Toward Transparent AI: A Survey on Interpreting the Inner Structures of Deep Neural Networks

Tilman Räuker*  
traeuker@gmail.com

Anson Ho*
Epoch  
anson@epoch.ai

Stephen Casper*
MIT CSAIL  
scasper@mit.edu

Dylan Hadfield-Menell  
MIT CSAIL

Abstract—The last decade of machine learning has seen drastic increases in scale and capabilities. Deep neural networks (DNNs) are increasingly being deployed in the real world. However, they are difficult to analyze, raising concerns about how they function. Effective tools for interpreting them will be important for building more trustworthy AI by helping to identify problems, fix bugs, and improve basic understanding. In particular, “inner” interpretability techniques, which focus on explaining the internal components of DNNs, are well-suited for developing a mechanistic understanding, guiding manual modifications, and reverse engineering solutions.

Much recent work has focused on DNN interpretability, and rapid progress has thus far made a thorough systematization of methods difficult. In this survey, we review over 300 works with a focus on inner interpretability tools. We introduce a taxonomy that classifies methods by what part of the network they help to explain (weights, neurons, subnetworks, or latent representations) and whether they are implemented during (intrinsic) or after (post hoc) training. To our knowledge, we are also the first to survey a number of connections between interpretability research and work in adversarial robustness, continual learning, and benchmarking, guiding manual modifications, and reverse engineering solutions.

We define an interpretability method as any process by which an AI system’s computations can be characterized in human-understandable terms. This encompasses a broad set of techniques in the literature on DNNs, but in this paper, we focus specifically on methods for understanding internal structures and representations (i.e. not data, inputs, outputs, or the model as a whole). We call these inner interpretability methods. We introduce a taxonomy for these methods, providing an overview of the literature, highlight key connections between interpretability and other topics in deep learning, and conclude with directions for continued work. Our central goals are twofold: (1) to provide a thorough resource for existing inner interpretability work and (2) to propose directions for continued research.

I. INTRODUCTION

A defining feature of the last decade of deep learning is drastic increases in scale and capabilities \[143, 264\], with the training compute for machine learning systems growing by ten orders of magnitude from 2010 to 2022 \[263\]. At the same time, deep neural networks (DNNs) are increasingly being deployed in the real world. If rapid progress continues, broad-domain artificial intelligence could be highly impactful \[39, 59, 209, 227, 246, 277\].

Given this potential, it is important that practitioners can understand how AI systems make decisions, especially their issues. Models are most typically evaluated by their performance on a test set for a particular task. This raises concerns because a black box performing well on a test set does not imply that the learned solution is adequate. Testing sets typically fail to capture the full deployment distribution, including potential adversarial inputs. They also fail to reveal problems with a model that do not directly relate to test performance (e.g. learning harmful biases). Moreover, even if a user is aware of inadequacies, the black-box nature of a system can make it difficult to fix issues. Thus, a key step to building safe and trustworthy AI systems is to have an expanded toolbox for detecting and addressing problems.

Open-ended evaluation: Short of actually deploying a system, any method of evaluating its performance can fundamentally only be a proxy for its performance. In particular, test sets can fail to identify—and often incentivize—undesirable solutions such as dataset bias, socially harmful biases, or developing deceptive solutions. Thus, it is important to have additional ways of rigorously evaluating systems’ performance. One of the most important advantages of interpretability techniques lies in their unique ability to, unlike standard evaluation methods, allow humans to more open-endedly study a model and search for flaws.

Fixing bugs: By understanding a problem and/or producing examples that exploit it, a network can be redesigned, probed,
Fig. 1. A taxonomy of inner interpretability methods and hazards associated with them. This mirrors our organization of Sections II-V. We organize methods first by what part of the network’s computational graph they help to explain: weights, neurons, subnetworks, or latent representations. Second, we organize approaches by whether they are intrinsic (implemented during training), post hoc (implemented after training), or can rely on a mix of intrinsic and post hoc techniques. Finally, we also discuss prominent hazards for different approaches.

fine-tuned, and/or adversarially-trained to better align it with the user’s goals.

Determining accountability: Properly characterizing behavior is essential for establishing responsibility in the case of misuse or deployment failures.

Improvements in basic understanding: By offering users more basic insights on models, data, and/or algorithms, interpretability techniques could be useful for reducing risks in deployed systems or better forecasting progress in AI. However, improved basic understanding could also be harmful if it causes advancements in risky capabilities to outpace effective oversight. We discuss this in Section VII.

“Microscope” AI: Rigorously understanding how an AI system accomplishes a task may provide additional domain knowledge. This could include insights about solving the task as a whole or the properties of specific examples. This goal has been referred to as “microscope” AI [133], and it could allow for reverse engineering more understandable or verifiable solutions. This may be especially valuable for studying systems with superhuman performance.

B. Scope

Inner Interpretability: Our focus is on inner interpretability methods for DNNs. Black-box techniques, adversarial techniques, input attribution methods, neurosymbolic methods, and “good old-fashioned AI” are all valuable but beyond the scope of this survey. This is not to say that they are of less value to building safer AI than the methods we focus on – many of them have major advantages, and a diverse interpretability toolbox is important. However, we focus on inner interpretability methods because (1) there is a great deal of current interest in them and (2) they are well-equipped for certain goals such as guiding manual modifications, reverse engineering solutions, and detecting inner “latent” knowledge which may contribute to deceptive behavior.

Contrasts with past survey works: See also several previous surveys and critiques of interpretability work that overlap with ours [3], [69], [71], [79], [112], [136], [155], [178], [200], [202]-[204], [243], [244], [251], [254], [255], [314]. Unlike this survey, [79], [155], [178], [200], [243] are critique/position papers that do not extensively survey existing work, [3], [71], [112], [203], [204], [244], [254], [255] focus mostly or entirely on approaches out of the scope of this work (e.g. non-DNNs or feature attribution), [69], [136] only survey methods for language models, [314] only focuses on convolutional networks, [251] only surveys single-neuron methods, [203], [254] only focus on post-hoc methods, and [3], [79], [112], [178], [200], [243], [255], [314] are relatively old (before 2020). This survey is also distinct from all of the above in its focus on inner interpretability, implications for diagnostics and debugging, and the intersections between interpretability and a number of other research paradigms.
C. Taxonomy

Our taxonomy divides inner interpretability techniques by what part of the DNN’s computational graph they explain: weights, neurons, subnetworks, or latent representations. We dedicate Sections II-IV respectively to each of these approaches. Interpretability techniques can also be divided by whether they are used during or after training. Intrinsic interpretability techniques involve training models to be easier to study or come with natural interpretations. Post hoc techniques aim to interpret a model after it has been trained. We divide methods by whether they are intrinsic or post hoc at the subsection level.

Fig. 1 depicts our taxonomy and previews the organization of Sections II-IV. Note that this taxonomy sometimes divides related methods. For example, continual learning methods for weights (Section II-A) and neurons (Section III-A) are conceptually similar, and methods for interpreting subnetworks (Section IV) frequently involve variations or applications of methods for weights II) or neurons III). We note these connections as we discuss the families of methods below. However, we divide methods first by what part of the network they target because how a technique operates on a network typically matters more for goal-oriented engineering than whether it occurs during or after training.

II. Weights

A. Continual Learning (Intrinsic):

One research paradigm in deep learning is to train systems that can learn new tasks without forgetting old ones. This is known as continual learning or avoiding catastrophic forgetting[72], [267]. Some techniques are based on the principle of having weights specialize for particular types of input data, updating more for some than others [8], [11], [150], [175], [191], [278], [312]. This offers a natural way to characterize weights based on the tasks or classes that they specialize in. Unfortunately, current research on these methods has not been done with an emphasis on improving interpretations of weights or subnetworks. This may be a useful direction for future work. See also methods for continual learning that operate on neurons in Section III-A.

