TOWARDS INTERPRETABLE AND RELIABLE DEEP NEURAL NETWORKS FOR VISUAL INTELLIGENCE

A Dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy

by

NING XIE
M.S., Wright State University, 2018
B.S., Hebei University of Technology, China, 2014

2020
Wright State University

WRIGHT STATE UNIVERSITY

GRADUATE SCHOOL

July 23, 2020

I HEREBY RECOMMEND THAT THE DISSERTATION PREPARED UNDER MY SUPERVISION BY Ning Xie ENTITLED Towards Interpretable And Reliable Deep Neural Networks for Visual Intelligence BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF Doctor of Philosophy.

Derek Doran, Ph.D.
Dissertation Director

Yong Pei, Ph.D.
Director, Computer Science and Engineering
Ph.D. Program

Barry Milligan, Ph.D.
Interim Dean of the Graduate School

Committee on
Final Examination

Derek Doran, Ph.D.

Michael Raymer, Ph.D.

Tanvi Banerjee, Ph.D.

Pascal Hitzler, Ph.D.

ABSTRACT

Xie, Ning. Ph.D., Department of Computer Science and Engineering, Wright State University, 2020. *Towards Interpretable And Reliable Deep Neural Networks for Visual Intelligence.*

Deep Neural Networks (DNNs) are powerful tools blossomed in a variety of successful real-life applications. While the performance of DNNs is outstanding, their opaque nature raises a growing concern in the community, causing suspicions on the reliability and trustworthiness of decisions made by DNNs. In order to release such concerns and towards building reliable deep learning systems, research efforts are actively made in diverse aspects such as model interpretation, model fairness and bias, adversarial attacks and defenses, and so on.

In this dissertation, we focus on the research topic of DNN interpretations for visual intelligence, aiming to unfold the black-box and provide explanations for visual intelligence tasks in a human-understandable way. We first conduct a categorized literature review, systematically introducing the realm of explainable deep learning. Following the review, two specific problems are tackled, explanations of Convolutions Neural Networks (CNNs), which relates the CNN decisions with input concepts, and interpretability of multi-model interactions, where an explainable model is built to solve a visual inference task. Visualization techniques are leveraged to depict the intermediate hidden states of CNNs and attention mechanisms are utilized to build an instinct explainable model. Towards increasing the trustworthiness of DNNs, a certainty measurement for decisions is also proposed as an extensive exploration of this study. To show how the introduced techniques holistically realize a contribution to interpretable and reliable deep neural networks for visual intelligence, further experiments and analyses are conducted for visual entailment task at the end of this dissertation.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Acknowledgment

The past five years have been a wonderful experience in my life. I still remember the winter of 2015 when I first came to the states to start my journey of pursuing my PhD degree, being excited, a little concerned, and also cannot wait to explore the unknown future. Four years is just like a blink, while looking back to my past life and studies at Wright State, I feel so lucky to come here and I am deeply appreciated to the amazing experience and strong support from my advisor, my professors, my mentors, my friends, etc.

First and foremost, my greatest thanks go to my advisor Dr. Derek Doran. When I first joint Wright State, I honestly had little research experience, nor do I had enough confidence to do it well. It is Dr. Doran, who lead me to the wonderland of academia, encouraging me to start my own exploration of real research. I feel extremely lucky to have Dr. Doran as my PhD advisor, without whose great support I can hardly go this far. Thank you so much Dr. Doran!

I would also love to take this opportunity to extend my thanks to Dr. Pascal Hitzler, Dr. Michael Raymer, and Dr. Tanvi Banerjee. Thank you so much for your great support and for being a member of my PhD dissertation committee! Without Dr. Hitzler, my journey at Wright State can hardly begin. I remember it was October 2015, when I wrote my first letter to Dr. Hitzler, inquiring the opportunities of PhD positions. Although there was no availability at Dr. Hitzler's lab, he kindly referred me to Dr. Doran, then the amazing story began at that time. As our PhD academic advisor, Dr. Raymer has offered me kind and continuous support during the past four years. Whenever I have a blur on the PhD program and refer to Dr. Raymer, Dr. Raymer is always able to provide me with very clear and professional explanations in patience. It is Dr. Banerjee, who taught me my first machine learning course, which opened the wonderful world of machine learning for me. I feel lucky to be a student in her class, and her teaching skills are awesome!

During my PhD, I have done two wonderful internships at NEC labs America and Amazon.com. I appreciate my mentors Farley Lai and Asim Kadav at NEC labs, who guided me on how to conduct top tier research, and also provided me with constructive advices towards improvement, which really benefit my future career. I appreciate the support from my mentor Jinmiao Fu and manager Yi Sun during my internship at Amazon, who lead me through the whole pipeline on how to conduct a product project at Amazon. I also want to extend my thanks to my friends, especially Fang Yu, who always stand by my side whenever I refer to her. Thank you so much everyone for the support and guidance, which really help me on becoming a more skilled, professional, and qualified researcher!

Dedicated to my parents

Introduction

1.1 Motivation

1.1.1 The Power and Limits of DNNs

Artificial Intelligence powered by deep learning is thriving in a wide range of real-life applications. Recently a variety of DNN usage scenarios are actively being developed, benefiting our daily life in diverse aspects. Deep Neural Networks (DNNs) are successfully utilized for mobile payment systems [Raut et al., 2018], screen unlocking [Chu and Feng, 2018], driving assistant [Olabiyi et al., 2017], cancer diagnostics [Munir et al., 2019], voice assistant [Coucke et al., 2018], automatic machine translation [Srivastava et al., 2018], recommendation systems [Zhang et al., 2019b], predicting earthquakes [DeVries et al., 2018], analyzing trading strategies [Lv et al., 2019], etc. Besides, novel techniques and applications on deep learning are being rapidly explored, leading to a group of very promising research topics and potential industrial attempts.

Along with the success of DNNs, there raises an increasing concern on the reliability of DNNs and the trustworthiness of their model predictions [Jiang et al., 2018a, Heo et al., 2018, Cortes Ciriano and Bender, 2019]. Such concern seems more urgent for some scenarios, where improper model predictions would result in undesirable severe consequences. For those scenarios, users tend to seek explanations w.r.t. DNN decisions, in order to decide whether or not to accept such decisions generated by the model. For instance, for a model utilized in financial area that predicts to deny a load request, bankers want to know how this decision is made (is it made based on a comprehensive consideration or just based on some biased input features such as the applicant's gender or race?), and how reliable this decision is (bankers tend to minimize the risk thus won't take unreliable decisions into consideration). In medical applications such as cancer prediction, model decisions should be carefully audited before releasing to the patients, and the decision explanations are important to serve doctors as references. The demanding of model explanations are also defined in legal terms [Goodman and Flaxman, 2017]. According to the European Unionś new General Data Protection Regulation (GDPR), users own the "right to explanation", which means a machine learning model should not only generate decisions, but also needs to provide evidence correspondingly.

The opaque nature of DNNs is becoming an obstacle towards its potential future prosperity on real-life applications, especially for critical scenarios where decision justifications are urgently required, which attracts increasing attention in the community. A variety of studies on DNN interpretability and reliability are being actively explored [Xie et al., 2020, Ras et al., 2018, Montavon et al., 2018, Zhang and Zhu, 2018, Samek et al., 2017, Erhan et al., 2010], which will be introduced in detail in Part I. We notice that besides DNN interpretability, there exists a large group of interpretability research studies for other black-box machine learning models [Miller, 2018, Samek et al., 2017, Guidotti et al., 2018, Lipton, 2018, Liu et al., 2017, Došilović et al., 2018]. Such studies are out of the main scope of this dissertation, however may also be covered when the techniques or discussions are general enough to be applied to the realm of interpretable and reliable DNNs.

1.1.2 The Traits of an Explanation

A central tenet in explainable machine learning is that the algorithm must emit information allowing a user to relate characteristics of input features with its output[1]. It is thus worth noting that DNNs are not inherently "explainable". The limited information captured in a DNN's parameters associated with input features becomes entangled and compressed into a single value via a non-linear transform of a weighted sum of feature values. This compression occurs multiple times with different weight vectors depending on the number of activations in the first hidden layer. Subsequent layers then output non-linear transforms of weighted sums of these compressions, and so forth, until a decision is made based on the output of the DNN. Hence, it is exceedingly difficult to trace how particular stimulus properties drive this decision.

The unexplainable nature of DNNs is a significant impediment[2] to the wide-spread adoption of DNNs we are beginning to see in society. DNN-powered facial recognition systems, for example, are now associating people with locations and activities under wide-spread surveillance activities with opaque intent [Masi et al., 2018]. People analytics and human resource platforms now tout the ability to predict employee performance and time to resignation, and to automatically scan the CV of job applicants [Zhao et al., 2018, Qin et al., 2018]. These examples foretell a future where DNN technology will make countless recommendations and decisions that more directly, and perhaps more significantly, impact people and their well-being in society.

[1]Portions of this section have been published [Xie et al., 2020] and is currently under review at Journal of Artificial Intelligence Research.

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

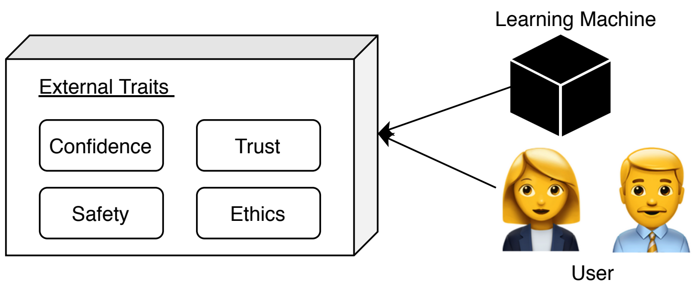


Figure 1.1: Necessary external traits of an explanation.

The present art develops ways to promote traits that are *associated with* explainability. A trait represents a property of a DNN necessary for a user to evaluate its output [Lipton, 2018]. Traits, therefore, represent a particular objective or an evaluation criterion for explainable deep learning systems. We can say that a DNN **promotes explainability** if the system exhibits any trait that is justifiably related to explainability. This exhibition may be self-evident (e.g., in an NLP task, visualizations highlighting keywords or phrases that suggest a reasonable classification of a sentence), measurable (based on a trait-specific "error" metric), or evaluated through system usability studies. We discuss the four traits in Figure 1.1 that are the theme of much of the explainable deep learning literature.

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

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

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

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

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

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

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

1.2 Dissertation Summary

This dissertation is about the interpretability[3] and reliability of model decisions for *visual intelligence* tasks, where interpretability is our main focus and reliability is an extensive exploration of our studies. In this dissertation, visual intelligence is referred to as the ability of a model to take visual related data as input, and generate reasonable model output that is competitive to human comprehensions. Such types of data include but are not limited to images, videos, or data that contains visual elements such as (image, text) pairs. Specifically, in this dissertation, we explore the model interpretability and reliability for two types of visual intelligence tasks, image classification (Chapter 4 and 6) and visual inference (Chapter 5).

The summary of this dissertation is as follows. A categorized literature review is provided in Part I, in order to offer the readers a comprehensive and clear overview of our research topic. The review is presented in two folds, methods for explaining DNNs (Chapter 2) and topics closely associated with interpretable and reliable DNNs (Chapter 3). For

[3]In the dissertation, we use explainability and interpretability interchangeably.

methods on explaining DNNs (Chapter 2), we illustrate methods that provide explanations for given model predictions in mainly three aspects, i) visualization (Section 2.1), where explanations are provided by visualizing DNN hidden states, ii) model distillation (Section 2.2), where distillation techniques are leverages either locally (for instance, learning a simple linear classifier using data around a given decision for local explanation) or globally (translating the whole black-box model into a transparent one, or extracting semantic concepts for the whole model for sake of human understanding), and iii) intrinsic methods (Section 2.3), where the DNNs are intrinsically explainable by design.

Besides the aforementioned methods to explain DNN decisions, there exist other closely associated topics (Chapter 3). Learning mechanism (Section 3.1) includes methods that explore the learning mechanisms of DNNs, and the main difference between learning mechanisms and our focus is the former targets on the general learning strategies of DNNs while the latter focus on specific model decisions that are given. Model debugging (Section 3.2) is to probe and revise the model and its related procedures such as the training process, towards a more desirable model performance. We consider that the study of our research focus (interpretable and reliable DNNs) would shed light on potential model debugging solutions, and some techniques on model debugging may also introduce novel strategies for our studies. Towards building a more robust and reliable DNNs, adversarial techniques, both adversarial attack and defense (Sections 3.3) are covered in our literature review, indicating current threatens on DNN applications and the appealing on building transparent and reliable DNN solutions. Last but not least, fairness and bias (Section 3.4) introduces that some DNN decisions may be affected by undesirable input features causing unfairness to some groups. For instance, a DNN may deny a load request just based on the applicant's race or gender. Such cases should be avoided and we believe our study, on DNN interpretability and reliability, could play an important role in proposing promising future solutions to reduce such model biases.

Having the landscape of current related research in mind, we further introduce our solutions towards the goal of interpretable and reliable DNNs for visual intelligence in Part II. For DNN interpretability, we propose two types of methods that generate explanations for model decisions, through hidden state visualization (Chapter 4) and by leveraging attention to make a DNN intrinsically interpretable (Chapter 5). In Chapter 4, relating input concepts to CNN decisions, a Convolutional Neural Network (CNN) is utilized as an image scene classifier, and our goal is to explain why the CNN classifier predicts an image as, for instance, a bedroom scene [Xie et al., 2017b]. Deconvolution [Zeiler and Fergus, 2014a], a visualization technique, is used to find evidence (concepts) in the given input image, and a greedy algorithm is designed to associate the evidence with model predictions. Chapter 5, attention mechanism for intrinsic DNN explanations, achieves the goal of interpretability from an intrinsic perspective. A DNN is designed to solve visual entailment [Xie et al., 2019], which is a novel visual inference task we proposed that involves the interaction between computer vision and natural language processing. In order to depict how the input image interacts with the input natural sentence, attention mechanism is leveraged, showing that given the input sentence, which image areas are critical towards generating the model decision. The image attention heatmaps are visualized for a clear illustration.

For an extension of the main focus of this dissertation and towards DNN reliability, an intrinsic measurement, called "certainty score", for DNN decisions is designed (Chapter 6), based on which the users could decide whether or not to adopt the given DNN decision. We call the certainty score intrinsic in the sense that it considers the collection of activations of each intermediate layer in the deep neural network. The certainty score measures the class similarity on the prediction for a given input against labels of examples in the training set whose layer-wise activation patterns are close to the activations of the given input. Details on how the certainty score is defined and its usability will be discussed in Chapter 6.

In order to show how the introduced techniques (Chapter 4, 5, and 6) holistically realize a contribution to explainable deep learning, further experiments and analyses are conducted for visual entailment task in Chapter 7.

Part I

A Categorized Literature Review

Methods for Explaining DNNs

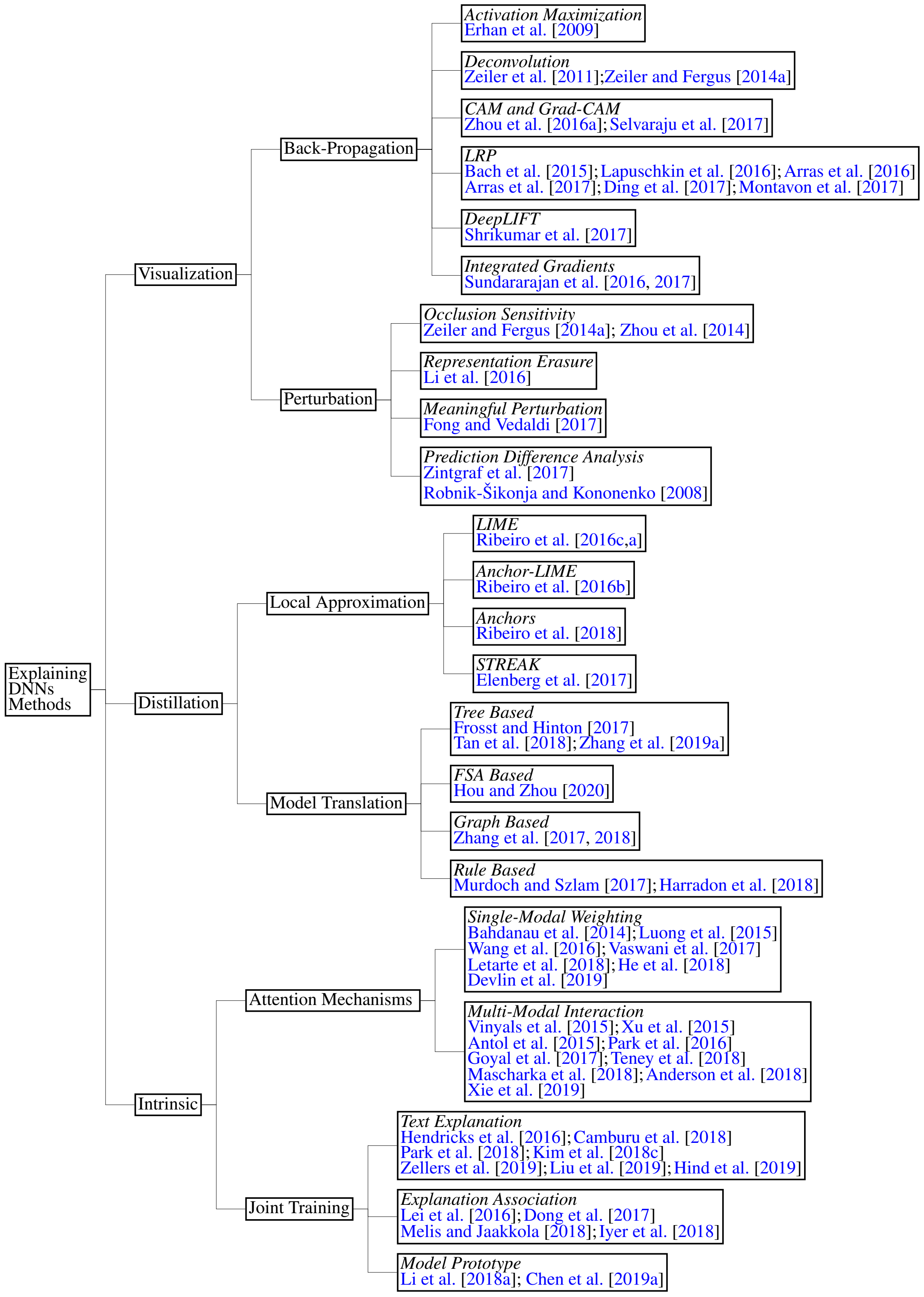
A detailed description of methods that relate to the topic of “decision explanation” is discussed in this chapter[1]. The methods are categorized and defined in three types: visualization, model distillation, and intrinsic methods, as shown below,

- **Visualization methods**: Methods that leverage visualization techniques to associate model decisions to input characteristics.
- **Model distillation**: Methods that use an auxiliary model to create explanation.
- **Intrinsic methods**: Methods architected and/or trained to produce “natural” explanations.

