Demysifying Deep Neural Networks Through Interpretation: A Survey

Giang Dao & Minwoo Lee
Department of Computer Science
University of North Carolina at Charlotte
gdao@uncc.edu, minwoo.lee@uncc.edu

Abstract

Modern deep learning algorithms tend to optimize an objective metric, such as minimize a cross entropy loss on a training dataset, to be able to learn. The problem is that the single metric is an incomplete description of the real world tasks. The single metric cannot explain why the algorithm learn. When an erroneous happens, the lack of interpretability causes a hardness of understanding and fixing the error. Recently, there are works done to tackle the problem of interpretability to provide insights into neural networks behavior and thought process. The works are important to identify potential bias and to ensure algorithm fairness as well as expected performance.

1 Introduction

Deep neural networks have shown a broad range of success in multiple domains including image recognition tasks, natural language tasks, recommendation systems, security, and data science Pouyanfar et al. [2018]. Despite the success, there is a general mistrust about the system results. Neural network prediction can be unreliable and contain biases Geman et al. [1992]. Deep neural networks are easy to be fooled to output wrong predictions in image classification task Nguyen et al. [2015]. Not only in the image recognition task, adversarial attack can be applied in natural language processing tasks Jia and Liang [2017]. The problem becomes worse in security applications to secure against trojan attacks Liu et al. [2017]. Even though there have been discrimination methods developed to defend such adversarial attacks Madry et al. [2017]; Carlini and Wagner [2017], the unintuitive errors, which cannot fool human perception, still remain as a big problem in neural networks. The need for demystifying neural networks has arisen to understand the neural network's unexpected behavior.

With the demand for understanding neural networks, some existing deployed systems are required to be interpretable by regulations. The European Union has adopted the General Data Protection Regulation (GDPR) which became law in May 2018. The GDPR stipulated "a right of interpretability" in the clauses on automated decision-making. The inequality or bias, the safety of human users, industrial liability, and ethics are endangered without establishing trustworthiness based on interpretation (thus understanding) of the systems. Therefore, the demand for interpretability created a new line of research to understand *why* a neural network makes a decision. Reflecting on the needs, the number of neural networks interpretability research has been growing fast since AlexNet Krizhevsky et al. [2012] came out in 2012[1].

In this survey, we review existing study to interpret neural networks to help human understand what a neural network has learned and why a decision is made. For this, we define interpretability, restate

[1] Google Scholar found about 18,500 results of 'neural networks interpretability' from 2012 to 2020 (accessed in Feb. 17, 2020).

the significance, and compile them with a high-level categorization in Section 2. We review the interpretation methods in each category in Section 4. In Section 5, we highlight different ways to evaluate a interpretable neural network framework. We discuss new challenges and conclude in Section 6, draw conclusion in Section 7, and propose the future directions for the field in Section 8.

2 Definition & Importance of Neural Network Interpretability

Interpretation is defined as *the action of explaining the meaning of something*[2]. In the context of this paper, we slightly modify the definition of interpretation as *the action of explaining what the neural networks have learned in understandable terms to human* that anyone without deep knowledge in neural networks can understand why the neural networks make a decision. The understandable terms are tied to knowledge, cognition, and bias of humans. The interpretable system needs to provide information in a simple and meaningful manner.

Why is it important to understand or interpret a neural network model when it is performing well on a test dataset? Most of the time we don't certainly know if the dataset is generalized or covering all possibilities. For example, self-driving car technology needs to learn a lot of accident cases to be able to generalize and perform well in the real world situation, but there can be infinite possibilities of cases that are impossible to fully collect or synthesize. A correct prediction should be derived from a proper understanding of the original problem. Therefore, we need to explore and understand why a neural network model makes certain decisions. Knowing 'why' helps us learn about the problem, the data, and the reasons why the model might succeed or fail.

Doshi and Kim Doshi-Velez and Kim [2017] provided reasons that drive the demand for interpretability:

1. There is a big wave of change from qualitative to quantitative and toward deep neural networks with the increasing amount of data. In order to gain *scientific understanding*, we need to make the model as the source of knowledge instead of the data.
2. Deploying neural networks model for automation has been increasing in real world practices. Therefore, monitoring the *safety* of the model is necessary to ensure the model operates without harming the environment.
3. Despite the complexity of neural networks, encoding fairness into neural networks might be too abstract. Microsoft has announced the bias and discrimination problem of facial recognition[3]. Ensuring the model *ethics* can increase trust from users.
4. The neural networks may optimize an *incomplete objective*. Most of the deep neural networks minimize cross-entropy loss for classification task. However, the cross-entropy loss is known to be vulnerable to adversarial attacks Nar et al. [2019].

3 Related Work

Some previous papers have surveyed on interpreting machine learning in different domains. The trend in interpretable artificial intelligence in human-computer interface research by reviewing a large number of publication records Abdul et al. [2018]. Reviewing a large number of articles, the authors emphasized the lack of methods being applied to interpretability and encouraged a broader interpretability methods to current research. The interpretation of a black box model has been surveyed Guidotti et al. [2018]. The authors divided the interpretable methods based on the types of problems: interpreting a black box model, interpreting black box outcomes, inspecting a black box, and designing a transparent box model. The authors acknowledge that some approaches have attempted to tackle interpretability problems but some important scientific questions still remain unanswered.

From analyzing the related works, we recognize that researchers have been focusing on interpreting deep neural network model in the modern works because deep neural network uses a lot of parameters and operations to derive a prediction with a low error rate. For example, ResNet He et al. [2016] holds around 50 million parameters and performs around 100 billion operations to classify an

[2]Accessed from Google dictionary in Feb. 17, 2020.

[3]https://blogs.microsoft.com/on-the-issues/2018/12/06/facial-recognition-its-time-for-action/

image Canziani et al. [2016]. This complex system makes the neural network difficult to interpret. Therefore, interpretation of neural networks becomes an exciting area of research. With the challenge in interpretibility of neural networks, we focus on surveying methods of how to interpret a neural network model to fully understand why the neural network makes its decision. We go deeper and highlight different methods with their advantages and disadvantages in the sub-fields of neural networks interpretation in the next sections. We also provide an overview of how we can evaluate an interpretation system and propose new challenges in the interpretation field.

4 Approaches

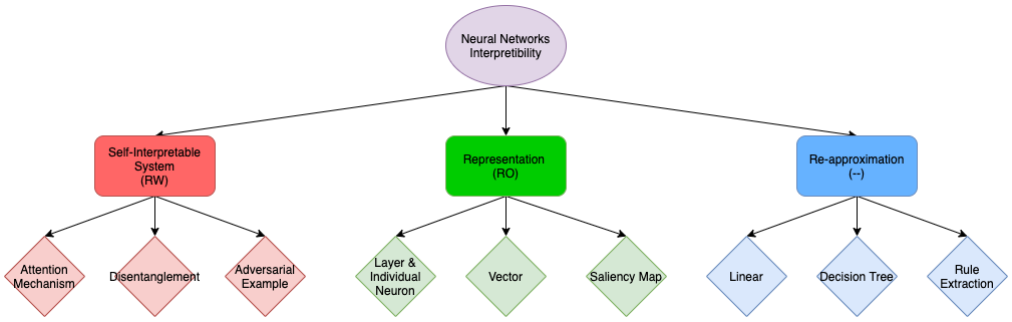


Figure 1: Splitting neural networks interpretability approaches into sub-categories and its methods of interpretation. We denote the required the accessibility to the model for interpretation: RW means read/write, RO means read-only, and – means no access requirement.