B. Weight-Masking (Post Hoc):

In contrast to intrinsic methods, one can also train weight masks over a network to determine which weights are essential for which tasks [65], [304], [319]. For example, a mask over a classifier’s weights can be trained to cover as many as possible while preserving performance on a subset of data. The resulting mask identifies a subset of weights (and a corresponding subnetwork) that can be causally understood as specializing in that subtask. This approach also works for identifying subnetworks that specialize in a task (Section IV).

C. Frivolous Weights (Hazard):

A difficulty in interpreting weights is that many are often unimportant to the network. Past works have shown that networks can often be pruned to contain a very small fraction of their original weights with little to no loss in performance (though sometimes with fine-tuning) [33], [102], [283]. See also frivolous neurons (Section III-G).

III. Individual Neurons

As is common in the literature, we use “neuron” to refer both to units in dense layers and elements of feature maps in convolutional layers.

A. Continual Learning (Intrinsic):

Just as continual learning [72], [267] can be facilitated via specialization among weights (see Section II-A), the same can be done with neurons. Unlike weight-based continual learning methods, which have weights update more for some tasks than others, neuron-based ones typically rely on adding new neurons to the architecture upon encountering a new task [167], [247], [310]. This discourages neurons from learning to simultaneously detect features from multiple unrelated tasks and allows for natural interpretations of neurons in terms of what subtasks they specialize in. As with continual learning methods that operate on weights, current research on these methods has not been done with an emphasis on improving interpretations of neurons or subnetworks. This may be a useful direction for future work. See also Section IV-B which discusses methods for modularity among neurons.
Fig. 3. Inner interpretability methods for individual neurons can focus on (1) continual learning techniques that make neurons specialize in particular tasks, (2) dataset-based techniques that aim to find which neurons respond to which features, (3) feature synthesis to construct inputs that excite individual neurons, (4) perturbation or ablation of neurons coupled with analysis of changes to network behavior, and (5) gradient-based attribution methods that analyze partial derivatives of outputs w.r.t. neural activations.

B. Dataset-Based (Post Hoc):

A simple way to characterize the role of individual neurons is to use a dataset to analyze which types of inputs they respond to. Perhaps the simplest example of this is searching through a dataset and selecting the inputs that maximally excite a given neuron [323]. A more sophisticated technique known as network “dissection” uses a richly-labeled dataset of semantic concepts to analyze neural responses [25]–[27]. A neuron can then be characterized based on how well its activations align with different types of input. This line of work has been extended to assign descriptions to neurons using compositional logic expressions on a set of labels [208]. This allows the interpretability of neurons to be quantified as the intersection over union for a logical formula on input features and the neuron’s activations. This has been further extended to develop natural language explanations by using captioning methods to describe a set of image patches that activate a neuron [125], [219]. These approaches have proven useful for identifying undesirable biases in networks [125], [208]. Dissection has also been used to analyze what types of neurons are exploited by adversarial examples [306], identify failure modes for text-to-image models [57], and probe neural responses in transformers to isolate where specific information is stored [68], [98], [105], [106], [197], [288]. This can then be followed by improving the model by editing a learned fact (such as an undesirable bias) [68], [197]. Unfortunately, all dataset-based methods are limited by the diversity of examples in the dataset used and the quality of labels. See also probing methods in Section V-E.

C. Feature Synthesis (Post Hoc):

This approach is based on synthesizing inputs with the goal of maximally (or minimally) activating a given neuron or combination of neurons. Synthesis methods come with the advantage of not being limited to a particular dataset. Several works have taken this approach, optimizing inputs to excite particular neurons [189], [213], [223]. One can use a distance measure in the optimization objective to synthesize a batch of inputs to be diverse [223]. There has also been work on using generative models instead of directly optimizing over input features [49], [51], [211], [212]. A broader survey of these types of methods is provided by [214]. However [38] finds that natural exemplars which strongly-activate individual neurons can be more useful for helping humans predict neural responses to data than synthesized features.

D. Neural Perturbation and Ablation (Post Hoc):

By analyzing a DNN’s behavior under perturbation to a neuron, one can gain insight into the type of information it processes. For example, if a neuron in an image classifier robustly and uniquely detects dogs, one should expect performance on dog classification to worsen when that neuron is ablated (i.e. dropped out). A key benefit of these methods is that they allow for testing counterfactuals, helping establish a causal rather than a correlational relationship between neural activations and the behavior of the network. Works in this area have used neural ablation [127], [325], subspace ablation [207], [236], and non-ablation perturbations [24], [84]. Notably, the net effect of perturbing a neuron can vary by context and which others, if any, are also perturbed. To account for this, one can compute Shapley values for neurons to measure their importance averaged over the ablation of other neurons [111].
defines and detects two distinct types: prunable
G. Frivolous Neurons (Hazard):

polysemanticity to layers. Each compress networks by eliminating frivolous neurons.

A solution. For example, [122], [131], [188], [250], [268], [318]. Most of this work has been done to study attributions on inputs and is outside the scope of this survey. However, the same type of approach has been applied for attribution with internal neurons. [273] introduce an approach for this using gradients along with runtime tests for sensitivity and invariance to evaluate the quality of interpretations. Building off this, several works have found gradient-based attribution useful in large language models [14], [81], [187], including to guide a search for where certain facts are stored [68]. However, these methods are limited in that explanations are only as valid as the local linear approximation the gradient is based on, and they cannot directly provide causal explanations.

E. Gradient-Based Attribution (Post Hoc):

Much work has been done on gradient-based feature attribution to study which features are influential for neural responses or model outputs. There are several surveys and critiques of feature attribution methods in particular [4], [5], [15], [74], [77], [99], [129], [139], [141], [216], [266], [318]. This would suggest that polysemantic neurons might be useful for model performance. However, they also pose a significant challenge for two reasons. First, interpretations of polysemantic neurons might be useful for model performance. However, they also pose a significant challenge for two reasons. First, interpretations of polysemantic neurons are more likely to be incorrect or incomplete. Second, it has been shown that they can be exploited for adversarial attacks [225], [208]. See also Section V-C for a discussion of entanglement, which generalizes the notion of polysemanticity to layers.

F. Polysemantic Neurons (Hazard):

Polysemantic neurons are activated by multiple unrelated features. They have been discovered via dataset-based methods [100], [125], [208], various forms of visual feature synthesis [114], [213], [221], [290], and feature attribution [85]. How and why they form remains an open question. However, [221] observed a tendency for monosemantic neurons to become polysemantic over the course of training and hypothesized that it is associated with representing information more efficiently. This would suggest that polysemantic neurons might be useful for model performance. However, they also pose a significant challenge for two reasons. First, interpretations of polysemantic neurons are more likely to be incorrect or incomplete.

G. Frivolous Neurons (Hazard):

Frivolous neurons are not important to a network. [48] defines and detects two distinct types: prunable neurons, which can be removed from a network by ablation, and redundant neurons, which can be removed by refactoring weight matrices. They pose a challenge for interpretability because a frivolous neuron’s contribution to the network may be meaningless or difficult to detect with certain methods (e.g., neural perturbation). Network compression may offer a solution. For example, [122], [131], [188], [250], [268] each compress networks by eliminating frivolous neurons. Compression and the interpretability of neurons are linked. After compressing a network, [174] found that the remaining neurons were more interpretable with only marginal change in performance, and [308] used proxies for neuron interpretability to guide neuron-level pruning. Additionally, the motivation for pruning to increase interpretability is closely-related to intrinsically interpretable layer representations. See also Section II-C on frivolous weights and Section V-C on neural disentanglement.