2.1 Visualization Methods

Visualization methods associate the degree to which a DNN considers input features to a decision. A common explanatory form of visualization methods is *saliency maps*. These maps identify input features that are most salient, in the sense that they cause a maximum response or stimulation influencing the model’s output [Yosinski et al., 2015, Ozbulak, 2019, Olah et al., 2017, 2018, Carter et al., 2019]. We break down visualization methods

[1]Portions of this chapter have been published [Xie et al., 2020] and is currently under review at Journal of Artificial Intelligence Research.



Explaining DNNs Methods
Visualization
Back-Propagation
Activation Maximization
Erhan et al. [2009]
Deconvolution
Zeiler et al. [2011];Zeiler and Fergus [2014a]
CAM and Grad-CAM
Zhou et al. [2016a]; Selvaraju et al. [2017]
LRP
Bach et al. [2015]; Lapuschkin et al. [2016]; Arras et al. [2016]
Arras et al. [2017]; Ding et al. [2017]; Montavon et al. [2017]
DeepLIFT
Shrikumar et al. [2017]
Integrated Gradients
Sundararajan et al. [2016, 2017]
Perturbation
Occlusion Sensitivity
Zeiler and Fergus [2014a]; Zhou et al. [2014]
Representation Erasure
Li et al. [2016]
Meaningful Perturbation
Fong and Vedaldi [2017]
Prediction Difference Analysis
Zintgraf et al. [2017]
Robnik-Šikonja and Kononenko [2008]
Distillation
Local Approximation
LIME
Ribeiro et al. [2016c,a]
Anchor-LIME
Ribeiro et al. [2016b]
Anchors
Ribeiro et al. [2018]
STREAK
Elenberg et al. [2017]
Model Translation
Tree Based
Frosst and Hinton [2017]
Tan et al. [2018]; Zhang et al. [2019a]
FSA Based
Hou and Zhou [2020]
Graph Based
Zhang et al. [2017, 2018]
Rule Based
Murdoch and Szlam [2017]; Harradon et al. [2018]
Intrinsic
Attention Mechanisms
Single-Modal Weighting
Bahdanau et al. [2014]; Luong et al. [2015]
Wang et al. [2016]; Vaswani et al. [2017]
Letarte et al. [2018]; He et al. [2018]
Devlin et al. [2019]
Multi-Modal Interaction
Vinyals et al. [2015]; Xu et al. [2015]
Antol et al. [2015]; Park et al. [2016]
Goyal et al. [2017]; Teney et al. [2018]
Mascharka et al. [2018]; Anderson et al. [2018]
Xie et al. [2019]
Joint Training
Text Explanation
Hendricks et al. [2016]; Camburu et al. [2018]
Park et al. [2018]; Kim et al. [2018c]
Zellers et al. [2019]; Liu et al. [2019]; Hind et al. [2019]
Explanation Association
Lei et al. [2016]; Dong et al. [2017]
Melis and Jaakkola [2018]; Iyer et al. [2018]
Model Prototype
Li et al. [2018a]; Chen et al. [2019a]

Figure 2.1: Methods for explaining DNNs.

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

Table 2.1: Visualization methods.

into two types, namely *back-propagation* and *perturbation-based* visualization. The types are summarized in Table 2.1 and will be discussed further below.

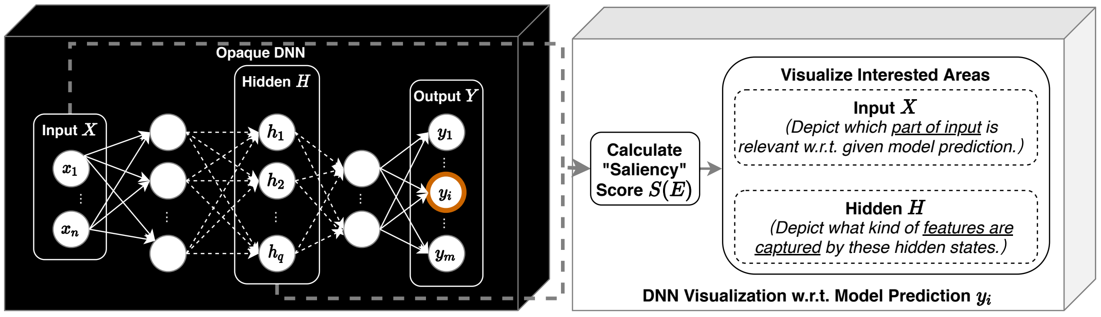

Opaque DNN
Input X
x_1
x_n
Hidden H
h_1
h_2
h_q
Output Y
y_1
y_i
y_m
Calculate "Saliency" Score $S(E)$
Visualize Interested Areas
Input X
(Depict which part of input is relevant w.r.t. given model prediction.)
Hidden H
(Depict what kind of features are captured by these hidden states.)
DNN Visualization w.r.t. Model Prediction y_i

Figure 2.2: Visualization methods. The to-be-visualized element E can be either the model input X or hidden states H. Visualization is based on the calculated saliency score $S(E)$, which varies along with different visualization methods.

Backpropagation-based methods

Backpropagation-based methods identify the saliency of input features based on some evaluation of gradient signals passed from output to input during network training. A baseline gradient-based approach visualizes the partial derivative of the network output with respect to each input feature scaled by its value [Simonyan et al., 2013, Springenberg et al., 2014], thus quantifying the “sensitivity” of the network’s output with respect to input features. In a scene recognition task, for example, a high relevance score for pixels representing a bed in a CNN that decides the image is of class "bedroom" may suggest that the decision made by the CNN is highly sensitive to the presence of the bed in the image. Other gradient-based methods may evaluate this sensitivity with respect to the output, but from different collections of feature maps at intermediate CNN network layers [Zeiler and Fergus, 2014a, Bach et al., 2015, Montavon et al., 2017, Shrikumar et al., 2017]. We describe some foundational gradient-based methods below.

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

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

$$\boldsymbol{A}^{\ell}, s^{\ell} = \texttt{maxpool}(ReLU(\boldsymbol{A}^{\ell-1} * \boldsymbol{K}^{\ell} + \boldsymbol{b}^{\ell})) \tag{2.1}$$

where ℓ indicates the respective layer, $\boldsymbol{A}^{\ell}$ is the output of the previous layer, $\boldsymbol{K}$ is the learned filter, and $\boldsymbol{b}$ is the bias. If the outputs from the ReLU are passed through a local max-pooling function, it additionally stores the output s^{ℓ} containing the indices of the maximum values for a later unpooling operation. In the original paper, the set of s^{ℓ}'s are referred to as *switches*. A deconvolutional neural network, referred to as `DeCNN`, consists of the inverse operations of the original convolutional network `CNN`. `DeCNN` takes the output of `CNN` as its input. In other words, `DeCNN` runs the `CNN` in reverse, from top-down. This is why the deconvolution method is classified as a back-propagation method. The convolutional layers in `CNN` are replaced with *deconvolutions* and the max-pooling layers are replaced with *unpooling* layers. A deconvolution is also called a *transposed convolution*, meaning that the values of $\boldsymbol{K}^{\ell}$ are transposed and then copied to the deconvolution filters $\boldsymbol{K}^{\ell^T}$. If `CNN` included max-pooling layers, they are replaced with unpooling layers which approximately upscales the feature map, retaining only the maximum values. This is done by retrieving the indices stored in s^{ℓ} at which the maximum values were located when the max-pooling was originally applied in `CNN`. As an example let us see the calculations involved in deconvolving Equation 2.1:

$$\boldsymbol{A}^{\ell-1} = \texttt{unpool}(ReLU((\boldsymbol{A}^{\ell} - \boldsymbol{b}^{\ell}) * \boldsymbol{K}^{\ell^T}), s^{\ell}) \tag{2.2}$$

Using Equation 2.2 one or multiple learned filters $\boldsymbol{K}$ in any layer of the network can be visualized by reverse propagating the values of $\boldsymbol{K}$ all the way back to the input space. Finally, this study also describes how the visualizations can be used for architecture selection.

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

$$map_c = \sum_{k}^{K} w_{k,c} \boldsymbol{A}_k \tag{2.3}$$

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

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

is computed and averaged to get an importance score $\alpha_{k,c}$ for feature map $\boldsymbol{A}_k$:

$$\alpha_{k,c} = \frac{1}{m \cdot n} \sum_{i=1}^{m} \sum_{j=1}^{n} \frac{\partial y_c}{\partial \boldsymbol{A}_{k,i,j}} \tag{2.4}$$

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

$$map_c = ReLU\Big(\sum_{k}^{K} \alpha_{k,c} \boldsymbol{A}_k\Big) \tag{2.5}$$

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

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

$$\begin{aligned} f(\boldsymbol{x}) &= f(\hat{\boldsymbol{x}}) + \nabla_{\hat{\boldsymbol{x}}} f \cdot (\boldsymbol{x} - \hat{\boldsymbol{x}}) + \epsilon \\ &= \sum_{i}^{N} \frac{\partial f}{\partial x_i}(\hat{x_i}) \cdot (x_i - \hat{x_i}) + \epsilon \end{aligned} \tag{2.6}$$

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

$$r_i = \frac{\partial f}{\partial x_i}(\hat{x_i}) \cdot (x_i - \hat{x_i}) \tag{2.7}$$

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

$$r_i^{\ell} = \sum_{j}^{M} r_{i,j}^{\ell} \tag{2.8}$$

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

DeepLIFT. Deep Learning Important FeaTures (DeepLIFT) is another important back-propagation based approach, proposed by Shrikumar et al. [2017]. It assigns relevance scores to input features based on the difference between an input $\boldsymbol{x}$ and a "reference" input

$\boldsymbol{x}'$. The reference should be chosen according to the problem at hand and can be found by answering the question *"What am I interested in measuring differences against?"*. In an example using MNIST the reference chosen is an input of all zeros as this is the background value in the images. Define $\Delta t = f(\boldsymbol{x}) - f(\boldsymbol{x}')$ as the difference-from-reference of an interested neuron output of the network between $\boldsymbol{x}$ and reference $\boldsymbol{x}'$, and $\Delta\boldsymbol{x} = \boldsymbol{x} - \boldsymbol{x}'$ as the difference between $\boldsymbol{x}$ and $\boldsymbol{x}'$. DeepLIFT assigns a relevance score $R_{\Delta x_i \Delta t}$ for input feature x_i:

$$\Delta t = \sum_{i=1}^{N} R_{\Delta x_i \Delta t} \tag{2.9}$$

where N is the number of input neurons that are necessary to compute t. In this formulation, $R_{\Delta x_i \Delta t}$ can be thought of as a weight denoting how much influence Δx_i had on Δt. According to Equation 2.9 the sum of the all weights is equal to the difference-from-reference output Δt. The relevance score can be calculated via the *Linear* rule, *Rescale* rule, or *RevealCancel* rule, as elaborated in their study. A multiplier $m_{\Delta\boldsymbol{x}\Delta t}$ is defined as

$$m_{\Delta\boldsymbol{x}\Delta t} = \frac{R_{\Delta\boldsymbol{x}\Delta t}}{\Delta\boldsymbol{x}} \tag{2.10}$$

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

$$m_{\Delta x_i \Delta t} = \sum_{j=1}^{K} m_{\Delta x_i \Delta a_j^\ell} m_{\Delta a_j^\ell \Delta t} \tag{2.11}$$

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

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

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

$$IG_i(\boldsymbol{x}) = (x_i - x_i') \int_0^1 \frac{\partial f(\boldsymbol{x}' + \alpha(\boldsymbol{x} - \boldsymbol{x}'))}{\partial x_i} d\alpha \tag{2.12}$$

where α is associated with the path from $\boldsymbol{x}'$ to $\boldsymbol{x}'$, and is smoothly distributed in range $[0, 1]$. An interpretation of IG_i is the cumulative sensitivity of f to changes in feature i in all inputs on a straight line between $\boldsymbol{x}'$ to $\boldsymbol{x}$ going in direction i. Intuitively, x_i should have increasing relevance if gradients are large between a "neutral" baseline point $\boldsymbol{x}'$ and $\boldsymbol{x}$ along the direction of x_i. IG can be approximated by a Riemann summation of the integral:

$$IG_i(\boldsymbol{x}) \approxeq (x_i - x_i') \sum_{k=1}^{M} \frac{\partial F(\boldsymbol{x}' + \frac{k}{M}(\boldsymbol{x} - \boldsymbol{x}'))}{\partial x_i} \frac{1}{M} \tag{2.13}$$

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

Perturbation-based Methods

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

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

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

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

$$B(\boldsymbol{x}; f) = \{\boldsymbol{x} \in \boldsymbol{X}_c \Leftrightarrow f(\boldsymbol{x}) = +1\} \tag{2.14}$$

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

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

$$p(c \mid \boldsymbol{x}_{-i}) = \sum_{x_i} p(x_i \mid \boldsymbol{x}_{-i}) p(c \mid \boldsymbol{x}_{-i}, x_i) \tag{2.15}$$

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

$$\text{Diff}_i(c \mid \boldsymbol{x}) = \log_2(\texttt{odds}(c \mid \boldsymbol{x})) - \log_2(\texttt{odds}(c \mid \boldsymbol{x}_{-i})) \tag{2.16}$$

where $\texttt{odds}(c \mid \boldsymbol{x}) = \frac{p(c \mid \boldsymbol{x})}{1 - p(c \mid \boldsymbol{x})}$. The magnitude of $\text{Diff}_i(c \mid \boldsymbol{x})$ measures the importance of feature x_i. $\text{Diff}_i(c \mid \boldsymbol{x})$ measures the influence direction of feature x_i, where a positive value means *for* decision c and a negative value means *against* decision c. Compared to Robnik-

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

Table 2.2: Model distillation.

Šikonja and Kononenko [2008], Zintgraf *et al.* improves prediction difference analysis in three ways: by i) sampling patches instead of pixels given the high pixel dependency nature of images; ii) removing patches instead of individual pixels to measure the prediction influence given the robustness nature of neutral networks on individual pixels; iii) adapting the method to measure the effect of intermediate layers by changing the activations of a given intermediate layer and evaluate the influence on down-streaming layers.

2.2 Model Distillation

In this review we use the term *model distillation* to refer to a class of post-training explanation methods where the knowledge encoded within a trained DNN is *distilled* into a representation amenable for explanation by a user. The reader should be aware that Hinton et al. [2015] outlines a method, with the same name, that implements a specific form of model distillation, namely distilling the knowledge learned by an ensemble of DNNs into a single DNN. An entire class of explainable deep learning techniques have emerged which are based on the notion of model distillation. In this setting, as illustrated in Figure 2.3, an inherently transparent or white box machine learning model g is trained to mimic the input/output behavior of a trained opaque deep neural network f so that $g(\boldsymbol{y}) \approx f(\boldsymbol{x})$. Subsequent explanation of how g maps inputs to outputs may serve as a surrogate explanation of f's mapping.

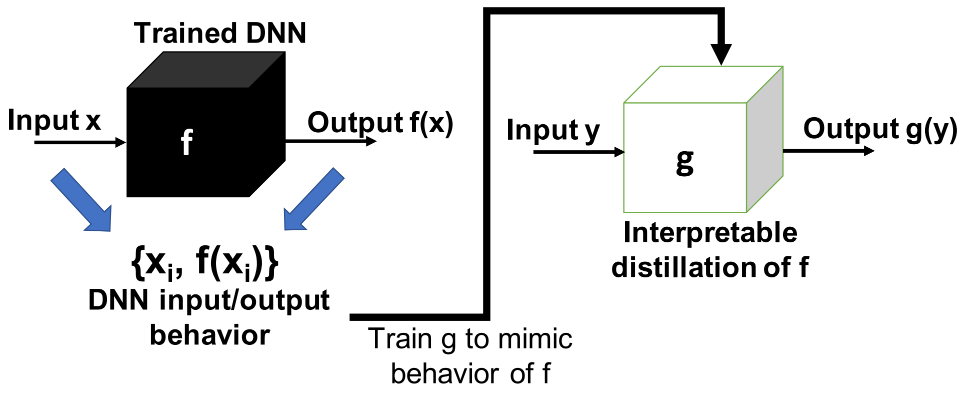

Trained DNN
Input x
f
Output f(x)
{x_i, f(x_i)}
DNN input/output behavior
Train g to mimic behavior of f
Input y
g
Output g(y)
Interpretable distillation of f

Figure 2.3: Model distillation. The behavior of a trained deep learning model f used as training data for an explainable model g.

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

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

- **Local Approximation.** A local approximation method learns a simple model whose input/output behavior mimics that of a DNN for a small subset of the input data. This method is motivated by the idea that the model a DNN uses to discriminate within a local area of the data manifold is simpler than the discriminatory model over the entire surface. Given a sufficiently high local density of input data to approximate the local manifold with piecewise linear functions, the DNN's behavior in this local area may be distilled into a set of explainable linear discriminators.

- **Model Translation.** Model translations train an alternative smaller model that mimics the input/output behavior of a DNN. They contrast local approximation methods in replicating the behavior of a DNN across an entire dataset rather than small subsets. The smaller models may be directly explainable, may be smaller and easier to deploy, or could be further analyzed to gain insights into the causes of the input/output behavior that the translated model replicates.

Local Approximation

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

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

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

$$\arg\min_{g_{\boldsymbol{x}} \in G} \mathcal{L}(f, g_{\boldsymbol{x}}, \Pi_{\boldsymbol{x}}) + \Omega(g_{\boldsymbol{x}}) \tag{2.17}$$

$\Omega(g_{\boldsymbol{x}})$ thus serves as a type of complexity regularization, or as a guarantee that the returned model will not be too complex when $\Omega(g_{\boldsymbol{x}})$ codes a hard constraint. So that LIME remains model agnostic, $\mathcal{L}$ is approximated by uniform sampling over the non-empty space of $\mathbb{R}^{d'}$. For each sampled data point $\boldsymbol{y}' \in \mathbb{R}^{d'}$, LIME recovers the $\boldsymbol{x} \in \mathbb{R}^{d}$ corresponding to $\boldsymbol{y}'$, computes $f(\boldsymbol{x})$, and compares this to $g_{\boldsymbol{x}}(\boldsymbol{y}')$ using $\mathcal{L}$. To make sure that the $g_{\boldsymbol{x}}$ minimizing Equation 2.17 fits well in the area local to a reference point $\boldsymbol{x}$, the comparison of $f(\boldsymbol{y})$ to $g_{\boldsymbol{x}}(\boldsymbol{y}')$ in $\mathcal{L}$ is weighted by $\Pi_{\boldsymbol{x}}(\boldsymbol{y})$ such that sampled data farther from $\boldsymbol{x}$ has lower contribution to loss.