Fig. 1 depicts a high-level view of interpretability research in neural networks. There exists three main approaches to interpret neural networks. We categorize these three main branches by how much accessibility and permission a method needs to have to interpret a neural network model: requiring full access and modification (*Self-interpretable System*), requiring full access without modification (*Representation Analysis*), or requiring no access or modification privilege (*Re-approximation*) as follows:

1. *Self-Interpretable System* is a method that designs a neural network in a way that it can somewhat explain its decision. This approach requires to fully access the model to be able to modify and architect the neural network.

2. *Representation Analysis* is an approach to understand individual sub-system inside the neural network by simply observing the weights and gradient updates. As it is not necessary to modify the neural network model, only full read access is enough for methods in this category.

3. *Re-approximation* uses genuinely interpretable models to understand the neural networks. This approach does not read or modify the model to understand it. It simply monitors input and output of the model and re-approximates for interpretation.

We compiled all approaches and methods that we reviewed with advantages and disadvantages in Table 1.

We split the interpretibility system into three main branches because of the user accessibility to the neural networks. For example, a neural network's creator can use all of the three branches to explain their model which they can modify the model to have better understanding. Users, who download models online for their application, cannot modify the model but can access the internal to understand the model's weights. Application programing interference (API) users, who call a neural networks API to get a result, can only understand the model by approximating it.

We summarize the splitted approaches and the methods with each own advantage and disadvantages in 1.

Approach	Method	Advantage	Disadvantage
Self-Interpretable System (RW)	Attention Mechanism	• Easy to interpret which input information is relevant to output	• Create more parameters for training • Model design is required
	Disentanglement	• Easy to understand from low dimension	• Limited knowledge in feature roles without examining
	Adversarial Example	• Understand neural network's vulnerability	• Hard to understand the meaning of the added noise
Representation Analysis (RO)	Layers & Individual Neurons Analysis	• Visualizing what features have been learned	• Too many visualizations for analyzing one sample
	Vectors Analysis	• Easy to understand sample distribution from visualization	• Current methods are not good enough • Might not explain why and how each data is clustered
	Saliency Map	• Highlight important input information	• Noisy interpretation of input features
Re-approximation (–)	Linear Approximation	• Simple to implement	• Slow to train for a single sample $\rightarrow$ hard to scale • Lower performance to neural network
	Decision Tree	• Easy to follow tree to get answer and understand process	• Complex tree structure with deep networks $\rightarrow$ hard to scale
	Rules Extraction	• Straight forward to analyze a sample	• Complex rules are hard to keep track $\rightarrow$ hard to scale

Table 1: Full approaches and methods with their advantages and disadvantages.

4.1 Self-Interpretable System

Several efforts have been taken to design a neural network model that is able to interpret its decisions after well-trained. There are three main methods to design an interpretable neural networks model: *attention mechanism*, *disentanglement learning*, and *adversarial examples*. An output of a specifically designed layer in the self-interpretable system can be easily understood because it is represented as a probability distribution in attention mechanism, vector space in disentanglement learning, and sample representation in adversarial examples.

4.1.1 Attention Mechanism

Attention mechanism attempts to understand the relationship between information. Attention in deep learning is a vector of importance weights which shows how an input element correlates to target output. Attention weights can be formulated as a probability distribution of correlation between a target with other sources. A higher probability results from a higher correlation between a target and a source. There are two types of attention mechanisms: hard-attention and soft-attention. Hard-attention strictly enforce attention weights to either 0 for non-correlated or 1 for correlated (Bernoulli distributions). Soft-attention represents attention weights with more flexible probability distributions. With the flexibility, soft-attention recently dominates over hard-attention in most of the applications. An example of computing soft-attention weights is using softmax function to compute the correlation between a target with other sources:

$$\alpha_{ts} = \frac{exp(score(h_t, \bar{h}_s))}{\sum_{s'=1}^{S} exp(score(h_t, \bar{h}_s))}.$$

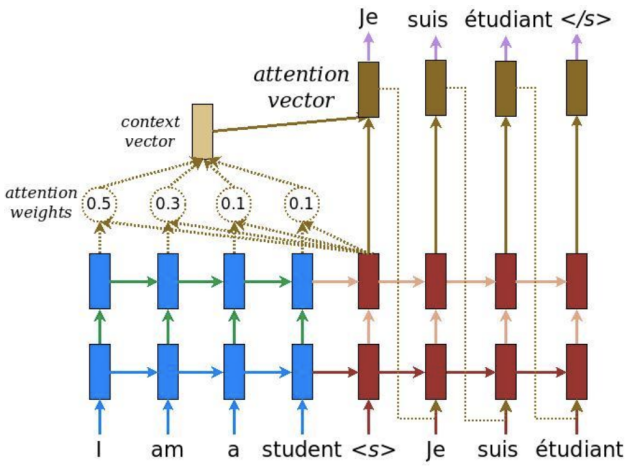


Figure 2: An example of translating from English to French showing the attention weights of the word "Je" in French has highest correlation probability with the word "I" in English using soft-attention from Luong et al. [2015] method.

Attention mechanism has achieved remarkable success in natural language translation with different score functions as well as other optimization tricks Graves et al. [2014]; Bahdanau et al. [2014]; Luong et al. [2015]; Canziani et al. [2016]. A TensorFlow tutorial[4] shows an example of attention mechanism in a machine translation task in Fig. 2. Not only showing the capability of self-interpretability in natural language processing tasks, attention mechanisms can also be designed to interpret neural network decision by looking at the attention pixels in different tasks: image classification Xiao et al. [2015]; Wang et al. [2017], image segmentation Chen et al. [2016a], and image captioning Xu et al. [2015]; Lu et al. [2016, 2017]; Anderson et al. [2018]. The neural networks error prediction can be interpreted by attention mechanism shown in Fig. 3.

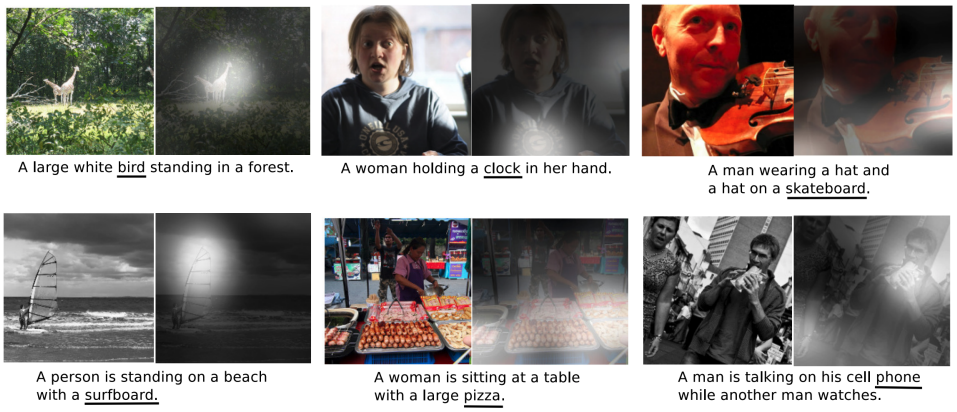


Figure 3: Visual examples interpreting why image captioning produces error by looking at the attention region proposed by Xu et al. [2015].

Even though attention units reveal interpretable information, they are hardly evaluated because of the robustness in the comparison process. Therefore, Das et al. Das et al. [2017] has created human attention datasets to compare the attention between neural networks and humans to see if they look at the same regions when making a decision. To enforce the neural networks to look at the same region as human and to have similar human behavior, a method to train attention mechanisms explicitly through supervised learning with the attention datasets by constraining the machine attention to be similar to human attention in the loss function was proposed Ross et al. [2017].