IV. SUBNETWORKS

Note that many of the methods used for analyzing subnetworks supervise on techniques for weights (Section II) or neurons (Section III).

A. Sparsity (Intrinsic):

Sparse weights inside of DNNs allow for much simpler analysis of relationships between neurons. In some cases, sparsification can reduce the number of weights by almost two orders of magnitude while causing little to no tradeoff with performance [102]. Sparsity-aided interpretability has been explored through pruning [29], [101], [205], [303] regularization [242], and sparse attention [196]. In particular, [303] demonstrates how sparsity can be paired with post-hoc techniques for neuron analysis to help a human to edit a model. This has direct implications for safety and debiasing. Pruning portions of the network architecture can also be guided by measures of interpretability [296], [309]. Meanwhile, as an alternative to conventional sparsity, [305] introduce a method to regularize the behavior of a neural network to mimic that of a decision tree.

While sparsity simplifies the analysis of subnetworks, it may not improve the interpretability of individual neurons. [101] find no increase in their interpretability through the dissection of pruned networks, and [196] fail to find evidence of improved interpretability of individual neurons with sparse attention.

B. Modularity (Intrinsic):

Modularity is a common principle of engineered systems and allows for a model to be understood by analyzing its parts separately. At a high level, [13] offers a survey of DNN modularization techniques, and [7], [201] study the capabilities and generality of modular networks compared to monolithic ones. The simplest way to design a modular DNN is to use an explicitly modular architecture. This can be considered “hard” modularity. This can be a form of “model-aided deep learning” [265] if domain-specific considerations are used to guide the design. Modular architectures were studied by [291] who analyzed the extent to which neurons in a branched architecture learned to process different features from those in other branches, and [307] who experimented with distinct neural modules that were trained to execute algorithmic subroutines.

Aside from branched architectures, a “softer” form of modularity can be achieved if neurons in different modules are connected to each other but must compete for access to information. This can allow for end-to-end differentiability,
yet sparse information flow between modules. Methods for soft modularity have been studied via initialization [96], regularization [96], a controller [140], [151], or sparse attention [16], [85], [115], [261]. Notably [261] used attention to both induce specialization among neurons and reduce catastrophic forgetting. See also methods for avoiding catastrophic forgetting by having subsets of neurons specialize in a given task in Section III-A.

C. Modular Partitionings (Post Hoc):

A post hoc way of understanding a DNN in terms of modules is to partition the neurons into a set of subnetworks, each composed of related neurons. Toward this goal, [96], [298], [299] divide neurons into modules based on graphical analysis of the network’s weights and analyze how distinct the neurons in each module are. These methods involve no data or runtime analysis. In contrast, [18], [50], [127], [163], [297] each perform partitioning and cluster analysis based on how neurons associate with inputs and/or outputs. In particular, [127] present a statistical pipeline for estimating the interpretability of neuron clusters without a human in the loop. Overall, however, these methods have had very limited success in finding highly-composite partitionings in models. A useful direction for future work may be to combine this approach with intrinsic modularity methods.

D. Circuits Analysis (Post Hoc):

Instead of analyzing an entire partition of a network, a much simpler approach is to study individual subnetworks inside of it. These have often been referred to as neural “circuits” which can be as small as just a few neurons and weights. This has been done with weight masking [65], [295], data-based methods [94], [257], [280], [315], [316], feature synthesis [45], [220], [224], [233], [260], [290], and neural ablation [118], [198]. However, many of the successes of circuits analysis to date have focused on toy models and involved intensive effort from human experts. To be useful for improving models in practical applications, future methods will likely need to leverage automation. [239] make progress on this by distilling a DNN into a sparse symbolic causal graph on a set of concepts and providing theoretical guarantees for the faithfulness of the graph. See also Section V-D for a discussion of circuits in transformers.

V. INTERNAL REPRESENTATIONS

A. Self-Explaining Models (Intrinsic):

Most methods in the literature used for understanding DNNs aim to help a human “open up” the network and study parts of it. If one wants to understand another human’s reasoning, the analogous techniques would involve studying their brain directly. These are sometimes useful, but in most cases, simply asking another human for an explanation of what they are thinking is much more effective. Self-explaining AI systems are meant to provide such explanations of internal reasoning in an analogous way to how humans can provide them. Competing definitions are offered in the literature, but we will use one based on [87], which simply requires that a model produces an explanation for its reasoning that can be easily be understood by a human, ideally paired with a confidence estimate.

In computer vision, one approach has been to classify images based on their similarity to a set of learned “prototypes” [12], [53], [145], [171], [248], [313]. Prototype-based classification has also been studied in language models [92]. These methods allow the model to attribute its outputs to a set of exemplary datapoints, allowing its decision to be explained as “this input resembles these other examples.”

Another self-explaining AI strategy has been to supervise human-understandable explanations for model outputs that are computed from the same inner representations. In computer vision, this has been done for classification and question answering [9], [123], [124], [149], [230]. In natural language processing, this has been done for question answering and natural language inference with explanations [43], [160], [162], [320]. For large language models that have sufficiently general language capabilities, explanations can also simply be elicited via prompts (e.g., [40], [58]). However, the extent to which these explanations accurately explain the model’s decision making is very unclear [142].

[12] argues that explanations should meet three criteria: (1) Explicitness: are explanations direct and understandable? (2) Faithfulness: do they correctly characterize the decision? And
(3) Stability: how consistent are they for similar examples? It has been shown that explanations from such models can be unfaithful [12], [28, 44, 221] or vulnerable to adversarial examples [44, 128, 221], so producing self-explaining models that meet these remains an open challenge. Toward fixing these issues, [75] introduces an NLP benchmark, and [36] provides an interactive debugging method for prototype networks.

B. Adversarial Training (Intrinsic)

[88] found that adversarially trained classifiers exhibited improvements in a number of interpretability-related properties, including feature synthesis for neurons (see Section III-C). It has also been found that these adversarially trained networks produce better representations for transfer learning [253], image synthesis [49, 51, 256], and for modeling the human visual system [89]. Unfortunately, robustness may be at odds with accuracy [252], potentially due to predictive but “nonrobust” features in a dataset [135]. This had led to an understanding that adversarial examples can be used to help understand what types of useful or exploitable features a network detects and represents [49, 51].

C. Disentanglement (Intrinsic):

During a pass through a network, each layer’s activations can be represented as a vector in latent space. The goal of disentanglement [30] is ensuring that features can be more easily identified from studying a latent vector by encouraging a more bijective relationship between neurons and a set of interpretable concepts. See also Section III-F for a discussion of polysemantic neurons. Disentanglement can be done in a supervised manner by encouraging neurons to align to a set of predetermined concepts. [56] did this by applying a whitening operation to decorrelate features followed by a learned orthogonal transformation to produce latent activations that could be supervised. Similarly, inner supervision was used by [152, 183, 184] to train a ‘bottleneck’ layer to separate features, and by [272] to learn sparse, interpretable embeddings. However, [190] discusses challenges with these models, particularly the problem of “leakage” in which undesired information nonetheless makes it through the bottleneck.