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

Model Translation

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

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

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

Distillation into Graphs. Both Zhang et al. [2017] and Zhang et al. [2018] build an object parts graph for a pre-trained CNN to provide model explanations. Similar to Zhang et al. [2019a], the authors first extract semantic patterns in the input and then gradually construct the graph for explanation. Each node in the graph represents a part pattern, while each edge represents co-activation or spatial adjacent between part patterns. The explanatory graph explains the knowledge hierarchy inside of the model, which can depict which nodes/part patterns are activated as well as the location of the parts in the corresponding feature maps.

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

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

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

Table 2.3: Intrinsic methods.

approximating the output of the LSTM model.

2.3 Intrinsic Methods

Ideally, we would like to have models that provide explanations for their decisions as part of the model output, or that the explanation can easily be derived from the model architecture. In other words, explanations should be *intrinsic* to the process of designing model architectures and during training. The ability for a network to intrinsically express an explanation may be more desirable compared to post-hoc methods that seek explanations of models that were never designed to be explainable in the first place. This is because an intrinsic model has the capacity to learn not only accurate outputs per input but also outputs expressing an explanation of the network's action that is optimal with respect to some notion of explanatory fidelity. Ras et al. [2018] previously defined a category related to this approach as *intrinsic methods* and identified various methods that offer explainable extensions of the model architecture or the training scheme. In this section, we extend the notion of intrinsic explainability with models that actually provide an explanation for their decision even as they are being trained.

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

Attention Mechanisms

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

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

Multi-Modal Interaction. In multi-modal interaction tasks, such as image captioning [Vinyals et al., 2015, Xu et al., 2015], visual question answering [Antol et al., 2015, Goyal et al., 2017, Johnson et al., 2017a, Teney et al., 2018] or visual entailment [Xie et al., 2019], attention mechanisms play an important role in feature alignment and fusion across different feature spaces (for instance, between text and images). For example, Park *et al.* propose the Pointing and Justification model (PJ-X) that uses multiple attention mechanisms to explain the answer of a VQA task with natural language explanations and image region alignments [Park et al., 2016]. Xie *et al.* use attention mechanisms to recover semantically meaningful areas of an image that correspond to the reason a statement is, is not, or could be entailed by the image's conveyance [Xie et al., 2019]. Mascharka et al. [2018] aims to close the gap between performance and explainability in visual reasoning by introducing a neural module network that explicitly models an attention mechanism in image space. By passing attention masks between modules it becomes explainable by being able to directly visualize the masks. This shows how the attention of the model shifts as it considers the different components of the input.

Joint Training

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

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

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

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

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

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

Melis and Jaakkola [2018] proposes an intrinsic method which associates input features with semantically meaningful concepts and regards the coefficient as the importance of such concepts during inference. A regularization based general framework for creating self-explaining neural networks (SENNs) is introduced. Given raw input, the network jointly learns to generate the class prediction and to generate explanations in terms of an input feature-to-concept mapping. The framework is based on the notion that linear regression models are explainable and generalizes the respective model definition to encompass complex classification functions, such as a DNN. A SENN consists of three components: i) A "concept encoder" that transforms the raw input into a set of explainable concepts. Essentially this encoder can be understood as a function that transforms low-level input into high-level meaningful structure, which predictions and explanations can be built upon. ii) An "input-dependent parametrizer", which is a procedure to get the coefficient of explainable concepts, learns the relevance of the explainable concepts for the class predictions.

The values of the relevance scores quantify the positive or negative contribution of the concept to the prediction. iii) Some "aggregation function" (e.g. a sum) that combines the output of the concept encoder and the parametrizer to produce a class prediction.

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

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

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

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

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

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

Topics Associated with Interpretable and Reliable DNNs

We next review research topics closely aligned with interpretable deep learning[1]. A survey, visualized in Figure 3.1, identifies four broad related classes of research. Work on **learning mechanism (Section 3.1)** investigates the backpropagation process to establish a theory around weight training. These studies, in some respects, try to establish a theory to explain how and why DNNs converge to some decision-making process. Research on **model debugging (Section 3.2)** develops tools to recognize and understand the failure modes of a DNN. It emphasizes the discovery of problems that limit the training and inference process of a DNN (e.g., dead ReLUs, mode collapse, etc.). Techniques for **adversarial attack and defense (Section 3.3)** search for differences between regular and unexpected activation patterns. This line of work promotes deep learning systems that are robust and trustworthy; traits that also apply to explainability. Research on **fairness and bias in DNNs (Section 3.4)** is related to the ethics trait discussed above, but more narrowly concentrates on ensuring DNN decisions do not over-emphasize undesirable input data features. We elaborate on the connection between these research areas and explainable DNNs next.

[1]Portions of this chapter have been published [Xie et al., 2020] and is currently under review at Journal of Artificial Intelligence Research.

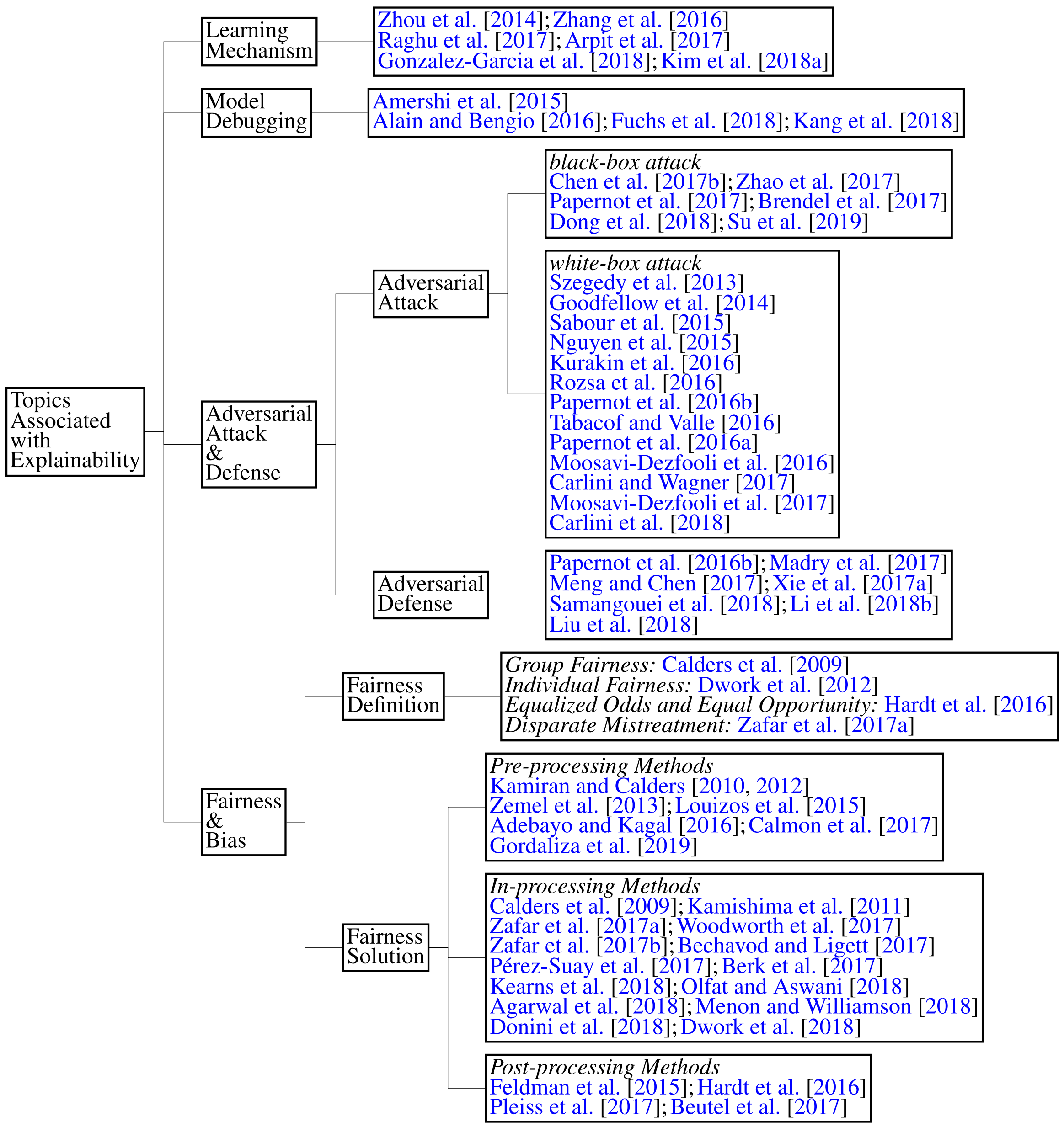

Topics Associated with Explainability
Learning Mechanism
Zhou et al. [2014]; Zhang et al. [2016]
Raghu et al. [2017]; Arpit et al. [2017]
Gonzalez-Garcia et al. [2018]; Kim et al. [2018a]
Model Debugging
Amershi et al. [2015]
Alain and Bengio [2016]; Fuchs et al. [2018]; Kang et al. [2018]
Adversarial Attack & Defense
Adversarial Attack
black-box attack
Chen et al. [2017b]; Zhao et al. [2017]
Papernot et al. [2017]; Brendel et al. [2017]
Dong et al. [2018]; Su et al. [2019]
white-box attack
Szegedy et al. [2013]
Goodfellow et al. [2014]
Sabour et al. [2015]
Nguyen et al. [2015]
Kurakin et al. [2016]
Rozsa et al. [2016]
Papernot et al. [2016b]
Tabacof and Valle [2016]
Papernot et al. [2016a]
Moosavi-Dezfooli et al. [2016]
Carlini and Wagner [2017]
Moosavi-Dezfooli et al. [2017]
Carlini et al. [2018]
Adversarial Defense
Papernot et al. [2016b]; Madry et al. [2017]
Meng and Chen [2017]; Xie et al. [2017a]
Samangouei et al. [2018]; Li et al. [2018b]
Liu et al. [2018]
Fairness & Bias
Fairness Definition
Group Fairness: Calders et al. [2009]
Individual Fairness: Dwork et al. [2012]
Equalized Odds and Equal Opportunity: Hardt et al. [2016]
Disparate Mistreatment: Zafar et al. [2017a]
Fairness Solution
Pre-processing Methods
Kamiran and Calders [2010, 2012]
Zemel et al. [2013]; Louizos et al. [2015]
Adebayo and Kagal [2016]; Calmon et al. [2017]
Gordaliza et al. [2019]
In-processing Methods
Calders et al. [2009]; Kamishima et al. [2011]
Zafar et al. [2017a]; Woodworth et al. [2017]
Zafar et al. [2017b]; Bechavod and Ligett [2017]
Pérez-Suay et al. [2017]; Berk et al. [2017]
Kearns et al. [2018]; Olfat and Aswani [2018]
Agarwal et al. [2018]; Menon and Williamson [2018]
Donini et al. [2018]; Dwork et al. [2018]
Post-processing Methods
Feldman et al. [2015]; Hardt et al. [2016]
Pleiss et al. [2017]; Beutel et al. [2017]

Figure 3.1: Topics associated with explainability.

3.1 Learning Mechanism

The investigation of the learning mechanism tries to derive principles explaining the evolution of a model's parameters during training. Many existing approaches can be categorized as being semantics-related, in that the analysis tries to associate a model's learning process with concepts that have a concrete semantic meaning. They generally assign semantic concepts to a DNNs' internal filters (weights) or representations (activations), in order to uncover a human-interpretable explanation of the learning mechanism. Semantically interpretable descriptions are rooted in the field of neuro-symbolic computing [Garcez et al., 2012]. An early work is Zhou et al. [2014] which assigns semantic concepts, such as objects, object parts, etc, to the internal filters of a convolutional neural network (CNN) image scene classifier. Those semantic concepts are generated based on the visualization of receptive fields of each internal unit in the given layers. The authors also discovered that object detectors are embedded in a scene classifier without explicit object-level supervision for model training. Gonzalez-Garcia et al. [2018] further explores this problem in a quantitative fashion. Two quantitative evaluations are conducted to study whether the internal representations of CNNs really capture semantic concepts. Interestingly, the authors' experimental results show that the association between internal filters and semantic concepts is modest and weak. But this association improves for deeper layers of the network, matching the conclusion of Zhou et al. [2014]. Kim et al. [2018a] quantifies the importance of a given semantic concept with respect to a classification result via Testing with Concept Activation Vector (TCAV), which is based on multiple linear classifiers built with internal activations on prepared examples. The prepared examples contain both positive examples representing a semantic concept and randomly sampled negative examples that do not represent the concept. Directional derivatives are used to calculate TCAV, which measures the proportion of examples that belong to a given class that are positively influenced by a given concept.

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

3.2 Model Debugging

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

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

3.3 Adversarial Attack and Defense

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

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

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

Adversarial defense is challenging due to the diversity of the adversarial example crafting processes and a DNN's high-dimensional feature space. There exist two typical groups of adversarial defense methods, i) adversarial training [Madry et al., 2017, Goodfellow et al., 2014, Szegedy et al., 2013], which is to augment the training dataset with generated adversarial examples such that the trained model is more robust against adversarial attack, and ii) removal perturbations [Samangouei et al., 2018, Meng and Chen, 2017], which dismisses adversarial perturbations from input data. Madry et al. [2017] integrates the PGD attack into the model training process, such that the model is optimized on both benign examples and challenging adversarial examples. The optimization is conducted in a min-max fashion, where the loss for adversarial attack process is maximized in order to generate strong adversarial examples, while the loss for the classification process is minimized, in order to get a robust and well-performed model. Samangouei et al. [2018], on the other hand, tackles the adversarial defense problem by filtering out adversarial perturbations. Generative Adversarial Networks (GANs) are leveraged to project a given input image, potentially polluted by adversarial perturbations, into a pseudo original image, where adversarial artifacts are diminished. The model decision is made based on the GAN generated "original" image, and experiments indicate this defense technique is effective against both black-box and white-box attacks.

3.4 Fairness and Bias

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

The fairness problem is currently addressed by three types of methods [Calmon et al., 2017]: (i) *pre-processing* methods revise input data to remove information correlated to sensitive attributes; (ii) *in-process* methods add fairness constraints into the model learning process; and (iii) *post-process* methods adjust model predictions after the model is trained. *Pre-processing* methods [Kamiran and Calders, 2010, 2012, Zemel et al., 2013, Louizos et al., 2015, Adebayo and Kagal, 2016, Calmon et al., 2017, Gordaliza et al., 2019] learn an alternative representation of the input data that removes information correlated to the sensitive attributes (such as race or gender) while maintaining the model performance as much as possible. For example, Calmon et al. [2017] proposes a probabilistic framework to transform input data to prevent unfairness in the scope of supervised learning. The input transformation is conducted as an optimization problem, aiming to balance discrimination control (group fairness), individual distortion (individual fairness), and data utility. *In-process* methods [Calders et al., 2009, Kamishima et al., 2011, Zafar et al., 2017a, Woodworth et al., 2017, Zafar et al., 2017b, Bechavod and Ligett, 2017, Kearns et al., 2018, Pérez-Suay et al., 2017, Berk et al., 2017, Olfat and Aswani, 2018, Agarwal et al., 2018, Menon and Williamson, 2018, Donini et al., 2018, Dwork et al., 2018] directly introduce fairness learning constraints to the model in order to punish unfair decisions during training. Kamishima et al. [2011] achieves the fairness goal by adding a fairness regularizer, for example, such that the influence of sensitive information on model decisions is reduced. *Post-process* methods [Feldman et al., 2015, Hardt et al., 2016, Pleiss et al., 2017, Beutel et al., 2017] are characterized by adding ad-hoc fairness procedures to a trained model. One example is Hardt et al. [2016] which constructs non-discriminating predictors as a post-processing step to achieve equalized odds and equal opportunity (two fairness notions proposed in their study). They introduce the procedure to construct non-discriminating predictors for two scenarios of the original model, binary predictor and score function, where in the latter scenario the original model generates real score values in range $[0, 1]$. For the latter scenario, a non-discriminating predictor is constructed for each protected group, and a threshold is chosen to achieve a defined fairness goal.

Part II

Model Interpretability and Reliability

Towards DNN interpretability for visual intelligence, we propose two types of methods, relating input concepts to DNN decisions (Chapter 4), and designing an intrinsic interpretable DNN model via attention (Chapter 5). In order to achieve DNN reliability for visual intelligence, an intrinsic measurement on DNN decisions is defined in Chapter 6. To show how the aforementioned techniques holistically realize a contribution to interpretable and reliable deep learning for visual intelligence, further experiments and analyses are conducted for visual entailment task in Chapter 7. We now discuss the contributions of each of our works below.

Our method in Chapter 4 is built upon DNN visualization techniques for computer vision tasks, which could reveal concepts (parts, objects, etc, that exist in the input image) captured by the hidden states of the DNN. Most existing visualization methods hypothesize that the recognition of these concepts are instrumental in the decision a CNN reaches, however the nature of this relationship has not been well explored. The contribution of our work in Chapter 4 is to address this gap, by examining the quality of a concept's recognition by a CNN and the degree to which the recognitions are associated with CNN decisions.

In Chapter 5, we first propose a novel visual inference task called visual entailment, and a corresponding dataset, which requires real-life fine-grained scene reasoning compared to previous visual reasoning tasks such as VQA [Antol et al., 2015, Goyal et al., 2017] and CLEVR [Johnson et al., 2017a]. For this task, we aim to construct a transparent model, which is achieved by leveraging the attention mechanism, that generates model explanations along with the model prediction without explicit post processing. Our contributions in this work are, i) we propose a novel inference task, Visual Entailment, which requires a systematic cross-modal understanding between vision and natural languages, and ii) an intrinsic interpretable DNN model is proposed, which provides a transparent solution to tackle our visual entailment task.

In Chapter 6, we introduce a quantitative intrinsic notion of the certainty of DNN decisions as an extension of this dissertation study. The measure is based on an evaluation of whether the internal activations at each layer of a deep neural network of input are “close" to the activations produced by training examples that have the same class label as the prediction for the input. Such measurement can be used to assess the confidence or trustworthiness a user should have in the decisions made by a deep neural network, towards achieving the goal of DNN reliability.

In Chapter 7, we first discuss the demand for an integrated solution for interpretable and reliable DNNs, where three aspects are proposed for an ideal integrated solution. An integrated solution for interpretable and reliable DNNs for visual entailment task is introduced, which offers both technically useful and layman-friendly explanations, and potentially leverages the proposed certainty measurement to quantify the level of trustworthiness for visual entailment model decisions.

Relating Input Concepts to CNN Decisions

In this chapter, we will introduce a method we propose [Xie et al., 2017b] to explain decisions made by a CNN on a scene classification task[1]. The background information is covered in the background (Section 4.1), including the scene classification task and the interpretability target we want to achieve. Detailed methodologies are included in Section 4.2 where a greedy algorithm is designed to associate input concepts (generated via a visualization method) to CNN decisions. Related discussions and analysis are covered in Section 4.3, explaining the influences of input concepts, could be positive or negative influences, that result in the model behaviors.

