conf. Mach. Learn.*, 2017, pp. 3145–3153.
- [113] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, and W. Samek, “On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation,” *PLoS one*, vol. 10, 2015.
- [114] M. Sundararajan, A. Taly, and Q. Yan, “Axiomatic attribution for deep networks,” in *Proc. 34th Int. Conf. Mach. Learn.*, 2017, vol. 70, pp. 3319–332.
- [115] A. Ghorbani, J. Wexler, J. Y. Zou, and B. Kim, “Towards automatic concept-based explanations,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 9277–9286.
- [116] S. Salman, S. N. Payrovnaziri, X. Liu, P. Rengifo-Moreno, and Z. He, “Deepconsensus: Consensus-based interpretable deep neural networks with application to mortality prediction,” in *Proc. Int. Joint Conf. Neural Netw.*, 2020, pp. 1–8.
- [117] C.-K. Yeh, J. Kim, I. E.-H. Yen, and P. K. Ravikumar, “Representer point selection for explaining deep neural networks,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2018.
- [118] M. Wu *et al.*, “Regional tree regularization for interpretability in deep neural networks,” *AAAI*, 2020, pp. 6413–6421.
- [119] M. Wu, M. C. Hughes, S. Parbhoo, M. Zazzi, V. Roth, and F. Doshi-Velez, “Beyond sparsity: Tree regularization of deep models for interpretability,” in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018.
- [120] G. Plumb, M. Al-Shedivat, A. A. Cabrera, A. Perer, E. Xing, and A. Talwalkar, “Regularizing black-box models for improved interpretability,” *Adv. Neural Inf. Process. Syst.*, vol. 33, 2020.
- [121] E. Weinberger, J. Janizek, and S.-I. Lee, “Learning deep attribution priors based on prior knowledge,” *Adv. Neural Inf. Process. Syst.*, vol. 33, 2020.
- [122] M. Wojtas and K. Chen, “Feature importance ranking for deep learning,” *Adv. Neural Inf. Process. Syst.*, vol. 33, 2020.
- [123] C. Chen, O. Li, D. Tao, A. Barnett, C. Rudin, and J. K. Su, “This looks like that: Deep learning for interpretable image recognition,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2019.
- [124] S. Wachter, B. Mittelstadt, and C. Russell, “Counterfactual explanations without opening the black box: Automated decisions and the GDPR,” *Harvard J. Law Technol.*, 2018.
- [125] M. W. Craven and J. W. Shavlik, “Using sampling and queries to extract rules from trained neural networks,” in *Proc. Mach. Learn.*, 1994.
- [126] R. Andrews, J. Diederich, and A. B. Tickle, “Survey and critique of techniques for extracting rules from trained artificial neural networks,” *Knowl.-Based Syst.*, vol. 8, 1995.
- [127] A. B. Tickle, R. Andrews, M. Golea, and J. Diederich, “The truth will come to light: Directions and challenges in extracting the knowledge embedded within trained artificial neural networks,” *IEEE Trans. Neural Netw.*, vol. 9, no. 6, pp. 1057–1068, Nov. 1998.
- [128] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees*. Wadsworth & Brooks/Cole Advanced Books & Software, 1984.
- [129] J. R. Quinlan, “C4.5: Programs for machine learning,” *Morgan Kaufmann Ser. Mach. Learn.*, San Mateo, CA: Morgan Kaufmann, 1993.
- [130] G. D. Plotkin, “A note on inductive generalization,” *Mach. Intell.*, vol. 5, 1970.
- [131] S. Mitra and Y. Hayashi, “Neuro-fuzzy rule generation: Survey in soft computing framework,” *IEEE Trans. Neural Netw.*, vol. 11, no. 3, pp. 748–768, May 2000.
- [132] I. Goodfellow *et al.*, “Generative adversarial nets,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2014.
- [133] A. Radford, L. Metz, and S. Chintala, “Unsupervised representation learning with deep convolutional generative adversarial networks,” *ICLR*, 2016.
- [134] C. Ledig *et al.*, “Photo-realistic single image super-resolution using a generative adversarial network,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 4681–4690.
- [135] T. Minematsu, A. Shimada, and R.-i. Taniguchi, “Analytics of deep neural network in change detection,” in *Proc. 14th IEEE Int. Conf. Adv. Video Signal Based Surveill.*, 2017, pp. 1–6.
- [136] T. Minematsu, A. Shimada, H. Uchiyama, and R.-i. Taniguchi, “Analytics of deep neural network-based background subtraction,” *J. Imag.*, 2018.
- [137] D. Bachrens, T. Schroeter, S. Harmeling, M. Kawanabe, K. Hansen, and K.-R. Mäžller, “How to explain individual classification decisions,” *J. Mach. Learn. Res.*, vol. 11, 2010.
- [138] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, “Learning deep features for discriminative localization,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 2921–2929.
- [139] M. Ancona, E. Ceolini, C. Öztireli, and M. Gross, “Towards better understanding of gradient-based attribution methods for deep neural networks,” in *Proc. Int. Conf. Learn. Representations*, 2018.
- [140] A. Ghorbani, A. Abid, and J. Zou, “Interpretation of neural networks is fragile,” in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, 2019.
- [141] A.-K. Dombrowski, M. Alber, C. Anders, M. Ackermann, K.-R. Müller, and P. Kessel, “Explanations can be manipulated and geometry is to blame,” *Adv. Neural Inf. Process. Syst.*, vol. 32, 2019.
- [142] J. Heo, S. Joo, and T. Moon, “Fooling neural network interpretations via adversarial model manipulation,” in *Adv. Neural Inf. Process. Syst.*, 2019.
- [143] P.-J. Kindermans *et al.*, “The (un)reliability of saliency methods,” in *Proc. Explainable AI: Interpreting, Explaining Visualizing Deep Learn.*, 2019.
- [144] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, “Distributed representations of words and phrases and their compositionality,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2013.
- [145] L. S. Shapley, “A value for n-person games,” *Contributions Theory Games*, vol. 2, 1953.
- [146] R. Caruana, H. Kangaroo, J. Dionisio, U. Sinha, and D. Johnson, “Case-based explanation of non-case-based learning methods,” in *Proc. AMIA Symp.*, 1999, p. 212.
- [147] J. Bien, R. Tibshirani *et al.*, “Prototype selection for interpretable classification,” *Ann. Appl. Statist.*, vol. 5, 2011.
- [148] B. Zhou, D. Bau, A. Oliva, and A. Torralba, “Interpreting deep visual representations via network dissection,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 9, pp. 2131–2145, Sep. 2019.
- [149] W. Samek, A. Binder, G. Montavon, S. Lapuschkin, and K.-R. Müller, “Evaluating the visualization of what a deep neural network has learned,” *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 11, pp. 2660–2673, Nov. 2017.
- [150] S. Hooker, D. Erhan, P.-J. Kindermans, and B. Kim, “A benchmark for interpretability methods in deep neural networks,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2019.