Disentanglement can also be done in an unsupervised manner. A partial example of this is dropout [269] which prevents co-adaptation among neurons, though at the cost of increasing redundancy. Other works have explored using lateral inhibition between neurons in a layer to make them compete for activation [46, 83, 157, 272], designing a “capsule” based architecture in which a group of neurons have activations that each represent a specific feature [73, 249], aligning activations to components of variation in data [161], using a mutual information loss [55], using an inter-class activation entropy-based loss [317], regularizing the Hessian of the network’s outputs w.r.t a layer [231], training a classifier and autoencoder from the same latents [259], or learning a mask over features [121]. Other works have focused specifically on autoencoders, training them to have more independently-activated neurons [41, 54, 126, 148, 158]. However, in a survey of these methods [181, 182] prove an impossibility result for unsupervised disentanglement without inductive biases on both the models and data.

D. Tokens and Attention (Intrinsic and Post Hoc):

Transformer architectures process data by passing token representations through attention and feed forward layers in alternating fashion. These architectural building blocks pose
unique opportunities for studying the network’s internal representations.

First, the tokens can be studied. This can be done by interpreting token representations in transformers directly [84], [108], [109], [169], [210], [225] or analyzing how fully-connected layers process them [110], [210].

Second, key-query products are computed inside of an attention layer and represent how much each inner token is attending to others. This notion of studying relations between token representations has similarities to circuits analysis covered in Section V-D In their seminal work, [20] showed that an attentional alignment appeared to show the expected attention patterns for translation. Other recent works have used this approach more systematically [2], [62], [119], [285] including for the study of harmful biases [72]. And [240] introduced a “ConceptTransformer” whose outputs can be explained as an attention map over user-defined concepts much like a concept bottleneck network [152]. Interactive tools for visual analysis of attentional attribution are provided by [166], [179], [271], [287]. And [52], [84], [225] expanded on this approach toward the goal of multi-step attribution across multiple layers. Importantly, the analysis of attention may not always suggest faithful explanations, and an over-reliance on them for interpretation can be hazardous [138], [262], [301]. Finally, transformers may have many frivolous, prunable attention heads [289], suggesting a further need for caution because not all heads may be worth interpreting.

E. Concept Vectors (Post Hoc):

While disentanglement aims to align concepts with individual neurons, methods for analyzing concept vectors are post hoc solutions to the same problem. Here, the goal is to associate directions in latent space with meaningful concepts. Several works have done this by analysis of activations induced by images from a dataset of concepts [100], [146], [185], [186], [238], [324] including [1], [311] who used it explicitly for debugging. A contrasting approach was used by [259]. Rather than beginning with concepts and then identifying directions for them, they first identified directions using a generator and a “layer selectivity” heuristic, and then sought to find post hoc explanations of what they encoded. A debugging-oriented approach was taken by [137], [302] who classified and clustered embeddings of data examples that were incorrectly labeled by a classifier, including cases due to demographic biases. This allowed for detection, interpretation, and intervention for potentially difficult inputs for the model as well as a way to identify underrepresented subcategories of data. Unfortunately for these approaches, there is evidence that networks learn to represent many more useful concepts than can linearly independently be represented by their internal layers [86]. And single directions in activation space can correspond to unrelated concepts depending on the activation vector’s magnitude [32].

F. Probing (Post Hoc):

Given some way of embedding data, the goal of probing is to understand whether or not that embedding captures a certain type of information. Probing leverages transfer learning as a test for whether embeddings carry information about a target task. The three steps to probing are to (1) obtain a dataset that contains examples capturing variation in some quality of interest, (2) embed the examples, and (3) train a model on those embeddings to see if it can learn the quality of interest. Any inner representation from any model can be used, making this a versatile technique. A survey of probing works is provided by [28]. The simplest example of probing is to use an unsupervised learning algorithm as the probe [130]. Additional work has been done with linear probes for image classifiers [10]. However, probing has most commonly been done in language models [6], [63], [91], [117], [153], [168], [170], [177], [199], [217], [232], [252], [274]. Notably, a form of contrastive probing was used by [42] for detecting deception in language models. While versatile, probing is imperfect [17]. One issue is that a probe’s failure to learn to represent the desired quality in data is not necessarily an indicator that it is not represented. For example, this may be the case with an underparameterized probe. On the other hand, a successful probe does not necessarily imply that the model being probed actually uses that information about the data. This was demonstrated by [237] who argued for the use of rigorous controls when probing. In a subsequent paper, [83] aimed to address this problem by pairing probing with experiments that manipulated the data in order to analyze the causal influence of perturbations on performance. See also dataset-based methods for characterizing neurons in Section III-B.

G. Representation Comparison (Post Hoc):

A somewhat indirect way of characterizing the representations learned by a DNN is to estimate the similarity between its inner representations and those of another DNN. This is challenging to quantify because networks are highly nonlinear and represent concepts in complex ways that may not reliably align with neurons or directions in activation space. Nonetheless, a set of works have emerged to address this problem with a variety of both linear and nonlinear methods. These include single-neuron alignment [113], [172], [173], [276], vector-space alignment [294], canonical correlation analysis [206], singular vector canonical correlation analysis [234], centered kernel alignment [154], [215], [235], [275], deconfounded representation similarity [66], layer reconstruction [176], model stitching [21], [64], [194], representational similarity analysis [195], representation topology divergence [22], and probing [93]. Methods like these may aid in a better basic understanding of what features networks learn and how. However, different methods often disagree about the extent to which layers are similar. [76] argue that these methods should be sensitive to changes that affect functional behavior and invariant to ones that do not. They introduce a benchmark for evaluating similarity measures and show that two of the most
common methods, canonical correlation analysis and centered kernel alignment, each fail in one of these respects.

VI. DISCUSSION

Interpretability is closely linked with adversarial robustness research. There are several connections between the two areas, including some results from non-inner interpretability research. (1) More interpretable DNNs are more robust to adversaries [141]. A number of works have studied this connection by regularizing the input gradients of networks to improve robustness [37], [80], [90], [97], [120], [144], [147], [192], [218], [241], [258]. Aside from this, [82] use lateral inhibition, and [251] use a second-order optimization technique, each to improve both interpretability and robustness. Furthermore, emulating properties of the human visual system in a convolutional neural network improves robustness [70]. (2) More robust networks are more interpretable [19], [86], [88], [228]. Adversarially trained networks also produce better representations for transfer learning [7], [253], image generation [49], [51], [256], modeling the human visual system [89], and fitting symbolic graphs [239]. (3) Interpretability tools can be used to design adversaries. Doing so is a way to rigorously demonstrate the usefulness of the interpretability tool. This has been done by [47], [49], [51], [125], [208] and has been used to more effectively generate adversarial training data [327]. As a means of debugging models, [132] argues for using “relaxed” adversarial training, which can rely on interpretability techniques to discover distributions of inputs or latents which may cause a model to fail. (4) Adversarial examples can be interpretability tools [49], [51], [78], [135], [279] including adversarial trojan detection methods [104], [110], [116], [180], [187], [292], [293], [322].

Interpretability is also closely linked with continual learning, modularity, network compression, and semblance to the human visual system. Continual learning methods involving parameter isolation, and/or regularization make neurons and weights more intrinsically interpretable. Sections II-A and III-A discussed how these methods suggest intrinsic interpretations for weights and/or neurons. Thus, they allow for each weight or neuron to be understood as having partial memberships in a set of task-defined modules. Aside from this, a number of other intrinsic modularity techniques were the focus of Section IV-B. And as discussed in Section IV-C, networks can also be interpreted by partitioning them into modules and studying each separately. Moreover, “frivolous” neurons, as discussed in Section III-C, can include sets of redundant neurons that can be interpreted as modules. Networks with frivolous neurons are compressible, and compression can guide interpretations, and interpretations can guide compression, as discussed in Section III-C. Finally, structuring networks to be more similar to the human visual system, including having convolutional filters that represent easily-describable patterns also improves robustness [70].