4.1 Background

Many current methods to interpret CNNs use visualization techniques to highlight concepts, defined as semantically meaningful image areas, of the input seemingly relevant to a CNN's decision. The methods hypothesize that the recognition of these concepts are instrumental in the decision a CNN reaches, however the nature of this relationship has not been well explored. To address this gap, we examine the quality of a concept's recognition

[1]Portions of this chapter have been published [Xie et al., 2020] and is accepted by NIPS IEVDL 2017.

by a CNN and the degree to which the recognitions are associated with CNN decisions.

We specifically consider input concepts and decisions under a scene recognition task over the ADE20k dataset [Zhou et al., 2017]. The study is powered by a novel algorithm to compute how well any concept is recognized across the feature maps of a convolutional layer. Analysis along with concept types, including those that appear often within a scene, often across multiple scenes, and those unique to a scene reveal a weak relationship between correct decision making and concept recognition. Our study finds that the relationship is dampened by the recognition of 'sparse' concepts that seldom appear in the images of a scene and by 'misleading' concepts that appear often across the images of many different scenes. However, the recognition of concepts that are unique to the images of specific scenes promotes correct CNN decisions.

4.2 Concept Recognition

Studying the relationship between input concepts and CNN decisions requires a measure of how well such concepts are recognized by a CNN. We define a concept as being 'recognized' if there are a set of late stage convolutional layer nodes that only activate over the input because of the concept's presence. Whereas much of the research assumes that these nodes must lie within the same CNN feature map [Bau et al., 2017, Zintgraf et al., 2017], we assert that concept recognition could occur in a *distributed way*, across many feature maps at a convolutional layer. Past studies have suggested and demonstrated that neural networks learn a representation of input features in a distributed fashion [Carpenter and Grossberg, 1988, Bengio et al., 2003, Hinton, 1986]; thus, we do not consider the possibility that input concepts can only be recognized within a single feature map.

In the context of scene classification, the recognition of a concept (e.g. an annotated object) would be manifested by a set of (distributed) nodes (across multiple feature maps) that collectively respond to the input pixels representing the concept. If the set of nodes is a “good” recognizer of the concept, they should collectively respond to all pixels representing the concept, and over no pixels not representing the concept. We call a node activated if it takes on a non-zero value under a sigmoid or tanh non-linearity, or is > 0 under a ReLU non-linearity.

The deconvolution of a feature map recovers the pixels of an input image causing its nodes to activate [Zeiler and Fergus, 2014b, Zeiler et al., 2011, Yosinski et al., 2015]. Deconvolutions thus seem like a natural way to identify if input concepts in scenes are represented by a feature map: if the deconvolution of the feature map covers most pixels of a concept, we may consider it as ‘recognized’ by the feature map. However, patterns activating nodes in a feature map are not always consistent from image to image. We illustrate this point in Figure 4.1 where a feature map, taken from the last convolutional layer of AlexNet trained for object recognition, has its deconvolution computed for different input images. The deconvolution over the first cat image suggests that the feature map recognizes the facial features of a cat, or the texture of a cat’s fur. The deconvolution over the second image, however, recognizes nothing about the cat, and it is unclear if any concept in the third image is recognized by the feature map. Recent approaches for concept recognition find that only a limited number of feature maps consistently recognize a specific concept [Bau et al., 2017].

Instead of focusing on concept recognitions localized to a single feature map, Figure 4.2 summarizes our approach to find and evaluate concepts recognized *across* multiple feature maps in a convolutional layer. Given a binary segmentation mask of the concept and the deconvolutions of feature maps in the latest stage convolutional layer, a greedy algorithm selects the subset of feature maps that collectively “best" recognize the given concept according to a scoring function. The selected feature maps and a recognition quality score is then returned to the user. The specifics of the recognition scoring and the greedy algorithm are discussed next.

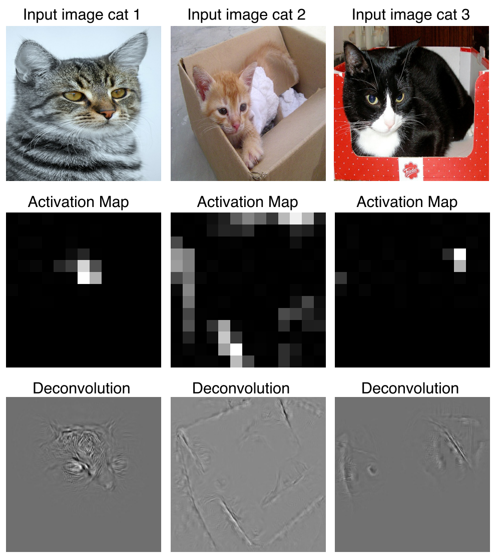


Figure 4.1: Deconvolutions of different cat images over the same feature map

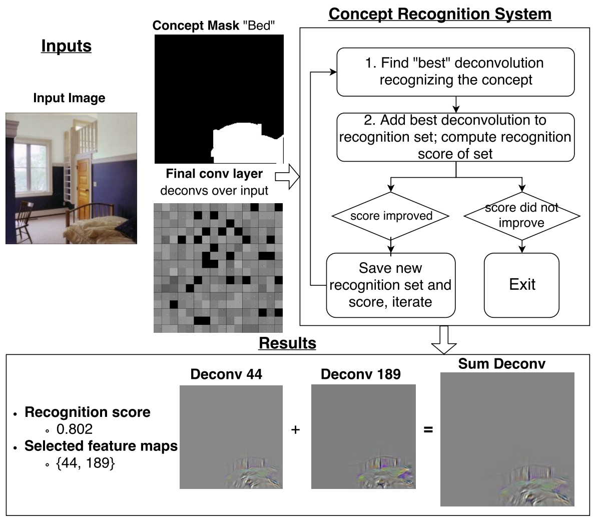

Inputs
Input Image
Concept Mask "Bed"
Final conv layer
deconvs over input
Concept Recognition System
1. Find "best" deconvolution recognizing the concept
2. Add best deconvolution to recognition set; compute recognition score of set
score improved
score did not improve
Save new recognition set and score, iterate
Exit
Results
Recognition score
0.802
Selected feature maps
{44, 189}
Deconv 44
+
Deconv 189
=
Sum Deconv

Figure 4.2: Concept recognition across feature maps

4.2.1 Recognition scoring

Ideally, the pixel area for a given concept should be covered by the deconvolutions of the selected feature maps as precisely as possible. The score should thus consider the combined coverage of the deconvolutions of the chosen feature maps over and not over the pixels of a concept. Based on this idea, we evaluate how well a set of feature maps $G_\mathfrak{c}$ recognizes a concept $\mathfrak{c}$ in an image ξ using a binary segmentation mask $M_\mathfrak{c}(\xi)$ that denotes the pixel positions of $\mathfrak{c}$ in ξ. We assume that $M_\mathfrak{c}(\xi)$ is available in a dataset or can be generated via object segmentation methods [Chen et al., 2016]. From the set of deconvolutions $D_\mathfrak{c}(\xi) = \{D_i(\xi)\}$ of $G_\mathfrak{c}$ with respect to ξ and their combined sum $D_\mathfrak{c}^{\text{sum}}(\xi) = \sum D_\mathfrak{c}(\xi)$, we define $\mathcal{D}_\mathfrak{c}(\xi)$ as the set of the positions of the pixels of $D_\mathfrak{c}^{\text{sum}}(\xi)$ representing node activations across $G_\mathfrak{c}$. Then a concept recognition score $S_\mathfrak{c}(G_\mathfrak{c}, \xi)$ is defined with a Jaccard like similarity measure similar to Bau et al. [2017]:

$$S_\mathfrak{c}(G_\mathfrak{c}, \xi) = \frac{|M_\mathfrak{c}(\xi) \cap \mathcal{D}_\mathfrak{c}(\xi)|}{|M_\mathfrak{c}(\xi) \cup \mathcal{D}_\mathfrak{c}(\xi)|}$$

Table 4.1: Scene classes considered

Label	Class Name	Num. images	Label	Class Name	Num. images
0	bathroom	671	8	mountain snowy	132
1	street	2038	9	conference room	168
2	office	112	10	skyscraper	320
3	building facade	228	11	corridor	110
4	airport terminal	107	12	bedroom	1389
5	game room	99	13	dining room	412
6	living room	697	14	highway	295
7	hotel room	160	15	kitchen	652

4.2.2 Recognition algorithm

We devise a greedy algorithm to identify the $G_\mathfrak{c}$ that best recognizes $\mathfrak{c}$ listed as Algorithm 1. The intuition behind the greedy approach is to find a set of feature maps that recognizes $\mathfrak{c}$ well, is as small as possible, and is composed of feature maps that minimally 'overlap', e.g. recognizes the same parts or qualities of a concept. The latter two criteria capture the idea that a good distributed representation is one where the nodes of each feature map in the set activate over different and significant parts of the concept. Thus, in each greedy iteration, the algorithm searches for the feature map whose addition to $G_\mathfrak{c}$ would yield the largest improvement in recognition score $S_\mathfrak{c}(G_\mathfrak{c}, \xi)$. Large improvements would only be possible if the newly added feature map activates over pixels representing $\mathfrak{c}$ that no other feature map in $G_\mathfrak{c}$ activates over. Moreover, this feature map cannot have significant activations over pixels that do not represent $\mathfrak{c}$ without reducing $S_\mathfrak{c}$. Greedy iterations continue until there is no feature map whose inclusion would yield an improvement in score greater than Δ. $\Delta = 0.01$ is used in the experiments below.

4.3 Recognition analysis

We use Algorithm 1 to recognize each concept in each given input image, and study the relationship between its recognition quality and a CNN's scene classification accu-

Algorithm 1 Concept Localization

```
1: procedure GREEDY_SELECTION($G$, $D$, $M_\mathfrak{c}(\xi)$, $\Delta$)
2:     $S_\mathfrak{c} \leftarrow 0$                        ▷ Score of the selected set of feature maps
3:     $G_\mathfrak{c} \leftarrow \{\}$                       ▷ Set of selected feature maps
4:     while True do
5:         $tmp_s \leftarrow 0$
6:         $g \leftarrow$ null
7:         for $k = 1$ to $|G|$ do
8:             $K = G_\mathfrak{c} \cup G^k$      ▷ Add candidate feature map $G^k \in G$ to the selected set
9:             $D^K(\xi) = \sum_{k \in K} D^k(\xi)$   ▷ Sum the deconvolutions $D^k$ of the feature maps in $K$
10:            $S_\mathfrak{c}(K, \xi) = \frac{|M_\mathfrak{c}(\xi) \cap \mathcal{D}^K(\xi)|}{|M_\mathfrak{c}(\xi) \cup \mathcal{D}^K(\xi)|}$   ▷ Find the new recognition score after adding $G^k$
11:            if $S_\mathfrak{c}(K, \xi) > tmp_s$ then     ▷ Is $G^k$ better than the best candidate found so far?
12:                $tmp_s \leftarrow S_\mathfrak{c}(K, \xi)$
13:                $g \leftarrow G^k$
14:        $G.remove(g)$                        ▷ Remove the selected feature map from $G$
15:        if $tmp_s - S_\mathfrak{c} > \Delta$ then   ▷ Does adding $g$ improve the score by more than $\Delta$?
16:            $S_\mathfrak{c} \leftarrow tmp_s$
17:            $G_\mathfrak{c}.append(g)$                ▷ Add $g$ to the feature map set and repeat
18:        else
19:            return $S_\mathfrak{c}, G_\mathfrak{c}$
```

racy. We consider an AlexNet [Krizhevsky et al., 2012] CNN model trained over the Places365 [Zhou et al., 2016b] scene dataset and fine-tune network weights using ADE20k [Zhou et al., 2017]. We only consider the subset of scenes in ADE20k having at least 99 example images. We choose this subset to ensure a sufficient number of examples are available for CNN training and to be able to take representative measurements of the CNN's ability to classifying a scene correctly. The 16 (out of the 1000+) scenes in ADE20k having at least 99 example images and are listed in Table 4.1[2]. 60% of the images from each class are randomly sampled as training data during fine-tuning and 40% for testing. The fine-tuned CNN achieves a 74.9% top-1 classification accuracy over the testing images after 30 training epochs, which is higher than the performance of other CNN scene classifiers [Zhou et al., 2016b], but we note that we only test over scenes that have an abundance of images in the ADE20K's training data.

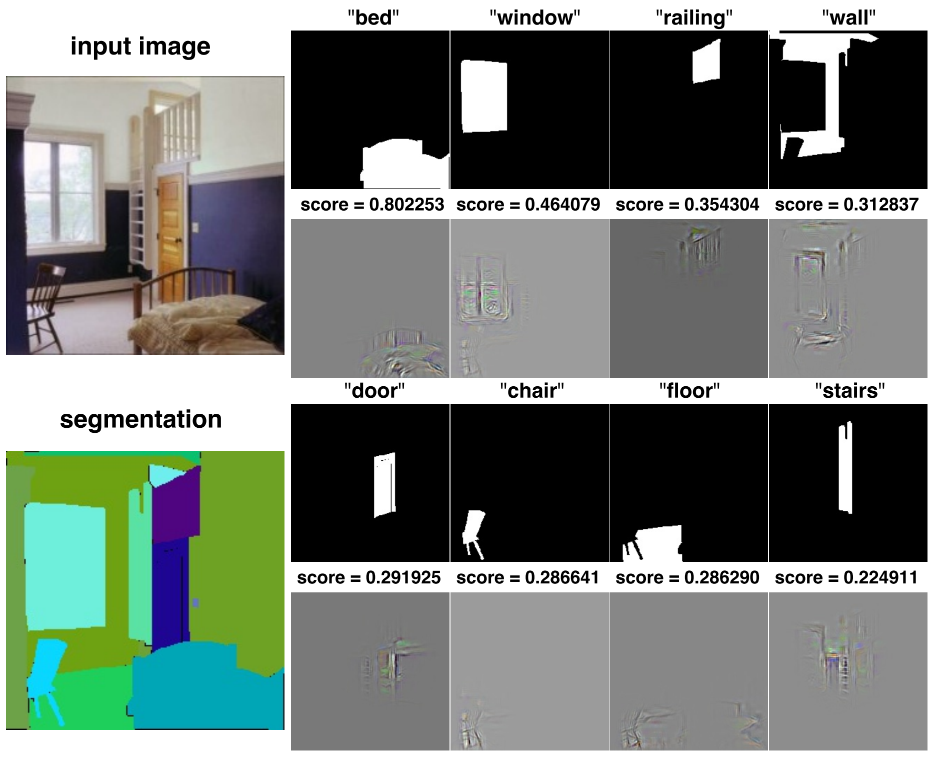


Figure 4.3: Concept recognition results for a given image

[2]We also omit the 'misc' class of ADE20k as it is a catch-all for hard to describe scenes, even though it has over 99 images.

We then randomly choose 50 images from each class and compute how well their concepts are recognized by the 256 feature maps in the last convolutional layer of the CNN. This sample of $50 \times 16 = 800$ images feature 370 distinct concepts. To get a sense of whether a recognition score is relatively “low" or “high", we plot the score distribution across all concepts in the sampled images in Figure 4.4. We note that the mean recognition score is 0.315 with a median 0.284, and the lower and upper quartiles are 0.174 and 0.429 respectively. Figure 4.3 illustrates the output of Algorithm 1 in a sampled bedroom scene. For the eight concepts annotated in this image, the binary segmentation mask, its label, a visualization of the sum of deconvolutions chosen by our greedy algorithm, and the recognition score are presented. The highest quality recognition is of the `bed` concept, with a score (0.802) well above the upper quartile of the recognition score distribution across all concepts, a summed deconvolution that captures texture information about the bed and the shape and patterning of the bed frame, and activates over few pixels that do not represent the bed concept. The `chair` concept has a lower recognition score (0.287) that happens to be close to the median of the concept recognition score distribution. In this case, the selected feature maps are able to recognize most parts of the chair, including its legs and back, but also happens to activate over some of the straight line and texture patterns of the wall and floor surrounding the chair. The `stairs` concept has the lowest score (0.225), caused by the feature maps' inability to activate over all pixels of the concept and also activate across pixels representing the nearby concepts (`wall` and `door`).

4.3.1 Recognition versus performance

We now explore the relationship between concept recognition and CNN performance. For each scene and its sampled images, we compare the average recognition score of concepts within a scene's images against the CNN's average classification accuracy of the scene. Figure 4.5 shows only a weak linear relationship (Pearson's correlation $\rho = 0.187$), al-

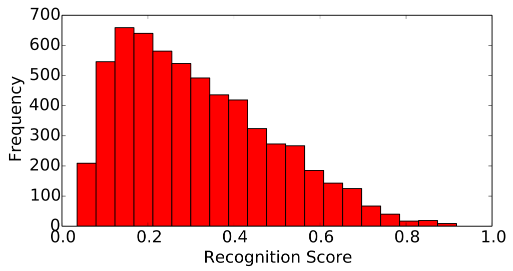


Figure 4.4: Recognition score distribution

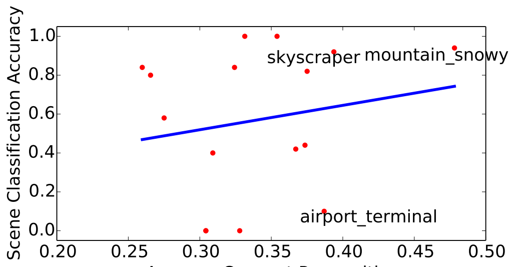


Figure 4.5: Recognition quality vs CNN's accuracy

though there are interesting observations for some scenes. The two scenes with the best classification and recognition scores are `skyscraper` and `mountain_snowy`, which are scenes whose images include concepts that are especially emblematic. For example, the `mountain` concept is captured well across `mountain_snowy` scenes ($\bar{S}^{\texttt{mountain\_snowy}}_{\texttt{mountain}} = 0.562$ where $\bar{S}^{\mathfrak{s}}_{\mathfrak{c}}$ denotes the average recognition of concept $\mathfrak{c}$ across the sampled scenes of $\mathfrak{s}$) and concepts like `skyscraper`, `sky`, and `building` are identified well in `skyscraper` scenes ($\bar{S}^{\texttt{skyscraper}}_{\texttt{sky}} = 0.532$, $\bar{S}^{\texttt{skyscraper}}_{\texttt{building}} = 0.362$, $\bar{S}^{\texttt{skyscraper}}_{\texttt{skyscraper}} = 0.407$). `airport_terminal` is a challenging scene for the CNN to identify despite achieving high average concept recognition. This may be due to strong recognitions for concepts like `floor` and `ceiling` ($\bar{S}^{\texttt{airport\_terminal}}_{\texttt{floor}} = 0.585$, $\bar{S}^{\texttt{airport\_terminal}}_{\texttt{ceiling}} = 0.559$) that appear in at least 45 of the 50 sampled `airport_terminal` images, but these concepts are generic and could apply to any kind of indoor scene. Concepts better capturing the notion of an airport terminal are also recognized, e.g., `armchair` ($\bar{S}^{\texttt{airport\_terminal}}_{\texttt{armchair}} = 0.555$) and `shops` ($\bar{S}^{\texttt{airport\_terminal}}_{\texttt{shops}} = 0.548$), but they emerge in only one of the sampled images.

