

Explainable Deep Learning: A Field Guide for the Uninitiated

Ning Xie*

XIE.25@WRIGHT.EDU

*Department of Computer Science and Engineering
Wright State University
Dayton, OH 45435, USA*

Gabriëlle Ras*

G.RAS@DONDEERS.RU.NL

*Donders Institute for Brain, Cognition and Behaviour
Radboud University Nijmegen
6525 HR Nijmegen, the Netherlands*

Marcel van Gerven

M.VANGERVEN@DONDEERS.RU.NL

*Donders Institute for Brain, Cognition and Behaviour
Radboud University Nijmegen
6525 HR Nijmegen, the Netherlands*

Derek Doran

DEREK.DORAN@WRIGHT.EDU

*Department of Computer Science and Engineering
Wright State University
Dayton, OH 45435, USA*

Abstract

The deep neural network (DNN) is an indispensable machine learning tool for achieving human-level performance on many learning tasks. Yet, due to its black-box nature, it is inherently difficult to understand which aspects of the input data drive the decisions of the network. There are various real-world scenarios in which humans need to make actionable decisions based on the output of a decision support system that makes use of DNNs. These decision support systems can be found in critical domains, such as legislation, law enforcement, and healthcare. It is important that the humans making high-level decisions can be sure that the DNN decisions are driven by combinations of data features that are appropriate in the context of the deployment of the decision support system and that the decisions made are legally or ethically defensible. Due to the incredible pace at which DNN technology is being developed and adopted, the development of new methods and studies on explaining the decision-making process of DNNs has blossomed into an active research field. A practitioner beginning to study explainable deep learning may be intimidated by the plethora of orthogonal directions the field is taking. This complexity is further exacerbated by the general confusion that exists in defining what it means to be able to explain the actions of a deep learning system and to evaluate a system’s “ability to explain”. To alleviate this problem, this article offers a “field guide” to deep learning explainability for those uninitiated in the field. The field guide: i) Discusses the traits of a deep learning system that researchers enhance in explainability research, ii) places explainability in the context of other related deep learning research areas, and iii) introduces three simple dimensions defining the space of foundational methods that contribute to explainable deep learning. The guide is designed as an easy-to-digest starting point for those just embarking in the field.

*. Ning Xie and Gabriëlle Ras are co-first authors on this work.

1. Introduction

Artificial intelligence (AI) systems powered by deep neural networks (DNNs) are pervasive across society: they run in our pockets on our cell phones (Georgiev et al., 2017), in cars to help avoid car accidents (Jain et al., 2015), in banks to manage our investments (Chong et al., 2017) and evaluate loans (Pham & Shen, 2017), in hospitals to help doctors diagnose disease symptoms (Nie et al., 2015), at law enforcement agencies to help recover evidence from videos and images to help law enforcement (Goswami et al., 2014), in the military of many countries (Lundén & Koivunen, 2016), and at insurance agencies to evaluate coverage suitability and costs for clients (Dong et al., 2016; Sirignano et al., 2016). But when a person’s future is on the line, when a medical treatment is to be assigned, when a major financial decision must be made, when a military decision is to be reached, and when a risky choice having security ramifications is under consideration, it is understandable that we want AI to *suggest* or *recommend* a course of action with reasonable evidence, rather than to merely *prescribe* one. For the human ultimately responsible for the action taken, the use of present-day DNNs leaves an important question unanswered: *how can one who will be held accountable for a decision trust a DNNs recommendation, and justify its use?*

Achieving trust and finding justification in a DNN’s recommendation can hardly be achieved if the user does not have access to a satisfactory explanation for the process that led to its output. Consider for example a hypothetical scenario in which there exists a medical system running a DNN in the backend. Assume that the system makes life-altering predictions about whether or not a patient has a terminal illness. It is desirable if this system could also provide a rationale behind its predictions. More importantly, it is desirable if this system can give a rationale that both physicians and patients can understand and trust. Trust in a decision is built upon a rationale that is: (i) easily interpretable; (ii) relatable to the user; (iii) connects the decision with contextual information about the choice or to the user’s prior experiences; (iv) reflects the intermediate thinking of the user in reaching a decision. Given the qualitative nature of these characteristics, it may come as no surprise that there is great diversity in the definitions, approaches, and techniques used by researchers to provide a rationale for the decisions of a DNN. This diversity is further compounded by the fact that the form of a rationale often conforms to a researcher’s personal notion of what constitutes an “explanation”. For a newcomer to the field, whether they are seasoned AI researchers or students of an unrelated discipline that DNN decision-making stands to disturb, jumping into the field can be a daunting task.

This article offers a much-needed starting point for researchers and practitioners who are embarking into the field of explainable deep learning. This “field guide” is designed to help an uninitiated researcher understand:

- Traits can be thought of as a simple set of qualitative target contributions that the explainable DNN field tries to achieve in the results. (**Section 2**).
- Complementary research **topics** that are aligned with explainability. Topics that are complementary to explainability may involve the inspection of a DNN’s weights or activations, the development of mechanisms that mathematically explain how DNNs learn to generalize, or approaches to reduce a DNN’s sensitivity to particular input features. Such topics are indirectly associated with explainability in the sense that

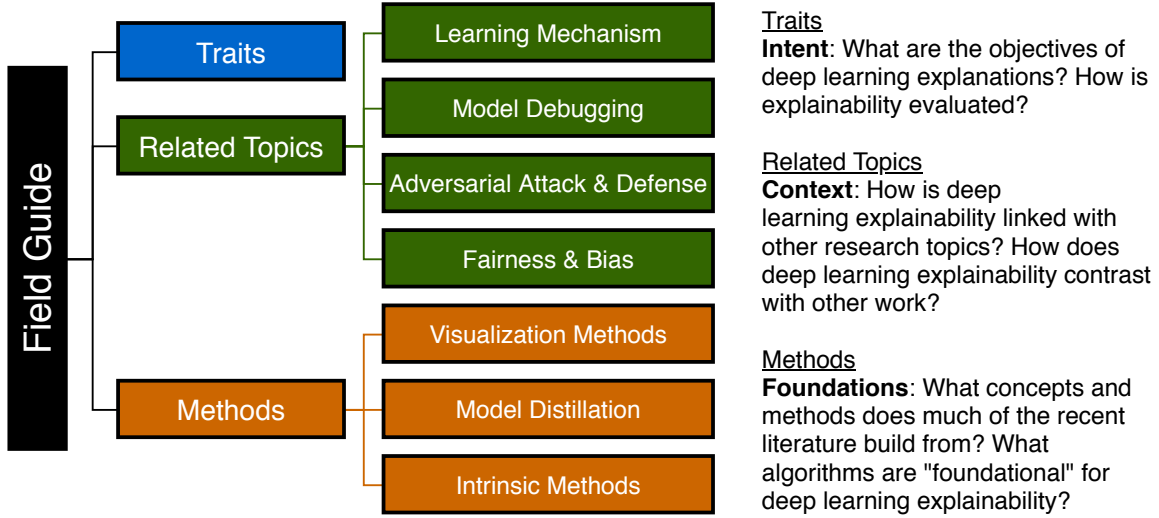


Figure 1: Outline of the field guide.

they investigate *how* a DNN learns or performs inference, even though the intention of the work is not directly to investigate explanations (**Section 3**).

- A set of **dimensions** that characterize the space of work that constitutes foundational work in explainable deep learning, and a description of such methods. This space summarizes the core aspects of explainable DNN techniques that a majority of present work is inspired by or built from (**Section 4**).
- The **considerations** of a designer developing an explainable DNN system (**Section 5**).
- **Future directions** in explainability research (**Section 6**).

The aims of the field are established by understanding the traits desired in explainable DNNs. Complementary DNN topics are reviewed and the relationships between explainable DNNs and other related research areas are developed. Our taxonomy of explainable DNN techniques clarifies the technical ideas underpinning most modern explainable deep learning techniques. The discussion of fundamental explainable deep learning methods, emblematic of each framework dimension, provides further context for the modern work that builds on or takes inspiration from them. The field guide then turns to essential considerations that need to be made when building an explainable DNN system in practice (which could include multiple forms of explanation to meet requirements for both users and DNN technicians). Finally, the overview of our current limitations and seldom-looked at aspects of explainable deep learning suggests new research directions. This information captures what a newcomer needs to know to successfully navigate the current research literature on explainable deep learning and to identify new research problems to tackle.

There are a large number of existing reviews on the topic of model explainability. Most of them focus on explanations of general artificial intelligence methods (Arrieta et al., 2019; Carvalho et al., 2019; Mueller et al., 2019; Tjoa & Guan, 2019; Gilpin et al., 2018; Adadi & Berrada, 2018; Miller, 2018; Guidotti et al., 2018; Lipton, 2018; Liu et al., 2017; Došilović

et al., 2018; Doshi-Velez & Kim, 2017), and some on deep learning (Ras et al., 2018; Montavon et al., 2018; Zhang & Zhu, 2018; Samek et al., 2017b; Erhan et al., 2010). Given the existing reviews, the **contributions** of our article are as follows.

- Our article is specifically targeting on deep learning explanation while existing reviews either focus on explanations of general artificial intelligence methods or are less as detailed and comprehensive as ours.
- We provide a detailed field guide for researchers who are uninitiated in explainable deep learning, aiming to lower or even eliminate the bar for them to come into this field.
- We propose a novel categorization scheme to systematically organize numerous existing methods on explainable deep learning, depicting the field in a clear and straightforward way.
- A review of related topics that are closely related to the realm of explainable deep learning is elaborated. Such a review would help the uninitiated researchers thoroughly understand the connections between the related fields and explainable deep learning, and how jointly those fields help shape the future of deep learning towards transparency, robustness, and reliability.

2. The Traits of an Explanation

A central tenet in explainable machine learning is that the algorithm must emit information allowing a user to relate characteristics of input features with its output. It is thus worth noting that DNNs are not inherently “explainable”. The limited information captured in a DNN’s parameters associated with input features becomes entangled and compressed into a single value via a non-linear transform of a weighted sum of feature values. This compression occurs multiple times with different weight vectors depending on the number of activations in the first hidden layer. Subsequent layers then output non-linear transforms of weighted sums of these compressions, and so forth, until a decision is made based on the output of the DNN. Hence, it is exceedingly difficult to trace how particular stimulus properties drive this decision.

The unexplainable nature of DNNs is a significant impediment¹ to the wide-spread adoption of DNNs we are beginning to see in society. DNN-powered facial recognition systems, for example, are now associating people with locations and activities under wide-spread surveillance activities with opaque intent (Masi et al., 2018). People analytics and human resource platforms now tout the ability to predict employee performance and time to resignation, and to automatically scan the CV of job applicants (Zhao et al., 2018; Qin et al., 2018). These examples foretell a future where DNN technology will make countless recom-

1. There are common counter-arguments to call this limitation “significant”. Some argue that many successful applications of DNNs do not require explanations (LeCun et al., 2017), and in these instances, enforcing constraints that provide explainability may hamper performance. Others claim that, because DNNs are inherently not explainable, an ascribed explanation is at best a plausible story about how the network processes an input that cannot be proven (Rudin, 2019).

mendations and decisions that more directly, and perhaps more significantly, impact people and their well-being in society.

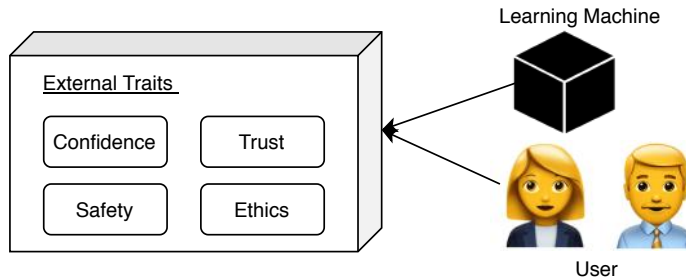


Figure 2: Necessary external traits of an explanation.

The present art develops ways to promote traits that are *associated with* explainability. A trait represents a property of a DNN necessary for a user to evaluate its output (Lipton, 2018). Traits, therefore, represent a particular objective or an evaluation criterion for explainable deep learning systems. We can say that a DNN **promotes explainability** if the system exhibits any trait that is justifiably related to explainability. This exhibition may be self-evident (e.g., in an NLP task, visualizations highlighting keywords or phrases that suggest a reasonable classification of a sentence), measurable (based on a trait-specific “error” metric), or evaluated through system usability studies. We discuss the four traits in Figure 2 that are the theme of much of the explainable deep learning literature.

- **Confidence.** Confidence grows when the “rationale” of a DNN’s decision is congruent with the thought process of a user. Of course, a DNN’s output is based on a deterministic computation, rather than a logical rationale. But by associating the internal actions of a DNN with features of its input or with the environment it is operating in, and by observing decisions that match what a rational human decision-maker would decide, a user can begin to align a DNN’s processing with her own thought process to engender confidence.

For example, saliency maps of attention mechanisms on image (Park et al., 2018; Hudson & Manning, 2018) or text (Vaswani et al., 2017; Luong et al., 2015; Letarte et al., 2018; He et al., 2018) inputs reassure a user that the same semantically meaningful parts of the input she would focus on to make a classification decision are being used. Observing how the actions of a trained agent in a physical environment mimic the actions a human would give some confidence that its action choice calculus is aligned with a rational human. The saliency maps and the observation of the agents in these examples may constitute a suitable “form of explanation”.

Confidence must be developed by observing a DNN when its decisions are both correct and incorrect. Eschewing observations of incorrect decisions means a user will never be able to identify when she should not be confident, and hence not rely on a DNN. Users must be able to use their confidence to measure the operational boundaries of a DNN to be able to intuitively answer the question: When does this DNN work or not work?

- **Trust.** DNNs whose decision-making process need not be validated are trustworthy. Recent research (Jiang et al., 2018; Baum et al., 2017; Varshney & Alemzadeh, 2017; Amodei et al., 2016; Pieters, 2011; Lee & See, 2004) explores the model trustworthiness problem, which studies whether or not a model prediction is safe to be adopted. Note that a prediction with high probability does not guarantee its trustworthiness, as shown in recent adversarial studies (Goodfellow et al., 2014; Nguyen et al., 2015; Moosavi-Dezfooli et al., 2016; Yuan et al., 2019). Trust in a DNN is best developed in two ways: (i) *Satisfactory testing*. Under *ideal* conditions, the network’s performance on test data should well approximate their performance in practice. The test accuracy of a model can thus be thought of as a direct measure of trust: a model with a perfect performance during the testing phase *may* be fully trusted to make decisions; lower performance degrades trust proportionally. (ii) *Experience*. A user does not need to inspect or validate the actions of a DNN as long as the network’s input/output behavior matches expectations. For example, a DNN’s ability to predict handwritten digits from MNIST is beyond question (LeCun et al., 1995, 1998), and may thus be regarded as a trustworthy system to sort postal mail by location code. A system that consistently performs poorly in practice may even be “un-trusted” indicating that the DNN should not be used.

Trust is a difficult trait to evaluate. Most deep learning studies include some evaluation component over test data, but it is seldom the case that the evaluation is ideal. Without careful sampling procedures, test data can be biased towards a particular class or have feature distributions that do not match the general case (Tommasi et al., 2017). It may also be the case that a model can perform poorly in practice over time as characteristics of the data evolve or drift. Therefore, the best way to evaluate trust is with system observations (spanning both output and internal processing) over time. Explainability research allowing users to evaluate these observations (through an interpretation of activations during a DNN’s forward pass, for example) is one avenue for enhancing trust.

- **Safety.** DNNs whose decisions (in)directly lead to an event impacting human life, wealth, or societal policy should be *safe*. The definition of safety is multi-faceted. A safe DNN should: (i) consistently operate as expected; (ii) given cues from its input, guard against choices that can negatively impact the user or society; (iii) exhibit high reliability under both standard and exceptional operating conditions; (iv) provide feedback to a user about how operating conditions influence its decisions. The first aspect of safety aligns this trait with trust since trust in a system is a prerequisite to consider it safe to use. The second and third aspects imply that safe systems possess mechanisms that augment its decision-making process to steer away from decisions with negative impact, or consider its operating environment as part of its decision-making process. The fourth aspect gives necessary feedback to the user to assess safety. The feedback may include an evaluation of its environment, the decision reached, and how the environment and the input data influence the decision made. This allows the user to verify the rationality of the decision making process with respect to the environment that the system is operating in.

- **Ethics.** A DNN behaves ethically if its decisions and decision-making process does not violate a code of moral principles defined by the user. The right way to evaluate if a DNN is acting ethically is a topic of debate. For example, different users assume their own unique code of ethics, ethical decisions in one culture may be unethical in another, and there may be instances where no possible decision is consistent with a set of moral principles. Thus, rather than making DNNs inherently ethical, this trait can be expressed by some notion of an “ethics code” that the system’s decisions are formed under. This allows users to individually assess if the reasoning of a DNN is compatible with the moral principles it should operate over. The field of ethical decision making in AI is growing as a field in and of itself (see Section 3.4).

3. Topics Associated with Explainability

We next review research topics closely aligned with explainable deep learning. A survey, visualized in Figure 3, identifies four broad related classes of research. Work on **learning mechanism (Section 3.1)** investigates the backpropagation process to establish a theory around weight training. These studies, in some respects, try to establish a theory to explain how and why DNNs converge to some decision-making process. Research on **model debugging (Section 3.2)** develops tools to recognize and understand the failure modes of a DNN. It emphasizes the discovery of problems that limit the training and inference process of a DNN (e.g., dead ReLUs, mode collapse, etc.). Techniques for **adversarial attack and defense (Section 3.3)** search for differences between regular and unexpected activation patterns. This line of work promotes deep learning systems that are robust and trustworthy; traits that also apply to explainability. Research on **fairness and bias in DNNs (Section 3.4)** is related to the ethics trait discussed above, but more narrowly concentrates on ensuring DNN decisions do not over-emphasize undesirable input data features. We elaborate on the connection between these research areas and explainable DNNs next.