Interpretability techniques should scale to large models. Small networks and simple tasks such as MNIST classification [165] are often used for testing methods. However, simple networks performing simple tasks can only be deployed in a limited number of real-world settings, and they are sometimes easy to replace with other intrinsically interpretable, non-network models. As a result, the scalability of a technique is strongly related to its usefulness. For example, capsule networks [249] achieve impressive performance on MNIST classification and have intrinsic interpretability properties that convolutional networks lack. However, they are much less parameter efficient and have thus far not achieved competitive performance beyond the CIFAR-10 [156] level, let alone the ImageNet [245] level [229]. Methods like these may offer excellent inspiration for future work, but if they fail to be tractable for large models, they will be of limited direct value for practical interpretability. We urge researchers to detail computational requirements and test the scalability of their methods.

Interpretability techniques generate hypotheses – not conclusions. Producing merely-plausible explanations is insufficient. Evaluating validity and uncertainty are key. Mistaking hypotheses for conclusions is a pervasive problem in the interpretability literature [178], [200], [243]. Consider the goal of explaining a particular neuron. There exist several methods to do so (Section III). However, if such an approach suggests that the neuron has a particular role, this does not offer any guarantee that this explanation is complete and faithful to its true function. Often, very plausible-seeming explanations do not pass simple sanity checks [4] or are very easy to find counterexamples for [35], [128], [221]. A great number of works in interpretability have failed to go beyond simply inspecting the results of a method. More care is needed. Interpretability techniques can only be evaluated to the extent that they help users make testable predictions. They can only genuinely be useful inasmuch as those predictions validate. And the validity of an interpretation is only granted on the distribution of data for which validating tests were conducted – extrapolating interpretations is risky (e.g., [35]). Developing specific methods for evaluating interpretability techniques is discussed later in Section VII.

In addition to validity, it is important to quantify uncertainty. Ideally, interpretations should be paired with confidence estimates. How to measure certainty depends on the method at hand, but some approaches have been used such as supervising explanations (e.g., [123]), conducting multiple trials (e.g., [221]), comparisons to random baselines (e.g., [127], [237]), comparisons to other simple methods [4], or searching for cases in which an interpretation fails (e.g., [25], [35], [128]).

Cherry-picking is harmful and pervasive. Evaluation of methods should not fixate on best-case performance. Due to the inherent difficulty of interpreting DNNs, many works in the literature showcase individual, highly successful applications of their method, often in toy models. This can be useful for providing illustrative examples or specific insights. But the evaluation of interpretability techniques should not be biased toward their best-case performance. One hazard of doing this...
could be from overestimating the value of techniques. And in fact, some works have found that certain methods only tend to perform well on a fraction of examples (e.g., [25], [35], [45], [48], [85], [127], [181], [182], [196], [197], [226], [289]).

Another harm of cherry-picking might come from biasing progress in interpretability toward methods that fail to explain complex subprocesses. Some methods are better equipped for this than others. For example, attributing a feature’s representation to a linear combination of neurons is strictly more general than attributing it to a single neuron. It is likely that some kinds of features or computations in DNNs are more naturally human-understandable than others, so methods that are only useful for explaining simple subprocesses may be poorly-equipped for studying networks in general.

Works should evaluate how their techniques perform on randomly or adversarially sampled tasks. For example, a work on characterizing neural circuits should not focus only on presenting results from circuits that were particularly amenable to interpretation. It should also aim to explain the role of randomly or adversarially sampled neurons inside of circuits or find circuits that can explain how the network computes randomly or adversarially selected subtasks. If a method like this only succeeds in limited cases, this should be explicitly stated.

Ideally, progress in interpretability should neither decrease performance in general nor increase certain risky capabilities. On one hand, interpretable AI techniques should maintain competitiveness. It is key to avoid costs such as degraded task performance, increases in bias, higher compute demands, or difficulty to use in modern deep learning frameworks. Competitive shortcomings like these could lead to “value erosion” [67] in which safer, more interpretable AI practices are not adopted in favor of more competitive approaches.

On the other hand, certain types of performance improvements from interpretability research may also be undesirable. Interpretability work should also not lead to increased capabilities if they make safety-related oversight more difficult. For example, advances toward general intelligence may lead to serious harm if not managed appropriately [39], [59], [209], [227], [246], [277]. One risky possibility is if interpretability is a byproduct of increased general capabilities. For example, large language models can often be prompted to “explain” their reasoning, but only as a result of having advanced, broad-domain abilities. Another way for this to occur is if interpretability leads to advancements in capabilities via basic model insights. From the perspective of avoiding risks from advanced AI systems, neither of these is ideal. A focus on improving interpretability techniques without commensurate increases in capabilities offers the best chance of preventing advancements in AI from outpacing our ability for effective oversight. From this perspective, we argue that improvements in safety rather than capabilities should be the principal goal for future work in interpretability.

VII. Future Work

The connections between interpretability, modularity, adversarial robustness, continual learning, network compression, and semblance to the human visual system should be better understood. One of the most striking findings of modern interpretability work is its connections with other paradigms in deep learning. One of the central goals of this survey has been to highlight these connections (see Section VI). Currently, the intersections in the literature between interpretability and these other areas are relatively sparse. Moving forward, an interdisciplinary understanding of interpretability may lead to insights and advancements spanning multiple domains.

Scaling requires efficient human oversight. Many explanations obtained by state of the art interpretability techniques have involved a degree of human experimentation and creativity in the loop. In some cases, many hours of meticulous effort from experts have been required to explain models or subnetworks performing very simple tasks (e.g., [45], [210]). But if the goal is to obtain a thorough understanding of large systems, human involvement must be efficient. Ideally, humans should be used for screening interpretations instead of generating them. Solutions can include using active learning (e.g., [103]), weak supervision (e.g., [34]), implicit supervision using proxy models trained on human-labeled data (e.g., [49], [51]), and/or rigorous statistical analysis of proxies (e.g., [127], [326]) to reduce the need for human involvement. Toward this end, obtaining additional high-quality datasets (e.g., [25]) with richly-labeled samples may be valuable.

Focus on discovering novel behaviors – not just analyzing them. Many existing methods are only well-equipped to study how models behave in limited settings. For example, any interpretability method that relies on a dataset is limited to characterizing the model’s behavior on that data distribution. But ideally, methods should not be limited to a given dataset or to studying potential failures when the failure modes are already known. For instance, an important practical problem is the detection of offensive or toxic speech, but no dataset contains examples of all types of offensive sentences, and having a human hand-specify a function to perfectly identify offensive from inoffensive speech is intractable. Humans can, however, usually identify offensiveness when they see it with ease.

This highlights a need for techniques that allow a user to discover failures that may not be in a typical dataset or easy to think of in advance. This represents one of the unique potential benefits of interpretability methods compared to other ways of evaluating models such as test performance. Toward this end, some inner interpretability methods that generate abstract understandings of subnetworks have proven to be useful (e.g., [68], [125], [197], [208], [257]). However, methods based on synthesizing adversarial examples may offer a particularly general approach for discovering novel failure modes (e.g., [49], [51], [116], [292]).
Interpretability work may help better understand convergent learning of representations. Some works have hypothesized that similar features or concepts tend to occur across different model instances or architectures \cite{221, 300}. Better understanding the extent to which systems learn similar concepts would lead to a more basic understanding of their representations and how interpretable we should expect them to be. If these hypotheses are true, interpreting one model in depth may be much more likely to lead to generalizable insights. Continued work on measuring representational similarity between neural networks (see Section V-G) may be well-suited for making progress toward this goal.