4.3.2 Sparse concepts

The `airport_terminal` example suggests that there may be particular types of concepts that have stronger or weaker relationships to a CNN's decisions. We first consider 'sparse' concepts, which are concepts appearing in a small number of images within a scene (we quantify this notion with a *popularity* score in the sequel). Sparse concepts may not appear often enough during training for a CNN to learn to recognize well or to relate with a particular scene. For example, while the CNN is able to recognize the `armchair` and `shops` concepts in an `airport_terminal` well, their infrequency could mean the CNN does not have enough observations to establish a relationship between these concepts and the scene label.

Figure 4.6 explores the prevalence of concepts and how well they are recognized across each of the 16 scene classes. It illustrates that, for every class, there are a majority of concepts that emerge in less than 10 of the 50 images sampled from each scene. Scenes that are relatively uniform in the way they look, for instance `skyscraper`, `mountain_snowy`, and `street` scene, have fewer sparse concepts. Moreover, such scenes tend to have their non-sparse concepts recognized strongly by the CNN (reflected by the steeper slopes of the linear fits in their scatter plots). Scenes that are non-uniform in what they could look like, for example `bedroom`, `hotel_room`, and `dining_room` images that depict different styles and design, tend to exhibit a larger number of sparse concepts. But some of these sparse concepts have high recognition scores (resulting in shallower slopes of the linear fits in their scatter plots), suggesting that the CNN learns to recognize them. This may be because a sparse concept could be observed across a large number of different scenes. For example, although not every `bedroom` has a `chair`, one can imagine a `chair` to appear across a variety of different scenes, giving a CNN enough examples to learn to recognize this concept.

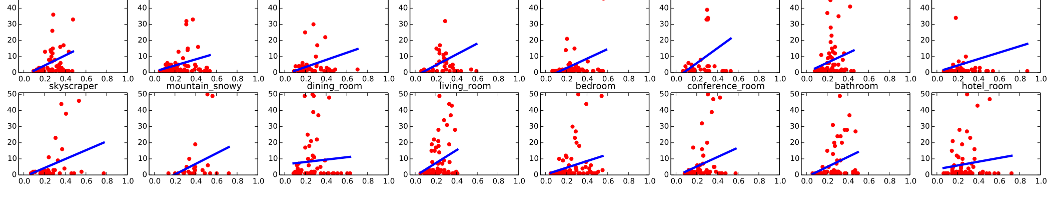


Figure 4.6: Average concept recognition (x-axis) vs. number of concept occurrences (y-axis) per scene

The figure and discussion suggest the following hypothesis: the fewer the number of sparse concepts present and the greater the number of well recognized non-sparse concepts appear across the images of a scene, the higher the chance is that the CNN can correctly identify the scene. Moreover, scenes whose images are dominated by a variety of sparse concepts should prove to be more challenging for the CNN to classify. To test this, we plot the slope of the linear fit of each scatter plot from Figure 4.6 against the CNN's accuracy for each scene in Figure 4.7. The moderate linear relationship (Pearson's $\rho = 0.444$) suggests that many non-sparse, well recognized concepts are associated with correct CNN decisions, lending support for the hypothesis.

4.3.3 Unique and misleading concepts

We now investigate non-sparse concepts further. Intuitively, non-sparse concepts may have greater benefit to correct CNN decisions if they appear across a smaller number of different types of scenes. For example, concepts like `sand` and `shell` may be present in many beach scenes, are closely associated with the notion of beach, and are unlikely to appear in other types of scenes. Thus, high quality recognition of `sand` and `shell` concepts would help a CNN to classify `beach` scenes correctly. On the other hand, non-sparse concepts emerging across a variety of scenes may be less helpful. For example, since we expect most images of indoor scenes to include concepts like `wall`, `floor`, or `ceiling`, their recognition may not help a CNN differentiate between different indoor scenes. In fact, these recognitions may be of limited help in the best case and could confuse or mislead a CNN to make a wrong classification in the worst case.

To explore these ideas, we compute a *uniqueness* score of a concept that reflects the variety of scenes it appears in. The uniqueness $U(\mathfrak{c})$ of a concept $\mathfrak{c}$ is calculated as:

$$U(\mathfrak{c}) = 1 - \frac{\text{\# of scene classes } \mathfrak{c} \text{ appears}}{\text{\# of scene classes}}$$

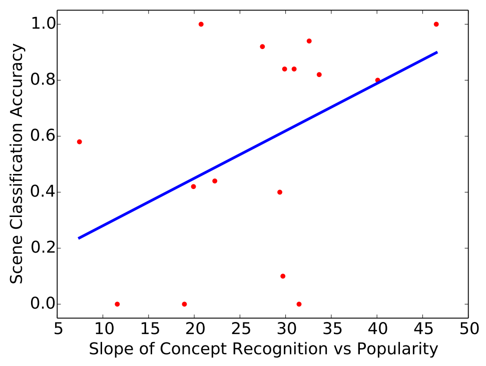


Figure 4.7: Slope of sparse concept recognition (Figure 4.6) vs CNN's accuracy

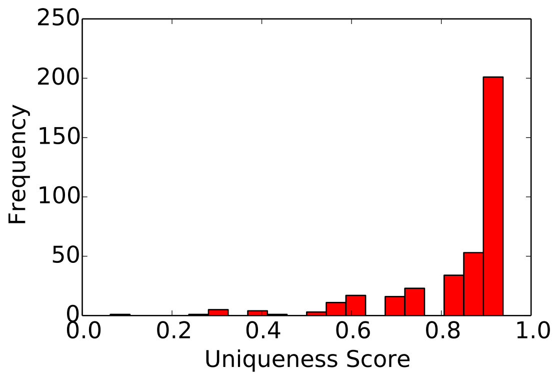


Figure 4.8: Uniqueness score distribution

Figure 4.8 gives the distribution of the uniqueness scores of each concept. It is skewed, with its average uniqueness score at 0.845, and its lower quartile, median, and upper quartile is 0.8125, 0.9375, and 0.9375 respectively. 210 of the 370 concepts appear in only one scene class, although many of these concepts are likely to be sparse. Following the fact that many of the scenes used in our analysis (listed in Table 4.1) are indoors, concepts with the least unique scores pertain to generic aspects of a room. For example, the concepts having the three lowest uniqueness scores are $U(\texttt{wall}) = 0.063$, $U(\texttt{floor}) = 0.25$, and $U(\texttt{door}) = U(\texttt{plant}) = U(\texttt{window}) = U(\texttt{ceiling}) = U(\texttt{picture}) = 0.3125$.

We hypothesize that the recognition of unique concepts helps a CNN make correct classifications, and that concepts with low uniqueness scores may 'mislead' a CNN. We evaluate this hypothesis by comparing the CNN's classification accuracy to the average recognition score calculated on "unique" concepts and "misleading" concepts respectively. A concept $\mathfrak{c}$ is labeled as "unique" if its uniqueness score $U(\mathfrak{c}) > \alpha$ for a uniqueness threshold α. However, we recall from Figure 4.6 that a number of unique concepts are likely to be 'sparse', thus hindering classification accuracy (Figure 4.7). We thus filter away sparse concepts by defining a *popularity* score $P(\mathfrak{c})$ with respect to some scene by:

$$P(\mathfrak{c}) = \frac{\text{\# of images } \mathfrak{c} \text{ appears in a scene class}}{\text{\# of images sampled from a scene class}}$$

and only consider concepts whose $P(\mathfrak{c}) > \beta$ for a popularity threshold β.

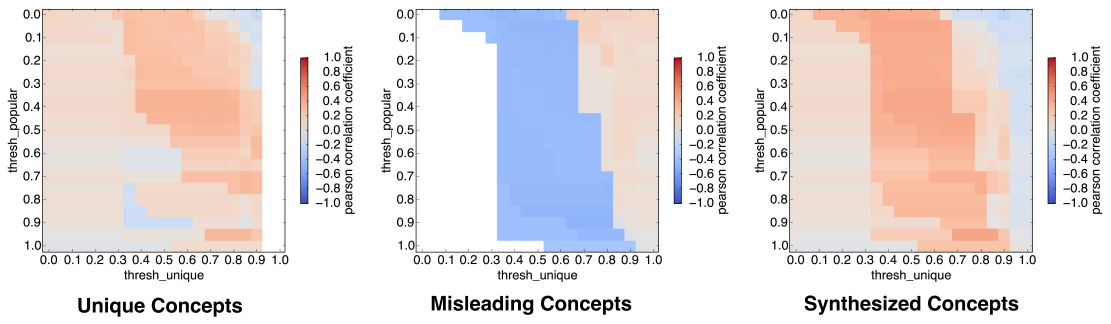


Figure 4.9: Heatmap for PCC calculated upon "unique" concept, "misleading" concept, and "synthesized" of unique and misleading concepts using different thresholds.

We then compute Pearson's correlation coefficient ρ between the CNN's accuracy over each scene class against the average recognition score on "unique" and "misleading" concepts respectively for various values of α and β. Figure 4.9 presents ρ over a grid of the two thresholds, varying their values in increments of 0.05 between 0 and 1. The left heatmap shows ρ when only unique concepts are considered. Most of the area shows a positive relationship between the unique concepts recognition quality and CNN accuracy. Larger uniqueness and popularity thresholds α and β, making the set of unique concepts even smaller, lead to an even stronger relationship. Note that there is no concept having $U(\mathfrak{c}) \geq 0.95$, causing empty cells in the rightmost two columns. The middle heatmap only considers misleading concepts. The shaded blue areas indicate a negative relationship between the misleading concepts recognition quality and the model performance. For most valid settings of β, when $U(\mathfrak{c}) < 0.7$, there exists a moderate strong negative correlation. This provides some evidence that the recognition of misleading concepts, e.g. those concepts appearing across many different scene types, may be hindering a CNN's ability to classify scenes correctly. The right heatmap reports ρ using a "synthesized" average concept recognition score, which is defined for each scene class by $S_{\text{syn}} = {(S_{\text{unique}}+1.0-S_{\text{mislead}})}/{2}$ where S_{unique} is the average concept recognition score over the unique concepts and S_{mislead} is the same but over misleading concepts. This synthetic score unifies the results from the unique and misleading heatmaps together in search of threshold settings that maximize ρ over unique concepts and minimize ρ over misleading concepts. We find the highest positive correlation of $\rho = 0.521$ using the synthetic scores when $\beta = 0.4$ and $\alpha = 0.55$. At these thresholds, we find $\rho = 0.454; (p = 0.078)$ over the unique concepts and $\rho = -0.528; (p = 0.036)$ on the misleading concepts. The p-values for these correlation scores, computed over $n = 16$ classes, indicate a significant negative correlation between misleading concept recognition and CNN's accuracy, and a moderate positive correlation between unique concept recognition and CNN's accuracy.

4.4 Conclusion

We investigated the relationship between CNN's recognition of input concepts and classification accuracy. A novel approach was developed to quantify how well a concept (specifically, an object in an image) is recognized across the latest convolutional layer of a CNN. Analysis using image object annotations in the ADE20k scene dataset revealed a weak relationship between the average recognition of image concepts in a scene and classification accuracy. We found evidence to suggest that the relationship is hindered by recognized concepts that are "sparse", or appear in a small number of images of a scene and by "misleading" concepts that appear in many images across many different scenes. Recognizing "unique" concepts, which appear often but in a limited set of scenes, is moderately positively correlated with CNN's classification accuracy.

Achieving Intrinsic DNN Explanations via Attention Mechanism

In this chapter, we propose an intrinsic DNN explanation method via leveraging attention mechanism[1]. By intrinsic we mean the model is designed in an intrinsic way such that it could provide explanations to its decisions naturally without adding extra process. The interpretable DNN model that we designed solves a novel task called visual entailment, involving both computer vision and natural languages, which will be introduced in Section 5.1 and 5.2. Details for the interpretable model we propose, Explainable Visual Entailment (EVE), are covered in Section 5.3. The experiment details and how to generate explanations via the EVE model are discussed in Section 5.4 and Section 5.5 respectively.

5.1 Introduction

The pursuit of "visual intelligence" is a long lasting theme of the machine learning community. While the performance of image classification and object detection has significantly improved in recent years [Krizhevsky et al., 2012, Simonyan and Zisserman, 2014, Szegedy et al., 2015, He et al., 2016], progress in higher-level scene reasoning tasks such as scene understanding is relatively limited [Wu et al., 2017].

[1]Portions of this chapter have been published [Xie et al., 2018, 2019] and is accepted by NeurIPS ViGIL 2018.

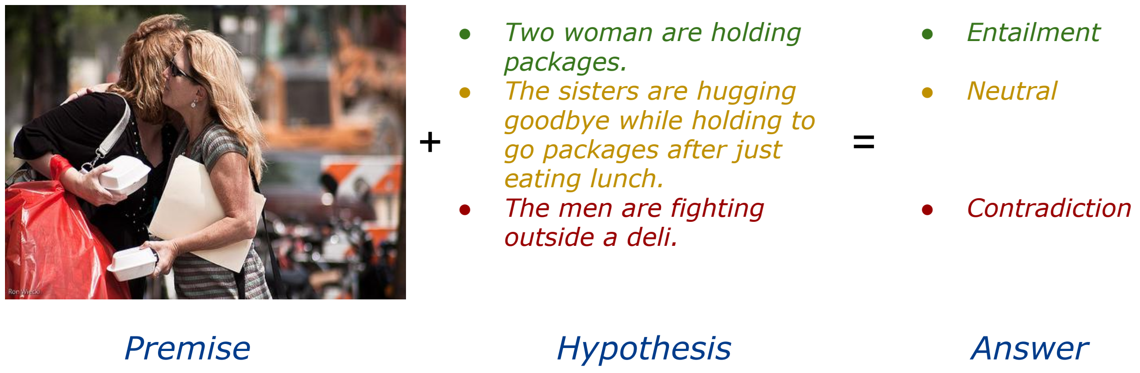
Recently, several datasets, such as VQA-v1.0 [Antol et al., 2015], VQA-v2.0 [Goyal et al., 2017], CLEVR [Johnson et al., 2017a], Visual7w [Zhu et al., 2016], Visual Genome [Krishna et al., 2016], COCO-QA [Ren et al., 2015], and models [Johnson et al., 2017b, Santoro et al., 2017, Hudson and Manning, 2018, Jiang et al., 2018b, Anderson et al., 2018, Teney et al., 2017, Fukui et al., 2016, Kim et al., 2018b] have been used to measure the progress in understanding the interaction between vision and language modalities. However, the quality of the widely used VQA-v1.0 dataset [Antol et al., 2015] suffers from a natural bias [Goyal et al., 2017]. Specifically, there is a long tail distribution of answers and also a question-conditioned bias where, questions may hint at the answers, such that the correct answer may be inferred without even considering the visual information. Besides, many questions in the VQA-v1.0 dataset are simple and straightforward and do not require compositional reasoning from the trained model. VQA-v2.0 [Goyal et al., 2017] has been proposed to reduce the dataset "bias" considerably in VQA-v1.0 by associating each question with relatively balanced different answers. However, the questions are rather straight-forward and require limited fine-grained reasoning. CLEVR dataset [Johnson et al., 2017a], is designed for fine-grained reasoning and consists of compositional questions such as "What size is the cylinder that is left of the brown metal thing that is left of the big sphere?". This kind of question requires learning fine-grained reasoning based on visual information. However, CLEVR is a synthetic dataset, and visual information and sentence structures are very similar across the dataset. Hence, models that provide good performance on CLEVR dataset may not generalize to real-world settings.

To address the above limitations, we propose a novel inference task, *Visual Entailment* (VE), which requires fine-grained reasoning in real-world settings. The design is derived from Text Entailment (TE) [Dagan et al., 2006] task. In our VE task, a real world image premise P_{image} and a natural language hypothesis H_{text} are given, and the goal is to determine if H_{text} can be concluded given the information provided by P_{image}. Three labels *entailment*, *neutral* or *contradiction* are assigned based on the relationship conveyed by the

(P_{image}, H_{text}).

- *Entailment* holds if there is enough evidence in P_{image} to conclude that H_{text} is true.
- *Contradiction* holds if there is enough evidence in P_{image} to conclude that H_{text} is false.
- Otherwise, the relationship is *neutral*, implying the evidence in P_{image} is insufficient to draw a conclusion about H_{text}.

The main difference between VE and TE task is, the premise in TE in a natural language sentence P_{text}, instead of an image premise P_{image}. Note that the existing of "neutral" makes the VE task more challenging compared to previous "yes-no" VQA tasks, since "neutral" requires the model to conclude the uncertainty between "entailment (yes)" and "contradiction (no)". Figure 5.1 illustrates a VE example, which is from the SNLI-VE dataset we propose below, that given an image premise, the three different text hypotheses lead to different labels.



- *Two woman are holding packages.*
- *The sisters are hugging goodbye while holding to go packages after just eating lunch.*
- *The men are fighting outside a deli.*

- *Entailment*
- *Neutral*
- *Contradiction*

Figure 5.1: An example from SNLI-VE dataset

5.2 Visual Entailment: A Novel Visual Reasoning Task

5.2.1 Formal Definition

We introduce a dataset $\mathcal{D}$ for VE task structured as $\{(i_1, h_1, l_1), \ldots (i_n, h_{m_n}, l_{m_n})\}$, where (i_k, h_s, l_s) is an instance from $\mathcal{D}$, with i_k, h_s, and l_s denoting an image premise, a text hypothesis and a class label, respectively. It is worth noting that each image i_k is used multiple times with different labels given distinct hypotheses $\{h_{m_k}\}$.

Three labels e, n, or c are assigned based on the relationship conveyed by (i_k, h_s). Specifically, i) e (entailment) is assigned if $i_k \models h_s$, ii) n (neutral) is assigned if $i_k \not\models h_s \wedge i_k \not\models \neg h_s$, iii) c (contradiction) is assigned if $i_k \models \neg h_s$.