3.1 Learning Mechanism

The investigation of the learning mechanism tries to derive principles explaining the evolution of a model’s parameters during training. Many existing approaches can be categorized as being semantics-related, in that the analysis tries to associate a model’s learning process with concepts that have a concrete semantic meaning. They generally assign semantic concepts to a DNNs’ internal filters (weights) or representations (activations), in order to uncover a human-interpretable explanation of the learning mechanism. Semantically interpretable descriptions are rooted in the field of neuro-symbolic computing (Garcez et al., 2012). An early work is Zhou et al. (2014) which assigns semantic concepts, such as objects, object parts, etc, to the internal filters of a convolutional neural network (CNN) image scene classifier. Those semantic concepts are generated based on the visualization of receptive fields of each internal unit in the given layers. The authors also discovered that object detectors are embedded in a scene classifier without explicit object-level supervision for model training. Gonzalez-Garcia et al. (2018) further explores this problem in a quantitative fashion. Two quantitative evaluations are conducted to study whether the internal representations of CNNs really capture semantic concepts. Interestingly, the authors’ experimental results show that the association between internal filters and semantic concepts is

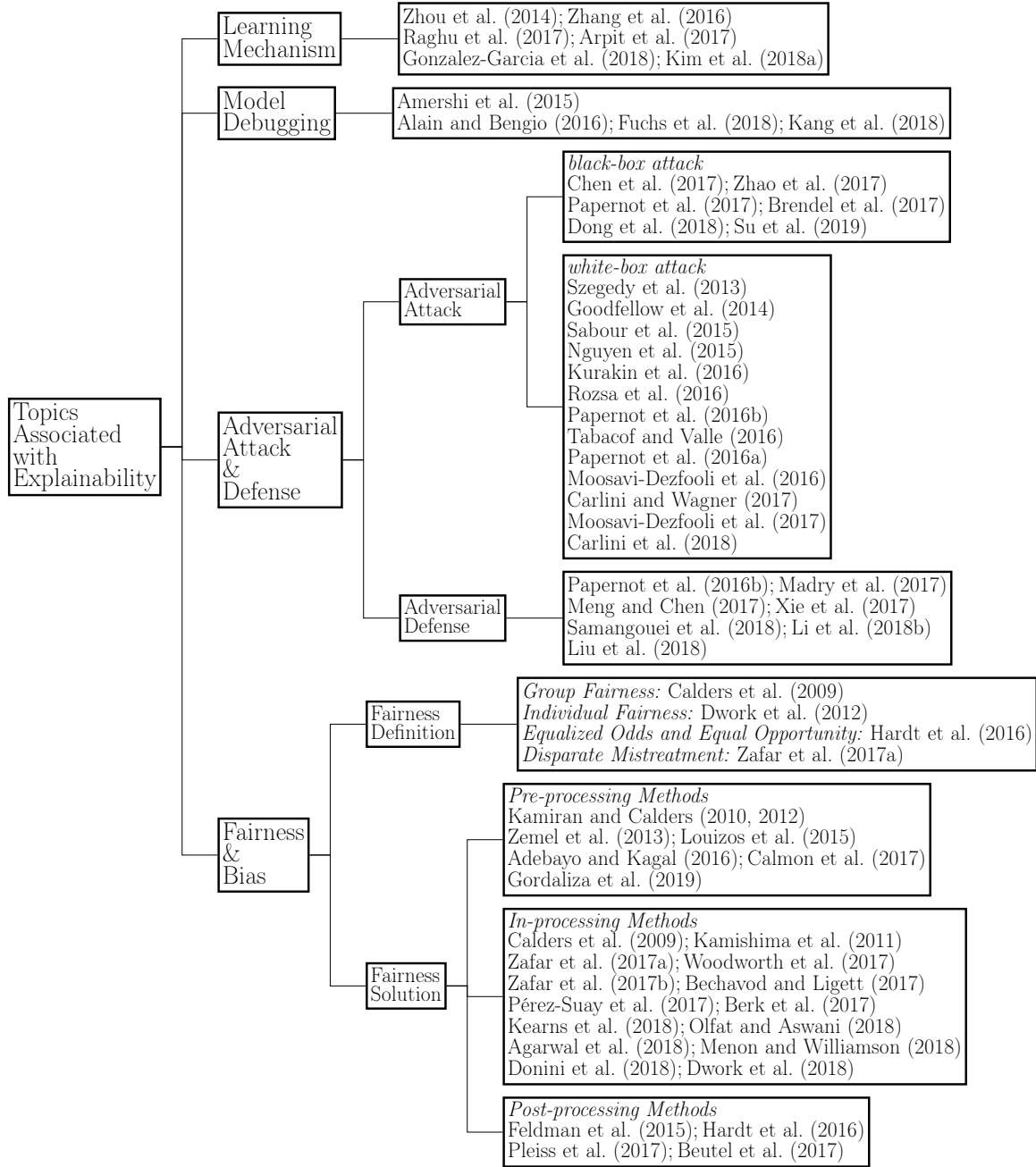


Figure 3: Topics associated with explainability.

modest and weak. But this association improves for deeper layers of the network, matching the conclusion of Zhou et al. (2014). Kim et al. (2018a) quantifies the importance of a given semantic concept with respect to a classification result via Testing with Concept Activation Vector (TCAV), which is based on multiple linear classifiers built with internal activations on prepared examples. The prepared examples contain both positive examples representing

a semantic concept and randomly sampled negative examples that do not represent the concept. Directional derivatives are used to calculate TCAV, which measures the proportion of examples that belong to a given class that are positively influenced by a given concept.

Other methods to interpret the learning process of a DNN searches for statistical patterns indicative of convergence to a learned state. Those learning patterns include but are not limited to: i) how layers evolve along with the training process (Raghu et al., 2017); ii) the convergence of different layers (Raghu et al., 2017); and iii) the generalization and memorization properties of DNNs (Zhang et al., 2016; Arpit et al., 2017). In studying the learning dynamics during training, Raghu et al. (2017) makes a comparison between two different layers or networks via Singular Vector Canonical Correlation Analysis (SVCCA). For a neuron in a selected layer of a DNN, the neuron’s vector representation is generated in a “global fashion”, i.e. all examples from a given finite dataset are used, and each element in the neuron’s vector representation is an activation for an example. The vector representations for all neurons in a selected layer form a vector set, representing this layer. To compare two layers, SVCCA takes the vector set of each layer as input and calculates a canonical correlation similarity to make the alignment. The nature of SVCCA makes it a useful tool to monitor how layer activations evolve along with the training process. The authors further discover that earlier layers converge faster than later layers. Thus, the weights for earlier layers can be frozen earlier to reduce computational cost during training. Layer-wise convergence is also studied in work such as Zhang et al. (2016) using systematic experimentation. Keeping the model structure and hyper-parameters fixed, the authors’ experiments are conducted only with different input modification settings, either on input labels or image pixels. The experimental results indicate that DNNs can perfectly fit training data with both random feature values and labels, while the degree of generalization on testing data reduces as randomness increases. The authors also hypothesize that explicit regularization (such as dropout, weight decay, data augmentation, etc.) *may* improve generalization and stochastic gradient descent could act as an implicit regularizer for linear models. In a similar study, Arpit et al. (2017) examines memorization by DNNs via quantitative experiments with real and random data. The study finds that DNNs do not simply memorize all real data; instead, patterns that are commonly shared among the data are leveraged for memorization. Interestingly, the authors claim that explicit regularization does make a difference in the speed of memorization for random data, which is different from the conclusions in Zhang et al. (2016). Besides the aforementioned research work, we would like to refer readers to a recent review paper Bahri et al. (2020), which covers the intersection between statistical mechanics and deep learning, and derives the success of deep learning from a theoretical perspective.

3.2 Model Debugging

Similar to the concept of software debugging, the concept of model debugging applies techniques from traditional programming to find out when and where model-architecture, data processing, and training related errors occur. A “probe” is leveraged to analyze the internal pattern of a DNN, to provide further hints towards performance improvement. A probe is usually an auxiliary model or a structure such as a linear classifier, a parallel branch of the model pipeline, etc. The training process of the probe is usually independent of the training

process of the master model (a DNN) that the probe serves for. Regardless of the form of the probe, the ultimate goal is model improvement.

Kang et al. (2018) uses *model assertions*, or Boolean functions, to verify the state of the model during training and run time. The assertions can be used to ensure the model output is consistent with meta observations about the input. For example, if a model is detecting cars in a video, the cars should not disappear and reappear in successive frames of the video. Model debugging is thus implemented as a verification system surrounding the model and is implicitly model-agnostic. The model assertions are implemented as user-defined functions that operate on a recent history of the model input and output. The authors explore several ways that model assertions can be used during both run-time and training time, in correcting wrong outputs and in collecting more samples to perform active learning. Amershi et al. (2015) proposes *ModelTracker*, a debugging framework revolving around an interactive visual interface. This visual interface summarizes traditional summary statistics, such as AUC and confusion matrices, and presents this summary to the user together with a visualization of how close data samples are to each other in the feature space. The interface also has an option to directly inspect prediction outliers in the form of the raw data with its respective label, giving users the ability to directly correct mislabeled samples. The goal of this framework is to provide a unified, model-agnostic, inspection tool that supports debugging of three specific types of errors: mislabeled data, inadequate features to distinguish between concepts and insufficient data for generalizing from existing examples. Alain and Bengio (2016) uses linear classifiers to understand the predictive power of representations learned by intermediate layers of a DNN. The features extracted by an intermediate layer of a deep classifier are fed as input to the linear classifier. The linear classifier has to predict which class the given input belongs to. The experimental results show that the performance of the linear classifier improves when making predictions using features from deeper layers, i.e., layers close to the final layer. This suggests that task-specific representations are encoded in the deeper layers. Fuchs et al. (2018) proposes the idea of *neural stethoscopes*, which is a general-purpose framework used to analyze the DNN learning process by quantifying the importance of specific influential factors in the DNN and influence the DNN learning process by actively promoting and suppressing information. Neural stethoscopes extend a DNN’s architecture with a parallel branch containing a two-layer perceptron. It is important to note that the main network branch does not need to be changed to be able to use the neural stethoscope. This parallel branch takes the feature representation from an arbitrary layer from the main network as input and is trained on a supplemental task given known complementary information about the dataset. Specifically, in this study the experiments are conducted on the ShapeStacks dataset (Groth et al., 2018), which introduces a vision-based stability prediction task for block towers. The dataset provides information on both the local and global stability of a stack of blocks. In this specific study the stethoscope was used to investigate the internal representations contained in the network layers that lead to the prediction of the global stability of a stack of blocks, with local stability as complementary information. The stethoscope can be tuned to three different modes of operation: analytic, auxiliary, and adversarial. Each mode determines how the stethoscope loss L_S is propagated, e.g., in the analytical mode, L_S is not propagated through the main network. The auxiliary and adversarial modes are used to promote and

suppress information respectively. The paper shows that the method was successful in improving network performance and mitigating biases that are present in the dataset.

3.3 Adversarial Attack and Defense

An adversarial example is an artificial input engineered to intentionally disturb the judgment of a DNN (Goodfellow et al., 2014). Developing defenses to adversarial examples requires a basic understanding of the space that inputs are taken from and the shape and form of boundaries between classes. Interpretations of this space inform the construction of defenses to better discriminate between classes and forms the basis of explaining input/output behavior. Moreover, an “explanation” from a model that is not reasonable given its input and output may be indicative of an adversarial example.

The study of adversarial examples (Yuan et al., 2019; Zhang et al., 2019b) are from the perspective of attack and defense. **Adversarial attack** is about generating adversarial examples, which can fool a DNN. From the model access perspective, there are two main types of adversarial attack: *black-box* (Chen et al., 2017; Zhao et al., 2017; Papernot et al., 2017; Brendel et al., 2017; Dong et al., 2018; Su et al., 2019) and *white-box* (Szegedy et al., 2013; Goodfellow et al., 2014; Sabour et al., 2015; Nguyen et al., 2015; Kurakin et al., 2016; Rozsa et al., 2016; Papernot et al., 2016a; Moosavi-Dezfooli et al., 2016; Tabacof & Valle, 2016; Kurakin et al., 2016; Carlini & Wagner, 2017; Moosavi-Dezfooli et al., 2017; Carlini et al., 2018; Eykholt et al., 2018) attacks. In the black-box setting the attacker has no access to the model parameters or intermediate gradients whereas these are available for the white-box settings. **Adversarial defense** (Madry et al., 2017; Papernot et al., 2016b; Meng & Chen, 2017; Xie et al., 2017; Samangouei et al., 2018; Li et al., 2018b; Liu et al., 2018), on the other hand, is to come up with solutions to make a DNN robust against generated adversarial examples.

Recent work on adversarial attack reveals vulnerabilities by perturbing input data with imperceptible noise (Goodfellow et al., 2014; Carlini & Wagner, 2017; Madry et al., 2017) or by adding “physical perturbations” to objects under analysis (i.e. black and white stickers on objects captured by computer vision systems) (Eykholt et al., 2018). Among numerous adversarial attack methods, the C&W attack (Carlini & Wagner, 2017) and PGD attack (Madry et al., 2017) are frequently used to evaluate the robustness of DNNs. C&W attack (Carlini & Wagner, 2017) casts the adversarial attack task as an optimization problem and is originally proposed to challenge an adversarial defense method called defensive distillation (Papernot et al., 2016b). Variants of C&W attacks are based on the distance metrics (ℓ_0 , ℓ_2 , or ℓ_∞). Carlini and Wagner (2017), for example, can successfully defeat defensive distillation with high-confidence adversarial examples generated via C&W attack. Projected Gradient Descent (PGD) attack (Madry et al., 2017) in brief is an iterative version of an early stage adversarial attack called Fast Gradient Sign Method (FGSM) (Goodfellow et al., 2014). As indicated in its name, PGD attack generates adversarial examples based on the gradients of the loss with respect to the input. PGD attack is more favorable than C&W attack when direct control of input distortion is needed (Liu et al., 2018).

Adversarial defense is challenging due to the diversity of the adversarial example crafting processes and a DNN’s high-dimensional feature space. There exist two typical groups of adversarial defense methods, i) adversarial training (Madry et al., 2017; Goodfellow et al.,

2014; Szegedy et al., 2013), which is to augment the training dataset with generated adversarial examples such that the trained model is more robust against adversarial attack, and ii) removal perturbations (Samangouei et al., 2018; Meng & Chen, 2017), which dismisses adversarial perturbations from input data. Madry et al. (2017) integrates the PGD attack into the model training process, such that the model is optimized on both benign examples and challenging adversarial examples. The optimization is conducted in a min-max fashion, where the loss for adversarial attack process is maximized in order to generate strong adversarial examples, while the loss for the classification process is minimized, in order to get a robust and well-performed model. Samangouei et al. (2018), on the other hand, tackles the adversarial defense problem by filtering out adversarial perturbations. Generative Adversarial Networks (GANs) are leveraged to project a given input image, potentially polluted by adversarial perturbations, into a pseudo original image, where adversarial artifacts are diminished. The model decision is made based on the GAN generated “original” image, and experiments indicate this defense technique is effective against both black-box and white-box attacks.

3.4 Fairness and Bias

Model fairness aims to build DNN models that objectively consider each input feature and is not unduly biased against a particular subset of the input data. Although a firm definition of what it means for a DNN to be “fair” is evolving, common themes are emerging in the literature (Heidari et al., 2018). *Group fairness* (Calders et al., 2009), also called *demographic parity* or *statistical parity*, focuses on fairness with respect to a group (based on race, gender, etc.). The goal of group fairness is to ensure each group receives equalized percentage of benefit. Consider a loan application as an example. Suppose we are monitoring the loan approval situation of two cities, city A and city B. The population of city A is twice as much as that of city B. Based on the definition of group fairness, twice as many loan applications should be approved in A compared to city B. *Individual fairness* (Dwork et al., 2012) aims to treat similar inputs similarly based on a metric to measure the closeness of their features. To compare group fairness and individual fairness, let’s return to the loan request example. Under the restriction of group fairness, an individual from city A may not be approved for a loan request just because of the group percentage limitation, even though this individual is more qualified based on economic metrics than other approved ones from city B. However, individual fairness requires that individuals with similar characteristics should have the same chance to be approved for a loan request, regardless of which city individuals come from. This is in antithesis with group fairness. Further notions of fairness, such as *equalized odds* and *equal opportunity* (Hardt et al., 2016), avoiding *disparate mistreatment* (Zafar et al., 2017a), and others (Heidari et al., 2018; Woodworth et al., 2017) are also documented in the literature.

The fairness problem is currently addressed by three types of methods (Calmon et al., 2017): (i) *pre-processing* methods revise input data to remove information correlated to sensitive attributes; (ii) *in-process* methods add fairness constraints into the model learning process; and (iii) *post-process* methods adjust model predictions after the model is trained. *Pre-processing* methods (Kamiran & Calders, 2010, 2012; Zemel et al., 2013; Louizos et al., 2015; Adebayo & Kagal, 2016; Calmon et al., 2017; Gordaliza et al., 2019) learn an alter-

native representation of the input data that removes information correlated to the sensitive attributes (such as race or gender) while maintaining the model performance as much as possible. For example, Calmon et al. (2017) proposes a probabilistic framework to transform input data to prevent unfairness in the scope of supervised learning. The input transformation is conducted as an optimization problem, aiming to balance discrimination control (group fairness), individual distortion (individual fairness), and data utility. *In-process* methods (Calders et al., 2009; Kamishima et al., 2011; Zafar et al., 2017a; Woodworth et al., 2017; Zafar et al., 2017b; Bechavod & Ligett, 2017; Kearns et al., 2018; Pérez-Suay et al., 2017; Berk et al., 2017; Olfat & Aswani, 2018; Agarwal et al., 2018; Menon & Williamson, 2018; Donini et al., 2018; Dwork et al., 2018) directly introduce fairness learning constraints to the model in order to punish unfair decisions during training. Kamishima et al. (2011) achieves the fairness goal by adding a fairness regularizer, for example, such that the influence of sensitive information on model decisions is reduced. *Post-process* methods (Feldman et al., 2015; Hardt et al., 2016; Pleiss et al., 2017; Beutel et al., 2017) are characterized by adding ad-hoc fairness procedures to a trained model. One example is Hardt et al. (2016) which constructs non-discriminating predictors as a post-processing step to achieve equalized odds and equal opportunity (two fairness notions proposed in their study). They introduce the procedure to construct non-discriminating predictors for two scenarios of the original model, binary predictor and score function, where in the latter scenario the original model generates real score values in range $[0, 1]$. For the latter scenario, a non-discriminating predictor is constructed for each protected group, and a threshold is chosen to achieve a defined fairness goal.

4. Methods for Explaining DNNs

There are countless surveys on explainable AI (Arrieta et al., 2019; Carvalho et al., 2019; Mueller et al., 2019; Tjoa & Guan, 2019; Gilpin et al., 2018; Adadi & Berrada, 2018; Miller, 2018; Guidotti et al., 2018; Lipton, 2018; Liu et al., 2017; Došilović et al., 2018; Doshi-Velez & Kim, 2017) and explainable deep learning (Ras et al., 2018; Montavon et al., 2018; Zhang & Zhu, 2018; Samek et al., 2017b; Erhan et al., 2010). The surveys represent a large body of work that could be difficult to navigate and synthesize into a broad view of the field. Instead, this section investigates a simple space of **foundational** explainable DNN methods. We say a method is foundational if it is often used in practice or if it introduces a concept that modern work builds upon. Understanding this smaller space of foundational methods can thus help a reader appreciate and identify the key concepts behind modern approaches. Our space is three-dimensional and it encompasses:

- **Visualization methods:** Visualization methods express an explanation by highlighting, through a scientific visualization, characteristics of an input that strongly influence the output of a DNN.
- **Model distillation:** Model distillation develops a separate, “white-box” machine learning model that is trained to mimic the input-output behavior of the DNN. The white-box model, which is inherently explainable, is meant to identify the decision rules or input features influencing DNN outputs.