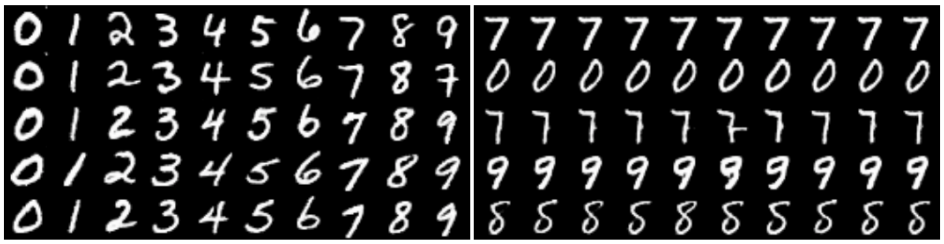
[4]https://www.tensorflow.org/tutorials/text/nmt_with_attention

Despite the advantage of easy to interpret which input information is highly correlated to a target output, the attention mechanism caries two disadvantages. One is creating more parameters for training with more complex computation graph. The second disadvantage is that it requires the full accessibility to the model.

4.1.2 Disentanglement Learning

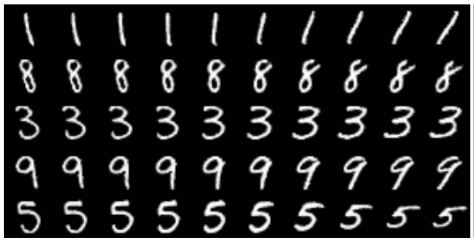
Disentanglement learning is a method to understand a high level concepts from low level information. Disentanglement learning is a learning process that learns disentangled representations in lower dimensional latent vector space where each latent unit represents a meaningful and independent factor of variation. For example, an image contains a black hair man will have representation of gender: male, and hair color: black encoded in the latent vector space. A disentangled representation can be learned explicitly from training a deep neural network. There are two different ways that can be considered to learn disentangled representation. The disentangled representation can be learned through generative adversarial networks (GAN) Goodfellow et al. [2014a] and variational autoencoder (VAE) Kingma and Welling [2013].

GAN contains 2 main parts (generator and discriminator) which learns to map a vector representation into higher dimensional data. The generator takes a vector representation to generate a data point. The vector representation usually has lower dimension than the generated data point. The discriminator takes a data point and outputs true if the data is real and false if the data is generated. After the learning process, the vector representation usually provides high level information of the data. InfoGAN Chen et al. [2016b] is a scalable unsupervised approach to increase the disentanglement by maximizing the mutual information between subsets of latent variables and observations within the generative adversarial network. Auxiliary classifier GAN Odena et al. [2017] extends InfoGAN by controlling a latent unit with actual categorical classes. This is simply adding a controllable disentangled unit with a known independent factor. Fig. 4 shows the output is varied when tuning only one latent unit of InfoGAN.



(a) Varying c_1 on InfoGAN (Digit type)

(b) Varying c_1 on regular GAN (No clear meaning)



(c) Varying c_2 from -2 to 2 on InfoGAN (Rotation)

(d) Varying c_3 from -2 to 2 on InfoGAN (Width)

Figure 4: Interpretation result of InfoGAN Chen et al. [2016b] by adjusting different parameters in latent dimension with different effects on the produced images. The figure shows the first latent unit corresponds for different digit type (a), the second latent unit handles the rotation of the digit (c), and the third latent unit manages the width of the digit (d). The authors also compared with the original GAN to shows the interpretibility by manipulating latent dimension (b).

Instead of learning to map a vector representation into a data point, VAE learns to map a data point to a lower vector representation. VAE minimizes a loss function:

$$\mathcal{L}(\theta, \phi, x) = \frac{1}{L} \sum_{l=1}^{L} (log p_\theta(x|z^l)) - D_{KL}(q_\phi(z|x)||p_\theta(z)),$$

has been shown as a promising direction to explicitly learn disentanglement latent units with β-VAE Higgins et al. [2016]. β-VAE magnifies the KL divergence term with a factor $\beta > 1$:

$$\mathcal{L}(\theta, \phi, x) = \frac{1}{L}\sum_{l=1}^{L}(logp_\theta(x|z^l)) - \beta D_{KL}(q_\phi(z|x)||p_\theta(z)),$$

Further experiment Burgess et al. [2018] showed the disentangled and proposed modification of KL divergence term in the loss function to get improvement in reconstruction:

$$\mathcal{L}(\theta, \phi, x) = \frac{1}{L}\sum_{l=1}^{L}(logp_\theta(x|z^l)) - \beta|D_{KL}(q_\phi(z|x)||p_\theta(z)) - C|,$$

with C is a gradually increasing number to a large enough value to produce good reconstructions. The first term, $\frac{1}{L}\sum_{l=1}^{L}(logp_\theta(x|z^l))$, is an expected negative reconstruction error, while the second term, Kullback-Leibler divergence of approximate posterior from the prior $D_{KL}(q_\phi(z|x)||p_\theta(z))$, acts as a regularizer. The β magnifies the KL divergence term to have better constrain on the prior and the posterior. Since KL divergence term can grow to infinity, the gradually increasing number C makes the term stay numerically computable.

Both GAN and VAE methods can be trained in such a way that each individual latent unit is corresponding to a specific feature. van Steenkiste et al. [2019] observed the disentangle learning leads to a better abstract reasoning. Graph construction (Zhang et al. [2017]) and decision trees (see more in Section 4.3) are additional methods using disentangle latent dimensions. High-level concepts can also be represented by organizing the disentanglement with capsule networks by Sabour et al. [2017]. Disentanglement learning is not only designed for interpretability, it recently shows huge improvement in unsuppervised learning tasks via encoding information (Oord et al. [2018]; Löwe et al. [2019]).

The disentanglement learning has an advantage of low dimensional representation (or interpretation) which is straightforward to understand. However, limited knowledge in the role of each dimension requires manual inspection for interpretation. For example, we cannot know exactly what the first latent unit is representing the digit type in InfoGAN without doing a repeated experiment.

4.1.3 Adversarial Examples

Adversarial examples can be used for interpretation of neural networks bu revealing the vulnerability of the neural networks. An adversarial attack is a method to deceive a neural network model. The main idea is to slightly perturb the input data to get a false prediction from the neural networks model, although the perturbed sample makes no different to human perception. Early work has been proposed Szegedy et al. [2013] to find the perturbation noise by minimizing a loss function:

$$\mathcal{L} = loss(\hat{f}(x + \eta), l) + c \cdot |\eta|,$$

where η is the perturbed noise, l is the desired deceived target label to deceive the neural networks, and c is a constant to balance the original image and the perturbed image. Goodfellow et al. Goodfellow et al. [2014b] proposed a fast gradient method to find η by the gradient of the loss w.r.t to the input data: $\eta = \epsilon \cdot sign(\nabla_x \mathcal{L}(x, l))$. However, the two methods require a lot of pixels to be changed. Yousefzadeh and O'Leary Yousefzadeh and O'Leary [2019] reduced the number of pixels using flip points. It is al possible to deceive a neural network classifier with only one pixel change Su et al. [2019].

Fig. 5 shows how a neural networks can be deceived by changing a digital image. However, it is hard to intentionally modify a digital image when the image is captured by a camera without hacking into a system. A method to print stickers that can fool a neural networks classifier Brown et al. [2017] was designed. Similarly, the usage of 3D printer to print a turtle but is classified as a rifle Athalye et al. [2017] has also implemented.

Differently to the other neural network interpretability methods, adversarial examples focus on interpreting the vulnerability of the neural networks. Through different methods to generate adversarial examples, researchers observe that the neural networks are vulnerable to the adversarial examples with a small noise addition while human perception is not deceived by the adversarial examples. The known and discovered vulnerabilities help to enhance and to strengthen neural network decision

Figure 5: Left images are the original images, middle images are perturbation noise, and the right images are the perturbed images. The upper images are done by Szegedy et al. [2013], and the lower images are done by Goodfellow et al. [2014b]. There is no different in human perception. However, The perturbed images are classified wrong by the neural networks with the desired deceived predictions.

boundaries Miyato et al. [2018]; Douzas and Bacao [2018]. One disadvantage of adversarial example is that the meaning of the added noise is unclear to human perception and why the added noise changes the prediction of the neural network.