5.2.2 Visual Entailment Dataset

Dataset criteria

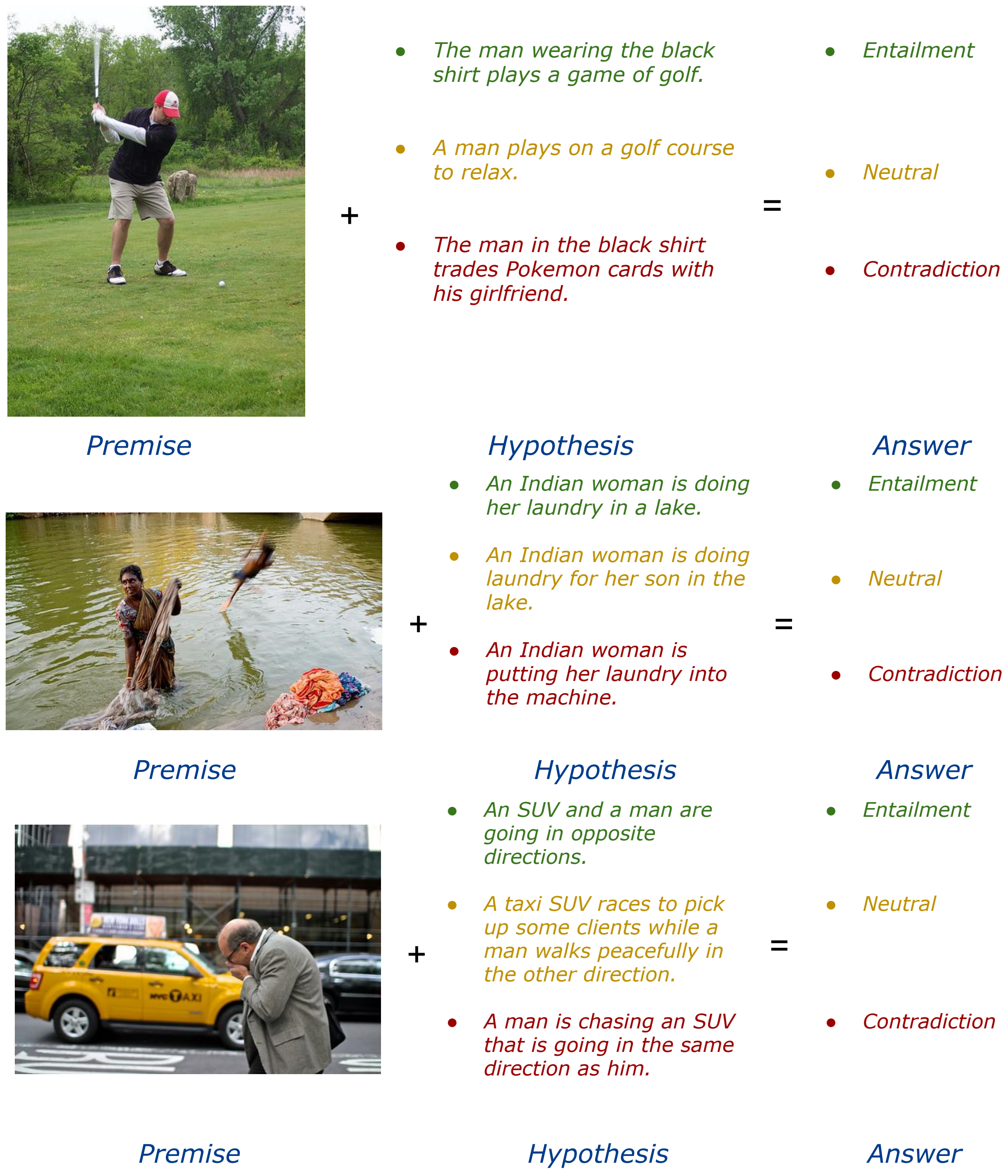
Based on the vision community's experience with SNLI, VQA-v1.0, VQA-v2.0, and CLEVR, there are four *criteria* in developing an effective dataset:

1. *Structured set of real-world images.* The dataset should be based on real-world images and the same image can be paired with different hypotheses to form different labels.

2. *Fine-grained.* The dataset should enforce fine-grained reasoning about subtle changes in hypotheses that could lead to distinct labels.

3. *Sanitization.* No instance overlapping across different dataset partitions. One image can only exist in a single partition.

4. *Account for any bias.* Measure the dataset bias and provide baselines to serve as the performance lower bound for potential future evaluations.


- The man wearing the black shirt plays a game of golf. = Entailment
- A man plays on a golf course to relax. = Neutral
- The man in the black shirt trades Pokemon cards with his girlfriend. = Contradiction

- An Indian woman is doing her laundry in a lake. = Entailment
- An Indian woman is doing laundry for her son in the lake. = Neutral
- An Indian woman is putting her laundry into the machine. = Contradiction

- An SUV and a man are going in opposite directions. = Entailment
- A taxi SUV races to pick up some clients while a man walks peacefully in the other direction. = Neutral
- A man is chasing an SUV that is going in the same direction as him. = Contradiction

Figure 5.2: More examples from SNLI-VE dataset

SNLI-VE Construction

We now describe how we construct SNLI-VE, which is a dataset for VE tasks. We build the dataset SNLI-VE based on two existing datasets, Flickr30k [Young et al., 2014] and SNLI

[Bowman et al., 2015]. **Flickr30k** is a widely used image captioning dataset containing 31,783 images and 158,915 corresponding captions. The images in Flickr30k consist of everyday activities, events and scenes [Young et al., 2014], with 5 captions per image generated via crowdsourcing. **SNLI** is a large annotated TE dataset built upon Flickr30k captions. Each image caption in Flickr30k is used as a text premise in SNLI. The authors of SNLI collect multiple hypotheses in the three classes - *entailment*, *neutral*, and *contradiction* - for a given premise via Amazon Mechanical Turk [Turk, 2012], resulting in about 570K (P_{text}, H_{text}) pairs. Data validation is conducted in SNLI to measure the label agreement. Specifically, each (P_{text}, H_{text}) pair is assigned a *gold label*, indicating the label is agreed by a majority of crowdsourcing workers (at least 3 out of 5). If such a consensus is not reached, the gold label is marked as "-".

Since SNLI was constructed using Flickr30k captions, for each (P_{text}, H_{text}) pair in SNLI, it is feasible to find the corresponding Flickr30k image through the annotations in SNLI. This enables us to create a structured VE dataset based on both. Specifically, for each (P_{text}, H_{text}) pair in SNLI with an agreed gold label, we replace the text premise with its corresponding Flickr30k image, resulting in a (P_{image}, H_{text}) pair in SNLI-VE. Figures 5.1 and 5.2 illustrate examples from the SNLI-VE dataset. SNLI-VE naturally meets the aforementioned *criterion 1* and *criterion 2*. Each image in SNLI-VE are real-world ones and is associated with distinct labels given different hypotheses. Furthermore, Flickr30k and SNLI are well-studied datasets, allowing the community to focus on the new task that our paper introduces, rather than spending time familiarizing oneself with the idiosyncrasies of a new dataset.

A sanity check is applied to SNLI-VE dataset partitions in order to guarantee the *criterion 3*. We notice the original SNLI dataset partitions does not consider the arrangement of the original caption images. If SNLI-VE directly adopts the original partitions from SNLI, all images in validation or testing partitions exist in the training partitions, violating *criterion 3*. To amend this, we disjointedly partition SNLI-VE by images following the

	Training	**Validation**	**Testing**
#Image	29,783	1,000	1,000
#Entailment	176,932	5,959	5,973
#Neutral	176,045	5,960	5,964
#Contradiction	176,550	5,939	5,964
Vocabulary Size	29,550	6,576	6,592

Table 5.1: SNLI-VE dataset

	SNLI-VE	**VQA-v2.0**	**CLEVR**
Partition Size:			
Training	529,527	443,757	699,989
Validation	17,858	214,354	149,991
Testing	17,901	555,187	149,988
Question Length:			
Mean	7.4	6.1	18.4
Median	7.0	6.0	17.0
Mode	6	5	14
Max	56	23	43
Vocabulary Size	32,191	19,174	87

Table 5.2: Dataset comparison summary

partition in Gong et al. [2014] and make sure instances with different labels are of similar numbers across training, validation, and testing partitions as shown in Table 5.1.

Regarding *criterion 4*, since SNLI has already been extensively studied, we are aware that there exists a hypothesis-conditioned bias in SNLI as recently reported by Gururangan *et al.* [Gururangan et al., 2018]. Though the labels in SNLI-VE are distributed evenly across dataset partitions, SNLI-VE still inevitably suffers from this bias inherently. Therefore, we provide a hypothesis-only baseline in Section 5.4.1 to serve as a performance lower bound.

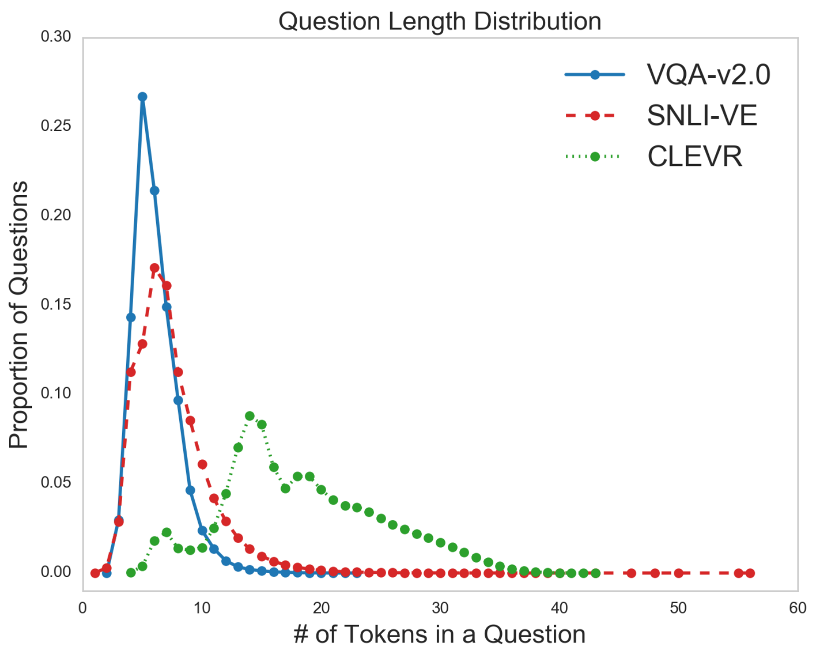


Figure 5.3: Question length distribution

5.2.3 SNLI-VE and VQA Datasets

We further compare our SNLI-VE dataset with the two widely used VQA datasets, VQA-v2.0 and CLEVR. The comparison focuses on the *questions* (for SNLI-VE dataset, we consider a hypothesis as a question). Table 5.2 is a statistical summary about the questions from three datasets. Before generating Table 5.2, questions are prepossessed by three steps: *i)* split into words, *ii)* lower case all words, *iii)* removing punctuation symbols {‘’“”,.-?!}. Figure 5.3 depicts a detailed question length distribution.

According to Table 5.2, among the three datasets, our SNLI-VE dataset, which contains the smallest total number of questions (summing up training, validation and testing partitions), has the largest vocabulary size. The maximum question length in SNLI-VE is 56, which is the largest among these three datasets, and represents real-world descriptions. Both the mean and median lengths are larger than VQA-v2.0 dataset. The question length distribution of SNLI-VE, as shown in Figure 5.3, is quite heavy-tailed in contrast to the others. These observations indicate that the text in SNLI-VE may be difficult to handle compared to VQA-v2.0 for certain models. As for CLEVR dataset, even though most sentences are much longer than SNLI-VE as shown in Figure 5.3, the vocabulary size is only 87. We believe this is due to the synthetic nature of CLEVR, which also indicates models that achieve high-accuracy on CLEVR may not be able to generalize to our SNLI-VE dataset.

5.3 EVE: Explainable Visual Entailment System

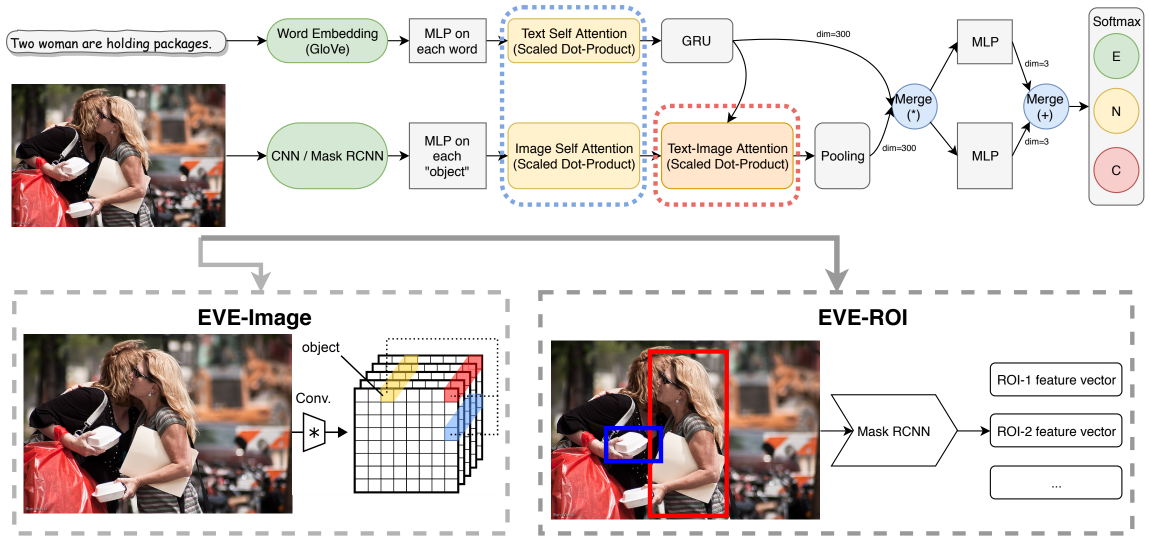


Figure 5.4: Our model EVE combines image and ROI information to model fine-grained cross-modal information

In order to tackle the VE task we propose, we design an intrinsic explainable VE architecture, as shown in Figure 5.4, that is based on the Attention Top-Down/Bottom-Up model [Anderson et al., 2018]. Similar to the Attention Top-Down/Bottom-Up, our EVE architecture is composed of a text and an image branch. The text branch extracts features from the input text hypothesis H_{text} through an RNN. The image branch generates image features from P_{image}. The features produced from the two branches are then fused and projected through fully-connected (FC) layers towards predicting the final conclusion. The image features can be configured to take the feature maps from a pre-trained convolutional neural network (CNN) or ROI-pooled image regions from a region of interest (ROI) proposal network (RPN).

We build two model variants, *EVE-Image* and *EVE-ROI*, for image and ROI features, respectively. EVE-Image incorporates a pre-trained ResNet101 [He et al., 2016], which generates k feature maps of size $d \times d$. For each feature map position, the feature vector across all the k feature maps is considered as an *object*. As a result, there are a total number of $d \times d$ objects of feature size k for an input image. In contrast, the EVE-ROI variant takes ROIs as objects extracted from a pre-trained Mask R-CNN [Matterport].

In order to accurately solve this cross-model VE task, we need: both a mechanism to identify the salient features in images and text inputs and a cross-modal embedding to effectively learn the image-text interactions, which are addressed by employing *self-attention* and *text-image attention* techniques in the EVE model respectively. We next describe the design and implementation of the mechanisms in EVE model.

5.3.1 Self-Attention

EVE utilizes self-attention [Vaswani et al., 2017] in both text and image branches as highlighted with the dotted blue frame in Figure 5.4. Since the hypothesis in SNLI-VE can be relatively long and complex, self-attention helps focus on important keywords in a sentence that relate to each other. The text branch applies self-attention to the projected word embeddings from a multi-layer perceptron (MLP). It is worth noting that although word embeddings, either from GloVe or other existing models, may be fixed, the MLP transformation is able to be trained to generate adaptive projected word embeddings. Similarly, the image branch applies the self-attention to projected image regions either from the aforementioned feature maps or ROIs in expectation of capturing the hidden relations between elements in the same feature space.

Specifically, we use the scaled dot product (SDP) attention in [Vaswani et al., 2017] to capture this hidden information:

$$Att_{\text{sdp}} = \text{softmax}(\frac{RQ^T}{\sqrt{d_k}})) \tag{5.1}$$

$$Q_{Att} = Att_{\text{sdp}}Q \tag{5.2}$$

where $Q \in \mathbb{R}^{M \times d_k}$ is the *query* feature matrix and $R \in \mathbb{R}^{N \times d_k}$ is the *reference* feature matrix. M and N represent the number of features vectors in matrix Q and R respectively, and d_k denotes the dimension of each feature vector. $Att_{\text{sdp}} \in \mathbb{R}^{N \times M}$ is the resulting attention mask for Q given R. Each element a_{ij} in Att_{sdp} represents how much weight (before scaled by $\frac{1}{\sqrt{d_k}}$ and normalized by softmax) the model should put on each query feature vector $q_{j \in \{1,2,\ldots,M\}} \in \mathbb{R}^{d_k}$ in Q w.r.t. each reference feature vector $r_{i \in \{1,2,\ldots,N\}} \in \mathbb{R}^{d_k}$ in R. The attended query feature matrix $Q_{Att} \in \mathbb{R}^{N \times d_k}$ is the weighted and fused version of the original query feature matrix Q, calculated by the matrix dot product between the attention mask Att_{sdp} and the query feature matrix Q. Note that for the self-attention, the query matrix $Q \in \mathbb{R}^{M \times d_k}$ and the "reference" matrix $R \in \mathbb{R}^{N \times d_k}$ are the same matrix.

5.3.2 Text-Image Attention

Multi-modal tasks such as phrase grounding [Chen et al., 2017a] demonstrate that high-quality cross-modal feature interactions improve the overall performance. The dotted red frame highlighted area in Figure 5.4 shows that EVE incorporates the text-image attention to relevant image regions based on the text embedding from the GRU. The feature interaction between the text and image regions are computed using the same SDP technique introduced in Section 5.3.1, serving as the attention weights. The weighted features of image regions are then fused with the text features for further decision making. Specifically, for the text-image attention, the query matrix $Q \in \mathbb{R}^{M \times d_k}$ is the image features while the "reference" matrix $R \in \mathbb{R}^{N \times d_k}$ is the text features. Note that although Q and R are from different feature spaces, the dimension of each feature vector is projected to be the same d_k in respective branches for ease of the attention calculation.

5.4 Experiment Details

In this section, we evaluate EVE as well as several other baseline models on SNLI-VE. Most of the baselines are existing or previous well-performed VQA architectures. The performance results of all models are listed in Table 5.3.