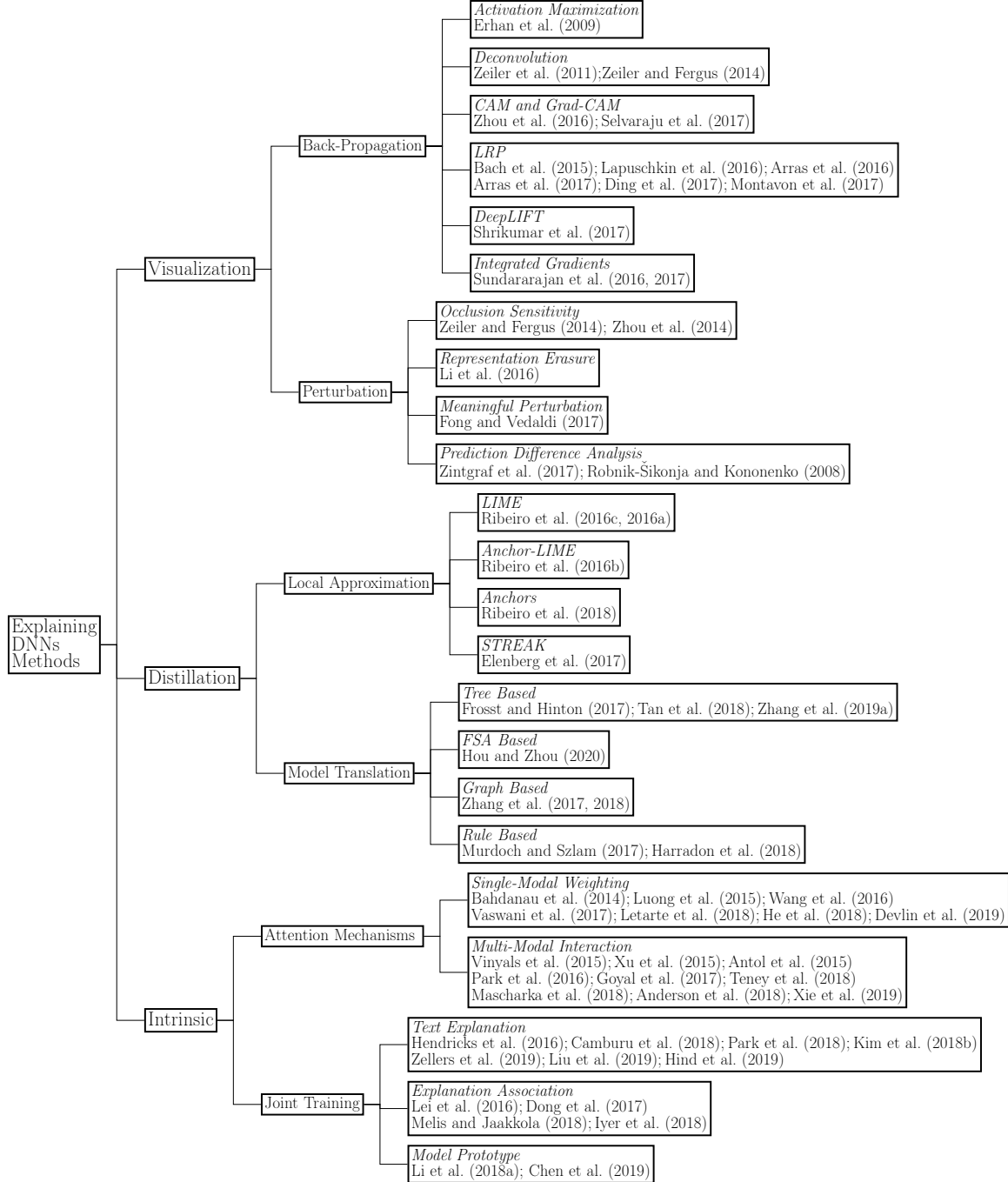


Figure 4: Methods for explaining DNNs.

Visualization Methods	Summary	References
Backpropagation-based	Visualize feature relevance based on volume of gradient passed through network layers during network training.	Erhan et al. (2009), Zeiler et al. (2011), Zeiler and Fergus (2014), Zhou et al. (2016), Selvaraju et al. (2017), Bach et al. (2015), Lapuschkin et al. (2016), Arras et al. (2016, 2017), Ding et al. (2017), Montavon et al. (2017), Shrikumar et al. (2017), Sundararajan et al. (2017, 2016)
Perturbation-based	Visualize feature relevance by comparing network output between an input and a modified copy of the input.	Zeiler and Fergus (2014), Zhou et al. (2014), Li et al. (2016), Fong and Vedaldi (2017), Robnik-Šikonja and Kononenko (2008), Zintgraf et al. (2017)

Table 1: Visualization methods.

- **Intrinsic methods:** Intrinsic methods are DNNs that have been specifically created to render an explanation along with its output. As a consequence of its design, intrinsically explainable deep networks can jointly optimize both model performance and some quality of the explanations produced.

4.1 Visualization Methods

Visualization methods associate the degree to which a DNN considers input features to a decision. A common explanatory form of visualization methods is *saliency maps*. These maps identify input features that are most salient, in the sense that they cause a maximum response or stimulation influencing the model’s output (Yosinski et al., 2015; Ozbulak, 2019; Olah et al., 2017, 2018; Carter et al., 2019). We break down visualization methods into two types, namely *back-propagation* and *perturbation-based* visualization. The types are summarized in Table 1 and will be discussed further below.

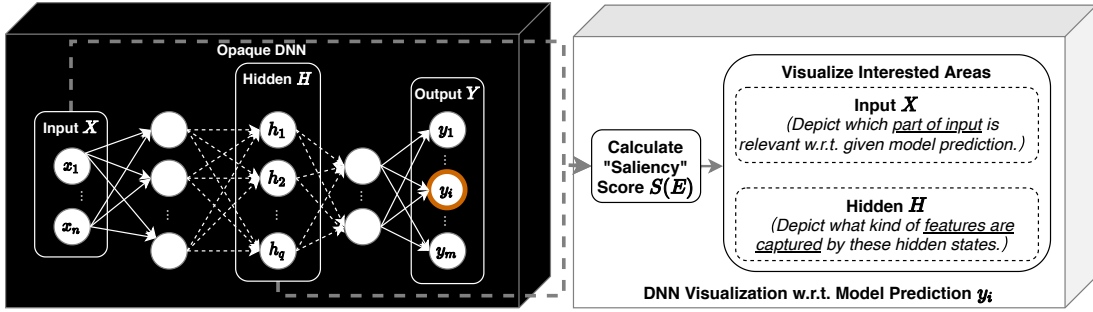


Figure 5: Visualization Methods. The to-be-visualized element E can be either the model input X or hidden states H . Visualization is based on the calculated saliency score $S(E)$, which varies along with different visualization methods.

4.1.1 BACKPROPAGATION-BASED METHODS

Backpropagation-based methods identify the saliency of input features based on some evaluation of gradient signals passed from output to input during network training. A baseline gradient-based approach visualizes the partial derivative of the network output with respect

to each input feature scaled by its value (Simonyan et al., 2013; Springenberg et al., 2014), thus quantifying the “sensitivity” of the network’s output with respect to input features. In a scene recognition task, for example, a high relevance score for pixels representing a bed in a CNN that decides the image is of class “bedroom” may suggest that the decision made by the CNN is highly sensitive to the presence of the bed in the image. Other gradient-based methods may evaluate this sensitivity with respect to the output, but from different collections of feature maps at intermediate CNN network layers (Zeiler & Fergus, 2014; Bach et al., 2015; Montavon et al., 2017; Shrikumar et al., 2017). We describe some foundational gradient-based methods below.

Activation maximization. One of the earliest works on visualization in deep architectures is proposed by Erhan et al. (2009). This seminal study introduces the activation maximization method to visualize important features in any layer of a deep architecture by optimizing the input \mathbf{X} such that the activation a of the chosen unit i in a layer j is maximized. Parameters θ of a trained network are kept fixed during activation maximization. The optimal \mathbf{X} is found by computing the gradient of $a_j^i(\mathbf{X}, \theta)$ and updating \mathbf{X} in the direction of the gradient. The practitioner decides the values of the hyperparameters for this procedure, i.e., the learning rate and how many iterations to run. The optimized \mathbf{X} will be a representation, in the input space, of the features that maximize the activation of a specific unit, or if the practitioner chooses so, multiple units in a specific network layer.

Deconvolution. Deconvolution was originally introduced as an algorithm to learn image features in an unsupervised manner (Zeiler et al., 2011). However, the method gained popularity because of its applications in visualizing higher layer features in the input space (Zeiler & Fergus, 2014), i.e., visualizing higher layer features in terms of the input. Deconvolution assumes that the model being explained is a neural network consisting of multiple convolutional layers. We will refer to this model as CNN. The consecutive layers of this network consist of a convolution of the previous layer’s output (or the input image in the case of the first convolutional layer) with a set of learned convolutional filters, followed by the application of the rectified linear function (ReLU) $ReLU(\mathbf{A}) = \max(\mathbf{A}, 0)$ on the output of the aforementioned convolution

$$\mathbf{A}^\ell, s^\ell = \text{maxpool}(ReLU(\mathbf{A}^{\ell-1} * \mathbf{K}^\ell + \mathbf{b}^\ell)) \quad (1)$$

where ℓ indicates the respective layer, \mathbf{A}^ℓ is the output of the previous layer, \mathbf{K} is the learned filter, and \mathbf{b} is the bias. If the outputs from the ReLU are passed through a local max-pooling function, it additionally stores the output s^ℓ containing the indices of the maximum values for a later unpooling operation. In the original paper, the set of s^ℓ ’s are referred to as *switches*. A deconvolutional neural network, referred to as DeCNN, consists of the inverse operations of the original convolutional network CNN. DeCNN takes the output of CNN as its input. In other words, DeCNN runs the CNN in reverse, from top-down. This is why the deconvolution method is classified as a back-propagation method. The convolutional layers in CNN are replaced with *deconvolutions* and the max-pooling layers are replaced with *unpooling* layers. A deconvolution is also called a *transposed convolution*, meaning that the values of \mathbf{K}^ℓ are transposed and then copied to the deconvolution filters $\mathbf{K}^{\ell T}$. If CNN included max-pooling layers, they are replaced with unpooling layers which approximately

upscales the feature map, retaining only the maximum values. This is done by retrieving the indices stored in s^ℓ at which the maximum values were located when the max-pooling was originally applied in CNN. As an example let us see the calculations involved in deconvolving Equation 1:

$$\mathbf{A}^{\ell-1} = \text{unpool}(\text{ReLU}((\mathbf{A}^\ell - \mathbf{b}^\ell) * \mathbf{K}^{\ell T}), s^\ell) \quad (2)$$

Using Equation 2 one or multiple learned filters \mathbf{K} in any layer of the network can be visualized by reverse propagating the values of \mathbf{K} all the way back to the input space. Finally, this study also describes how the visualizations can be used for architecture selection.

CAM and Grad-CAM. Zhou et al. (2016) describes a visualization method for creating class activation maps (**CAM**) using global average pooling (GAP) in CNNs. Lin et al. (2013) proposes the idea to apply a global average pooling on the activation maps of the last convolutional layer, right before the fully connected (FC) output layer. This results in the following configuration at the end of the CNN: $\text{GAP}(\text{Conv}) \rightarrow \text{FC} \rightarrow \text{softmax}$. The FC layer has C nodes, one for each class. The CAM method combines the activations \mathbf{A} from **Conv**, containing K convolutional filters, and weights $w_{k,c}$ from **FC**, where the (k, c) pair indicates the specific weighted connection from **Conv** to **FC**, to create relevance score map:

$$\text{map}_c = \sum_k^K w_{k,c} \mathbf{A}_k \quad (3)$$

The map is then upsampled to the size of the input image and overlaid on the input image, very similar to a heat map, resulting in the class activation map. Each class has its own unique CAM, indicating the image regions that were important to the network prediction for that class. CAM can only be applied on CNNs that employ the $\text{GAP}(\text{Conv}) \rightarrow \text{FC} \rightarrow \text{softmax}$ configuration.

Gradient-weighted Class Activation Map (**Grad-CAM**) (Selvaraju et al., 2017) is a generalization of the CAM method that uses the gradients of the network output with respect to the last convolutional layer to achieve the class activation map. This allows Grad-CAM to be applicable to a broader range of CNNs compared to CAM, only requiring that the final activation function used for network prediction to be a differentiable function, e.g., softmax. Recall that the final convolutional layer has an output of K feature maps. For each feature map \mathbf{A}_k in the final convolutional layer of the network, a gradient of the score y_c (the value before softmax, also known as *logit*) of class c with respect to every node in \mathbf{A}_k is computed and averaged to get an importance score $\alpha_{k,c}$ for feature map \mathbf{A}_k :

$$\alpha_{k,c} = \frac{1}{m \cdot n} \sum_{i=1}^m \sum_{j=1}^n \frac{\partial y_c}{\partial \mathbf{A}_{k,i,j}} \quad (4)$$

where $\mathbf{A}_{k,i,j}$ is a neuron positioned at (i, j) in the $m \times n$ feature map \mathbf{A}_k . Grad-CAM linearly combines the importance scores of each feature map and passes them through a ReLU to obtain an $m \times n$ -dimensional relevance score map

$$\text{map}_c = \text{ReLU}\left(\sum_k^K \alpha_{k,c} \mathbf{A}_k\right) \quad (5)$$

The relevance score map is then upsampled via bilinear interpolation to be of the same dimension as the input image to produce the class activation map.

Layer-Wise Relevance Propagation. LRP methods create a saliency map that, rather than measuring sensitivity, represents the relevance of each input feature to the output of the network (Bach et al., 2015; Lapuschkin et al., 2016; Arras et al., 2016, 2017; Ding et al., 2017; Montavon et al., 2017). While *sensitivity* measures the change in response in the network’s output as a result of changing attributes in the input (Kindermans et al., 2019), *relevance* measures the strength of the connection between the input or pixel to the specific network output (without making any changes to the input or the components of the network). LRP methods decompose the output value $f(\mathbf{x})$ of a deep network f across input features $\mathbf{x} = (x_1, x_2, \dots, x_N)$, such that $f(\mathbf{x}) = \sum_i r_i$ where r_i is the relevance score of feature x_i . Perhaps the most generic type of LRP is called **Deep Taylor Decomposition** (Montavon et al., 2017). The method is based on the fact that f is differentiable and hence can be approximated by a Taylor expansion of f at some root $\hat{\mathbf{x}}$ for which $f(\hat{\mathbf{x}}) = 0$

$$\begin{aligned} f(\mathbf{x}) &= f(\hat{\mathbf{x}}) + \nabla_{\hat{\mathbf{x}}} f \cdot (\mathbf{x} - \hat{\mathbf{x}}) + \epsilon \\ &= \sum_i^N \frac{\partial f}{\partial x_i}(\hat{x}_i) \cdot (x_i - \hat{x}_i) + \epsilon \end{aligned} \quad (6)$$

where ϵ encapsulates all second order and higher terms in the Taylor expansion. A good root point is one that is as minimally different from \mathbf{x} and that causes the function $f(\mathbf{x})$ to output a different prediction. The relevance score for inputs can then be seen as the terms inside of the summation:

$$r_i = \frac{\partial f}{\partial x_i}(\hat{x}_i) \cdot (x_i - \hat{x}_i) \quad (7)$$

To extend this idea to a deep network, the deep Taylor decomposition algorithm considers a conservative decomposition of relevance scores across layers of the network, starting from the output, through each hidden layer, back to the input. Thus, the method requires that the relevance score of a node i at layer l , denoted r_i^l be decomposable into

$$r_i^\ell = \sum_j^M r_{i,j}^\ell \quad (8)$$

where the summation is taken over all M nodes in layer $\ell + 1$ that node i in layer ℓ connects or contributes to. This indicates that the relevance score of the later layers can be back-propagated to generate the relevance score of former layers. The relevance score with respect to the input space can thus be calculated by conducting this decomposition rule layer by layer. Further details can be found in the original paper (Montavon et al., 2017).

DeepLIFT. Deep Learning Important FeaTures (DeepLIFT) is another important back-propagation based approach, proposed by Shrikumar et al. (2017). It assigns relevance scores to input features based on the difference between an input \mathbf{x} and a “reference” input \mathbf{x}' . The reference should be chosen according to the problem at hand and can be found by answering the question “*What am I interested in measuring differences against?*”. In an example using

MNIST the reference chosen is an input of all zeros as this is the background value in the images. Define $\Delta t = f(\mathbf{x}) - f(\mathbf{x}')$ as the difference-from-reference of an interested neuron output of the network between \mathbf{x} and reference \mathbf{x}' , and $\Delta \mathbf{x} = \mathbf{x} - \mathbf{x}'$ as the difference between \mathbf{x} and \mathbf{x}' . DeepLIFT assigns a relevance score $R_{\Delta x_i \Delta t}$ for input feature x_i :

$$\Delta t = \sum_{i=1}^N R_{\Delta x_i \Delta t} \quad (9)$$

where N is the number of input neurons that are necessary to compute t . In this formulation, $R_{\Delta x_i \Delta t}$ can be thought of as a weight denoting how much influence Δx_i had on Δt . According to Equation 9 the sum of the all weights is equal to the difference-from-reference output Δt . The relevance score can be calculated via the *Linear* rule, *Rescale* rule, or *RevealCancel* rule, as elaborated in their study. A multiplier $m_{\Delta \mathbf{x} \Delta t}$ is defined as

$$m_{\Delta \mathbf{x} \Delta t} = \frac{R_{\Delta \mathbf{x} \Delta t}}{\Delta \mathbf{x}} \quad (10)$$

indicating the relevance of $\Delta \mathbf{x}$ with respect to Δt , averaged by $\Delta \mathbf{x}$. Given a hidden layer ℓ of nodes $\mathbf{a}^\ell = (a_1^\ell, a_2^\ell, \dots, a_K^\ell)$, whose upstream connections are the input nodes $\mathbf{x} = (x_1, x_2, \dots, x_N)$, and a downstream target node is t , the DeepLIFT paper proves the effectiveness of the “chain rule” as illustrated below:

$$m_{\Delta x_i \Delta t} = \sum_{j=1}^K m_{\Delta x_i \Delta a_j^\ell} m_{\Delta a_j^\ell \Delta t} \quad (11)$$

This “chain rule” allows for layer-by-layer computation of the relevance scores of each hidden layer node via backpropagation. The DeepLIFT paper and appendix specify particular rules for computing $m_{\Delta x_i \Delta a_j^\ell}$ based on the architecture of the hidden layer \mathbf{a}^ℓ .

Integrated Gradients. Integrated gradients (Sundararajan et al., 2017) is an “axiomatic attribution” map that satisfies two axioms for input feature relevance scoring on a network f . The first axiom is *sensitivity*: compared to some baseline input \mathbf{x}' , when input \mathbf{x} differs from \mathbf{x}' along feature x_i and $f(\mathbf{x}) \neq f(\mathbf{x}')$, then x_i should have a non-zero relevance score. The second axiom is *implementation invariance*: for two networks f_1 and f_2 whose outputs are equal for all possible inputs, the relevance score for every input feature x_i should be identical over f_1 and f_2 . The break of the second axiom may potentially result in the sensitivity of relevance scores on irrelevant aspects of a model.