“Mechanistic interpretability” and “microscope AI” are ambitious but potentially very valuable goals. One direction for interpretability research is mechanistic interpretability, which aims to gain an algorithmic-level understanding of a DNN’s computations. This can be operationalized as converting the DNN into some form of human-understandable pseudocode \cite{95}. This is related to the goal of microscope AI, which refers to gaining domain insights by thoroughly interpreting high-performing AI systems \cite{133}. These capabilities would have advantages, including predicting counterfactual behaviours and reverse engineering models. Thus far, there have been a limited number of attempts towards this goal that have had some success by using small models, simple tasks, and meticulous effort from human experts \cite{45, 83, 210, 286}. Future work in this direction may benefit by using techniques from program synthesis and analysis to automate the generation and validation of hypotheses.

Detecting deception and eliciting latent knowledge may be valuable for advanced systems. A system is deceptive if it passes false or incomplete information along some communication channel (e.g., to a human), despite having the capability to pass true and complete information. Relatedly, latent knowledge \cite{60} is something that a system “knows” but shows no signs of knowing. For example, a language model might babble a common misconception like “bats are blind” in some contexts despite having the “knowledge” that this is false. Hidden knowledge like this may lead to deceptive behavior. As an example, \cite{60} discusses a system that intentionally and deceptively manipulates the observations that a human sees for monitoring it. In this case, knowledge about the true nature of the observations is latent.

Being able to characterize deceptive behavior and latent knowledge has clear implications for safer highly intelligent AI by allowing humans to know when a model may be untrustworthy. But this may be difficult for several reasons including that (1) by definition, deceptive behavior and latent knowledge cannot be determined by observing the model’s deployment behavior alone, (2) any mismatches between the features/concepts used by humans and the model require a method for ontology translation, and (3) it is unclear the extent to which a human can interpret an AI system that is superhuman on some task. However, inner interpretability methods offer a unique approach to these challenges via scrutinizing parts of the model’s computational graph that may process latent knowledge. Probing has shown potential for this \cite{42}.

Rigorous benchmarks are needed. Ideally, they should measure how helpful methods are for producing useful insights that have relevance to engineers. These could involve rediscovering known flaws in networks. Interpretability work on DNNs is done with numerous techniques, not all of which have the same end goal. For example, some methods aim to explain how a DNN handles a single input while others are aimed at a more generalizable understanding of it. For these reasons, plus the rapid development of techniques, widely-accepted benchmarks for interpretability do not yet exist. This may be a limitation for further progress. A well-known example of a benchmark’s success at driving immense progress is how ImageNet \cite{245} invigorated work in supervised image classification.

The weakest form of evaluation for an interpretability method is by its ability to merely suggest a particular characterization. For example, if feature synthesis is used to visualize a neuron, having a human look at the visualization and say “this looks like X” is a fraught basis for concluding that the neuron is an X-detector. This would be to conflate a hypothesis with a conclusion. A somewhat more rigorous approach to evaluation is to make a simple testable prediction and validate it. For example, suppose the hypothesized X-detector activates more reliably for inputs that contain X than ones that do not. Another example is if the use of some method improves some quantitative proxy for interpretability (e.g., \cite{127}). This type of approach is valuable but still not ideal.

The end goal for interpretability tools should be to provide valid and useful insights, so methods for evaluating them ought to measure their ability to guide humans in doing useful things with models. In other words, interpretability tools should be useful for engineers, particularly ones who want to diagnose and debug models. Some works have made progress toward this. Examples include designing novel adversaries (e.g., \cite{42, 47, 49, 51, 107, 125, 135, 137, 164, 208, 302, 327}), manually editing a network to repurpose it or induce a predictable change in behavior (e.g., \cite{27, 68, 111, 197, 303}), or reverse-engineering a system using interpretability techniques (e.g., \cite{45, 84, 210}).

One tractable approach for benchmarking suggested by \cite{134} would be to evaluate interpretability techniques by their ability to help a human find flaws that an adversary has implanted in a model. Judging techniques by how well they help humans rediscover these flaws would offer a much more direct measure of their practical usefulness than ad hoc approaches to evaluation. Related techniques for feature attribution methods have been argued for \cite{129} and used \cite{5, 23, 74} but have not yet popularized. Competitions for implanting and rediscovering flaws (e.g., \cite{193}) hosted at well-known venues or platforms may be a useful way to drive progress in both techniques and benchmarking.

Combining techniques may lead to better results. Inter-
pretability techniques can often be combined. For example, almost any intrinsic method could be used with almost any post hoc one. However, the large majority of work in interpretability focuses on studying them individually. Studying the interplay between methods is relatively unexplored. Some works have identified useful synergies (e.g., [83], [303]). But to the best of our knowledge, there are no works dedicated to thoroughly studying interactions between different methods. We hope that new baselines and increased demand for rigorously interpretable systems will further incentivize results-oriented interpretability work.

Consider an example. The ImageNet benchmark was very effective at advancing the state of the art on image classification performance in the 2010s. Over this time period, improvements in classification performance were not due to single techniques, but a combination of breakthroughs – batch normalization, residual connections, inception modules, deeper architectures, etc. Similarly, we should not expect to best advance capabilities related to interpretability without a combination of methods.

Applying interpretability techniques for debugging and debiasing in the wild. Working to apply interpretability tools to find issues with real-world models (e.g., [61]) both helps for discovering issues in consequential applications and to test methods to see which ones may be the most practically useful. In doing so, researchers should be critical of the ethical frameworks used in machine learning and particularly how they may diverge from the interests of people—particularly from disadvantaged groups—who may be the most adversely affected by these technologies [31].

Growing the field of interpretability. Many ethical or safety concerns with AI systems can be reduced via tools to better understand how models make decisions and how they may fail. As a result, we argue that instead of being a separate interest, interpretability should be seen as a requirement for systems that are deployed in important settings. As discussed above, a compelling path forward is via benchmarking and thorough studying interactions between different methods.

A paradigm shift toward engineering. Currently, interpretability research produces few tools and insights that are useful in the real world. Being able to rigorously study the solutions learned by DNNs seems to have important potential for making DNNs safer, but they are currently rarely used for evaluation or engineering applications. As discussed above, works in the literature too often treat hypotheses as conclusions [178], [200], [243] and fail to connect a method to useful applications. Some amount of exploratory work is clearly valuable for generating insights, and it should continue. But the field has yet to produce many methods that are competitive in real applications. Motivations for interpretability work are “diverse and discordant” [178] and the term itself, as used in the literature, “lacks precise meanings when applied to algorithms” [155]. We join with [79], [243], [200], and [155] in calling for grounding interpretability in meaningful applications. If interpretability tools are ultimately meant to help engineers diagnose and debug DNNs, the field should design and evaluate methods based on this.

We argue that moving forward, the most pressing change that is needed in the field is a focus on producing tools that are useful to engineers. To better realize the potential of interpretability work for human-aligned AI, a more deliberate, interdisciplinary, and application-focused field will be important. It will be valuable to have more research that emphasizes diagnostics, debugging, adversaries, benchmarking, and leveraging useful combinations of different interpretability tools.

Acknowledgements

We thank Davis Brown, Erik Jenner, Jan Kirchner, Max Lamparth, Ole Jørgensen, Florian Dorner, and Peter Hase for their feedback. Tilman Räuber is supported by the Long-Term Future Fund.