4.2 Representation Analysis

Even though there are millions of parameters and billions of computing operations, deep neural networks are internally divided by smaller subcomponents. The subcomponents are *layers & individual neurons*, *vectors*, and *input information*. For example, ResNet50 can be organized into 50 layers, and each layer computes between 64 to 2048 neurons. The final layer of ResNet50 contains a vector of 2048 dimensions. Layer, individual neuron, vector representation, and input information can interpret the decision of the neural networks.

4.2.1 Layers & Individual Neurons Analysis

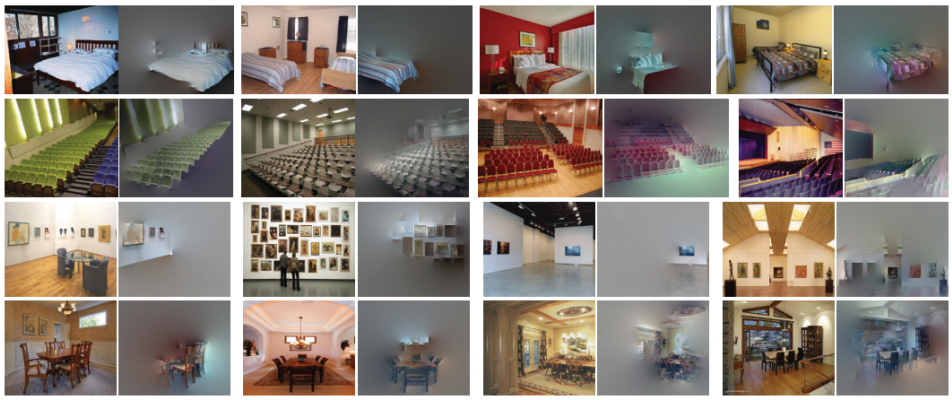


Figure 6: Pair of examples of simplified input information (right) from original input (left) image derived by Zhou et al. [2014] method.

Visualization of layer and individual neurons are helpful to understand which features have been learned. The information flows in neural networks can be subdivided into layers and individual neurons. A single individual neuron can be understood by visualizing the input patterns that maximize the neuron's response. With the neuron's responses visualization, researchers inpteret which information has been learned and passed through different layers and individual neurons of the neural networks.

We can directly visualize each individual neurons to observe the weights. By visualizing and observing each layers of a small neural network, the neural network is shown to learn from simple concepts to high level concepts through each layer Lee et al. [2009]. A neural network model first learns to detect edges, angles, contours, and corners in a different direction at the first layer, object parts at the second layer, and finally object category in the last layer. This sequence consistently happens during training different neural networks on different tasks.

Instead of visualizing neurons directly, researchers found out that the neurons' gradient can also be observed to reveal where important information parts come from. Gradient-based methods, which propagates through different layers and units Simonyan et al. [2013]; Olah et al. [2018], were proposed. The gradient of the layers and units highlights areas in an image which discriminate a given class. An input can also be simplified which only reveals important information Zhou et al. [2014]. Fig. 6 provides examples of original image and simplified images pair. A method to synthesize an input that highly maximizes a desired output neuron using activation maximization Nguyen et al. [2016] by utilizing gradients. For example, the method can synthesize an image of lighter that the neural network classifier would maximize the probability of the lighter. Mordvintsev et al. Mordvintsev et al. [2018] has successfully improved style transfer, which modifies a content image with a style of different image, by maximizing the activation difference of different layers. There is a survey of different methods for visualization of layer representations and diagnosed the representations Zhang and Zhu [2018]. By analyzing individual neurons from a small neural network, Fig. 7 pointed out a strategy of how neural networks learns by visualizing all neurons.

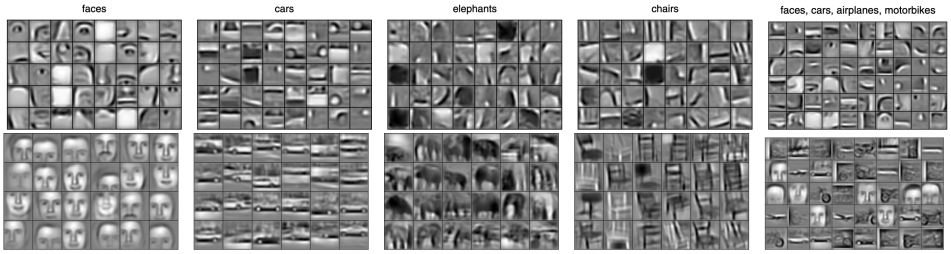


Figure 7: An illustration of what information is learned through different layers of neural networks in different tasks shown in Lee et al. [2009].

Another way to understand a single individual neuron and layers is to qualitatively validate its transferability to different tasks. A framework for quantifying the capacity of neural network transferability was introduced by comparing the generality versus the specificity of neurons in each layer Yosinski et al. [2014]. Network dissection method Bau et al. [2017] measures the ability of individual neurons by evaluating the alignment between individual neurons and a set of semantic concepts. By locating individual neurons to object, part, texture, and color concepts, network dissection can characterize the represented information from the neuron.

There is a possibility of solving the same problem with smaller neural networks in roughly similar architecture. Large neural networks can contain a successful sub-networks without several individual neurons connected. Pruning individual neurons is also an exciting area of research not only in understanding neural networks Frankle and Carbin [2018], but also improving the inference speed of the neural networks through quantization Jacob et al. [2018]. With the increase of complexity of neural network architecture to achieve state-of-the-art results, the number of layers and neurons also increases. More layers and neurons simply mean more human effort in validating more visualization.

4.2.2 Vectors Analysis

Vector representations are taken before applying a linear transformation to the output from a neural network model. However, the vector representation most likely to have more than three dimensions which are hard to be visualized by computer. Vector visualization methods aim to reduce the dimension of the vector to two or three dimensions to be able to visualize by computer. Reducing the vector to two or three dimensions to visualize is an interesting research area. PCA Frey and Pimentel [1978] designs an orthogonal transformation method to convert a set of correlated variables into another set of linearly uncorrelated variables (called principal components). The higher impact principal component has a larger variance. T-distribution stochastic neighbor embedding (t-SNE by Maaten and Hinton Maaten and Hinton [2008]) performs a non-linear dimension reduction for visualization in a low dimensional space of two or three dimensions. t-SNE constructs low dimensional space probability distribution over pairs of high dimensional objects and minimize KL divergence with respect to the locations of the points on the map.

Vector representation visualization methods are well known for helping humans understand high dimensional data. For example, if a neural network performs well in a classification task, the vector

representations need to be clustered together if they have a similar label. In order to ensure the vector representations are clustered, human needs to visualize the vector and validates the assumption, especially in unsupervised learning where no label is given. Both of the methods reduce high dimensional space to lower dimensions (usually two or three) for an easy visualization that helps human understand and validate the neural networks. PCA and t-SNE are widely used by researchers to visualize high dimension information. As we observe Fig. 8, although the t-SNE performs reasonable well to lower the dimensions, there are areas that it does not show full separation.