Given a deep network f whose codomain is $[0, 1]$, an input \mathbf{x} , and a baseline input \mathbf{x}' , the relevance of feature x_i of input \mathbf{x} over f is taken as the integral of the gradients of f along the straight line path from \mathbf{x}' to \mathbf{x} :

$$IG_i(\mathbf{x}) = (x_i - x'_i) \int_0^1 \frac{\partial f(\mathbf{x}' + \alpha(\mathbf{x} - \mathbf{x}'))}{\partial x_i} d\alpha \quad (12)$$

where α is associated with the path from \mathbf{x}' to \mathbf{x} , and is smoothly distributed in range $[0, 1]$. An interpretation of IG_i is the cumulative sensitivity of f to changes in feature i in all inputs on a straight line between \mathbf{x}' to \mathbf{x} going in direction i . Intuitively, x_i should have

increasing relevance if gradients are large between a “neutral” baseline point \mathbf{x}' and \mathbf{x} along the direction of x_i . IG can be approximated by a Riemann summation of the integral:

$$IG_i(\mathbf{x}) \cong (x_i - x'_i) \sum_{k=1}^M \frac{\partial F(\mathbf{x}' + \frac{k}{M}(\mathbf{x} - \mathbf{x}'))}{\partial x_i} \frac{1}{M} \quad (13)$$

where M is the number of steps in the Riemman approximation of this integral. In the original paper the authors propose setting M somewhere between 20 and 300 steps.

4.1.2 PERTURBATION-BASED METHODS

Perturbation-based methods compute input feature relevance by altering or removing the input feature and comparing the difference in network output between the original and altered one. Perturbation methods can compute the marginal relevance of each feature with respect to how a network responds to a particular input. Below are some visualization methods based on perturbation.

Occlusion Sensitivity. The approach proposed by Zeiler and Fergus (2014) sweeps a grey patch that occludes pixels over the image and sees how the model prediction varies as the patch covering different positions: when the patch covers a critical area, such as a dog’s face for the class *Pomeranian*, or a car’s wheel for the class *Car Wheel*, the prediction performance drops significantly. The visualization depicts the area sensitivity of an image with respect to its classification label. A variant of this method is implemented in (Zhou et al., 2014), where small gray squares are used to occlude image patches in a dense grid.

Representation Erasure. Li et al. (2016) focuses on providing explanations for natural language-related tasks. To measure the effectiveness of each input word or each dimension of intermediate hidden activations, the method erases the information by deleting a word or setting a dimension to zeros and observes the influences on model predictions correspondingly. Reinforcement learning is adopted to evaluate the influence of multiple words or phrases combined by finding the minimum changes in the text that causes a flipping of a neural network’s decision.

Meaningful Perturbation. Fong and Vedaldi (2017) formally defines an explanation as a meta-predictor, which is a rule that predicts the output of a black box f to certain inputs. For example, the explanation for a classifier that identifies a bird in an image can be defined as

$$B(\mathbf{x}; f) = \{\mathbf{x} \in \mathbf{X}_c \Leftrightarrow f(\mathbf{x}) = +1\} \quad (14)$$

where $f(\mathbf{x}) = +1$ means a bird is present and \mathbf{X}_c is the set of all images that the DNN predicts a bird exists. Given a specific image \mathbf{x}^0 and a DNN f , the visualization is generated via perturbation to identify sensitive areas of \mathbf{x}^0 with respect to the output $f(\mathbf{x}^0)$ formulated as a local explanation (“local” to \mathbf{x}^0) by the author. The author defines three kinds perturbations to delete information from image, i) *constant*, replacing region with a constant value ii) *noise*, adding noise to the region, and iii) *blur*, blurring the region area, and generating explainable visualization respectively.

Model Distillation	Comments	References
Local Approximation	Learns a simple model whose input/output behavior mimics that of a DNN for a small subset of input data.	Ribeiro et al. (2016c, 2016a, 2016b, 2018), Elenberg et al. (2017)
Model Translation	Train an alternative smaller model that mimics the input/output behavior of a DNN.	Frosst and Hinton (2017), Tan et al. (2018), Zhang et al. (2019a), Hou and Zhou (2020), Zhang et al. (2017, 2018), Harradon et al. (2018), Murdoch and Szlam (2017)

Table 2: Model distillation.

Prediction Difference Analysis. Zintgraf et al. (2017) proposes a rigorous approach to delete information from an input and measure its influence accordingly. The method is based on (Robnik-Šikonja & Kononenko, 2008), which evaluates the effect of feature x_i with respect to class c by calculating the prediction difference between $p(c \mid \mathbf{x}_{-i})$ and $p(c \mid \mathbf{x})$ using the marginal probability

$$p(c \mid \mathbf{x}_{-i}) = \sum_{x_i} p(x_i \mid \mathbf{x}_{-i}) p(c \mid \mathbf{x}_{-i}, x_i) \quad (15)$$

where \mathbf{x} denotes all input features, \mathbf{x}_{-i} denotes all features except x_i , and the sum iterates over all possible values of x_i . The prediction difference, also called *relevance value* in the paper, is then calculated by

$$\text{Diff}_i(c \mid \mathbf{x}) = \log_2(\text{odds}(c \mid \mathbf{x})) - \log_2(\text{odds}(c \mid \mathbf{x}_{-i})) \quad (16)$$

where $\text{odds}(c \mid \mathbf{x}) = \frac{p(c \mid \mathbf{x})}{1 - p(c \mid \mathbf{x})}$. The magnitude of $\text{Diff}_i(c \mid \mathbf{x})$ measures the importance of feature x_i . $\text{Diff}_i(c \mid \mathbf{x})$ measures the influence direction of feature x_i , where a positive value means *for* decision c and a negative value means *against* decision c . Compared to Robnik-Šikonja and Kononenko (2008), Zintgraf *et al.* improves prediction difference analysis in three ways: by i) sampling patches instead of pixels given the high pixel dependency nature of images; ii) removing patches instead of individual pixels to measure the prediction influence given the robustness nature of neural networks on individual pixels; iii) adapting the method to measure the effect of intermediate layers by changing the activations of a given intermediate layer and evaluate the influence on down-streaming layers.

4.2 Model Distillation

In this review we use the term *model distillation* to refer to a class of post-training explanation methods where the knowledge encoded within a trained DNN is *distilled* into a representation amenable for explanation by a user. The reader should be aware that Hinton et al. (2015) outlines a method, with the same name, that implements a specific form of model distillation, namely distilling the knowledge learned by an ensemble of DNNs into a single DNN. An entire class of explainable deep learning techniques have emerged which are based on the notion of model distillation. In this setting, as illustrated in Figure 6, an inherently transparent or white box machine learning model g is trained to mimic the input/output behavior of a trained opaque deep neural network f so that $g(\mathbf{y}) \approx f(\mathbf{x})$. Subsequent explanation of how g maps inputs to outputs may serve as a surrogate explanation of f 's mapping.

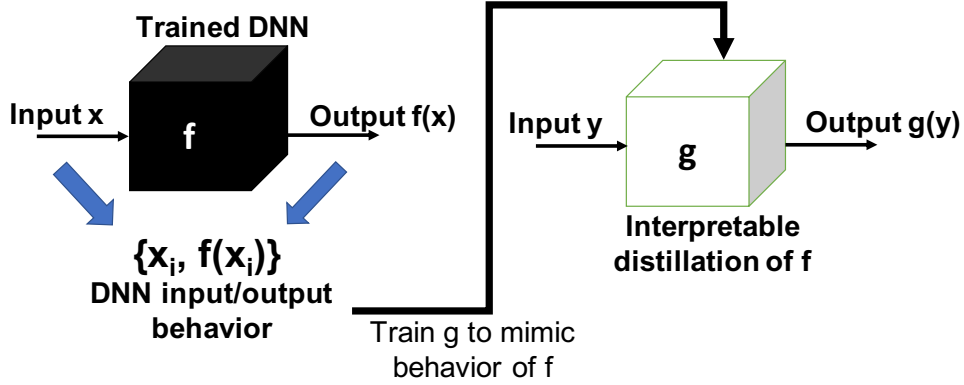


Figure 6: Model distillation. The behavior of a trained deep learning model f used as training data for an explainable model g .

A distilled model in general learns to imitate the actions or qualities of an opaque DNN on the same data used to train the opaque DNN. Distilled models, even if they are simpler, smaller, and possibly explainable, can still achieve reasonable performance while offering an explanation. There are two conceptual reasons for this. First, a distilled model has access to information from the trained DNN, including the input features it found to be most discriminatory and feature or output correlations relevant for classification. The distilled model can use this information directly during training, thus reducing the needed capacity of the distilled model. Second, the distilled model still takes the original data as input, and if explainable, thus develops a transparent model of how input features become related to the actions of the DNN. Interpreting the distilled model may thus not give very deep insights into the internal representation of the data a DNN learns, or say anything about the DNN’s learning process, but can at least provide insight into the features, correlations, and relational rules that explain how the DNN operates. Put another way, we can imagine that the explanation of a distilled model can be seen as a hypothesis as to why a DNN has assigned some class label to an input.

We organize model distillation techniques for explainable deep learning into the following two categories:

- **Local Approximation.** A local approximation method learns a simple model whose input/output behavior mimics that of a DNN for a small subset of the input data. This method is motivated by the idea that the model a DNN uses to discriminate within a local area of the data manifold is simpler than the discriminatory model over the entire surface. Given a sufficiently high local density of input data to approximate the local manifold with piecewise linear functions, the DNN’s behavior in this local area may be distilled into a set of explainable linear discriminators.
- **Model Translation.** Model translations train an alternative smaller model that mimics the input/output behavior of a DNN. They contrast local approximation methods in replicating the behavior of a DNN across an entire dataset rather than small sub-

sets. The smaller models may be directly explainable, may be smaller and easier to deploy, or could be further analyzed to gain insights into the causes of the input/output behavior that the translated model replicates.

4.2.1 LOCAL APPROXIMATION

A local approximation method learns a distilled model that mimics DNN decisions on inputs within a small subset of the input examples. Local approximations are made for data subsets where feature values are very similar, so that a simple and explainable model can make decisions within a small area of the data manifold. While the inability to explain *every* decision of a DNN may seem unappealing, it is often the case that an analyst or practitioner is most interested in interpreting DNN actions under a particular subspace (for example, the space of gene data related to a particular cancer or the space of employee performance indicators associated with those fired for poor performance).

The idea of applying local approximations may have originated from Baehrens et al. (2010). These researchers presented the notion of an “explainability vector”, defined by the derivative of the conditional probability a datum is of a class given some evidence x_0 by a Bayes classifier. The direction and magnitude of the derivatives at various points x_0 along the data space define a vector field that characterizes flow away from a corresponding class. The work imitates an opaque classifier in a local area by learning a Parzen window classifier that has the same form as a Bayes estimator for which the explanation vectors can be estimated.

Perhaps the most popular local approximation method is **LIME** (Local Interpretable Model-agnostic Explanations) developed by Ribeiro et al. (2016c). From a *global*, unexplainable model f and a datum of interest $\mathbf{x} \in \mathbb{R}^d$, LIME defines an interpretable model $g_{\mathbf{x}}$ from a class of *inherently* interpretable models $g_{\mathbf{x}} \in G$ with different domain $\mathbb{R}^{d'}$ that approximates f well in the local area around \mathbf{x} . Examples of models in G may be decision trees or regression models whose weights explain the relevance of an input feature to a decision. Note that the domain of $g_{\mathbf{x}}$ is different from that of f . $g_{\mathbf{x}}$ operates over some *interpretable representation* of the input data presented to the unexplainable model f , which could for example be a binary vector denoting the presence or absence of words in text input, or a binary vector denoting if a certain pixel or color pattern exists in an image input. Noting that $g_{\mathbf{x}}$ could be a decision tree with very high depth, or a regression model with many co-variate weights, an interpretable model that is overly complex may still not be useful or usable to a human. Thus, LIME also defines a measure of complexity $\Omega(g_{\mathbf{x}})$ on $g_{\mathbf{x}}$. $\Omega(g_{\mathbf{x}})$ could measure the depth of a decision tree or the number of higher order terms in a regression model, for example, or it could be coded as if to check that a hard constraint is satisfied (e.g., $\Omega(g_{\mathbf{x}}) = \infty$ if $g_{\mathbf{x}}$ is a tree and its depth exceeds some threshold). Let $\Pi_{\mathbf{x}}(\mathbf{z})$ be a similarity kernel between some data point \mathbf{z} and a reference data point $\mathbf{x} \in \mathbb{R}^d$ and a loss $\mathcal{L}(f, g_{\mathbf{x}}, \Pi_{\mathbf{x}})$ defined to measure how poorly $g_{\mathbf{x}}$ approximates f on data in the area $\Pi_{\mathbf{x}}$ around the data point \mathbf{x} . To interpret $f(\mathbf{x})$, LIME identifies the model $g_{\mathbf{x}}$ satisfying:

$$\arg \min_{g_{\mathbf{x}} \in G} \mathcal{L}(f, g_{\mathbf{x}}, \Pi_{\mathbf{x}}) + \Omega(g_{\mathbf{x}}) \quad (17)$$

$\Omega(g_{\mathbf{x}})$ thus serves as a type of complexity regularization, or as a guarantee that the returned model will not be too complex when $\Omega(g_{\mathbf{x}})$ codes a hard constraint. So that LIME remains

model agnostic, \mathcal{L} is approximated by uniform sampling over the non-empty space of $\mathbb{R}^{d'}$. For each sampled data point $\mathbf{y}' \in \mathbb{R}^{d'}$, LIME recovers the $\mathbf{x} \in \mathbb{R}^d$ corresponding to \mathbf{y}' , computes $f(\mathbf{x})$, and compares this to $g_{\mathbf{x}}(\mathbf{y}')$ using \mathcal{L} . To make sure that the $g_{\mathbf{x}}$ minimizing Equation 17 fits well in the area local to a reference point \mathbf{x} , the comparison of $f(\mathbf{y})$ to $g_{\mathbf{x}}(\mathbf{y}')$ in \mathcal{L} is weighted by $\Pi_{\mathbf{x}}(\mathbf{y})$ such that sampled data farther from \mathbf{x} has lower contribution to loss.

We mention LIME in detail because other popular local approximation models (Ribeiro et al., 2016a, 2016b, 2018; Elenberg et al., 2017) follow LIME’s template and make their extensions or revisions. One drawback of LIME, which uses a linear combination of input features to provide local explanations, is the precision and coverage of such explanations are not guaranteed. Since the explanations are generated in a locally linear fashion, for an unseen instance, which might lie outside of the region where a linear combination of input features could represent, it is unclear if an explanation generated linearly and locally still applies. To address this issue, *anchor* methods (Ribeiro et al., 2016b, 2018) extend LIME to produce local explanations based on if-then rules, such that the explanations are locally anchored upon limited yet sufficiently stable input features for the given instance and the changes to the rest of input features won’t make an influence. Another drawback of LIME is, the interpretable linear model is trained based on a large set of randomly perturbed instances and the class label of each perturbed instance is assigned by inevitably calling the complex opaque model, which is computationally costly. To reduce the time complexity, Elenberg et al. (2017) introduces STREAK, which is similar to LIME but limits the time of calling complex models, and thus runs much faster. Instead of randomly generating instances and training an interpretable linear model, STREAK directly selects critical input components (for example, superpixels of images) by greedily solving a combinatorial maximization problem. Taking the image classification task as an example, an input image which is predicted as a class by the opaque model, is first segmented into superpixels via image segmentation algorithm (Achanta et al., 2012). In every greedy step, a new superpixel is added to the superpixels set if by containing it in the image will maximize the probability of the opaque model on predicting the given class. The set of superpixels indicating the most important image regions of the given input image for the opaque model to make its decision. Despite the technical details, some common characteristics are shared among all aforementioned local approximation methods, i) input instance is segmented into semantic meaningful parts for selection, ii) function calls of the original opaque model is inevitable iii) the explanations and model behavior are explored in a local fashion.

4.2.2 MODEL TRANSLATION

Compared to local approximation methods, model translation replicates the behavior of a DNN across an *entire* dataset rather than small subsets, through a smaller model that is easier for explanation. The smaller model could be easier to deploy (Hinton et al., 2015), faster to converge (Yim et al., 2017), or even be easily explainable, such as a decision tree (Frosst & Hinton, 2017; Bastani et al., 2017; Tan et al., 2018), Finite State Automaton (FSA) (Hou & Zhou, 2020), graphs (Zhang et al., 2017, 2018), or causal and rule-based classifier (Harradon et al., 2018; Murdoch & Szlam, 2017). We highlight the diversity of model types DNNs have

been distilled into below.

Distillation to Decision Trees. Recent work has been inspired by the idea of tree-based methods for DNNs. Frosst and Hinton (2017) proposes “soft decision trees” which use stochastic gradient descent for training based on the predictions and learned filters of a given neural network. The performance of the soft decision trees is better than normal trees trained directly on the given dataset, but is worse compared to the given pre-trained neural networks. Another recent work is proposed by Tan et al. (2018) which generates global additive explanations for fully connected neural networks trained on tabular data through model distillation. Global additive explanations (Sobol, 2001; Hooker, 2004; Hoos & Leyton-Brown, 2014) have been leveraged to study complex models, including analyzing how model parameters influence model performance and decomposing black box models into lower-dimensional components. In this work, the global additive explanations are constructed by following previous work Hooker (2007) to decompose the black-box model into an additive model such as spline or bagged tree. Then follow Craven and Shavlik (1996) to train the additive explainable model. Zhang et al. (2019a) trains a decision tree to depict the reasoning logic of a pretrained DNN with respect to given model predictions. The authors first mine semantic patterns, such as objects, parts, and “decision modes” as fundamental blocks to build the decision tree. The tree is then trained to quantitatively explain which fundamental components are used for a prediction and the percentage of contribution respectively. The decision tree is organized in a hierarchical coarse-to-fine way, thus nodes close to the tree top correspond to common modes shared by multiple examples, while nodes at the bottom represent fine-grained modes with respect to specific examples.

Distillation to Finite State Automata. Hou and Zhou (2020) introduces a new distillation of RNNs to explainable Finite State Automata (FSA). An FSA consists of finite states and transitions between the states, and the transition from one state to another is a result of external input influence. FSA is formally defined as a 5-tuple $(\mathbb{E}, \mathbb{S}, s_0, \delta, \mathbb{F})$, where \mathbb{E} is a finite non-empty set of elements existing in input sequences, \mathbb{S} is a finite non-empty set of states, $s_0 \in \mathbb{S}$ is an initial state, $\delta : \mathbb{S} \times \mathbb{E} \rightarrow \mathbb{S}$ defines the state transmission function, and $F \subseteq \mathbb{S}$ is the set of final states. The transition process of FSA is similar to RNNs in the sense that both methods accept items from some sequence one by one and transit between (hidden) states accordingly. The idea to distillate an RNN to FSA is based on the fact that the hidden states of an RNN tend to form clusters, which can be leveraged to build FSA. Two clustering methods, *k-means++* and *k-means-x* are adopted to cluster the hidden states of RNN towards constructing the explainable FSA model. The authors follow the structure learning technique and translate an RNN into an FSA, which is easier to interpret in two aspects, i) FSA can be simulated by humans; ii) the transitions between states in FSA have real physical meanings. Such a model translation helps to understand the inner mechanism of the given RNN model.