Model Name	Val Acc Overall (%)	Val Acc Per Class (%) C	N	E	Test Acc Overall (%)	Test Acc Per Class (%) C	N	E
Hypothesis Only	66.68	67.54	66.90	65.60	66.71	67.60	67.71	64.83
Image Captioning	67.83	66.61	69.23	67.65	67.67	66.25	70.69	66.08
Relational Network	67.56	67.86	67.80	67.02	67.55	67.29	68.86	66.50
Attention Top-Down	70.53	70.23	68.66	72.71	70.30	69.72	69.33	71.86
Attention Bottom-Up	69.34	**71.26**	70.10	66.67	68.90	70.52	**70.96**	65.23
EVE-Image*	**71.56**	71.04	**70.55**	73.10	**71.16**	**71.56**	70.52	71.39
EVE-ROI*	70.81	68.55	68.78	**75.10**	70.47	67.69	69.45	**74.25**

Table 5.3: Model performance on SNLI-VE dataset

All models are implemented in PyTorch. We use the pre-trained GloVe.6B.300D for word embedding [Pennington et al., 2014], where 6B is the corpus size and 300D is the embedding dimension. Input hypotheses are padded to the maximum sentence length in a batch. Note we do not truncate the sentences because unlike VQA where the beginning of questions typically indicates what is asked about, labels of VE task may depend on keywords or small details at the end of sentences. For example, truncating the hypothesis "The person who is standing next to the tree and wearing a blue shirt is playing _____" inevitably loses the key detail and changes the conclusion. In addition, the maximum sentence length in SNLI is 56, which is much larger than 23 in VQA-v2.0 as shown in Table 5.2. Always padding to the dataset maximum is not necessarily efficient for training. As a consequence, we opt for padding to the batch-wise maximum sentence length.

Unless explicitly mentioned, all models are trained using a cross-entropy loss function optimized by the Adam optimizer with a batch size of 64. We use an adaptive learning rate scheduler which reduces the learning rate whenever no improvement on the validation dataset for a period of time. The initial learning rate and weight decay are both set to be $1e-4$. The maximum number of training epochs is set to 100. We save a checkpoint whenever the model achieves a higher overall validation accuracy. The final model checkpoint selected for testing is the one with the highest lowest per class accuracy in case the model performance is biased towards particular classes. The batch size is set as 32 for validation and testing. In the following, we discuss the details for each baseline.

5.4.1 Hypothesis Only

This baseline verifies the existing data bias in the SNLI dataset, as mentioned by Gururangan *et al.* [Gururangan et al., 2018] and Vu *et al.* [Vu et al., 2018], by using hypotheses only without the image premise information.

The model consists of a *text processing component* followed by two FC layers. The text processing component is used to extract the text feature from the given hypothesis. It first generates a sequence of word-embeddings for the given text hypothesis. The embedding sequence is then fed into a GRU [Chung et al., 2014] to output the text features of dimension 300. The input and output dimensions of the two FC layers are [300, 300] and [300, 3] respectively.

Without any premise information, this baseline is supposed to make a random guess out of the three classes but the resulting accuracy is up to 67%, implying the existence of a dataset bias. We do not intend to rewrite the hypotheses in SNLI to reduce the bias but instead, aim at using the premise (image) features to outperform the hypothesis only baseline.

5.4.2 Image Captioning

Since the original SNLI premises are image captions, a straightforward idea to address VE is to first apply an image caption generator to convert image premises to text premises and then followed by a TE classifier. Particularly, we adopt the PyTorch tutorial implementation [Choi] as a caption generator. A pre-trained ResNet152 serves as the image encoder while the caption decoder is a long short-term memory (LSTM) network. Once the image caption is generated, the image premise is replaced with the caption and the original VE task is reduced to a TE task. Similar to the Hypothesis-Only baseline, the TE classifier is composed of two text processing components to extract text features from both the premise and hypothesis. The text features are fused and go through two FC layers with input and output dimensions of [600, 300] and [300, 3] for the final prediction.

The resulting performance achieves a slightly higher accuracy of 67.83% and 67.67% on the validation and testing partitions over the Hypothesis-Only baseline, implying that the generated image caption premise does not improve much. We suspect that the generated captions may not cover the necessary information in the image as required by the hypothesis to make the correct conclusion. This is possible in a complex scene where exhaustive enumeration of captions may be needed to cover every detail potentially described by the hypothesis.

5.4.3 Relational Network

The Relational Network (RN) baseline is based on [Santoro et al., 2017] which is proposed to tackle the CLEVR dataset with high accuracy. There are an image branch and a text branch in the model. The image branch extracts image features in a similar manner as EVE, as described in Section 5.3, but without self-attention. The text branch generates the hypothesis embedding through an RNN. The highlight of RN is to capture pairwise feature interactions between image regions and the text embedding. Each pair of image region feature and question embedding goes through an MLP. The final classification takes the element-wise sum over the MLP output for each pair as input.

Despite the high accuracy on the synthetic dataset CLEVR, RN only achieves a marginal improvement on SNLI-VE at the accuracy of 67.56% and 67.55% on the validation and testing partitions. This may be attributed to the limited representational power of RN that fails to produce effective cross-modal feature fusion of the natural image premises and the free-form text hypothesis input from SNLI-VE.

5.4.4 Attention Top-Down and Bottom-Up

We consider the Attention Top-Down and Attention Bottom-Up baselines based on the winner of VQA challenge 2017 [Anderson et al., 2018]. Similar to the RN baseline, there is an image branch and a text branch. The difference between the image branches in Attention Top-Down and Attention Bottom-Up is similar to our EVE. The image features of Attention Top-Down come from the feature maps generated from a pre-trained CNN. As for Attention Bottom-Up, the image features are the top 10 ROIs extracted from a pre-trained Mask-RCNN implementation [He et al., 2017]. No self-attention is applied in both image and text branches. Moreover, the text-image attention is implemented by feeding the concatenation of both image and text features into an FC layer to derive the attention weights rather than using SDP as described in Section 5.3.1. Then the attended image features and text features are projected separately and fused by dot product. The fused features go through two different MLPs. The element-wise sum of both MLP output serves as the final features for classification.

The VQA 2017 winner model, Attention Top-Down, achieves an accuracy of 70.53% and 70.30% on the validation and testing partitions respectively, implying cross-modal attention is the key to effectively leveraging image premise features. The Attention Bottom-Up model using ROIs also achieves a good accuracy of 69.34% and 68.90% on the validation and testing partitions. The reason why Attention Bottom-Up performs worse than

Attention Top-Down could be possibly due to lack of background information in ROI features and ROI feature quality. It is not guaranteed that those top ROIs cover necessary details described by the hypothesis. However, even with more than 10 ROIs, we observe no significant improvement in performance.

5.4.5 EVE-Image and EVE-ROI

The details of our EVE architecture have been described in Section 5.3. EVE-Image achieves the best performance of 71.56% and 71.16% accuracy on the validation and testing partitions respectively. The performance of EVE-ROI is similar, with an accuracy of 70.81% and 70.47%, possibly suffering from similar issues as the Attention Bottom-Up model. However, the improvement is likely due to the introduction of self-attention and text-image attention through SDP that potentially captures the hidden relations in the same feature space and better attended cross-modal feature interaction.

5.5 Model Explanations

Our EVE model achieves the best performance compared to other baselines [Xie et al., 2019]. The explainability of EVE is attained using attention visualizations in the areas of interest in the image premise given the hypothesis. It is worth noting that attention is designed to be naturally embedded as a model component, and is jointly trained with the model and also directly utilized for generating model decisions, thus is considered as an intrinsic way to achieve model interpretability. Figure 5.5 and 5.6 illustrate two visualization examples of the *text-image attention* from EVE-Image and EVE-ROI respectively. The image premise of the EVE-Image example is shown on the left of Figure 5.5, and the corresponding hypothesis is *“A human playing guitar”*. On the right of Figure 5.5, our EVE-Image model successfully attends to the guitar area, leading to the correct conclusion: *entailment*. In Figure 5.6, our EVE-ROI focuses on the children and the sand area in the image premise, leading to the *contradiction* conclusion for the given hypothesis *"Two children are swimming in the ocean."*

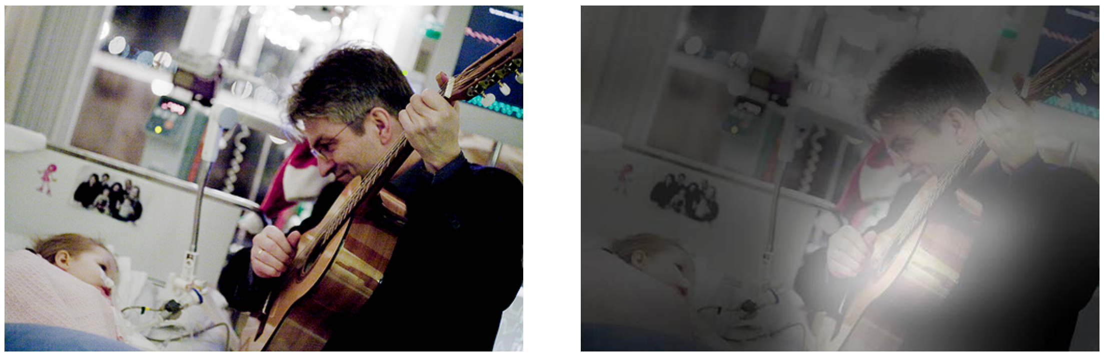


Figure 5.5: An attention visualization for EVE-Image

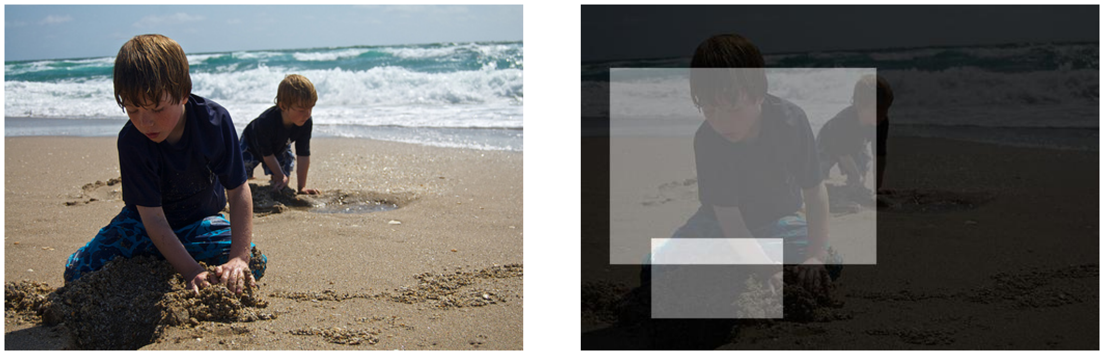


Figure 5.6: An attention visualization for EVE-ROI

An Intrinsic Certainty Measure for Deep Neural Network Decisions

In this chapter, we introduce a simple but effective measure to quantify the level with which an analyst can trust the decision of a deep neural network (DNN), as an extension of this dissertation study. The measure is intrinsic in the sense that it evaluates the characteristics of internal network activation patterns for an input, across multiple layers, to find anomalous activation patterns conditioned on the output of the network. Background knowledge and detailed methodology are included in Section 6.1 where the certainty score is designed upon intermediate activations of the network. Experiment results for MNIST, Fashion-MNIST, and Tiny-ImageNet across multiple deep learning architectures are discussed in Section 6.2. It demonstrates that the measure can promote the reliability of deep learning models by reporting the degree of trustworthiness for an input along with its decision.

6.1 Measuring Model Certainty

The pursuit of building powerful and reliable systems is a long-standing goal in the deep learning (DL) community. While modern DL systems are achieving impressive performance, there is limited work in measuring the *reliability* of a system's decision. Such a measure is increasingly important as vulnerabilities of ML have been demonstrated by recent work on adversarial attacks [Yuan et al., 2017, Akhtar and Mian, 2018, Athalye et al., 2018, Su et al., 2017, Chen et al., 2017b, Cisse et al., 2017, Zhao et al., 2017, Dong et al., 2018, Szegedy et al., 2013, Goodfellow et al., 2015, Kurakin et al., 2016, Papernot et al., 2016a, Carlini and Wagner, 2017, Moosavi-Dezfooli et al., 2016, 2017, Rozsa et al., 2016, Liu et al., 2016, Kurakin et al., 2016, Nguyen et al., 2015]. A DL system could also make mistakes on non-adversarial data, which may due to patterns of input data that occur during the testing time were seldomly seen in the training process. As DL becomes widely applied to many real-life applications, particularly those where DL decisions impact humans on mission critical systems, it is important for system users to know when DL decisions should or should not be trusted.

We propose an *intrinsic model certainty* score, which measures the degree to which a decision made by a DL system should be relied upon or is “trustworthy". The certainty score is intrinsic in the sense that it considers the collection of activations of each intermediate layer in the network. The certainty score measures the class similarity on the prediction for a given input against labels of examples in the training set whose layer-wise activation patterns are close to the activations of the input. In contrast to similar measures of model certainty Carrara et al. [2017], our measure intelligently combines scores from multiple layers, automatically infers a threshold to decide if a DL system's decision is trustworthy, and is shown to perform well on *both* adversarial examples and wrongly classified non-adversarial data.

Our model certainty score is calculated as a weighted sum of *purity* scores evaluated at each layer of a DL system. A purity score is similar to a k nearest neighbor (k-nn) score designed by Carrara et al. [2017]. Formally, let F_l be a set of activations at the l^{th} layer of the network, and $D^{\text{train}} = \{(x_i^{\text{train}}, c_i^{\text{train}})\}$ be the training data set where x_i^{train} is an input data, and c_i^{train} is a label. Given query data x^{query} with predicted class c^{query}, the purity of $(x^{\text{query}}, c^{\text{query}})$ at layer l is a function of the k nearest neighbors in the set of all activations at layer l over the training data. It is given as:

$$P_l(x^{\text{query}}, c^{\text{query}}) = \frac{\sum_{i \in \mathrm{K}(x^{\text{query}}, l)} w_i^{\text{train}} \mathbf{1}(c^{\text{query}} = c_i^{\text{train}})}{\sum_{i \in \mathrm{K}(x^{\text{query}}, l)} w_i^{\text{train}}} \tag{6.1}$$

where $\mathrm{K}(x^{\text{query}}, l)$ is the set of training data whose layer l activations lie in the k nearest neighbors of the layer l activation of x^{query} measured by Euclidean distance, w_i^{train} is the weight assigned to each k nearest neighbor, and $\mathbf{1}(c^{\text{query}} = c_i^{\text{train}})$ is the indicator function. $P_l(x^{\text{query}}, c^{\text{query}})$ thus measures the difference between the predicted class c^{query} with its nearest neighbors' labels $\{c_i^{\text{train}}\}$. Higher $P_l(x^{\text{query}}, c^{\text{query}})$ implies that that the activation of x^{query} is consistent with activations from the training data whose label is c^{query}.

The purity score for each layer is combined to produce a certainty score $C(x^{\text{query}}, c^{\text{query}})$. A straightforward combination, which is surprisingly effective, is to take a weighted sum of the layer-wise purity scores:

$$C(x^{\text{query}}, c^{\text{query}}) = \sum_{i=0}^{l} W_i P_i(x^{\text{query}}, c^{\text{query}})$$

where W_i is the fraction of k nearest neighbors in the training data activations of layer l for x^{query} whose class label is c^{query}. The intuition behind this strategy is, if the activations of x^{query} are close to many other activations from the training data of the same class, we should find the activations of x^{query} at layer l to increase the certainty we have on correctly generating the output for the given input.

We can compare $C(x^{\text{query}}, c^{\text{query}})$ against distribution of certainty scores over a reference dataset to determine if its value suggests that the model prediction should be trusted or not based on a cut off threshold $thresh$. The problem is thus turned into a binary classification problem, and the class label is 1 meaning the model prediction is "*untrustworthy*" if $C(x^{\text{query}}, c^{\text{query}}) < thresh$ and "*trustworthy*" otherwise. Our experiments below show, intuitively, that the trustworthiness of a model's decision is quite sensitive to this hyper parameter $thresh$. However, by observing the certainty score distributions for all wrongly predicted training data $D^{\text{train}}_{\text{wrong}}$ and sampled correctly predicted training data $D^{\text{train}}_{\text{correct}}$, the certainty score corresponding to the intersection of this two distributions is shown to be a high quality threshold.

6.2 Experiments and Result Analysis

We carry out experiments on **MNIST** LeCun [1998], **Fashion-MNIST** Xiao et al. [2017], and **Tiny-ImageNet** Hendrycks and Gimpel [2016] using three different neural network models: a *shallow artificial neural network* Yegnanarayana [2009], a *self defined convolutional neural network* Krizhevsky et al. [2012], and a revised *ResNet-50* He et al. [2016] respectively. Each training data set is randomly divided into an 80/20 split for training and validation data. Test data[1] for the models are taken from the published set of test examples for each dataset. All models are trained under a cross-entropy loss function using the RMSprop Tieleman and Hinton [2012] optimizer. Adaptive training learning rates are used, starting at 0.001.

A certainty score is calculated as a weighted sum over purity scores across three intermediate layers. The dimensionality of activations of an intermediate layer is reduced by half if the dimension exceeds 300 in order to generate purity scores efficiently. We consider the $k = 30$ nearest neighbors of the given example to derive purity scores. Besides evaluating certainty over the testing data, we also generated adversarial examples of correctly predicted testing data using the **FGSM** Goodfellow et al. [2015] and **Carlini & Wagner** Carlini and Wagner [2017] adversarial attack methods. We describe specific experimental settings for each dataset below:

MNIST. MNIST is an image digit dataset containing 10 class labels. It contains 60,000 training examples and 10,000 testing examples. The shallow artificial neural network ap-

[1]validation data is used for testing if the test labels are not publicly available

plied to this dataset contains three Fully Connected (FC) ReLU layers with (input, output) dimensions of (784, 300), (300, 100), and (100, 10) respectively. After training for 40 epochs, it achieves an accuracy of 97.28% on testing examples. The activations of three FC layers are used for calculating three purity scores respectively.

Fashion-MNIST. The dimension and size of Fashion-MNIST are identical to MNIST, except the images are grey-scale clothing pictures instead of digits. The convolutional neural network applied to this dataset has two convolutional layers with (input_channels, output_channels, kernel_size, stride) of (1, 16, 5, 1), and (16, 32, 5, 1) respectively, with ReLU activations and a dropout rate of 0.2. A single FC layer is appended to the two convolutional layers prior to softmax output. After training for 8 epochs, it achieves an accuracy of 88.37% on testing examples. The activations of two convolutional layers and one FC layer are used for calculating three purity scores respectively.

Tiny-ImageNet. Tiny-ImageNet is akin to ImageNet Deng et al. [2009] but with lower resolution images. It contains 200 classes with nearly 500 training examples and 50 testing examples per class. We use a ResNet-50 pretrained on full ImageNet and add three FC layers of dimension (2048, 1024), (1024, 512), and (512, 200) and finetune it for 102 epochs. It achieves an accuracy of 60.33% on testing examples. The activations of three FC layers are used for calculating three purity scores respectively.