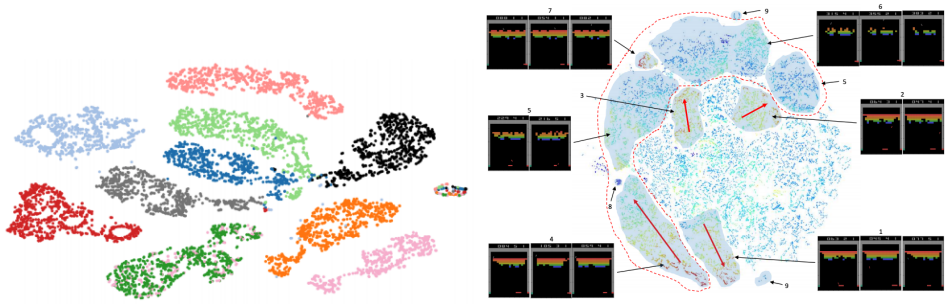


Figure 8: Examples of using t-SNE to reduce high dimension space into two dimensions to be visualizable. Left figure is showing clusters of different human voices by Oord et al. [2018]. Right figure is different regions of action decision from a reinforcement learning agent by Zahavy et al. [2016].

4.2.3 Saliency Map

Saliency map reveals significant information that affects the model decision. Zeiler and Furgus exemplified the saliency map by creating a map shows the influence of the input to the neural network output Zeiler and Fergus [2014]. There are different techniques built upon the saliency map which showing highly activated areas or highly sensitive areas. The saliency method requires the direct computation of gradient from the output of the neural network with respect to the input. However, such derivatives are not generalized and can miss important information flowing through the networks.

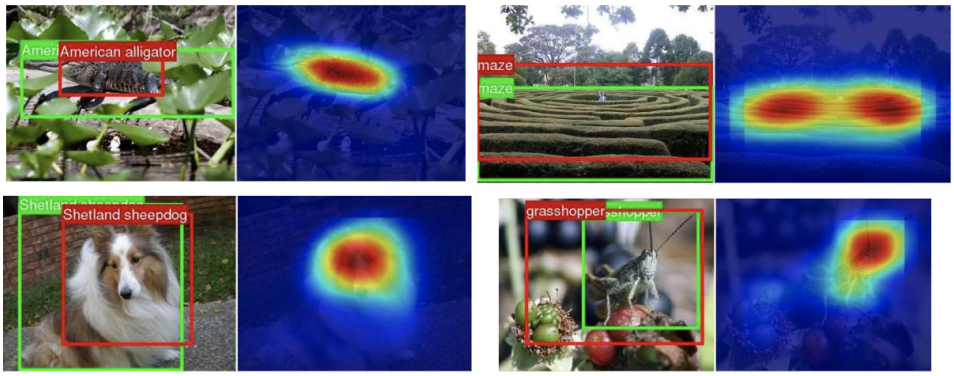


Figure 9: Pair of examples showing object detection (left) with green is ground truth and red is predicted by thresholding the salience map (right) from Zhou et al. [2016].

Researchers have been working on the solution to smoothly derive the required gradient for the saliency map. Layer-wise relevance propagation Bach et al. [2015] is a method to identify contributions of a single pixel by utilizing a bag-of-words features from neural network layers. By simply modifying the global average pooling layer combined with class activation mapping (CAM), a good saliency map is shown Zhou et al. [2016] comparable to an object detection method with interesting results as shown in Fig. 9. DeepLIFT Shrikumar et al. [2017] compares the activation of each neuron with reference activations and assigns contribution scores based on the difference. A weighted method is used for CAM to smooth the gradient Selvaraju et al. [2017]. An integrated gradient method is used to satisfy the sensitivity and implementation variance of the gradient Sundararajan et al. [2017].

De-noising the gradient by adding noise to perturb original input then average the saliency maps collected Smilkov et al. [2017] also shows a better saliency map. An application of using saliency map to interpret why a deep reinforcement learning agent behaves Greydanus et al. [2017]. The agent interpretable samples can be seen in Fig. 10 to understand the reason behind what strategy the agent has learned.

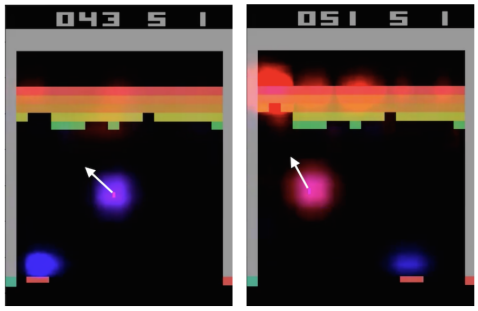


Figure 10: Greydanus et al. [2017] shows how a Breakout agent learns to tunnel for high reward regions. Blue areas interpret action related regions, and red area show the areas relation with a high reward.

4.3 Re-approximation with Interpretable Models

By reducing the complexity of a neural network model, the networks can be interpreted efficiently. This has been done mainly through **re-approximation** of the neural networks with existing interpretable models. The re-approximated model extracts the reasoning of what the neural networks have learned. This approach works regardless of the accessibility of the neural network models, i.e., only behavioral output is enough to prepare re-approximation model for interpretation. There are three main methods to perform the re-approximation: *linear approximation*, *decision tree*, and *rules extraction*.

4.3.1 Linear Approximation

A linear model can be he most simplified model that can provide interpretation of the observable outcomes. Linear model uses a set of weights w and bias b to make prediction: $\hat{y} = wx + b$. The linearity of the relationship between features, weights, and targets makes the interpretation easy. We can analyze the weights of the linear model to understand how an individual input feature impacts the decision.

(a) Original Image (b) Explaining *Electric guitar* (c) Explaining *Acoustic guitar* (d) Explaining *Labrador*

Figure 11: An example of LIME by Ribeiro et al. [2016] explains an image classification prediction from Google's Inception neural networks (Szegedy et al. [2015]) with the top 3 highest probability features: Electric guitar, acoustic guitar, and labrador.

Local Interpretable Model Agnostic (LIME) Ribeiro et al. [2016] exemplified the linear approximation approach to classification problems. LIME first perturbs input data to probe the behavior of the neural networks. A local linear model is trained through the perturbed input and neural network output on the neighborhood information of the input. Fig. 11 shows an example of LIME identifying regions of the input that influences the neural network decision.

With the simplicity in modeling, a linear approximation is by far the easiest method to implement to approximate a neural network. However, the linear model is hard to achieve the equivalent performance of the neural networks. Perturbing the neighborhood information can take a long time to train in high dimensional data. This makes the linear method hard to scale to the complex problems.

4.3.2 Decision Tree

Linear approximation assumes input features to be independent. Therefore, linear approximation fails when features interact with each other to form a non-linear relationship. Decision trees split the data multiple times according to certain cutoff values in the data features. The approach results in an algorithm similar to nested if-then-else statements to compare (smaller/bigger) input features with corresponding threshold numbers. The interpretation is fairly simple by following the instruction from the tree root node to the leaf node. All the edges are connected by 'AND' operation.

Artifitial Neural Networks - Decision Tree (ANN-DT) Schmitz et al. [1999] is an early work that converts a neural network into a decision tree. ANN-DT applied sampling methods to expand the training data using nearest neighbors to create the decision tree. Sato and Tsukimoto designed Continuous Rule Extractor via Decision tree (CRED) to interpret shallow networks Sato and Tsukimoto [2001]. By applying RxREN Augasta and Kathirvalavakumar [2012] to prune unnecessary input features and C4.5 algorithm Quinlan [2014] to create a parsimonious decision tree, an extension of CRED into DeepRED Zilke et al. [2016] is introduced to be able apply to deep neural networks. The decision tree method is also applied to interpret a reinforcement learning agent's decision making Bastani et al. [2018].

Although a decision tree can approximate the neural networks well to accomplish faithfulness, the constructed trees are quite large which cost time and memory to be able to scale. Furthermore, the input features of the decision tree are relatively simple that helps decision tree works. However, it is harder to approximate if the input data is in high dimensional space. Therefore, decision tree approach is hard to generalize to complex input data such as audio, images, or natural languages.