Distillation into Graphs. Both Zhang et al. (2017) and Zhang et al. (2018) build an object parts graph for a pre-trained CNN to provide model explanations. Similar to Zhang et al. (2019a), the authors first extract semantic patterns in the input and then gradually construct the graph for explanation. Each node in the graph represents a part pattern, while

Intrinsic Methods	Comments	References
Attention Mechanisms	Leverage attention mechanisms to learn conditional distribution over given input units, composing a weighted contextual vector for downstream processing. The attention visualization reveals inherent explainability.	Bahdanau et al. (2014), Luong et al. (2015), Vaswani et al. (2017), Wang et al. (2016), Letarte et al. (2018), He et al. (2018), Devlin et al. (2019), Vinyals et al. (2015), Xu et al. (2015), Antol et al. (2015), Goyal et al. (2017), Teney et al. (2018), Mascharka et al. (2018), Anderson et al. (2018), Xie et al. (2019), Park et al. (2016)
Joint Training	Add additional explanation “task” to the original model task, and <i>jointly train</i> the explanation task along with the original task.	Zellers et al. (2019), Liu et al. (2019), Park et al. (2018), Kim et al. (2018b), Hendricks et al. (2016), Camburu et al. (2018), Hind et al. (2019), Melis and Jaakkola (2018), Iyer et al. (2018), Lei et al. (2016), Dong et al. (2017), Li et al. (2018a), Chen et al. (2019)

Table 3: Intrinsic methods.

each edge represents co-activation or spatial adjacent between part patterns. The explanatory graph explains the knowledge hierarchy inside of the model, which can depict which nodes/part patterns are activated as well as the location of the parts in the corresponding feature maps.

Distillation into Causal and Rule-based Models. We also note work on distilling a DNN into symbolic rules and causal models. Harradon et al. (2018) constructs causal models based on concepts in a DNN. The semantics are defined over an arbitrary set of “concepts”, that could range from recognition of groups of neuron activations up to labeled semantic concepts. To construct the causal model, concepts of intermediate representations are extracted via an autoencoder. Based on the extracted concepts, a graphical Bayesian causal model is constructed to build association for the models’ inputs to concepts, and concepts to outputs. The causal model is finally leveraged to identify the input features of significant causal relevance with respect to a given classification result.

In another example, Murdoch and Szlam (2017) leverages a simple rule-based classifier to mimic the performance of an LSTM model. This study runs experiments on two natural language processing tasks, sentiment analysis, and question answering. The rule-based model is constructed via the following steps: i) decompose the outputs of an LSTM model, and generate important scores for each word; ii) based on the word level important score, important simple phrases are selected according to which jointly have high important scores; iii) The extracted phrase patterns are then used in the rule-based classifier, approximating the output of the LSTM model.

4.3 Intrinsic Methods

Ideally, we would like to have models that provide explanations for their decisions as part of the model output, or that the explanation can easily be derived from the model architecture. In other words, explanations should be *intrinsic* to the process of designing model architectures and during training. The ability for a network to intrinsically express an explanation may be more desirable compared to post-hoc methods that seek explanations of models that were never designed to be explainable in the first place. This is because an intrinsic model

has the capacity to learn not only accurate outputs per input but also outputs expressing an explanation of the network’s action that is optimal with respect to some notion of explanatory fidelity. Ras et al. (2018) previously defined a category related to this approach as *intrinsic methods* and identified various methods that offer explainable extensions of the model architecture or the training scheme. In this section, we extend the notion of intrinsic explainability with models that actually provide an explanation for their decision even as they are being trained.

We observe methods in the literature on intrinsically explainable DNNs to follow two trends: (i) they introduce *attention mechanisms* to a DNN, and the attention visualization reveals inherent explainability; (ii) they add additional explanation “task” to the original model task, and *jointly train* the explanation task along with the original task. We explain the trends and highlight the representative methods below.

4.3.1 ATTENTION MECHANISMS

DNNs can be endowed with attention mechanisms that simultaneously preserve or even improve their performance and have explainable outputs expressing their operations. An attention mechanism (Vaswani et al., 2017; Devlin et al., 2019; Teney et al., 2018; Xie et al., 2019) learns conditional distribution over given input units, composing a weighted contextual vector for downstream processing. The attention weights can be generated in multiple ways, such as by calculating cosine similarity (Graves et al., 2014), adding additive model structure, such as several fully connected layers, to explicitly generate attention weights (Bahdanau et al., 2014), leveraging the matrix dot-product (Luong et al., 2015) or scaled dot-product (Vaswani et al., 2017), and so on. Attention mechanisms have shown to improve DNN performance for particular types of tasks, including tasks on ordered inputs as seen in natural language processing (Vaswani et al., 2017; Devlin et al., 2019) and multi-modal fusion such as visual question answering (Anderson et al., 2018). It is worth noting that recently there appear some interesting discussions on whether or not attention can be counted as an explanation tool (Jain & Wallace, 2019; Wiegrefe & Pinter, 2019), however we would like to leave such discussions to the readers for further exploration.

Single-Modal Weighting. The output of attention mechanisms during a forward pass can inform a user about how strongly weighted that different input features are considered at different phases of model inference. In pure text processing tasks such as language translation (Bahdanau et al., 2014; Luong et al., 2015; Vaswani et al., 2017) or sentiment analysis (Wang et al., 2016; Letarte et al., 2018; He et al., 2018), attention mechanism allows the downstream modules, a decoder for language translation or fully connected layers for classification tasks, to concentrate on different words in the input sentence by assigning learned weights to them (Vaswani et al., 2017; Wang et al., 2016). To provide straightforward explanations, the attention weights can be visualized as heatmaps, depicting the magnitude and the sign (positive or negative) of each weight value, showing how input elements weighted combined to influence the model latter processing and the final decisions.

Multi-Modal Interaction. In multi-modal interaction tasks, such as image captioning (Vinyals et al., 2015; Xu et al., 2015), visual question answering (Antol et al., 2015;

Goyal et al., 2017; Johnson et al., 2017; Teney et al., 2018) or visual entailment (Xie et al., 2019), attention mechanisms play an important role in feature alignment and fusion across different feature spaces (for instance, between text and images). For example, Park *et al.* propose the Pointing and Justification model (PJ-X) that uses multiple attention mechanisms to explain the answer of a VQA task with natural language explanations and image region alignments (Park et al., 2016). Xie *et al.* use attention mechanisms to recover semantically meaningful areas of an image that correspond to the reason a statement is, is not, or could be entailed by the image’s conveyance (Xie et al., 2019). Mascharka et al. (2018) aims to close the gap between performance and explainability in visual reasoning by introducing a neural module network that explicitly models an attention mechanism in image space. By passing attention masks between modules it becomes explainable by being able to directly visualize the masks. This shows how the attention of the model shifts as it considers the different components of the input.

4.3.2 JOINT TRAINING

This type of intrinsic method is to introduce an additional “task” besides the original model task, and jointly train the additional task together with the original one. Here we generalize the meaning of a “task” by including preprocessing or other steps involved in the model optimization process. The additional task is designed to provide model explanations directly or indirectly. Such additional task can be in the form of i) text explanation, which is a task that directly provides explanations in natural language format; ii) explanation association, which is a step that associates input elements or latent features with human-understandable concepts or objects, or even directly with model explanations; iii) model prototype, which is to learn a prototype that has clear semantic meanings as a preprocessing step, and the model explanation is generated based on the comparison between the model behavior and the prototype.

Text Explanation. A group of recent work (Zellers et al., 2019; Liu et al., 2019; Park et al., 2018; Kim et al., 2018b; Hendricks et al., 2016; Camburu et al., 2018; Hind et al., 2019) achieve the explainable goal via augmenting the original DNN architecture with an explanation generation component and conducting joint training to provide natural language explanations along with the model decisions. Such explainable methods are quite straightforward and layman-friendly since the explanations are presented directly using natural language sentences, instead of figures or statistical data that usually require professional knowledge to digest. The explanation could be either generated word by word similar to a sequence generation task (Hendricks et al., 2016; Park et al., 2018; Camburu et al., 2018; Kim et al., 2018b; Liu et al., 2019), or be predicted from multiple candidate choices (Zellers et al., 2019). Despite how explanations are provided, there exist two facts that may potentially limit the usage of such intrinsic methods. First, these explainable methods require supervision on explanations during training, thus the dataset should have corresponding explanation annotations. However, such a dataset is relatively rare in real life and extra efforts should be paid to generate annotations. Second, a recent work (Oana-Maria et al., 2019) discovers there exists inconsistency in generated explanations, which undermines the

trust in the explanations provided by the model. Keeping both the merits and limitations in mind, we now introduce some related work below.

Hendricks et al. (2016) is an early work that provides text justifications along with its image classification results. The approach combines image captioning, sampling, and deep reinforcement learning to generate textual explanations. The class information is incorporated into the text explanations, which makes this method distinct from normal image captioning models that only consider visual information, via i) include class as an additional input for text generation and ii) adopt a reinforcement learning based loss that encourages generated sentences to include class discriminative information.

Liu et al. (2019) proposes a Generative Explanation Framework (GEF) for text classifications. The framework is designed to generate fine-grained explanations such as text justifications. During training, both the class labels and fine-grained explanations are provided for supervision, and the overall loss of GEF contains two major parts, classification loss and explanation generation loss. To make the generated explanations class-specific, “explanation factor” is designed in the model structure to associate explanations with classifications. The “explanation factor” is intuitively based on directly taking the explanations as input for classification and adding constraints on the classification softmax outputs. Specifically, “explanation factor” is formulated to minimize the pairwise discrepancy in softmax outputs for different input pairs, i) generated explanations and ground-truth explanations, and ii) generated explanations and original input text.

Unlike aforementioned methods which *generate* text explanations, Zellers et al. (2019) provides explanations in a *multichoice* fashion. They propose a visual reasoning task named Visual Commonsense Reasoning (VCR), which is to answer text questions based on given visual information (image), and provide reasons (explanations) accordingly. Both the answers and reasons are provided in a multichoice format. Due to the multichoice nature, reasonable explanations should be provided during testing, in contrast to other works which could generate explanations along with model decisions. Thus VCR is more suitable to be applied for prototype model debugging to audit model reasoning process, instead of real-life applications where explanations are usually lacking and remain to be generated.

Explanation Association. This type of joint training method associates input elements or latent features with human-understandable concepts or objects, or even directly with model explanations, which helps to provide model explanations intrinsically (Melis & Jaakkola, 2018; Iyer et al., 2018; Lei et al., 2016; Dong et al., 2017). Such methods usually achieve explanations by adding regularization term (Melis & Jaakkola, 2018; Lei et al., 2016; Dong et al., 2017) and/or revising model architecture (Melis & Jaakkola, 2018; Iyer et al., 2018; Lei et al., 2016). The explanations are provided in the form of i) associating input features or latent activations with semantic concepts (Melis & Jaakkola, 2018; Dong et al., 2017); ii) associating model prediction with a set of input elements (Lei et al., 2016); iii) associating explanations with object saliency maps in a computer vision task (Iyer et al., 2018). Regardless of the format of explanations and the technical details, methods belonging to this type commonly share the characteristics of associating hard-to-interpret elements to human-understandable atoms in an intrinsic joint training fashion.

Melis and Jaakkola (2018) proposes an intrinsic method which associates input features with semantically meaningful concepts and regards the coefficient as the importance of

such concepts during inference. A regularization based general framework for creating self-explaining neural networks (SENNs) is introduced. Given raw input, the network jointly learns to generate the class prediction and to generate explanations in terms of an input feature-to-concept mapping. The framework is based on the notion that linear regression models are explainable and generalizes the respective model definition to encompass complex classification functions, such as a DNN. A SENN consists of three components: i) A “concept encoder” that transforms the raw input into a set of explainable concepts. Essentially this encoder can be understood as a function that transforms low-level input into high-level meaningful structure, which predictions and explanations can be built upon. ii) An “input-dependent parametrizer”, which is a procedure to get the coefficient of explainable concepts, learns the relevance of the explainable concepts for the class predictions. The values of the relevance scores quantify the positive or negative contribution of the concept to the prediction. iii) Some “aggregation function” (e.g. a sum) that combines the output of the concept encoder and the parametrizer to produce a class prediction.

Iyer et al. (2018) introduces Object-sensitive Deep Reinforcement Learning (O-DRL), which is an explanation framework for reinforcement learning tasks that takes videos as input. O-DRL adds a pre-processing step (template matching) to recognize and locate specific objects in the input frame. For each detected object, an extra channel is added to the input frame’s RGB channels. Each object channel is a binary map that has the same height and width as the original input frame, 1’s encoding for the location of the detected object. The binary maps are later used to generate object saliency maps (as opposed to pixel saliency maps) that indicate the relevance of the object to action generation. It is argued that object saliency maps are more meaningful and explainable than pixel saliency maps since the objects encapsulate a higher-level visual concept.

Lei et al. (2016) integrates explainability in their neural networks for sentiment analysis by learning rationale extraction during the training phase in an unsupervised manner. Rationale extraction is done by allowing the network to learn to identify a small subset of words that all lead to the same class prediction as the entire text. They achieve this by adding mechanisms that use a combination of a generator and an encoder. The generator learns which text fragments could be candidate rationales and the encoder uses these candidates for prediction. Both the generator and the encoder are jointly trained during the optimization phase. The model explanation is provided by associating the model prediction with a set of critical input words.

Dong et al. (2017) focuses on providing intrinsic explanations for models on video captioning tasks. An interpretive loss function is defined to increase the visual fidelity of the learned features. This method is based on the nature of the used dataset, which contains rich human descriptions along with each video, and the rich text information can be leveraged to add constraint towards explainability. To produce an explanation, semantically meaningful concepts are first pre-extracted from human descriptions via Latent Dirichlet Allocation (LDA), which covers a variety of visual concepts such as objects, actions, relationships, etc. Based on the pre-extracted semantic topic, an interpretive loss is added to the original video captioning DNN model, for jointly training to generate video captions along with forcing the hidden neurons to be associated with semantic concepts.

Model Prototype. This type of intrinsic method is specifically for classification tasks, and is derived from a classical form of case-based reasoning (Kolodner, 1992) called prototype classification (Marchette & Socolinsky, 2003; Bien & Tibshirani, 2011; Kim et al., 2014). A prototype classifier generates classifications based on the similarity between the given input and each prototype observation in the dataset. In prototype classification applications, the word “prototype” is not limited to an observation in the dataset, but can be generalized to a combination of several observations or a latent representation learned in the feature space. To provide intrinsic explanations, the model architecture is designed to enable joint training the prototypes along with the original task. The model explainability is achieved by tracing the reasoning path for the given prediction back to each prototype learned by the model.

Li et al. (2018a) proposes an explainable prototype-based image classifier that can trace the model classification path to enable reasoning transparency. The model contains two major components; an autoencoder and a prototype classifier. The autoencoder, containing an encode and a decoder, is to transform raw input into a latent feature space, and the latent feature is later used by the prototype classifier for classification. The prototype classifier, one the other hand, is to generate classification via i) first calculating the distances in the latent space between a given input image and each prototype, ii) then passing through a fully-connected layer to compute the weighted sum of the distances, and iii) finally normalizing the weighted sums by the softmax layer to generate the classification result. Because the network learns *prototypes* during the training phase, each prediction always has an explanation that is faithful to what the network actually computes. Each prototype can be visualized by the decoder, and the reasoning path of the prototype classifier can be partially traced given the fully-connected layer weights and the comparison between input and each visualized prototype, providing intrinsic model explanations.



Chen et al. (2019) introduces an explainable DNN architecture called Prototypical Part Network (ProtoPNet) for image classification tasks. Similar to Li et al. (2018a), ProtoPNet also contains two components; a regular convolutional neural network and a prototype classifier. The regular convolutional neural network projects the raw image into hidden feature space, where prototypes are learned. The prototype classifier is to generate model predictions based on the weighted sum of each similarity score between an image patch and a learned prototype. Unlike Li et al. (2018a) where learned prototypes are corresponding to the entire image, the prototypes in (Chen et al., 2019) are more fine-grained and are latent representations of parts/patches of the image. To provide a model explanation, the latent representation of each prototype is associated with an image patch in the training set, shedding light on the reasoning clue of ProtoPNet.

4.4 Explanation Methods Lookup Table

Methods discussed in this field guide are categorized by distinct philosophies on eliciting and expressing an explanation from a DNN. This organization is ideal to understand the “classes” of methods that are being investigated in research and gradually implemented in practice. This does not, however, resolve an obvious question from a machine learning practitioner: *What is the “right” type of explanatory method I should use when building a model to solve my specific kind of problem?* It is difficult to match the methods with a particular situation because the type of explanation method suitable to that particular

situation is often dependent on many variables including the type of DNN architecture, data, problem, and desired form of explanation.

We propose Table 4 and Table 5 as a starting point in answering this question. All of the papers in Figure 4 are organized in Table 4 and Table 5. Each table is titled with the main category of explanation method. Both tables are organized into five columns. The first column indicates the subcategory of the explanation method and the second column displays the reference to the explanation paper. The following three columns contain summarized information taken directly from the explanation paper. The third and fourth columns contain one or more icons representing the type of data used and the type of problem(s) presented in the paper respectively. The meaning of the icons can be found at the bottom of the table. The final column displays information about the specific DNN model on which the explanation method has been used in the paper.

The practitioner can make use of Table 4 and Table 5 by considering what type of data, problem, and DNN architecture they are using in their specific situation. Then the practitioner can find an appropriate explanation method by matching the type of data, problem, and DNN architecture with the ones in the tables. For example, if a practitioner is using an image dataset to train a CNN on a classification problem, the practitioner can make use of all the explanation methods for which the  icon and the  icon and “CNN” are present in the respective rows. Note that in the “DNN Type” column we use “no specific requirements” to indicate that the DNN used in the respective paper does not need to meet any other specific requirements other than being a DNN. We use “model agnostic” to indicate that the type of model does not matter, i.e., the model does not have to be a DNN.

5. Designing Explanations for Users

The foundations of explaining DNNs discussed in this survey are seldom enough to achieve explanations useful to users in practice. ML engineers designing explainable DNNs in practice will thus often integrate an explanatory method into their DNN and then refine the presentation of the explanation to a form useful for the end-user. A *useful* explanation must conform to some definition of what constitutes a satisfactory explanation of the network’s inner workings depending on the user, the conditions of use, and the task at hand. These definitions are often qualitative (e.g., one user is better swayed by visual over textual explanations for a task). User requirements for an explanation may further vary by preferences between explanations that are of high fidelity versus those that are parsimonious. That the quality of an explanation thus depends on user-and context-specific utility makes their evaluation a difficult problem.