References

[1] Abubakar Abid, Mert Yükselgonul, and James Zou. Meaningfully debugging model mistakes using conceptual counterfactual explanations. In International Conference on Machine Learning, pages 66–88. PMLR, 2022.
[2] Samira Ahmar and Willem Zuidema. Quantifying attention flow in transformers. arXiv preprint arXiv:2005.06028, 2020.
[3] Amina Adadi and Mohammed Berrada. Peeking inside the black-box: a survey on explainable artificial intelligence (xai). IEEE access, 6:52138–52160, 2018.
[4] Julius Adebayo, Justin Gilmer, Michael Muelly, Ian Goodfellow, Moritz Hardt, and Been Kim. Sanity checks for saliency maps. Advances in neural information processing systems, 31, 2018.
[5] Julius Adebayo, Michael Muelly, Ilaria Liccardi, and Been Kim. Debugging tests for model explanations. arXiv preprint arXiv:2011.05429, 2020.
[6] Yossi Adi, Einat Kermany, Yonatan Belinkov, Ofer Lavi, and Yoav Goldberg. Fine-grained analysis of sentence embeddings using auxiliary prediction tasks. arXiv preprint arXiv:1608.04207, 2016.
[7] Atish Agarwala, Abhinavvya Das, Brendan Juba, Rina Panigrahy, Vatsal Sharan, Xin Wang, and Qiuqi Zhang. One network fits all? modular versus monolithic task formulations in neural networks. arXiv preprint arXiv:2103.15261, 2021.
[8] Hongjoon Ahn, Sungmin Cha, Donggyu Lee, and Taesup Moon. Uncertainty-based continual learning with adaptive regularization. Advances in Neural Information Processing Systems, 32, 2019.
[9] Zeynep Akata, Lisa Anne Hendricks, Stephan Alaniz, and Trevor Darrell. Generating post-hoc rationales of deep visual classification decisions. In Explainable and Interpretable Models in Computer Vision and Machine Learning, pages 135–154. Springer, 2018.
[10] Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes. arXiv preprint arXiv:1610.01644, 2016.
[11] Rahaf Aljundi, Klaas Kelchtermans, and Tinne Tuytelaars. Task-free continual learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 11254–11263, 2019.
[12] David Alvarez Melis and Tommi Jaakkola. Towards robust interpretability with self-explaining neural networks. Advances in neural information processing systems, 31, 2018.
[13] Mohammed Amer and Tomás Maud. A review of modularization techniques in artificial neural networks. Artificial Intelligence Review, 52(1):527–561, 2019.
[14] Marco Ancona, Enea Ceolini, Cengiz Öztireli, and Markus Gross. Towards better understanding of gradient-based attribution methods for deep neural networks. arXiv preprint arXiv:1711.06104, 2017.
Sawan Kumar and Partha Talukdar. Nile: Natural language inference with faithful natural language explanations. arXiv preprint arXiv:2001.00689, 2020.

Michael Lepori and R Thomas McCoy. Picking bert’s brain: Probing for linguistic dependencies in contextualized embeddings using representational similarity analysis. In Proceedings of the 28th International Conference on Computational Linguistics, pages 3637–3651, 2020.

Been Z Li, Maxwell Nye, and Iacob Andrias. Implicit representations of meaning in neural language models. arXiv preprint arXiv:2106.0737, 2021.

Jiaoda Li, Ryan Cotterell, and Minmiana Sachan. Probing via prompting, 2022.

Oscar Li, Hao Liu, Chaofan Chen, and Cynthia Rudin. Deep learning for case-based reasoning through prototypes: A neural network that explains its predictions. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 32, 2018.

Xuhong Li, Yves Grandvalet, Rémi Flamary, Nicolas Courty, and Dejing Dou. Representation transfer by optimal transport. arXiv preprint arXiv:2007.06737, 2020.

Yixuan Li, Jason Yosinski, Jeff Clune, Hod Lipson, and John Hopcroft. Convergent learning: Do different neural networks learn the same representations? arXiv preprint arXiv:1511.07543, 2015.

Yuchao Li, Shaohui Lin, Baochang Zhang, Jianzhuan Liu, David Doermann, Yongjian Wu, Feiyue Huang, and Rongrong Ji. Exploiting kernel sparsity and entropy for interpretable cnn compression. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 2800–2809, 2019.

Zhizhong Li and Derek Hoiem. Learning without forgetting. IEEE transactions on pattern analysis and machine intelligence, 40(12):2935–2947, 2017.

Ruofan Liang, Tianlin Li, Longfei Li, Jing Wang, and Quanshi Zhang. Knowledge consistency between neural networks and beyond. arXiv preprint arXiv:1908.01581, 2019.

Adam Dahlgren Lindström, Suna Bensch, Johanna Björklund, and Frank Drewes. Probing multimodal embeddings for linguistic properties: the visual-semantic case. arXiv preprint arXiv:2102.11115, 2021.

Zachary C Lipton. The myths of model interpretability: In machine learning, the concept of interpretability is both important and slippery. Queue, 16(3):31–57, 2018.

Shusen Liu, Tuo Li, Zhimin Li, Vivek Srikumar, Valerio Pascucci, and Peer-Timo Bremer. Visual interrogation of attention-based models through multilingual evaluation. 2015.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In International Conference on Machine Learning, pages 3519–3529. PMLR, 2019.

Maya Krishnan. Against interpretability: a critical examination of the interpretability problem in machine learning. Philosophy & Technology, 33(3):487–502, 2020.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Dmitry Krotov and John J Hopfield. Unsupervised learning by competing hidden units. Proceedings of the National Academy of Sciences, 116(16):7723–7731, 2019.

Abhishek Kumar, Prasanna Sattigeri, and Avinash Balakrishnan. Variational inference of disentangled latent concepts from unlabelled observations. arXiv preprint arXiv:1711.00848, 2017.

I Elizabeth Kumar, Suresh Venkatasubramanian, Carlos Scheidegger, and Sorelle Friedler. Problems with shapley-value-based explanations as feature importance measures. In International Conference on Machine Learning, pages 5491–5500. PMLR, 2020.

Sawan Kumar and Partha Talukdar. Nile: Natural language inference with faithful natural language explanations. arXiv preprint arXiv:2005.12116, 2020.

C-C Jay Kuo, Min Zhang, Siyang Li, Jiali Duan, and Yuern Chen. Interpretable convolutional neural networks via feedforward design. Journal of Visual Communication and Image Representation, 100:1-10, 2021.

Oscar Li, Ankik Mondal, Abhishek Chakraborty, Michael Zazuk, Nina Jacobsen, Daniel Xing, and Ankur Srivastava. A survey on neural trojans. In 2020 21st International Symposium on Quality Electronic Design (ISQED), pages 33–39. IEEE, 2020.

Francesco Locatello, Stefan Bauer, Mario Lucic, Gunnar Raetsch, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem. Challenging common assumptions in the unsupervised learning of disentangled representations. In international conference on machine learning, pages 4114–4124. PMLR, 2019.

Francesco Locatello, Stefan Bauer, Mario Lucic, Gunnar Raetsch, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem. Challenging common assumptions in the unsupervised learning of disentangled representations. In international conference on machine learning, pages 4114–4124. PMLR, 2019.

Francesco Locatello, Stefan Bauer, Mario Lucic, Gunnar Raetsch, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem. Challenging common assumptions in the unsupervised learning of disentangled representations. In international conference on machine learning, pages 4114–4124. PMLR, 2019.

Francesco Locatello, Stefan Bauer, Mario Lucic, Gunnar Raetsch, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem. Challenging common assumptions in the unsupervised learning of disentangled representations. In international conference on machine learning, pages 4114–4124. PMLR, 2019.
Alex Tamkin, Trisha Singh, Davide Giovanardi, and Noah Goodman. Investigating transferability in pretrained language models. arXiv preprint arXiv:2004.14975, 2020.