4.3.3 Rule Extraction

Similar to decision trees, rule extraction methods use nested if-then-else statements to approximate neural networks. While decision trees tell a user where to follow (left or right) in each node, the rule-based structures are sequences of logical predicates that are executed in order and apply if-else-then statements to make decisions. We can transform a decision tree to a rule-based structure and vice versa. Rule extraction is a well-studied approach in decision summarization from neural networks Andrews et al. [1995]. There are two main approaches to extract rules from neural networks: decompositional and pedagogical approaches.

Decompositional approaches mimics every individual unit behavior from neural networks by extracted rules. Knowledgetron (KT) method Fu [1994] sweeps through every neural unit to find different thresholds and apply if-then-else rules. The rules are generated based on input rather than the output of the preceding layer in a merging step. However, the KT method has an exponential time complexity and is not applicable to deep networks. The KT method was improved to achieve the polynomial time complexity Tsukimoto [2000]. Fuzzy rules was also created from neural network using the decompositional approach Benítez et al. [1997]. Towell et al. Towell and Shavlik [1993] proposed M-of-N rules which explain a single neural unit by clustering and ignoring insignificant units. Fast Extraction of Rules from Neural Networks (FERNN) Setiono and Leow [2000] tries to identify meaningful neural units and inputs. Unlike other reapproximation methods, the aforementioned decompositional approaches require a full access to the information of neural network models.

Pedagogical approaches are more straightforward than decompositional approaches by extracting rules directly from input and output space without sweeping through every layers and units. Validity interval analysis Thrun [1995] identifies stable intervals that have the most correlation between input and output to mimic behavior of the neural networks. the pedagogical approach can also use sampling methods Craven [1996]; Taha and Ghosh [1999]; Johansson et al. [2005] to extract the rules.

Similar to decision trees, rule extraction methods are easy to analyze a sample. However, the rule extraction methods can extract very complicated rules to explain a decision from deep neural networks. Therefore, rule extraction is also very hard to scale and generalize to the problems with complex input data.

5 How to Evaluate an Interpretable System?

Self-Interpretable System	Human evaluation Model bias
Representation	Performance by substitute task Model bias
Re-approximation	Performance to original model Performance by substitute task

Table 2: Evaluations for different interpretation approaches in our survey.

The three different categories of neural network interpretations have unique characteristics that are different from each other (e.g., the different level of accessibility to the networks). Therefore, there needs to be different evaluation criteria to explain how well the interpretation developed. Table 2 shows the suggested evaluations for each interpretation approach. In our survey, the four different evaluation metrics have appeared consistently:

1. *Performance to original model*: This metric is mostly applied in the re-approximate method to compare the performance of the replaced model against the original neural network model.
2. *Performance by substitute tasks*: Since some interpretation is not reflected by a neural network model, it requires different metrics to compare different attributes of the interpretations.
3. *Model bias*: We can detect the bias of neural networks by testing the sensitivity of a specific phenomenon. If the sensitivity is not consistent across different relevant input information, the neural network is considered biased to a specific pattern.
4. *Human evaluation*: Human is the most reliable evaluation metric. We can crosscheck the output of the interpretation method with human perception into the same problem. Human can also perform the previous three evaluation metrics.

Human evaluation and model bias are frequently used evaluation criteria for *self-interpretable system* approaches. Humans can double-check the result interpreted by the system to compare the interpretation with human perception. For example, attention mechanism can be used for comparing human attention to details; latent space can be evaluated its dimension effect with human analysis; human perception can be used for validating the vulnerability of the neural networks with adversarial examples. Since self-interpretable system is inside the neural networks, model bias evaluation can help the detection bias of the neural networks. For example, attention mechanism fails to translate languages because of the bias (high probability) of a specific pair.

Representation can be interpreted by the produced visualization or presentation. The methods can be evaluated by performance by a substitute task and model bias criteria. We can check the performance by substitute task by checking layers and individual neurons with different inputs to see how neural networks model performs. The same approach can be used for characterizing the layers and individual neurons' representation on a transfer task. For example, we can compare the sensitivity of the saliency maps with brute force measurement. The model bias method can be used to reveal models sensitivity to a specific phenomenon. The layers and individual neurons visualization can benefit from the model bias to examine if the neural network is relying or ignoring a pattern.

The *re-approximation* method can be interpreted by analyzing the weights of a linear model, tracing the nodes of a decision tree, and reasoning the rules. However, there is a trade-off between interpretability and performance in re-approximation method. An approximated model of a neural network needs to balance between simplicity (for interpretation) and accuracy (for resemblance via accurate approximation). Therefore, comparing the performance of the approximated model to the original neural network is a required evaluation criteria for *re-approximation* approach. Researchers also compare the performance by substitute tasks by comparing the trade-off between different re-approximation methods. Since the neural networks are much more complex than reapproximated methods, researchers tend to prefer approximate local behavior to be able to reduce the complexity of the neural networks.

6 Challenges

The trade-off of interpreting neural network exists between the accuracy and robustness of a neural network and the meaningful or simpleness of interpretation. The most accurate and robust model does not guarantee an interpretation of the network in an easy way. The simple and meaningful interpretation might not be easy to learn from a robust method. It is thus challenging when we do not have access to neural networks model to neither re-design nor extracting meaningful information from the model. Reviewing the interpretation methods, we identify two challenges for interpreting neural networks: *robust interpretation* and *sparsity of analysis*.

6.1 Robust Interpretation

Current approaches are too slow to produce robust interpretation in a timely manner. Self-interpretable systems, even though the interpretation is fast on inference, still need to be trained for a longer time. The representation systems need heavy computation in order to achieve visualization results. Re-approximation methods take a long time for both training to approximate neural networks as well as produce interpretation.

Noisy interpretation can severely harm trust of the model. A neural network is trained from the data, possibly training data often cause erroneous interpretation because of errors in labeling process. This phenomenon happens mostly with self-interpretable systems since the objective function designed to optimize the data-only, not the knowledge. The objective function might not be well-covered to interpret the problem that makes the interpretation harder. The representation methods can provide a lot of misleading information from layers and individual neurons, which are not related to human perceptions. Re-approximation methods have limited performance compared to the original neural networks model, so misleading towards the poor interpretation.

6.2 Sparsity of Analysis

For each method, interpretations are made from individual samples or a lot of different visualizations. If we scale up a problem with a large number of samples, a tremendous amount of observations and human effort are required. The problem becomes worse if we interpret samples not from the dataset. For example, in order to interpret the reasoning behind a neural network classifier, human needs to analyze different saliency maps from different input samples to validate the reasoning. With that being said, researchers should concern about sparsity of analysis by reducing the number of visualizations that human needs to analyze. The sparsity is one of the main challenge that we need to address to lessen human arduous effort in interpreting neural networks due to the large amount of data as well as computation units. We need to have a method to recognize a meaningful smaller subset of the whole dataset to interpret. From the meaningful subset, we also need figure out an interpretation between the relationship from different samples with different subsets.

7 Conclusion

Single metric to optimize in deep learning algorithm cannot reflex the complexity of the real world. Safety and ethic are also the concerns when deploying an intelligent system. In order to build safe and trustworthy intelligent system, we need to understand how and why a learning algorithm decides an action to help build better model understanding the real world around it. In order to gain scientific understanding we need to transform model into a source of knowledge.

In this work, we present an overview on interpretability of deep neural networks in general. The interpretability methods are split into three main branches according to the accessibility of users: (1) have access to model and able to modify, (2) have access to model but cannot modify, and (3) have no knowledge of the internal model. Four methods to evaluate the interpretibility system are introduced: (1) performance to original model, (2) performance by substitute task, (3) model bias, and (4) human evaluation. We also went deeper to explain the remaining challenges in the deep learning interpretation field.