This suggests that explanations, grounded in the methods discussed in this field guide, need to be designed by engineers on a case-by-case basis for the user and task at hand. This section describes the following important design questions when engineers apply the methods in this field guide in practice:

1. **Who is the end user?** The kind of end-user, and in particular their expertise in deep learning and their domain-specific requirements, define the appropriate trade-off between fidelity and parsimony in an explanation’s presentation.

a) Visualization Methods

	Explanation Paper	Data Type	Problem Type	DNN Type
Back-Propagation	Erhan et al. (2009)		©	classifier has to be differentiable
	Zeiler et al. (2011)		©	CNN with max-pooling + relu
	Zeiler and Fergus (2014)		©	CNN with max-pooling + relu
	Selvaraju et al. (2017)		©	CNN
	Zhou et al. (2016)		©	CNN with global average pooling + softmax output
	Bach et al. (2015)		©	multilayer network
	Lapuschkin et al. (2016)		©	CNN
	Arras et al. (2016)		©	CNN
	Arras et al. (2017)		©	CNN
	Ding et al. (2017)			attention-based encoder decoder
	Montavon et al. (2017)	agnostic	©	no specific requirements
	Shrikumar et al. (2017)		©	CNN
	Sundararajan et al. (2017)		©	no specific requirements
	Sundararajan et al. (2016)		©	no specific requirements
Perturbation	Zeiler and Fergus (2014)		©	CNN
	Li et al. (2016)		©	no specific requirements
	Fong and Vedaldi (2017)		©	model agnostic
	Zintgraf et al. (2017)		©	CNN
	Robnik-Šikonja and Kononenko (2008)		©	models has to output probabilities
	Dabkowski and Gal (2017)		©	classifier has to be differentiable

b) Model Distillation

Loc. Appr.	Ribeiro et al. (2016c)		©	model agnostic
	Ribeiro et al. (2016b)		©	model agnostic
	Ribeiro et al. (2018)		©	model agnostic
	Elenberg et al. (2017)		©	model agnostic
	Baehrens et al. (2010)		©	model agnostic
Model Translation	Hou and Zhou (2020)		©	RNN
	Murdoch and Szlam (2017)		©	LSTM
	Harradon et al. (2018)		©	CNN
	Frosst and Hinton (2017)		©	CNN
	Zhang et al. (2019a)		©	CNN
	Tan et al. (2018)		©	no specific requirements
	Zhang et al. (2017)		©	CNN
	Zhang et al. (2018)		©	CNN, GANs

Data Types	Problem Types
image	© classification
text	localization
molecular graph	visual question answering
DNA sequence	regression
embedding	language translation
categorical data	sequence tagging
tabular data	structured prediction
	text generation
	question answering
	captioning

Table 4: Lookup table for the (a) visualization and (b) model distillation methods.

Intrinsic Methods				
	Explanation Paper	Data Type	Problem Type	DNN Type
Attention Mechanisms	Vaswani et al. (2017)			transformer
	Devlin et al. (2019)			transformer
	Bahdanau et al. (2014)			RNN encoder-decoder
	Luong et al. (2015)			stacking LSTM
	Wang et al. (2016)			attention-based LSTM
	Letarte et al. (2018)			self-attention network
	He et al. (2018)			attention-based LSTM
	Teney et al. (2018)			CNN + GRU combination
	Mascharka et al. (2018)			various specialized modules
	Xie et al. (2019)			various specialized modules
	Park et al. (2016)			various specialized modules
	Vinyals et al. (2015)			CNN + LSTM combination
	Xu et al. (2015)			CNN + RNN combination
	Antol et al. (2015)			CNN + MLP, CNN + LSTM combinations
	Goyal et al. (2017)			CNN + LSTM combination
	Anderson et al. (2018)			region proposal network + resnet combo, LSTM
Joint Training	Camburu et al. (2018)			LSTM
	Hind et al. (2019)			model agnostic
	Hendricks et al. (2016)			CNN
	Zellers et al. (2019)			recognition to cognition network
	Liu et al. (2019)			encoder-predictor
	Park et al. (2018)			pointing and justification model
	Kim et al. (2018b)			CNN
	Lei et al. (2016)			encoder-generator
	Melis and Jaakkola (2018)			self-explaining neural network
	Iyer et al. (2018)			deep q-network
	Dong et al. (2017)			attentive encoder-decoder
	Li et al. (2018a)			autoencoder + prototype layer combination
	Chen et al. (2019)			prototypical part network

Data Types		Problem Types	
	image		classification
	text		visual question answering
	embedding		language translation
	categorical data		captioning
	tabular data		sentiment analysis
	video		visual entailment
			visual commonsense reasoning
			language understanding
			reinforcement learning
			control planning

Table 5: Lookup table for the intrinsic methods.

2. **How practically impactful are the decisions of the DNN?** Here impact corresponds to the consequence of right and wrong decisions on people and society. Time-critical scenarios require explanations that can be rapidly generated and processed by a user should there be a need to intervene (e.g., in self-driving cars). Decision-critical scenarios require explanations that are *trustworthy*, that is, an explanation that a user trusts to be faithful to the actual decision-making process of the DNN.
3. **How extendable is an explanation?** It is expensive to design a form of explanation for only a single type of user who faces a single type of problem. A good design should thus be grounded on a single user’s preferences that can be applied to multiple types of problems, or be flexible enough to appeal to multiple user types examining the same problem type. It may not be feasible to devise the presentation of an explanation that appeals to a broad set of users tailored to a diverse set of problems.

5.1 Understanding the End User

One of the primary tasks to design an explanation is to determine the type of end-user using the system. The literature has documented cases of designs that provide both low-level technical specific explanations targeting on deep learning experts (Zeiler & Fergus, 2014; Sundararajan et al., 2017; Anderson et al., 2018; Li et al., 2016; Fong & Vedaldi, 2017; Zintgraf et al., 2017), and high-level reasoning extracted explanations catering normal users (Harradon et al., 2018; Zhang et al., 2019a, 2017, 2018). DNN experts care mostly about technical details and potential hints for model revising and performance improvement. Ideal explanations for them could be in form of input feature influence analytics (Adler et al., 2018; Koh & Liang, 2017; Li et al., 2016; Fong & Vedaldi, 2017), hidden states interaction and visualizations (Anderson et al., 2018; Vaswani et al., 2017; Vinyals et al., 2015; Zeiler & Fergus, 2014; Selvaraju et al., 2017; Bach et al., 2015; Sundararajan et al., 2017), etc. DNN experts, for instance, could check if the model is emphasizing reasonable image areas (Zeiler & Fergus, 2014) or text elements/words (Vaswani et al., 2017; Sundararajan et al., 2017) towards generating corresponding model decisions, and propose model revision strategies accordingly. Normal users, on the other hand, mainly focus on the high-level functionality of the model, instead of technical details. Their main concern is if the model is working reasonably and not violating human logic. The explanation can be represented in the form of extracted reasoning logic (Harradon et al., 2018; Zhang et al., 2019a, 2017, 2018) or some easy to understandable input clues with respect to given prediction (Ribeiro et al., 2016c). If the decision is generated on unexpected input elements or not following a logical reasoning process, a doubt could be raised by the user to deny such a model decision. Considering the different user expertise level on DNN knowledge, the designing of a general model explanation system, which satisfies both DNN experts and normal users, is challenging and remains to be explored.

The domain a user operates in is another important consideration. For example, the explanation needs of a medical doctor require that the explanation representation be detailed enough such that the doctor can understand the reasoning process behind the specific diagnosis and be confident about said diagnosis (Lipton, 2017), e.g., the patient needs this specific treatment because it identifies features of cancer at a particular stage. But no explainable method is able to automatically tailor its explanations to end-users for a spe-

cific domain. One possible way to obtain such explanations using the present art is if the features of the input data that are expressed by an explanation method have an intuitive domain-specific interpretation, which is built upon a systematic knowledge base constructed by domain experts.

5.2 The Impact of DNN Decisions

The need for and characteristics of an explanation depends on the impact of a DNN’s operation on human life and society. This impact can be realized based on the impact and speed of a decision. In *time-critical* scenarios (Grigorescu et al., 2019) where users must process and react to DNN decisions in limited time, explanations must be produced that are simple to interpret and understand and are not computationally intense to perform. This is a particularly important aspect of explanations that is seldom investigated in the literature. For example, a DNN providing recommendations during a military operation, or sensing upcoming hazards on a vehicle, needs to support their output with explanations while giving the user enough time to process and react accordingly. In a *decision-critical* scenario (Grigorescu et al., 2019; Nemati et al., 2018; Ahmad et al., 2018), the ability to not only interpret but deeply inspect a decision grows in importance. Any user decision based on a DNN’s recommendation should be supported with evidence and explanations others can understand. At the same time, should a DNN’s recommendation turn out to be incorrect or lead to an undesirable outcome for the user, the model should be inspected post-hoc to hypothesize root causes and identify “bugs” in the DNN’s actions. Deep, technical inspections of the neural network guided by comprehensive interpretations of its inference and training actions are necessary for such post-hoc analysis.

Few current model explanations are designed with time- and decision-critical scenarios in mind. The computational cost of many model explanations tends to be high and may require extra human labor, which is undesirable if an automatic instant explanation is needed. For instance, for explanations presented in form of model visualization (Zeiler & Fergus, 2014; Selvaraju et al., 2017; Shrikumar et al., 2017; Sundararajan et al., 2017; Montavon et al., 2017; Zhou et al., 2016), extra human effort is needed for verification, which is potentially costly. Besides, some explanation methods with post-hoc training involved (Ribeiro et al., 2016c; Frosst & Hinton, 2017; Krakovna & Doshi-Velez, 2016; Hou & Zhou, 2018) may be limited in its utility on providing explanations for real-time input. The study for decision-critical scenarios is still under development. In order to increase the fidelity and reliability of model decisions, a variety of topics are explored besides model explanations, including model robustness (Papernot et al., 2016b; Meng & Chen, 2017; Xie et al., 2017; Samangouei et al., 2018; Li et al., 2018b), fairness and bias (Heidari et al., 2018; Calders et al., 2009; Hardt et al., 2016; Zafar et al., 2017a; Calmon et al., 2017; Gordaliza et al., 2019; Agarwal et al., 2018; Menon & Williamson, 2018; Donini et al., 2018; Dwork et al., 2018; Pleiss et al., 2017; Beutel et al., 2017), model trustworthiness (Jiang et al., 2018; Heo et al., 2018), etc. The study of the aforementioned topics, together with model explanations, may jointly shed light on potential new solutions for applications on decision-critical scenarios.

5.3 Design Extendability

Modularity and reusability are important extendability traits in the architecture of large-scale software systems: modularity promotes the ability of an engineer to replace and alter system components as necessary, while reusability promotes the use of already proven software modules. In a similar vein, highly reliable and performant DNN systems should also be constructed with reusable and highly modular components. Modularity is a trait of the functional units of a DNN that may be adaptable for multiple architectures. For example, the form of an attention mechanism suitable for any kind of sequential data processing (Vaswani et al., 2017; Devlin et al., 2019). A highly modular DNN architecture may be one that contains many “plug and play” components in each layer so that its complete design can be seen as a composition of interconnected functional units. Reusability speaks to complete DNN systems, perhaps already trained, that can be reused in multiple problem domains. One example of reusability is the common application of a pre-trained YOLO (Redmon et al., 2016) model for object localization in frames in a deep learning video processing pipeline.

DNN explanation methods that also exhibit these extendability traits are more likely to be broadly useful over a variety of DNN models and application domains. Modularized explainable models will crucially reduce the overhead in implementing and deploying explainability in new domains and may lead to explanatory forms a user is familiar with over multiple types of models. Reusability plays a role in risk control, such that the fidelity of an explanation remains consistent however the explanatory method is applied.

Neither modularity nor reusability is the focus of explainable methods in the literature. However, existing methods could be divided by how modular they potentially are. Model-agnostic methods (Ribeiro et al., 2016c, 2016b, 2016a; Fong & Vedaldi, 2017; Jha et al., 2017), which do not take the type of model into account, are modular by definition in the sense that the explanatory module is independent of the model it is producing explanations for. On the other hand, The second category contains explanation methods that are very specific to the model (Shrikumar et al., 2017; Zeiler & Fergus, 2014; Bach et al., 2015; Montavon et al., 2017; Zhou et al., 2016; Murdoch et al., 2018; Xie et al., 2017; Park et al., 2018; Hendricks et al., 2016). This aspect is important for expert users that are developing deep learning models and need to understand specifically which aspect of the deep learning model is influencing the predictions, e.g. in model debugging. However, these methods by their very nature lack modularity.

6. Future Directions

The field guide concludes by introducing research directions whose developments can contribute to improving explainable deep learning.

Developing a Systematic Theory of Model Explanation. Currently, there is still a lack of systematic general theory in the realm of DNN explanation (Arrieta et al., 2019; Díez et al., 2013). A systematic theory should be able to benefit the overall model explanation studies, and once formed, some current challenging explainable problems may be able to be handled properly by nature, and some novel directions may be proposed based on the systematic theories.

User-friendly Explanations. User-friendly explanations are needed to minimize the technical understanding of a user to correctly interpret explanations. As the concern of the opaque nature of DNNs is raising increasing attention in the society and even required by law, model explanations would inevitably be mandatory in a wide range of real-life applications (Goodman & Flaxman, 2017). Given the varied backgrounds of model users, the friendliness would be a future trend towards constructing explanations of high qualities. Most explainable methods are still catering towards expert users instead of laymen (Ras et al., 2018), in the sense that knowledge about the method, for instance the DNN process, is needed to understand the explanation. The requirement on model knowledge limits the wide usage of such explanation models, since in real scenarios the chance of the end-users being machine learning experts is very low. Assuming the end-user has been correctly determined, the next step is to determine *what* needs to be explained, i.e., which step or decision does the system need to explain?

Producing Explanations Efficiently. Time and decision-critical explanations (Grigorescu et al., 2019; Ahmad et al., 2018; Nemati et al., 2018), as discussed in Section 5.2, must be produced with enough time for a user to react to a DNN’s decision. An efficient manner to produce explanations further saves computational power, which is favorable in industrial applications or when explanations are required in environments with low computing resources.

Methods for DNN Model Debugging. To the best of our knowledge, the current literature on DNN model debugging is very limited. This is arguable, after the development of a systematic theory, among the most important directions that can support explainable deep learning. By creating good model debugging tools, DNN engineers will have essentially produced explainable deep learning techniques for technical experts able to reflect the influences of model errors and behaviors (Hall, 2019). This very low-level understanding of exactly how the errors are happening can form a basis to develop more intuitively interpretable explanations a layman can understand.

Developing Methods for Trustworthiness. The vulnerability of a DNN to adversarial examples (Yuan et al., 2019; Goodfellow et al., 2014; Carlini & Wagner, 2017; Madry et al., 2017) and poisoned training sets (Saha et al., 2019) raises much concern on trustworthiness. As more and more DNNs are leveraged in real-life applications, the demand for model trustworthiness would undoubtedly increase, especially for decision-critical scenarios where undesired decisions may cost severe consequences. This thread of research is only beginning to be developed (Jiang et al., 2018; Heo et al., 2018).

Evaluating Explanations. Compared to the booming of a variety of methods on explainable deep learning, the research progress on evaluations of such methods seems to slightly fall behind. Most current evaluation methods are mainly for explanations on general machine learning (Doshi-Velez & Kim, 2017; Narayanan et al., 2018; Mohseni et al., 2018; Yang et al., 2019; Cui et al., 2019), while a few is on deep learning (Samek et al., 2017a; Nie et al., 2018; Adebayo et al., 2018). One main challenge for explainable evaluations is the lack of ground truth for most cases. Besides, the favorable evaluation metric may vary a lot according to the specific evaluation goal and oriented user groups. Based on potential answers to the question “what makes a *good interpretation*? ”, different criteria are proposed as the evaluation goal. *Human-friendly* or *human-centered* (Narayanan et al., 2018) is one criterion, that the interpretations should be organized in a way that is easier for human

understanding. *Complexity* (Cui et al., 2019) measures the complexity of the explanations. The more complex explanations are, the less human-friendly it tends to be. *Accuracy*, or *correctness*, *fidelity* (Mohseni et al., 2018; Cui et al., 2019; Yang et al., 2019), requires the interpretations provided should accurately depict the internal decision making process. *Coverage*, or *completeness* (Cui et al., 2019) demands to provide complete information on the model interpretations such that it is reproducible. Besides the above criteria, Yang et al. (2019) also defines *generalizability* and *persuasibility*, where the former measures the generalization power of explanations and the latter is about how well human comprehend the explanations, which is similar to the human-friendly criterion. The evaluations on interpretation methods usually follow some of the aforementioned criteria as guidelines. Currently, many evaluations on DNN explanations, especially explanations through visualizations, are mainly focusing on the accuracy/correctness of the explanations (Nie et al., 2018; Adebayo et al., 2018).

7. Conclusions

The rapid advancements in deep neural networks have stimulated innovations in a wide range of applications such as facial recognition (Masi et al., 2018), voice assistance (Tulshan & Dhage, 2018), driving system (Jain et al., 2015), etc. The field of deep learning explainability is motivated by the opacity of DNN systems society increasingly relies on. Government policies, for instance the EU’s General Data Protection Regulation (GDPR) (Goodman & Flaxman, 2017), alludes to a future where the explainability aspects of deep networks will become a legal concern.

We hope this field guide has distilled the important topics, related work, methods, and concerns associated with explainable deep learning for an initiate. The necessary external traits of DNN explanations reveal the demand for this research topic in the community. Topics closely associated with DNN explainability, including model learning mechanism, model debugging, adversarial attack and defense, and model fairness and bias, are reviewed as related work. A wide range of existing methods on deep learning explainability is introduced and organized by a novel categorization scheme, depicting the field in a clear and straightforward way. A discussion on user-oriented explanation designing and future trends of this field is provided at the end of this survey, shedding light on potential directions on model explainability. Given the countless papers in this field and the rapid development of explainable methods, we admit that we are unable to cover every paper or every aspect that belongs to this realm. For the papers covered, we carefully designed hierarchical categories, such that the skeleton of this field is visualized.

In the end, the important thing is to explain the right thing to the right person in the right way at the right time.² We are excited to continue to observe how the field evolves to deliver the right explanation, to the right audience who need it the most. We hope the numerous solutions being actively explored will lead to the fairer, safer, and more confident use of deep learning across society.

Acknowledgments

2. Paraphrased from dr. Silja Renooij at BENELEARN2019

We would like to acknowledge support for this project from the Ohio Federal Research Network, the Multidisciplinary Research Program of the Department of Defense (MURI N00014-00-1-0637), and the organizers and participants of the Schloss Dagstuhl – Leibniz Center for Informatics Seminar 17192 on Human-Like Neural-Symbolic Computing for providing the environment to develop the ideas in this paper. Parts of this work was completed under a Fulbright-NSF Fellowship for Cyber Security and Critical Infrastructure. We would also like to thank Erdi Çallı for the helpful discussions and general support.

References

- Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., & Süsstrunk, S. (2012). SLIC superpixels compared to state-of-the-art superpixel methods. *IEEE transactions on pattern analysis and machine intelligence*, 34(11), 2274–2282.
- Adadi, A., & Berrada, M. (2018). Peeking inside the black-box: A survey on explainable artificial intelligence (XAI). *IEEE Access*, 6, 52138–52160.
- Adebayo, J., Gilmer, J., Muelly, M., Goodfellow, I., Hardt, M., & Kim, B. (2018). Sanity checks for saliency maps. In *Advances in Neural Information Processing Systems*, pp. 9525–9536.
- Adebayo, J., & Kagal, L. (2016). Iterative orthogonal feature projection for diagnosing bias in black-box models. *arXiv preprint arXiv:1611.04967*.
- Adler, P., Falk, C., Friedler, S. A., Nix, T., Rybeck, G., Scheidegger, C., Smith, B., & Venkatasubramanian, S. (2018). Auditing black-box models for indirect influence. *Knowledge and Information Systems*, 54(1), 95–122.
- Agarwal, A., Beygelzimer, A., Dudik, M., Langford, J., & Wallach, H. (2018). A reductions approach to fair classification. In *International Conference on Machine Learning*, pp. 60–69.
- Ahmad, M. A., Eckert, C., & Teredesai, A. (2018). Interpretable machine learning in health-care. In *Proceedings of the 2018 ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics*, pp. 559–560.
- Alain, G., & Bengio, Y. (2016). Understanding intermediate layers using linear classifier probes. *arXiv preprint arXiv:1610.01644*.
- Amershi, S., Chickering, M., Drucker, S. M., Lee, B., Simard, P., & Suh, J. (2015). Model-tracker: Redesigning performance analysis tools for machine learning. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pp. 337–346. ACM.
- Amodei, D., Olah, C., Steinhardt, J., Christiano, P., Schulman, J., & Mané, D. (2016). Concrete problems in AI safety. *arXiv preprint arXiv:1606.06565*.
- Anderson, P., He, X., Buehler, C., Teney, D., Johnson, M., Gould, S., & Zhang, L. (2018). Bottom-up and top-down attention for image captioning and visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6077–6086.