Yu Zhang received the B.Eng. degree from the Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen, China, in 2017. He is currently working toward the Ph.D. degree with the Department of Computer Science and Engineering, Southern University of Science and Technology, jointly with the School of Computer Science, University of Birmingham, Birmingham, U.K. His current research focuses on interpretable machine learning.



Peter Tino received the M.Sc. degree from the Slovak University of Technology, Bratislava, Slovakia and the Ph.D. degree from the Slovak Academy of Sciences, Bratislava, Slovakia. He was a Fulbright Fellow with the NEC Research Institute, Princeton, NJ, USA, and a Postdoctoral Fellow with the Austrian Research Institute for AI, Vienna, Austria and with Aston University, Birmingham, U.K. Since 2003, he has been with the School of Computer Science, University of Birmingham, Birmingham, U.K., where he is currently a Full Professor Chair of complex and

adaptive systems. His current research interests include dynamical systems, machine learning, probabilistic modelling of structured data, evolutionary computation, and fractal analysis. He was the recipient of the Fulbright Fellowship in 1994, the U.K.-Hong-Kong Fellowship for Excellence in 2008, three Outstanding Paper of the Year Awards from the IEEE TRANSACTIONS ON NEURAL NETWORKS in 1998 and 2011, and the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION in 2010, and the Best Paper Award at ICANN 2002. He is on the Editorial Boards of various journals.



Ke Tang (Senior Member, IEEE) received the B.Eng. degree from the Huazhong University of Science and Technology, Wuhan, China, in 2002 and the Ph.D. degree from Nanyang Technological University, Singapore, in 2007. From 2007 to 2017, he was with the School of Computer Science and Technology, University of Science and Technology of China, Hefei, China, first as an Associate Professor from 2007 to 2011 and later as a Professor from 2011 to 2017. He is currently a Professor with the Department of Computer Science and Engineering, Southern University

of Science and Technology, Shenzhen, China. He has more than 10 000 Google Scholar citation with an H-index of 48. He has authored or coauthored more than 70 journal papers and more than 80 conference papers. His current research interests include evolutionary computation, machine learning, and their applications. He was the recipient of the Royal Society Newton Advanced Fellowship in 2015 and the 2018 IEEE Computational Intelligence Society Outstanding Early Career Award. He is an Associate Editor for the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION and was a Member of Editorial Boards for a few other journals.



Aleš Leonardis is currently the Chair of robotics with the School of Computer Science, University of Birmingham, Birmingham, U.K. He is also a Professor of computer and information science with the University of Ljubljana, Ljubljana, Slovenia. He was a Visiting Researcher with the GRASP Laboratory, University of Pennsylvania, Philadelphia, PA, USA, a Postdoctoral Fellow with PRIP Laboratory, Vienna University of Technology, Vienna, Austria, and Visiting Professor with ETH Zurich, Zürich, Switzerland and University of Erlangen, Erlangen, Germany. His

research interests include computer vision, visual learning, and biologically motivated vision—all in a broader context of cognitive systems and robotics. He was the Program Co-Chair of the European Conference on Computer Vision 2006, and he is an Associate Editor for the IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE and IEEE ROBOTICS AND AUTOMATION LETTERS, an Editorial Board Member of the *Pattern Recognition* and *Image and Vision Computing*, and the Editor of the Springer book series *Computational Imaging and Vision*. In 2002, he coauthored a paper, *Multiple Eigenspaces*, which won the 29th Annual Pattern Recognition Society Award. He is a Fellow of the IAPR and in 2004 he was awarded one of the two most prestigious national (SI) awards for his research achievements.