8 Future Direction

As we mentioned two different challenges in interpreting a neural networks, we want to emphasize the gap in the current interpretibility approaches: robust interpretability, sparsity of analysis. In order to provide a fast and clear interpretation to human, the approach's robustness need to be ensured. Reducing the amount of analysis can be a good research question since it will also reduce human evaluation time. Dao et al. [2018] has proposed a statistical method to identify important moments in a reinforcement learning problem. A reinforcement learning agent might think differently than human but remains more effective, understanding the reason behind it can benefit a lot of areas with newly discovered knowledge.

The interpretability has been shown to be helpful to create better solutions to improve existing methods. For example, MEENA chatbot Adiwardana et al. [2020] achieved near human sensibleness and specificity understanding in natural language. The interpretability in the self-interpretable system and representation can help validating the neural network predictions. However, self-interpretable and representation systems require accessing and modifying neural networks. In order to trust the interpretation, understanding the networks without accessing it is neccessary. Therefore, we believe re-approximation with interpretable models is the most important approach needed to be improved in the future.

Another area we need to have an explanation in the learning model is reinforcement learning. Reinforcement learning (RL) has actively used deep neural networks and has successfully applied to many areas such as playing video games Mnih et al. [2015], robotics Chen et al. [2017], advertising Zhao et al. [2018], and finance Deng et al. [2016]. However, RL agents have not been able to give confidence to the users in the real world problems because of the lack of understanding (or interpretability). It is hard to convince to people to use an RL agent deployed in a real environment if the unexplained or not understandable behavior are repeated. For instance, in AlphaGo's game 2 against the world best GO player, Lee Sedol, the agent flummoxed with the 37th move, which were not easily explainable at the moment. There can be a huge risk applying a non-understandable RL agent into a business model, especially where human safety or cost for failure is high. There is a huge gap to fully understand why an RL agent decides to take an action and what an agent learns from training.

The interpretibility in RL can benefit humans to explore different strategies in solving problems. For example, DeepMind open-sourced unverified protein structures prediction for COVID-19 from their AlphaFold system Senior et al. [2020] in the middle of the epidemic. The system is confirmed to make accurate predictions with experimentally determined SARS-CoV-2 spike protein structure shared in the Protein Data Bank[5]. Understanding why the RL system makes such prediction can benefit bioinformatics researchers further understand and improve the existing techniques in protein structures to faster create better treatment before the epidemic happens.

References

A. Abdul, J. Vermeulen, D. Wang, B. Y. Lim, and M. Kankanhalli. Trends and trajectories for explainable, accountable and intelligible systems: An hci research agenda. In *Proceedings of the 2018 CHI conference on human factors in computing systems*, pages 1–18, 2018.

D. Adiwardana, M.-T. Luong, D. R. So, J. Hall, N. Fiedel, R. Thoppilan, Z. Yang, A. Kulshreshtha, G. Nemade, Y. Lu, et al. Towards a human-like open-domain chatbot. *arXiv preprint arXiv:2001.09977*, 2020.

P. Anderson, X. He, C. Buehler, D. Teney, M. Johnson, S. Gould, and L. Zhang. 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*, pages 6077–6086, 2018.

R. Andrews, J. Diederich, and A. B. Tickle. Survey and critique of techniques for extracting rules from trained artificial neural networks. *Knowledge-based systems*, 8(6):373–389, 1995.

A. Athalye, L. Engstrom, A. Ilyas, and K. Kwok. Synthesizing robust adversarial examples. *arXiv preprint arXiv:1707.07397*, 2017.

[5]https://deepmind.com/research/open-source/computational-predictions-of-protein-structures-associated-with-COVID-19 (Accessed Mar. 06, 2020).

M. G. Augasta and T. Kathirvalavakumar. Reverse engineering the neural networks for rule extraction in classification problems. *Neural processing letters*, 35(2):131–150, 2012.

S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, and W. Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, 10 (7), 2015.

D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

O. Bastani, Y. Pu, and A. Solar-Lezama. Verifiable reinforcement learning via policy extraction. In *Advances in neural information processing systems*, pages 2494–2504, 2018.

D. Bau, B. Zhou, A. Khosla, A. Oliva, and A. Torralba. Network dissection: Quantifying interpretability of deep visual representations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6541–6549, 2017.

J. M. Benítez, J. L. Castro, and I. Requena. Are artificial neural networks black boxes? *IEEE Transactions on neural networks*, 8(5):1156–1164, 1997.

T. Brown, D. Mane, A. Roy, M. Abadi, and J. Gilmer. Adversarial patch. 2017.

C. P. Burgess, I. Higgins, A. Pal, L. Matthey, N. Watters, G. Desjardins, and A. Lerchner. Understanding disentangling in beta-vae. *arXiv preprint arXiv:1804.03599*, 2018.

A. Canziani, A. Paszke, and E. Culurciello. An analysis of deep neural network models for practical applications. *arXiv preprint arXiv:1605.07678*, 2016.

N. Carlini and D. Wagner. Adversarial examples are not easily detected: Bypassing ten detection methods. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pages 3–14, 2017.

L.-C. Chen, Y. Yang, J. Wang, W. Xu, and A. L. Yuille. Attention to scale: Scale-aware semantic image segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3640–3649, 2016a.

X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *Advances in neural information processing systems*, pages 2172–2180, 2016b.

Y. F. Chen, M. Everett, M. Liu, and J. P. How. Socially aware motion planning with deep reinforcement learning. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1343–1350. IEEE, 2017.

M. W. Craven. Extracting comprehensible models from trained neural networks. Technical report, University of Wisconsin-Madison Department of Computer Sciences, 1996.

G. Dao, I. Mishra, and M. Lee. Deep reinforcement learning monitor for snapshot recording. In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 591–598. IEEE, 2018.

A. Das, H. Agrawal, L. Zitnick, D. Parikh, and D. Batra. Human attention in visual question answering: Do humans and deep networks look at the same regions? *Computer Vision and Image Understanding*, 163:90–100, 2017.

Y. Deng, F. Bao, Y. Kong, Z. Ren, and Q. Dai. Deep direct reinforcement learning for financial signal representation and trading. *IEEE transactions on neural networks and learning systems*, 28(3): 653–664, 2016.

F. Doshi-Velez and B. Kim. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*, 2017.

G. Douzas and F. Bacao. Effective data generation for imbalanced learning using conditional generative adversarial networks. *Expert Systems with applications*, 91:464–471, 2018.

J. Frankle and M. Carbin. The lottery ticket hypothesis: Training pruned neural networks. *arXiv preprint arXiv:1803.03635*, 2018.

D. Frey and R. Pimentel. Principal component analysis and factor analysis. 1978.

L. Fu. Rule generation from neural networks. *IEEE Transactions on Systems, Man, and Cybernetics*, 24(8):1114–1124, 1994.

S. Geman, E. Bienenstock, and R. Doursat. Neural networks and the bias/variance dilemma. *Neural computation*, 4(1):1–58, 1992.

I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014a.

I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014b.

A. Graves, G. Wayne, and I. Danihelka. Neural turing machines. *arXiv preprint arXiv:1410.5401*, 2014.

S. Greydanus, A. Koul, J. Dodge, and A. Fern. Visualizing and understanding atari agents. *arXiv preprint arXiv:1711.00138*, 2017.

R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, F. Giannotti, and D. Pedreschi. A survey of methods for explaining black box models. *ACM computing surveys (CSUR)*, 51(5):1–42, 2018.

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. 2016.