- Antol, S., Agrawal, A., Lu, J., Mitchell, M., Batra, D., Lawrence Zitnick, C., & Parikh, D. (2015). VQA: Visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pp. 2425–2433.
- Arpit, D., Jastrzebski, S., Ballas, N., Krueger, D., Bengio, E., Kanwal, M. S., Maharaj, T., Fischer, A., Courville, A., Bengio, Y., et al. (2017). A closer look at memorization in deep networks. In *International Conference on Machine Learning*, pp. 233–242.
- Arras, L., Horn, F., Montavon, G., Müller, K.-R., & Samek, W. (2016). Explaining predictions of non-Linear classifiers in NLP. In *Proceedings of the 1st Workshop on Representation Learning for NLP*, pp. 1–7.
- Arras, L., Horn, F., Montavon, G., Müller, K.-R., & Samek, W. (2017). "What is relevant in a text document?": An interpretable machine learning approach. *PloS one*, 12(8), e0181142.
- Arrieta, A. B., Díaz-Rodríguez, N., Del Ser, J., Bennetot, A., Tabik, S., Barbado, A., García, S., Gil-López, S., Molina, D., Benjamins, R., et al. (2019). Explainable artificial intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion*.
- Bach, S., Binder, A., Montavon, G., Klauschen, F., Müller, K.-R., & Samek, W. (2015). On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, 10(7), e0130140.
- Baehrens, D., Schroeter, T., Harmeling, S., Kawanabe, M., Hansen, K., & Müller, K.-R. (2010). How to explain individual classification decisions. *Journal of Machine Learning Research*, 11(Jun), 1803–1831.
- Bahdanau, D., Cho, K., & Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.
- Bahri, Y., Kadmon, J., Pennington, J., Schoenholz, S. S., Sohl-Dickstein, J., & Ganguli, S. (2020). Statistical mechanics of deep learning. *Annual Review of Condensed Matter Physics*.
- Bastani, O., Kim, C., & Bastani, H. (2017). Interpreting blackbox models via model extraction. *arXiv preprint arXiv:1705.08504*.
- Baum, K., Köhl, M. A., & Schmidt, E. (2017). Two challenges for CI trustworthiness and how to address them. In *Proceedings of the 1st Workshop on Explainable Computational Intelligence (XCI 2017)*.
- Bechavod, Y., & Ligett, K. (2017). Penalizing unfairness in binary classification. *arXiv preprint arXiv:1707.00044*.
- Berk, R., Heidari, H., Jabbari, S., Joseph, M., Kearns, M., Morgenstern, J., Neel, S., & Roth, A. (2017). A convex framework for fair regression. *arXiv preprint arXiv:1706.02409*.
- Beutel, A., Chen, J., Zhao, Z., & Chi, E. H. (2017). Data decisions and theoretical implications when adversarially learning fair representations. *arXiv preprint arXiv:1707.00075*.
- Bien, J., & Tibshirani, R. (2011). Prototype selection for interpretable classification. *The Annals of Applied Statistics*, 2403–2424.

- Brendel, W., Rauber, J., & Bethge, M. (2017). Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. *arXiv preprint arXiv:1712.04248*.
- Calders, T., Kamiran, F., & Pechenizkiy, M. (2009). Building classifiers with independency constraints. In *2009 IEEE International Conference on Data Mining Workshops*, pp. 13–18. IEEE.
- Calmon, F., Wei, D., Vinzamuri, B., Ramamurthy, K. N., & Varshney, K. R. (2017). Optimized pre-processing for discrimination prevention. In *Advances in Neural Information Processing Systems*, pp. 3992–4001.
- Camburu, O.-M., Rocktäschel, T., Lukasiewicz, T., & Blunsom, P. (2018). e-SNLI: Natural language inference with natural language explanations. In *Advances in Neural Information Processing Systems*, pp. 9560–9572.
- Carlini, N., Katz, G., Barrett, C., & Dill, D. L. (2018). Ground-truth adversarial examples..
- Carlini, N., & Wagner, D. (2017). Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy (SP)*, pp. 39–57. IEEE.
- Carter, S., Armstrong, Z., Schubert, L., Johnson, I., & Olah, C. (2019). Activation atlas. *Distill*. <https://distill.pub/2019/activation-atlas>.
- Carvalho, D. V., Pereira, E. M., & Cardoso, J. S. (2019). Machine learning interpretability: A survey on methods and metrics. *Electronics*, 8(8), 832.
- Chen, C., Li, O., Tao, D., Barnett, A., Rudin, C., & Su, J. K. (2019). This looks like that: Deep learning for interpretable image recognition. In *Advances in Neural Information Processing Systems*, pp. 8928–8939.
- Chen, P.-Y., Zhang, H., Sharma, Y., Yi, J., & Hsieh, C.-J. (2017). Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pp. 15–26. ACM.
- Chong, E., Han, C., & Park, F. C. (2017). Deep learning networks for stock market analysis and prediction: Methodology, data representations, and case studies. *Expert Systems with Applications*, 83, 187–205.
- Craven, M., & Shavlik, J. W. (1996). Extracting tree-structured representations of trained networks. In *Advances in neural information processing systems*, pp. 24–30.
- Cui, X., Lee, J. M., & Hsieh, J. (2019). An integrative 3C evaluation framework for explainable artificial intelligence..
- Dabkowski, P., & Gal, Y. (2017). Real time image saliency for black box classifiers. In *Advances in Neural Information Processing Systems*, pp. 6967–6976.
- Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4171–4186.
- Díez, J., Khalifa, K., & Leuridan, B. (2013). General theories of explanation: Buyer beware. *Synthese*, 190(3), 379–396.

- Ding, Y., Liu, Y., Luan, H., & Sun, M. (2017). Visualizing and understanding neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vol. 1, pp. 1150–1159.
- Dong, W., Li, J., Yao, R., Li, C., Yuan, T., & Wang, L. (2016). Characterizing driving styles with deep learning. *arXiv preprint arXiv:1607.03611*.
- Dong, Y., Liao, F., Pang, T., Su, H., Zhu, J., Hu, X., & Li, J. (2018). Boosting adversarial attacks with momentum. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 9185–9193.
- Dong, Y., Su, H., Zhu, J., & Zhang, B. (2017). Improving interpretability of deep neural networks with semantic information. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4306–4314.
- Donini, M., Oneto, L., Ben-David, S., Shawe-Taylor, J. S., & Pontil, M. (2018). Empirical risk minimization under fairness constraints. In *Advances in Neural Information Processing Systems*, pp. 2791–2801.
- Doshi-Velez, F., & Kim, B. (2017). Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*.
- Došilović, F. K., Brčić, M., & Hlupić, N. (2018). Explainable artificial intelligence: A survey. In *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pp. 0210–0215. IEEE.
- Dwork, C., Hardt, M., Pitassi, T., Reingold, O., & Zemel, R. (2012). Fairness through awareness. In *Proceedings of the 3rd innovations in theoretical computer science conference*, pp. 214–226. ACM.
- Dwork, C., Immorlica, N., Kalai, A. T., & Leiserson, M. (2018). Decoupled classifiers for group-fair and efficient machine learning. In *Conference on Fairness, Accountability and Transparency*, pp. 119–133.
- Elenberg, E., Dimakis, A. G., Feldman, M., & Karbasi, A. (2017). Streaming weak submodularity: Interpreting neural networks on the fly. In *Advances in Neural Information Processing Systems*, pp. 4044–4054.
- Erhan, D., Bengio, Y., Courville, A., & Vincent, P. (2009). Visualizing higher-layer features of a deep network. *Département d’Informatique et Recherche Opérationnelle, University of Montreal, QC, Canada, Tech. Rep, 1341*.
- Erhan, D., Courville, A., & Bengio, Y. (2010). Understanding representations learned in deep architectures. *Département d’Informatique et Recherche Opérationnelle, University of Montreal, QC, Canada, Tech. Rep, 1355*.
- Eykholt, K., Evtimov, I., Fernandes, E., Li, B., Rahmati, A., Xiao, C., Prakash, A., Kohno, T., & Song, D. (2018). Robust physical-world attacks on deep learning visual classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1625–1634.
- Feldman, M., Friedler, S. A., Moeller, J., Scheidegger, C., & Venkatasubramanian, S. (2015). Certifying and removing disparate impact. In *Proceedings of the 21th ACM SIGKDD*

- International Conference on Knowledge Discovery and Data Mining*, pp. 259–268. ACM.
- Fong, R. C., & Vedaldi, A. (2017). Interpretable explanations of black boxes by meaningful perturbation. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3429–3437.
- Frosst, N., & Hinton, G. (2017). Distilling a neural network into a soft decision tree. *arXiv preprint arXiv:1711.09784*.
- Fuchs, F. B., Groth, O., Kosoriek, A. R., Bewley, A., Wulfmeier, M., Vedaldi, A., & Posner, I. (2018). Neural stethoscopes: Unifying analytic, auxiliary and adversarial network probing. *arXiv preprint arXiv:1806.05502*.
- Garcez, A. S. d., Broda, K. B., & Gabbay, D. M. (2012). *Neural-symbolic learning systems: Foundations and applications*. Springer Science & Business Media.
- Georgiev, P., Bhattacharya, S., Lane, N. D., & Mascolo, C. (2017). Low-resource multi-task audio sensing for mobile and embedded devices via shared deep neural network representations. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 1(3), 50.
- Gilpin, L. H., Bau, D., Yuan, B. Z., Bajwa, A., Specter, M., & Kagal, L. (2018). Explaining explanations: An overview of interpretability of machine learning. In *2018 IEEE 5th International Conference on data science and advanced analytics (DSAA)*, pp. 80–89. IEEE.
- Gonzalez-Garcia, A., Modolo, D., & Ferrari, V. (2018). Do semantic parts emerge in convolutional neural networks?. *International Journal of Computer Vision*, 126(5), 476–494.
- Goodfellow, I. J., Shlens, J., & Szegedy, C. (2014). Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.
- Goodman, B., & Flaxman, S. (2017). European Union regulations on algorithmic decision-making and a "right to explanation". *AI magazine*, 38(3), 50–57.
- Gordaliza, P., Del Barrio, E., Fabrice, G., & Loubes, J.-M. (2019). Obtaining fairness using optimal transport theory. In *International Conference on Machine Learning*, pp. 2357–2365.
- Goswami, G., Bhardwaj, R., Singh, R., & Vatsa, M. (2014). MDLFace: Memorability augmented deep learning for video face recognition. In *IEEE International Joint Conference on Biometrics*, pp. 1–7. IEEE.
- Goyal, Y., Khot, T., Summers-Stay, D., Batra, D., & Parikh, D. (2017). Making the V in VQA matter: Elevating the role of image understanding in visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6904–6913.
- Graves, A., Wayne, G., & Danihelka, I. (2014). Neural turing machines. *arXiv preprint arXiv:1410.5401*.
- Grigorescu, S., Trasnea, B., Cocias, T., & Macesanu, G. (2019). A survey of deep learning techniques for autonomous driving. *Journal of Field Robotics*.

- Groth, O., Fuchs, F. B., Posner, I., & Vedaldi, A. (2018). Shapestacks: Learning vision-based physical intuition for generalised object stacking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 702–717.
- Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Giannotti, F., & Pedreschi, D. (2018). A survey of methods for explaining black box models. *ACM Computing Surveys (CSUR)*, 51(5), 93.
- Hall, P. (2019). Guidelines for responsible and human-centered use of explainable machine learning. *arXiv preprint arXiv:1906.03533*.
- Hardt, M., Price, E., Srebro, N., et al. (2016). Equality of opportunity in supervised learning. In *Advances in neural information processing systems*, pp. 3315–3323.
- Harradon, M., Druce, J., & Ruttenberg, B. (2018). Causal learning and explanation of deep neural networks via autoencoded activations. *arXiv preprint arXiv:1802.00541*.
- He, R., Lee, W. S., Ng, H. T., & Dahlmeier, D. (2018). Effective attention modeling for aspect-level sentiment classification. In *Proceedings of the 27th International Conference on Computational Linguistics*, pp. 1121–1131.
- Heidari, H., Ferrari, C., Gummadi, K., & Krause, A. (2018). Fairness behind a veil of ignorance: A welfare analysis for automated decision making. In *Advances in Neural Information Processing Systems*, pp. 1265–1276.
- Hendricks, L. A., Akata, Z., Rohrbach, M., Donahue, J., Schiele, B., & Darrell, T. (2016). Generating visual explanations. In *European Conference on Computer Vision*, pp. 3–19. Springer.
- Heo, J., Lee, H. B., Kim, S., Lee, J., Kim, K. J., Yang, E., & Hwang, S. J. (2018). Uncertainty-aware attention for reliable interpretation and prediction. In *Advances in Neural Information Processing Systems*, pp. 909–918.
- Hind, M., Wei, D., Campbell, M., Codella, N. C., Dhurandhar, A., Mojsilović, A., Nate-san Ramamurthy, K., & Varshney, K. R. (2019). TED: Teaching AI to explain its decisions. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, pp. 123–129. ACM.
- Hinton, G., Vinyals, O., & Dean, J. (2015). Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.
- Hooker, G. (2004). Discovering additive structure in black box functions. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 575–580.
- Hooker, G. (2007). Generalized functional anova diagnostics for high-dimensional functions of dependent variables. *Journal of Computational and Graphical Statistics*, 16(3), 709–732.
- Hoos, H., & Leyton-Brown, K. (2014). An efficient approach for assessing hyperparameter importance. In *International Conference on Machine Learning*, pp. 754–762.
- Hou, B.-J., & Zhou, Z.-H. (2018). Learning with interpretable structure from RNN. *arXiv preprint arXiv:1810.10708*.

- Hou, B.-J., & Zhou, Z.-H. (2020). Learning With interpretable structure from gated RNN. *IEEE Transactions on Neural Networks and Learning Systems*.
- Hudson, D. A., & Manning, C. D. (2018). Compositional attention networks for machine reasoning. *arXiv preprint arXiv:1803.03067*.
- Iyer, R., Li, Y., Li, H., Lewis, M., Sundar, R., & Sycara, K. (2018). Transparency and explanation in deep reinforcement learning neural networks. In *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, pp. 144–150. ACM.
- Jain, A., Koppula, H. S., Raghavan, B., Soh, S., & Saxena, A. (2015). Car that knows before you do: Anticipating maneuvers via learning temporal driving models. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3182–3190.
- Jain, S., & Wallace, B. C. (2019). Attention is not explanation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 3543–3556.
- Jha, S., Raman, V., Pinto, A., Sahai, T., & Francis, M. (2017). On learning sparse boolean formulae for explaining AI decisions. In *NASA Formal Methods Symposium*, pp. 99–114. Springer.
- Jiang, H., Kim, B., Guan, M., & Gupta, M. (2018). To trust or not to trust a classifier. In *Advances in neural information processing systems*, pp. 5541–5552.
- Johnson, J., Hariharan, B., van der Maaten, L., Fei-Fei, L., Lawrence Zitnick, C., & Girshick, R. (2017). CLEVR: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2901–2910.
- Kamiran, F., & Calders, T. (2010). Classification with no discrimination by preferential sampling. In *Proc. 19th Machine Learning Conf. Belgium and The Netherlands*, pp. 1–6. Citeseer.
- Kamiran, F., & Calders, T. (2012). Data preprocessing techniques for classification without discrimination. *Knowledge and Information Systems*, 33(1), 1–33.
- Kamishima, T., Akaho, S., & Sakuma, J. (2011). Fairness-aware learning through regularization approach. In *2011 IEEE 11th International Conference on Data Mining Workshops*, pp. 643–650. IEEE.
- Kang, D., Raghavan, D., Bailis, P., & Zaharia, M. (2018). Model assertions for debugging machine learning. In *NeurIPS MLSys Workshop*.
- Kearns, M., Neel, S., Roth, A., & Wu, Z. S. (2018). Preventing fairness gerrymandering: Auditing and learning for subgroup fairness. In *International Conference on Machine Learning*, pp. 2569–2577.
- Kim, B., Rudin, C., & Shah, J. A. (2014). The Bayesian case model: A generative approach for case-based reasoning and prototype classification. In *Advances in Neural Information Processing Systems*, pp. 1952–1960.

- Kim, B., Wattenberg, M., Gilmer, J., Cai, C., Wexler, J., Viegas, F., et al. (2018a). Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (TCAV). In *International Conference on Machine Learning*, pp. 2673–2682.
- Kim, J., Rohrbach, A., Darrell, T., Canny, J., & Akata, Z. (2018b). Textual explanations for self-driving vehicles. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 563–578.
- Kindermans, P.-J., Hooker, S., Adebayo, J., Alber, M., Schütt, K. T., Dähne, S., Erhan, D., & Kim, B. (2019). The (un)reliability of saliency methods. In *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, pp. 267–280. Springer.
- Koh, P. W., & Liang, P. (2017). Understanding black-box predictions via influence functions. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 1885–1894. JMLR. org.
- Kolodner, J. L. (1992). An introduction to case-based reasoning. *Artificial intelligence review*, 6(1), 3–34.
- Krakovna, V., & Doshi-Velez, F. (2016). Increasing the interpretability of recurrent neural networks using hidden Markov models. *arXiv preprint arXiv:1606.05320*.
- Kurakin, A., Goodfellow, I., & Bengio, S. (2016). Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*.
- Lapuschkin, S., Binder, A., Montavon, G., Muller, K.-R., & Samek, W. (2016). Analyzing classifiers: Fisher vectors and deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2912–2920.
- LeCun, Y., Bottou, L., Bengio, Y., Haffner, P., et al. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11), 2278–2324.
- LeCun, Y., Jackel, L., Bottou, L., Brunot, A., Cortes, C., Denker, J., Drucker, H., Guyon, I., Muller, U., Sackinger, E., et al. (1995). Comparison of learning algorithms for handwritten digit recognition. In *International conference on artificial neural networks*, Vol. 60, pp. 53–60. Perth, Australia.
- LeCun, Y., Weinberger, K., Simard, P., & Caruana, R. (2017). Debate: Interpretability is necessary in machine learning. <https://www.youtube.com/watch?v=2hW05ZfsUUo>.
- Lee, J. D., & See, K. A. (2004). Trust in automation: Designing for appropriate reliance. *Human factors*, 46(1), 50–80.
- Lei, T., Barzilay, R., & Jaakkola, T. (2016). Rationalizing neural predictions. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 107–117.
- Letarte, G., Paradis, F., Giguère, P., & Laviolette, F. (2018). Importance of self-attention for sentiment analysis. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pp. 267–275.
- Li, J., Monroe, W., & Jurafsky, D. (2016). Understanding neural networks through representation erasure. *arXiv preprint arXiv:1612.08220*.