Shuai Tang, Wesley J Maddox, Charlie Dickens, Tom Diethe, and Andreas Damianou. Similarity of neural networks with gradients. arXiv preprint arXiv:2003.11498, 2020.

Norman Tatro, Pin-Yu Chen, Payel Das, Igor Melnyk, Prasanna Sattigeri, and Rongjie Lai. Optimizing mode connectivity via neuron alignment. Advances in Neural Information Processing Systems, 33:15300–15311, 2020.

Max Tegmark. Life 3.0: Being human in the age of artificial intelligence. Vintage, 2017.

Michalis K Titsias, Jonathan Schwarz, Alexander G de G Matthews, Razvan Pascanu, and Yee Whye Teh. Functional regularisation for continual learning with gaussian processes. arXiv preprint arXiv:1901.11356, 2019.

Richard Tomsett, Amy Wididcombe, Tianwei Xing, Supriyo Chakraborty, Simon Julier, Prudhvi Gurram, Raghuviree Rao, and Mani Srivastava. Why the failure? how adversarial examples can provide insights for interpretable machine learning. In 2018 21st International Conference on Information Fusion (FUSION), pages 838–845. IEEE, 2018.

Joe Townsend, Theodoros Kasioumis, and Hiroya Inakoshi. Eric: extracting relationships inferred from convolutions. In Proceedings of the Asian Conference on Computer Vision, 2020.

Theodoros Tsiligkaridis and Jay Roberts. Second order optimization for adversarial robustness and interpretability. arXiv preprint arXiv:2009.04923, 2020.

Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy. arXiv preprint arXiv:1805.12152, 2018.

Sunil Vadera and Salem Ameen. Methods for pruning deep neural networks. arXiv preprint arXiv:2011.00241, 2020.

Marco Valentino, Ian Pratt-Hartmann, and André Freitas. Do natural language explanations represent valid logical arguments? verifying entailment in explainable nli gold standards. arXiv preprint arXiv:2005.01974, 2020.

Shikhah Vashisht, Shyam Upadhyay, Gaurav Singh Tomar, and Manaal Faruqui. Attention interpretability across nlp tasks. arXiv preprint arXiv:1909.11218, 2019.

Abhinav Verma, Vijayaraghavan Murali, Rishabh Singh, Pushmeet Kohli, and Swarat Chaudhuri. Programmatically interpretable reinforcement learning. In International Conference on Machine Learning, pages 5045–5054. PMLR, 2018.

Jesse Vig. A multiscale visualization of attention in the transformer. arXiv preprint arXiv:1906.05714, 2019.

Jesse Vig, Sebastian Gehrmann, Sharon Qian, Daniel Jie Wang, Ghulam Mubashar Hassan, and Naveed Akhtar. A survey of neural trojan attacks and defenses in deep learning. arXiv preprint arXiv:2105.01974, 2021.

Quanshi Zhang, Ying Nian Wu, and Song-Chun Zhu. Visual interpretability for deep neural networks by identifying critical data routing paths. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 8906–8914, 2018.

Quanshi Zhang, Xin Wang, Ruiming Cao, Ying Nian Wu, Feng Shi, and Song-Chun Zhu. Explanatory graphs for cnns. arXiv preprint arXiv:1708.01547, 2017.

Eli Yaffe, Yaron Singer, and Stuart Shieber. Investigating gender bias in neural language models using causal mediation analysis. Advances in Neural Information Processing Systems, 33:12388–12401, 2020.

Yang Yang, Hang Su, Bo Zhang, and Xiaolin Hu. Interpret neural networks with augmented memory. arXiv preprint arXiv:2205.15480, 2022.

Kaixuan Yao, Feilong Cao, Yee Leung, and Jiye Liang. Deep neural network compression through interpretability-based filter pruning. Pattern Recognition, 119:108056, 2021.

Seul-Ki Yeom, Philipp Seegerer, Sebastian Lupashcin, Alexander Binder, Simon Wiedemann, Klaus-Robert Müller, and Wojciech Samek. Pruning by explaining: A novel criterion for deep neural network pruning. Pattern Recognition, 115:107899, 2021.

Jaehong Yoon, Eunho Yang, Jeongtae Lee, and Sung Ju Hwang. Lifelong learning with dynamically expandable networks. arXiv preprint arXiv:1708.01547, 2017.

Chao Ye, Meng Yukegkonul, Maggie Wang, and James Zou. Post-hoc concept bottleneck models. arXiv preprint arXiv:2205.15480, 2022.

Friedemann Zenke, Ben Poole, and Surya Ganguli. Continuing learning through synaptic intelligence. In International Conference on Machine Learning, pages 3987–3995. PMLR, 2017.

Mengmi Zhang, Rohil Badkundi, Morgan B Talbot, Rushikesh Zawar, and Gabriel Kreiman. Hypothesis-driven online video stream learning with augmented memory. arXiv e-prints, pages arXiv–2104, 2021.

Quan-shi Zhang and Song-Chun Zhu. Visual interpretability for deep learning: a survey. Frontiers of Information Technology & Electronic Engineering, 19(1):27–39, 2018.

Quanshi Zhang, Xin Wang, Ruiming Cao, Ying Nian Wu, Feng Shi, and Song-Chun Zhu. Explanatory graphs for cnns. arXiv preprint arXiv:1812.07997, 2018.

Quanshi Zhang, Xin Wang, Ruiming Cao, Ying Nian Wu, Feng Shi, and Song-Chun Zhu. Extraction of an explanatory graph to interpret a cnn. IEEE transactions on pattern analysis and machine intelligence, 43(11):3863–3877, 2020.

Quanshi Zhang, Ying Nian Wu, and Song-Chun Zhu. Interpretable convolutional neural networks. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 8827–8836, 2018.
uncertainty in lime explanations. *arXiv preprint arXiv:1904.12991*, 2019.

[319] Mengjie Zhao, Tao Lin, Fei Mi, Martin Jaggi, and Hinrich Schütze. Masking as an efficient alternative to finetuning for pretrained language models. *arXiv preprint arXiv:2004.12406*, 2020.

[320] Xinyan Zhao and VG VydIslaran. Lirex: Augmenting language inference with relevant explanation. *arXiv preprint arXiv:2012.09157*, 2020.

[321] Haizhong Zheng, Earlene Fernandes, and Atul Prakash. Analyzing the interpretability robustness of self-explaining models. *arXiv preprint arXiv:1905.12429*, 2019.

[322] Songzhu Zheng, Yikai Zhang, Hubert Wagner, Mayank Goswami, and Chao Chen. Topological detection of trojaned neural networks. *Advances in Neural Information Processing Systems*, 34:17258–17272, 2021.

[323] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Object detectors emerge in deep scene cnns. *arXiv preprint arXiv:1412.6856*, 2014.

[324] Bolei Zhou, Yiyou Sun, David Bau, and Antonio Torralba. Interpretable basis decomposition for visual explanation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 119–134, 2018.

[325] Bolei Zhou, Yiyou Sun, David Bau, and Antonio Torralba. Revisiting the importance of individual units in cnns via ablation. *arXiv preprint arXiv:1806.02891*, 2018.

[326] Yilun Zhou, Marco Tulio Ribeiro, and Julie Shah. Exsum: From local explanations to model understanding. *arXiv preprint arXiv:2205.00130*, 2022.

[327] Daniel M Ziegler, Seraphina Nix, Lawrence Chan, Tim Bauman, Peter Schmidt-Nielsen, Tao Lin, Adam Scherlis, Noa Nabeshima, Ben Weinstein-Raun, Daniel de Haas, et al. Adversarial training for high-stakes reliability. *arXiv preprint arXiv:2205.01663*, 2022.