B. Jacob, S. Kligys, B. Chen, M. Zhu, M. Tang, A. Howard, H. Adam, and D. Kalenichenko. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2704–2713, 2018.

R. Jia and P. Liang. Adversarial examples for evaluating reading comprehension systems. *arXiv preprint arXiv:1707.07328*, 2017.

U. Johansson, R. Konig, and L. Niklasson. Automatically balancing accuracy and comprehensibility in predictive modeling. In *2005 7th International Conference on Information Fusion*, volume 2, pages 7–pp. IEEE, 2005.

D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proceedings of the 26th annual international conference on machine learning*, pages 609–616, 2009.

Y. Liu, S. Ma, Y. Aafer, W.-C. Lee, J. Zhai, W. Wang, and X. Zhang. Trojaning attack on neural networks. 2017.

S. Löwe, P. O'Connor, and B. Veeling. Putting an end to end-to-end: Gradient-isolated learning of representations. In *Advances in Neural Information Processing Systems*, pages 3033–3045, 2019.

J. Lu, J. Yang, D. Batra, and D. Parikh. Hierarchical question-image co-attention for visual question answering. In *Advances in neural information processing systems*, pages 289–297, 2016.

J. Lu, C. Xiong, D. Parikh, and R. Socher. Knowing when to look: Adaptive attention via a visual sentinel for image captioning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 375–383, 2017.

M.-T. Luong, H. Pham, and C. D. Manning. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015.

L. v. d. Maaten and G. Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9 (Nov):2579–2605, 2008.

A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

T. Miyato, S.-i. Maeda, M. Koyama, and S. Ishii. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE transactions on pattern analysis and machine intelligence*, 41(8):1979–1993, 2018.

V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

A. Mordvintsev, N. Pezzotti, L. Schubert, and C. Olah. Differentiable image parameterizations. *Distill*, 2018. doi: 10.23915/distill.00012. https://distill.pub/2018/differentiable-parameterizations.

K. Nar, O. Ocal, S. S. Sastry, and K. Ramchandran. Cross-entropy loss and low-rank features have responsibility for adversarial examples. *arXiv preprint arXiv:1901.08360*, 2019.

A. Nguyen, J. Yosinski, and J. Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 427–436, 2015.

A. Nguyen, A. Dosovitskiy, J. Yosinski, T. Brox, and J. Clune. Synthesizing the preferred inputs for neurons in neural networks via deep generator networks. In *Advances in neural information processing systems*, pages 3387–3395, 2016.

A. Odena, C. Olah, and J. Shlens. Conditional image synthesis with auxiliary classifier gans. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2642–2651. JMLR. org, 2017.

C. Olah, A. Satyanarayan, I. Johnson, S. Carter, L. Schubert, K. Ye, and A. Mordvintsev. The building blocks of interpretability. *Distill*, 3(3):e10, 2018.

A. v. d. Oord, Y. Li, and O. Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

S. Pouyanfar, S. Sadiq, Y. Yan, H. Tian, Y. Tao, M. P. Reyes, M.-L. Shyu, S.-C. Chen, and S. Iyengar. A survey on deep learning: Algorithms, techniques, and applications. *ACM Computing Surveys (CSUR)*, 51(5):1–36, 2018.

J. R. Quinlan. *C4. 5: programs for machine learning*. Elsevier, 2014.

M. T. Ribeiro, S. Singh, and C. Guestrin. " 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*, pages 1135–1144, 2016.

A. S. Ross, M. C. Hughes, and F. Doshi-Velez. Right for the right reasons: Training differentiable models by constraining their explanations. *arXiv preprint arXiv:1703.03717*, 2017.

S. Sabour, N. Frosst, and G. E. Hinton. Dynamic routing between capsules. In *Advances in neural information processing systems*, pages 3856–3866, 2017.

M. Sato and H. Tsukimoto. Rule extraction from neural networks via decision tree induction. In *IJCNN'01. International Joint Conference on Neural Networks. Proceedings (Cat. No. 01CH37222)*, volume 3, pages 1870–1875. IEEE, 2001.

G. P. Schmitz, C. Aldrich, and F. S. Gouws. Ann-dt: an algorithm for extraction of decision trees from artificial neural networks. *IEEE Transactions on Neural Networks*, 10(6):1392–1401, 1999.

R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017.

A. W. Senior, R. Evans, J. Jumper, J. Kirkpatrick, L. Sifre, T. Green, C. Qin, A. Žídek, A. W. Nelson, A. Bridgland, et al. Improved protein structure prediction using potentials from deep learning. *Nature*, pages 1–5, 2020.

R. Setiono and W. K. Leow. Fernn: An algorithm for fast extraction of rules from neural networks. *Applied Intelligence*, 12(1-2):15–25, 2000.

A. Shrikumar, P. Greenside, and A. Kundaje. Learning important features through propagating activation differences. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3145–3153. JMLR. org, 2017.

K. Simonyan, A. Vedaldi, and A. Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.

D. Smilkov, N. Thorat, B. Kim, F. Viégas, and M. Wattenberg. Smoothgrad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825*, 2017.

J. Su, D. V. Vargas, and K. Sakurai. One pixel attack for fooling deep neural networks. *IEEE Transactions on Evolutionary Computation*, 23(5):828–841, 2019.

M. Sundararajan, A. Taly, and Q. Yan. Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3319–3328. JMLR. org, 2017.

C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

I. A. Taha and J. Ghosh. Symbolic interpretation of artificial neural networks. *IEEE Transactions on knowledge and data engineering*, 11(3):448–463, 1999.

S. Thrun. Extracting rules from artificial neural networks with distributed representations. In *Advances in neural information processing systems*, pages 505–512, 1995.

G. G. Towell and J. W. Shavlik. Extracting refined rules from knowledge-based neural networks. *Machine learning*, 13(1):71–101, 1993.

H. Tsukimoto. Extracting rules from trained neural networks. *IEEE Transactions on Neural networks*, 11(2):377–389, 2000.

S. van Steenkiste, F. Locatello, J. Schmidhuber, and O. Bachem. Are disentangled representations helpful for abstract visual reasoning? In *Advances in Neural Information Processing Systems*, pages 14222–14235, 2019.

F. Wang, M. Jiang, C. Qian, S. Yang, C. Li, H. Zhang, X. Wang, and X. Tang. Residual attention network for image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3156–3164, 2017.

T. Xiao, Y. Xu, K. Yang, J. Zhang, Y. Peng, and Z. Zhang. The application of two-level attention models in deep convolutional neural network for fine-grained image classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 842–850, 2015.

K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pages 2048–2057, 2015.

J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328, 2014.

R. Yousefzadeh and D. P. O'Leary. Interpreting neural networks using flip points. *arXiv preprint arXiv:1903.08789*, 2019.

T. Zahavy, N. Ben-Zrihem, and S. Mannor. Graying the black box: Understanding dqns. In *International Conference on Machine Learning*, pages 1899–1908, 2016.

M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.

Q. Zhang, R. Cao, Y. N. Wu, and S.-C. Zhu. Growing interpretable part graphs on convnets via multi-shot learning. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

Q.-s. Zhang and S.-C. Zhu. Visual interpretability for deep learning: a survey. *Frontiers of Information Technology & Electronic Engineering*, 19(1):27–39, 2018.

J. Zhao, G. Qiu, Z. Guan, W. Zhao, and X. He. Deep reinforcement learning for sponsored search real-time bidding. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 1021–1030, 2018.

B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Object detectors emerge in deep scene cnns. *arXiv preprint arXiv:1412.6856*, 2014.

B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Learning deep features for discriminative localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2921–2929, 2016.

J. R. Zilke, E. L. Mencía, and F. Janssen. Deepred–rule extraction from deep neural networks. In *International Conference on Discovery Science*, pages 457–473. Springer, 2016.