- Li, O., Liu, H., Chen, C., & Rudin, C. (2018a). Deep learning for case-based reasoning through prototypes: A neural network that explains its predictions. In *Thirty-Second AAAI Conference on Artificial Intelligence*.
- Li, Y., Min, M. R., Yu, W., Hsieh, C.-J., Lee, T., & Kruus, E. (2018b). Optimal transport classifier: Defending against adversarial attacks by regularized deep embedding. *arXiv preprint arXiv:1811.07950*.
- Lin, M., Chen, Q., & Yan, S. (2013). Network in network. *arXiv preprint arXiv:1312.4400*.
- Lipton, Z. C. (2017). The doctor just won't accept that!. *arXiv preprint arXiv:1711.08037*.
- Lipton, Z. C. (2018). The mythos of model interpretability. *Communications of the ACM*, 61(10), 36–43.
- Liu, H., Yin, Q., & Wang, W. Y. (2019). Towards explainable NLP: A generative explanation framework for text classification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 5570–5581.
- Liu, S., Wang, X., Liu, M., & Zhu, J. (2017). Towards better analysis of machine learning models: A visual analytics perspective. *Visual Informatics*, 1(1), 48–56.
- Liu, X., Li, Y., Wu, C., & Hsieh, C.-J. (2018). Adv-bnn: Improved adversarial defense through robust bayesian neural network. *arXiv preprint arXiv:1810.01279*.
- Louizos, C., Swersky, K., Li, Y., Welling, M., & Zemel, R. (2015). The variational fair autoencoder. *arXiv preprint arXiv:1511.00830*.
- Lundén, J., & Koivunen, V. (2016). Deep learning for HRRP-based target recognition in multistatic radar systems. In *IEEE Radar Conference*, pp. 1–6. IEEE.
- Luong, M.-T., Pham, H., & Manning, C. D. (2015). Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*.
- Madry, A., Makelov, A., Schmidt, L., Tsipras, D., & Vladu, A. (2017). Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*.
- Marchette, C. E. P. D. J., & Socolinsky, J. G. D. D. A. (2003). Classification using class cover catch digraphs. *Journal of Classification*, 20, 3–23.
- Mascharka, D., Tran, P., Soklaski, R., & Majumdar, A. (2018). Transparency by design: Closing the gap between performance and interpretability in visual reasoning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4942–4950.
- Masi, I., Wu, Y., Hassner, T., & Natarajan, P. (2018). Deep face recognition: A survey. In *2018 31st SIBGRAPI conference on graphics, patterns and images (SIBGRAPI)*, pp. 471–478. IEEE.
- Melis, D. A., & Jaakkola, T. (2018). Towards robust interpretability with self-explaining neural networks. In *Advances in Neural Information Processing Systems*, pp. 7775–7784.
- Meng, D., & Chen, H. (2017). Magnet: A two-pronged defense against adversarial examples. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp. 135–147. ACM.

- Menon, A. K., & Williamson, R. C. (2018). The cost of fairness in binary classification. In *Conference on Fairness, Accountability and Transparency*, pp. 107–118.
- Miller, T. (2018). Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence*.
- Mohseni, S., Zarei, N., & Ragan, E. D. (2018). A survey of evaluation methods and measures for interpretable machine learning. *arXiv preprint arXiv:1811.11839*.
- Montavon, G., Lapuschkin, S., Binder, A., Samek, W., & Müller, K.-R. (2017). Explaining nonlinear classification decisions with deep Taylor decomposition. *Pattern Recognition*, 65, 211–222.
- Montavon, G., Samek, W., & Müller, K.-R. (2018). Methods for interpreting and understanding deep neural networks. *Digital Signal Processing*, 73, 1–15.
- Moosavi-Dezfooli, S.-M., Fawzi, A., Fawzi, O., & Frossard, P. (2017). Universal adversarial perturbations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1765–1773.
- Moosavi-Dezfooli, S.-M., Fawzi, A., & Frossard, P. (2016). Deepfool: A simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2574–2582.
- Mueller, S. T., Hoffman, R. R., Clancey, W., Emrey, A., & Klein, G. (2019). Explanation in human-AI systems: A literature meta-review, synopsis of key ideas and publications, and bibliography for explainable AI. *arXiv preprint arXiv:1902.01876*.
- Murdoch, W. J., Liu, P. J., & Yu, B. (2018). Beyond word importance: Contextual decomposition to extract interactions from LSTMs. *arXiv preprint arXiv:1801.05453*.
- Murdoch, W. J., & Szlam, A. (2017). Automatic rule extraction from long short term memory networks. *International Conference on Learning Representations*.
- Narayanan, M., Chen, E., He, J., Kim, B., Gershman, S., & Doshi-Velez, F. (2018). How do humans understand explanations from machine learning systems? An evaluation of the human-interpretability of explanation. *arXiv preprint arXiv:1802.00682*.
- Nemati, S., Holder, A., Razmi, F., Stanley, M. D., Clifford, G. D., & Buchman, T. G. (2018). An interpretable machine learning model for accurate prediction of sepsis in the ICU. *Critical care medicine*, 46(4), 547–553.
- Nguyen, A., Yosinski, J., & Clune, J. (2015). Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 427–436.
- Nie, L., Wang, M., Zhang, L., Yan, S., Zhang, B., & Chua, T.-S. (2015). Disease inference from health-related questions via sparse deep learning. *IEEE Transactions on Knowledge and Data Engineering*, 27(8), 2107–2119.
- Nie, W., Zhang, Y., & Patel, A. (2018). A theoretical explanation for perplexing behaviors of backpropagation-based visualizations. In *International Conference on Machine Learning*, pp. 3806–3815.

- Oana-Maria, C., Brendan, S., Pasquale, M., Thomas, L., & Phil, B. (2019). Make up your mind! Adversarial generation of inconsistent natural language explanations. *arXiv preprint arXiv:1910.03065*.
- Olah, C., Mordvintsev, A., & Schubert, L. (2017). Feature visualization. *Distill*. <https://distill.pub/2017/feature-visualization>.
- Olah, C., Satyanarayan, A., Johnson, I., Carter, S., Schubert, L., Ye, K., & Mordvintsev, A. (2018). The building blocks of interpretability. *Distill*. <https://distill.pub/2018/building-blocks>.
- Olfat, M., & Aswani, A. (2018). Spectral algorithms for computing fair support vector machines. In *International Conference on Artificial Intelligence and Statistics*, pp. 1933–1942.
- Ozbulak, U. (2019). PyTorch CNN visualizations. <https://github.com/utkuozbulak/pytorch-cnn-visualizations>.
- Papernot, N., McDaniel, P., Goodfellow, I., Jha, S., Celik, Z. B., & Swami, A. (2017). Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia conference on computer and communications security*, pp. 506–519. ACM.
- Papernot, N., McDaniel, P., Jha, S., Fredrikson, M., Celik, Z. B., & Swami, A. (2016a). The limitations of deep learning in adversarial settings. In *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*, pp. 372–387. IEEE.
- Papernot, N., McDaniel, P., Wu, X., Jha, S., & Swami, A. (2016b). Distillation as a defense to adversarial perturbations against deep neural networks. In *2016 IEEE Symposium on Security and Privacy (SP)*, pp. 582–597. IEEE.
- Park, D. H., Hendricks, L. A., Akata, Z., Rohrbach, A., Schiele, B., Darrell, T., & Rohrbach, M. (2018). Multimodal explanations: Justifying decisions and pointing to the evidence. In *31st IEEE Conference on Computer Vision and Pattern Recognition*.
- Park, D. H., Hendricks, L. A., Akata, Z., Schiele, B., Darrell, T., & Rohrbach, M. (2016). Attentive explanations: Justifying decisions and pointing to the evidence. *arXiv preprint arXiv:1612.04757*.
- Pérez-Suay, A., Laparra, V., Mateo-García, G., Muñoz-Marí, J., Gómez-Chova, L., & Camps-Valls, G. (2017). Fair kernel learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 339–355. Springer.
- Pham, T. T., & Shen, Y. (2017). A deep causal inference approach to measuring the effects of forming group loans in online non-profit microfinance platform. *arXiv preprint arXiv:1706.02795*.
- Pieters, W. (2011). Explanation and trust: what to tell the user in security and AI?. *Ethics and information technology*, 13(1), 53–64.
- Pleiss, G., Raghavan, M., Wu, F., Kleinberg, J., & Weinberger, K. Q. (2017). On fairness and calibration. In *Advances in Neural Information Processing Systems*, pp. 5680–5689.
- Qin, C., Zhu, H., Xu, T., Zhu, C., Jiang, L., Chen, E., & Xiong, H. (2018). Enhancing person-job fit for talent recruitment: An ability-aware neural network approach. In *The 41st*

- International ACM SIGIR Conference on Research & Development in Information Retrieval*, pp. 25–34. ACM.
- Raghu, M., Gilmer, J., Yosinski, J., & Sohl-Dickstein, J. (2017). SVCCA: Singular vector canonical correlation analysis for deep learning dynamics and interpretability. In *Advances in Neural Information Processing Systems*, pp. 6076–6085.
- Ras, G., van Gerven, M., & Haselager, P. (2018). Explanation methods in deep learning: Users, values, concerns and challenges. In *Explainable and Interpretable Models in Computer Vision and Machine Learning*, pp. 19–36. Springer.
- Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779–788.
- Ribeiro, M. T., Singh, S., & Guestrin, C. (2016a). Model-agnostic interpretability of machine learning. *arXiv preprint arXiv:1606.05386*.
- Ribeiro, M. T., Singh, S., & Guestrin, C. (2016b). Nothing else matters: Model-agnostic explanations by identifying prediction invariance. *arXiv preprint arXiv:1611.05817*.
- Ribeiro, M. T., Singh, S., & Guestrin, C. (2016c). Why should I trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1135–1144. ACM.
- Ribeiro, M. T., Singh, S., & Guestrin, C. (2018). Anchors: High-precision model-agnostic explanations. In *Thirty-Second AAAI Conference on Artificial Intelligence*.
- Robnik-Šikonja, M., & Kononenko, I. (2008). Explaining classifications for individual instances. *IEEE Transactions on Knowledge and Data Engineering*, 20(5), 589–600.
- Rozsa, A., Rudd, E. M., & Boulton, T. E. (2016). Adversarial diversity and hard positive generation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 25–32.
- Rudin, C. (2019). Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5), 206.
- Sabour, S., Cao, Y., Faghri, F., & Fleet, D. J. (2015). Adversarial manipulation of deep representations. *arXiv preprint arXiv:1511.05122*.
- Saha, A., Subramanya, A., & Pirsiavash, H. (2019). Hidden trigger backdoor attacks. *arXiv preprint arXiv:1910.00033*.
- Samangouei, P., Kabkab, M., & Chellappa, R. (2018). Defense-gan: Protecting classifiers against adversarial attacks using generative models. *arXiv preprint arXiv:1805.06605*.
- Samek, W., Binder, A., Montavon, G., Lapuschkin, S., & Müller, K.-R. (2017a). Evaluating the visualization of what a deep neural network has learned. *IEEE transactions on neural networks and learning systems*.
- Samek, W., Wiegand, T., & Müller, K.-R. (2017b). Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models. *arXiv preprint arXiv:1708.08296*.

- Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., & Batra, D. (2017). Grad-CAM: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 618–626.
- Shrikumar, A., Greenside, P., & Kundaje, A. (2017). Learning important features through propagating activation differences. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 3145–3153. JMLR. org.
- Simonyan, K., Vedaldi, A., & Zisserman, A. (2013). Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*.
- Sirignano, J., Sadhwani, A., & Giesecke, K. (2016). Deep learning for mortgage risk. *SSRN preprint*.
- Sobol, I. M. (2001). Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates. *Mathematics and computers in simulation*, 55(1-3), 271–280.
- Springenberg, J. T., Dosovitskiy, A., Brox, T., & Riedmiller, M. (2014). Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*.
- Su, J., Vargas, D. V., & Sakurai, K. (2019). One pixel attack for fooling deep neural networks. *IEEE Transactions on Evolutionary Computation*.
- Sundararajan, M., Taly, A., & Yan, Q. (2016). Gradients of counterfactuals. *arXiv preprint arXiv:1611.02639*.
- Sundararajan, M., Taly, A., & Yan, Q. (2017). Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 3319–3328. JMLR. org.
- Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., & Fergus, R. (2013). Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*.
- Tabacof, P., & Valle, E. (2016). Exploring the space of adversarial images. In *2016 International Joint Conference on Neural Networks (IJCNN)*, pp. 426–433. IEEE.
- Tan, S., Caruana, R., Hooker, G., Koch, P., & Gordo, A. (2018). Learning global additive explanations for neural nets using model distillation. *arXiv preprint arXiv:1801.08640*.
- Teney, D., Anderson, P., He, X., & van den Hengel, A. (2018). Tips and tricks for visual question answering: Learnings from the 2017 challenge. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4223–4232.
- Tjoa, E., & Guan, C. (2019). A survey on explainable artificial intelligence (XAI): Towards medical XAI. *arXiv preprint arXiv:1907.07374*.
- Tommasi, T., Patricia, N., Caputo, B., & Tuytelaars, T. (2017). A deeper look at dataset bias. In *Domain adaptation in computer vision applications*, pp. 37–55. Springer.
- Tulshan, A. S., & Dhage, S. N. (2018). Survey on virtual assistant: Google Assistant, Siri, Cortana, Alexa. In *International Symposium on Signal Processing and Intelligent Recognition Systems*, pp. 190–201. Springer.

- Varshney, K. R., & Alemzadeh, H. (2017). On the safety of machine learning: Cyber-physical systems, decision sciences, and data products. *Big data*, 5(3), 246–255.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. In *Advances in neural information processing systems*, pp. 5998–6008.
- Vinyals, O., Toshev, A., Bengio, S., & Erhan, D. (2015). Show and tell: A neural image caption generator. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3156–3164.
- Wang, Y., Huang, M., Zhao, L., et al. (2016). Attention-based LSTM for aspect-level sentiment classification. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pp. 606–615.
- Wiegrefe, S., & Pinter, Y. (2019). Attention is not not explanation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 11–20.
- Woodworth, B., Gunasekar, S., Ohannessian, M. I., & Srebro, N. (2017). Learning non-discriminatory predictors. *arXiv preprint arXiv:1702.06081*.
- Xie, C., Wang, J., Zhang, Z., Ren, Z., & Yuille, A. (2017). Mitigating adversarial effects through randomization. *arXiv preprint arXiv:1711.01991*.
- Xie, N., Lai, F., Doran, D., & Kadav, A. (2019). Visual entailment: A novel task for fine-grained image understanding. *arXiv preprint arXiv:1901.06706*.
- Xie, N., Sarker, M. K., Doran, D., Hitzler, P., & Raymer, M. (2017). Relating input concepts to convolutional neural network decisions. *arXiv preprint arXiv:1711.08006*.
- Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhudinov, R., Zemel, R., & Bengio, Y. (2015). Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pp. 2048–2057.
- Yang, F., Du, M., & Hu, X. (2019). Evaluating explanation without ground truth in interpretable machine learning. *arXiv preprint arXiv:1907.06831*.
- Yim, J., Joo, D., Bae, J., & Kim, J. (2017). A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. In *The IEEE Conference on Computer Vision and Pattern Recognition*.
- Yosinski, J., Clune, J., Nguyen, A., Fuchs, T., & Lipson, H. (2015). Understanding neural networks through deep visualization. *arXiv preprint arXiv:1506.06579*.
- Yuan, X., He, P., Zhu, Q., & Li, X. (2019). Adversarial examples: Attacks and defenses for deep learning. *IEEE transactions on neural networks and learning systems*.
- Zafar, M. B., Valera, I., Gomez Rodriguez, M., & Gummadi, K. P. (2017a). Fairness beyond disparate treatment & disparate impact: Learning classification without disparate mistreatment. In *Proceedings of the 26th International Conference on World Wide Web*, pp. 1171–1180. International World Wide Web Conferences Steering Committee.

- Zafar, M. B., Valera, I., Rodriguez, M., Gummadi, K., & Weller, A. (2017b). From parity to preference-based notions of fairness in classification. In *Advances in Neural Information Processing Systems*, pp. 229–239.
- Zeiler, M. D., & Fergus, R. (2014). Visualizing and understanding convolutional networks. In *European conference on computer vision*, pp. 818–833. Springer.
- Zeiler, M. D., Taylor, G. W., Fergus, R., et al. (2011). Adaptive deconvolutional networks for mid and high level feature learning. In *ICCV*, Vol. 1, p. 6.
- Zellers, R., Bisk, Y., Farhadi, A., & Choi, Y. (2019). From recognition to cognition: Visual commonsense reasoning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6720–6731.
- Zemel, R., Wu, Y., Swersky, K., Pitassi, T., & Dwork, C. (2013). Learning fair representations. In *International Conference on Machine Learning*, pp. 325–333.
- Zhang, C., Bengio, S., Hardt, M., Recht, B., & Vinyals, O. (2016). Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*.
- Zhang, Q.-s., & Zhu, S.-C. (2018). Visual interpretability for deep learning: A survey. *Frontiers of Information Technology & Electronic Engineering*, 19(1), 27–39.
- Zhang, Q., Cao, R., Shi, F., Wu, Y. N., & Zhu, S.-C. (2018). Interpreting CNN knowledge via an explanatory graph. *Proc. of AAAI Conference on Artificial Intelligence*.
- Zhang, Q., Cao, R., Wu, Y. N., & Zhu, S.-C. (2017). Growing interpretable part graphs on ConvNets via multi-shot learning. In *Proc. of AAAI Conference on Artificial Intelligence*, pp. 2898–2906.
- Zhang, Q., Yang, Y., Ma, H., & Wu, Y. N. (2019a). Interpreting CNNs via decision trees. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6261–6270.
- Zhang, W. E., Sheng, Q. Z., & Alhazmi, A. A. F. (2019b). Generating textual adversarial examples for deep learning models: A survey. *arXiv preprint arXiv:1901.06796*.
- Zhao, Y., Hryniewicki, M. K., Cheng, F., Fu, B., & Zhu, X. (2018). Employee turnover prediction with machine learning: A reliable approach. In *Proceedings of SAI intelligent systems conference*, pp. 737–758. Springer.
- Zhao, Z., Dua, D., & Singh, S. (2017). Generating natural adversarial examples. *arXiv preprint arXiv:1710.11342*.
- Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., & Torralba, A. (2014). Object detectors emerge in deep scene CNNs. *arXiv preprint arXiv:1412.6856*.
- Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., & Torralba, A. (2016). Learning deep features for discriminative localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2921–2929.
- Zintgraf, L. M., Cohen, T. S., Adel, T., & Welling, M. (2017). Visualizing deep neural network decisions: Prediction difference analysis. *International Conference on Learning Representations*.