We first evaluate the extent to which the certainty score differentiates different types of data samples. We show, in Figure 6.1, 6.2, and 6.3 the empirical density of certainty scores for training, testing, FGSM adversarial, and C&W adversarial examples that are predicted correctly and wrongly along with a KDE estimate. On MNIST, we find a a vast majority of correct train and test examples to have certainty at or near 1.0, whereas all incorrect examples (and even correctly classified adversarial examples) have a near uniform

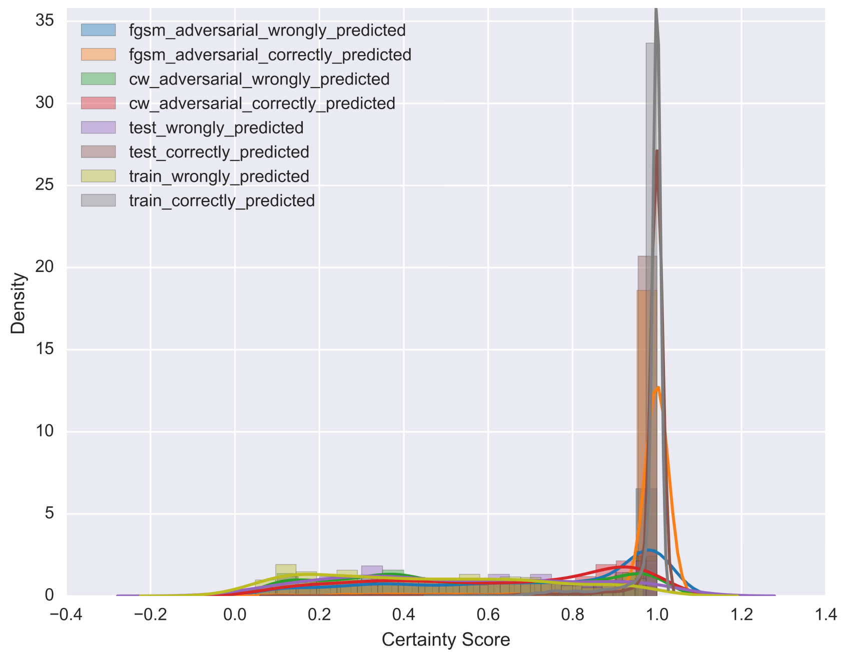


Figure 6.1: Certainty score distribution for MNIST dataset

distribution of scores. It is worth noting that wrongly predicted FGSM adversaries (e.g. successful attacks on the network) have a small mode near 1 as well. This pattern does not hold for certainty densities along the Fashion-MNIST distributions (Figure 6.2), however: correct train and test examples remain near 1, while all adversarial examples and incorrectly classified data have different density shapes. While the distribution of certainty scores for correct test and training examples have higher variance on the CNN for Tiny-ImageNet (Figure 6.3), it is encouraging that the certainty score for wrongly predicted adversarial examples of both times are concentrated between 0 and 0.2. As mentioned in section 6.1, a model certainty threshold can be inferred by the certainty score distribution over training examples. Shown in Figure 6.4, 6.5, and 6.6, we assign $thresh$ to be the certainty score at which the KDE of the two densities intersect.

We evaluate the thresholds using all data in the testing set plus adversarial examples

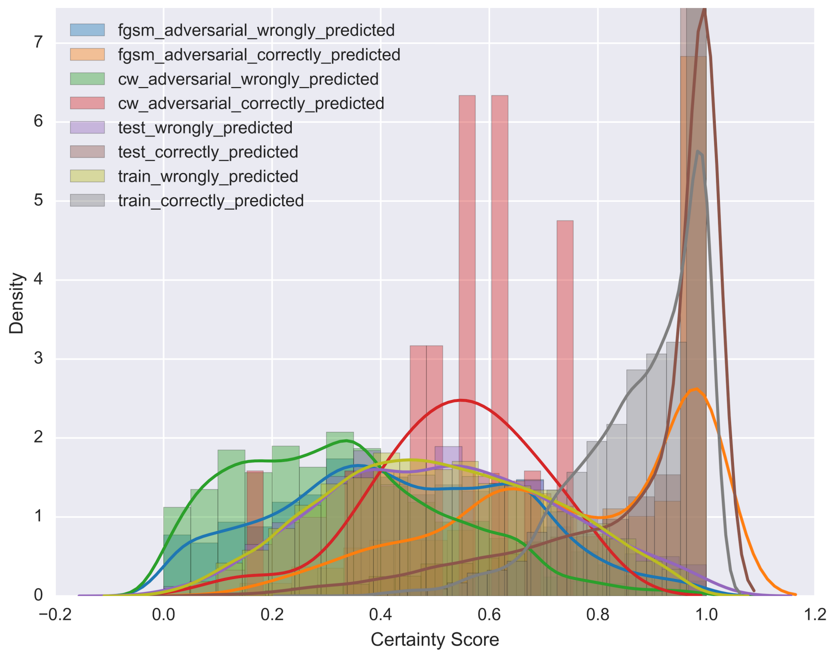


Figure 6.2: Certainty score distribution for Fashion-MNIST dataset

generated from each of the correctly classified test data using both FGSM and C&W methods. We consider the thresholds to be the decision boundary of a binary classifier: we predict positively if the certainty score of an example is below $thresh$ (hence a positive prediction means we consider the output of the network to be “untrustworthy") and predict negatively if the certainty score is above. An untrustworthy decision is correct if the data is wrongly classified by the original neural network model, and a trustworthy decision is correct if the data is correctly classified.

Figure 6.7, 6.8, and 6.9 shows the F_1 score for the trustworthiness evaluation for varying threshold values and when defining the certainty score using individual layer purity scores and when taking the weighted sum over all layer purity scores $\{P_l(x^{\text{query}}, c^{\text{query}})\}$. For the crossover $thresh$ points identified above, we find encouraging F_1 scores nearing 0.8 for MNIST, 0.9 for Fashion-MNIST, and 0.83 on Tiny-ImageNet. It is interesting to

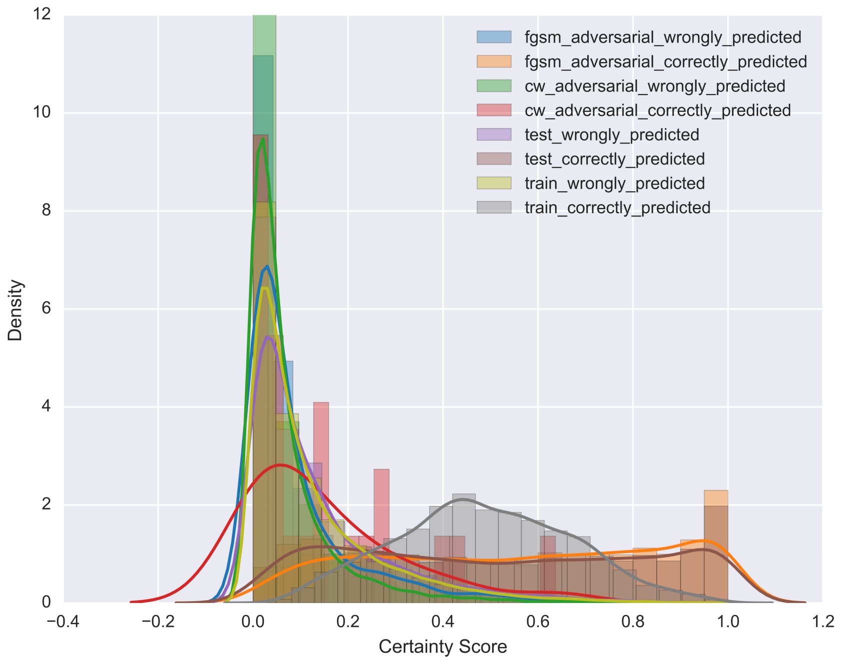


Figure 6.3: Certainty score distribution for Tiny-ImageNet dataset

note that our approach thus finds it harder to assess its trustworthiness of a decision on MNIST, the simplest of the three datasets. This may be because of the manifold of MNIST is (relatively) simplistic; hence there are a more limited number of ways that activation patterns between examples can vary. We further note that for both Fashion-MNIST and Tiny-Imagenet, the purity of activations at early layers are nearly as good as using the certainty score. This suggests that disagreement in the k nearest neighbors of activations early in the network are more predictive of whether an input will ultimately be classified correctly or wrongly.

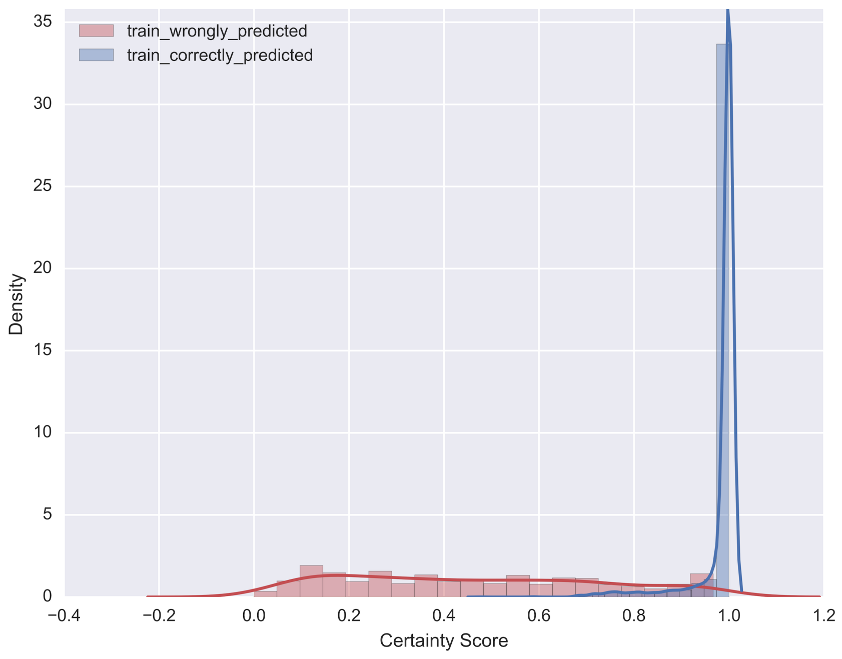


Figure 6.4: Certainty score distribution for MNIST dataset, training split

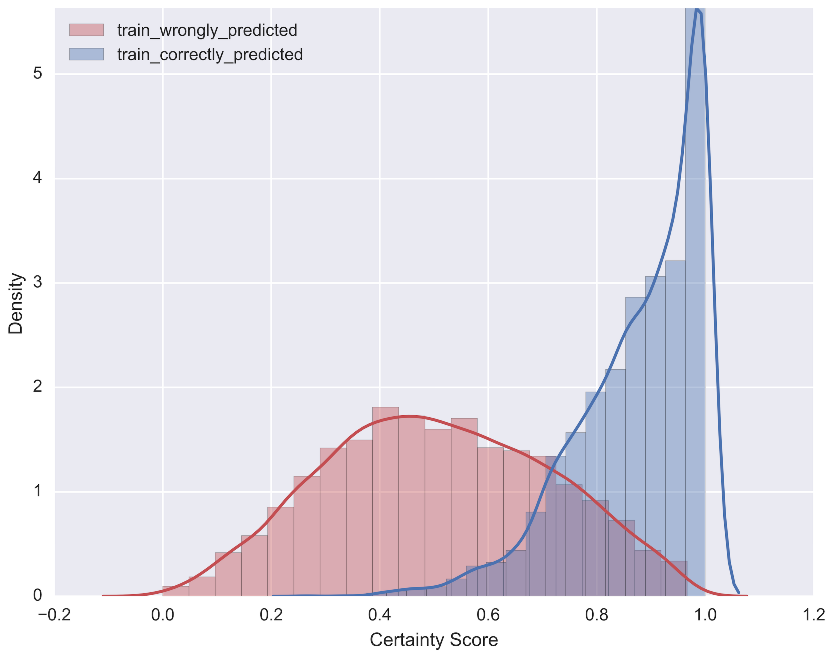


Figure 6.5: Certainty score distribution for Fashion-MNIST dataset, training split

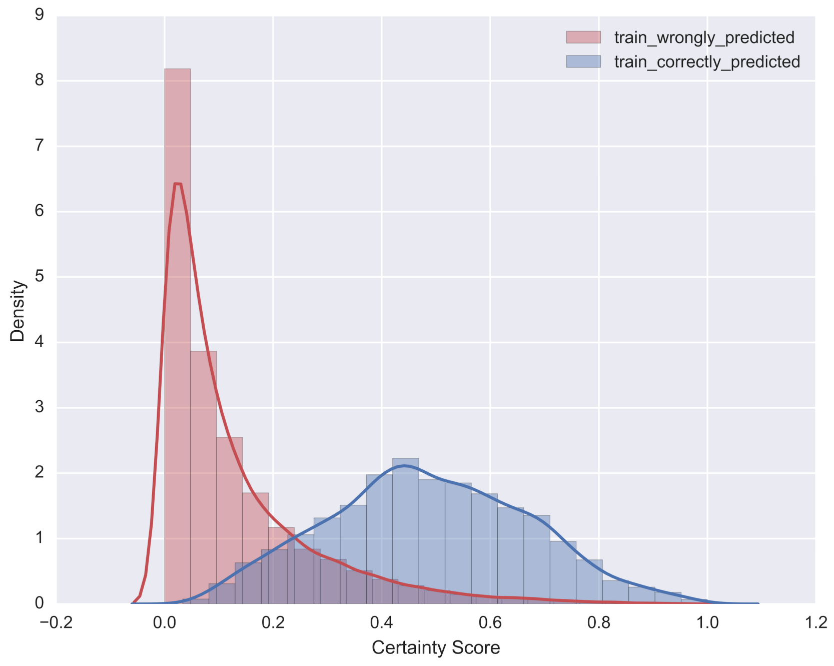


Figure 6.6: Certainty score distribution for Tiny-ImageNet dataset, training split

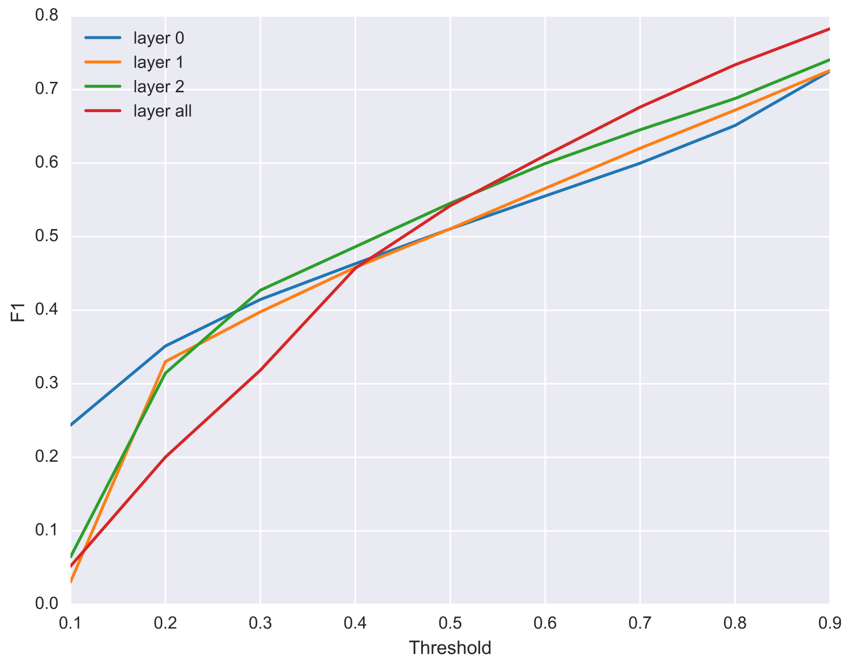


Figure 6.7: F1 score for MNIST dataset

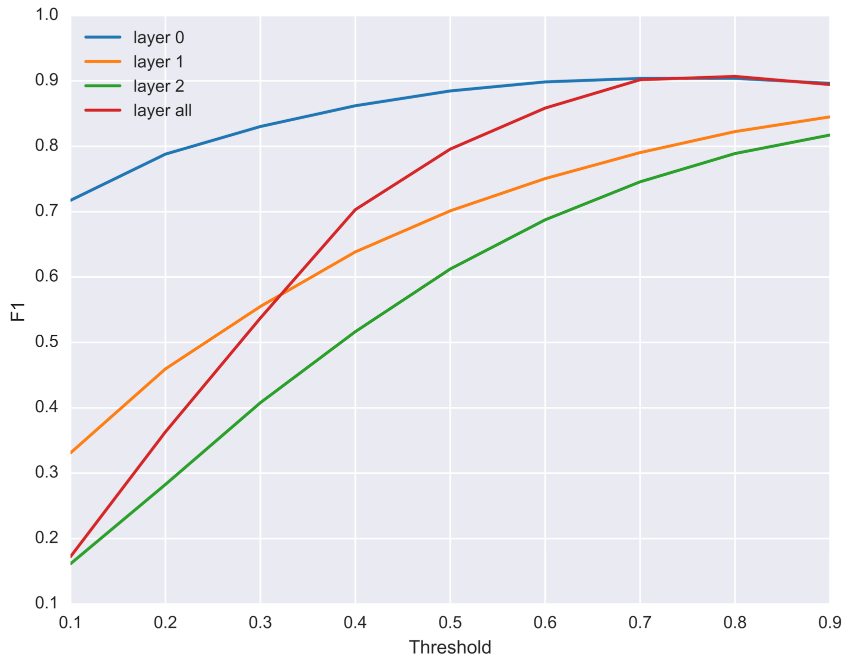


Figure 6.8: F1 score for Fashion-MNIST dataset

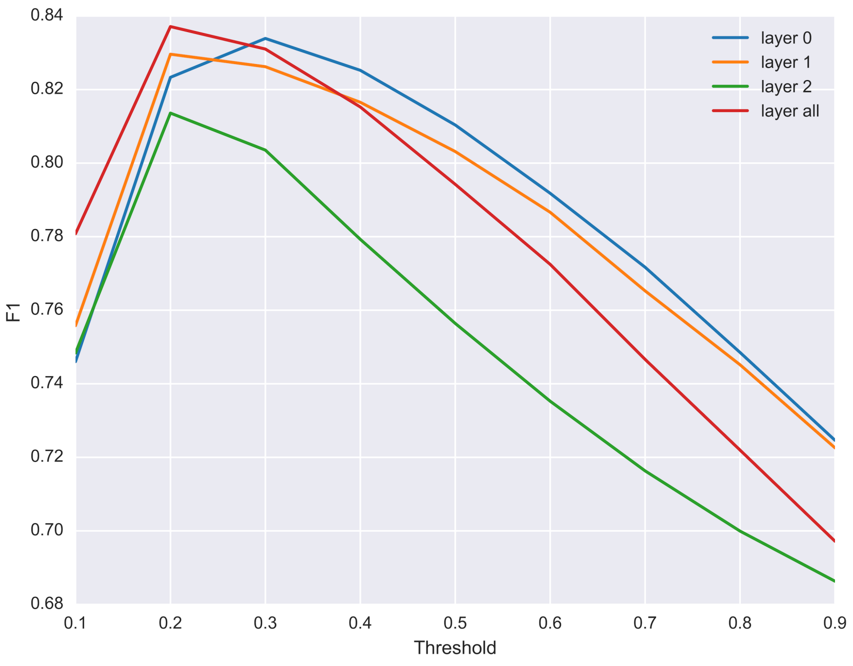


Figure 6.9: F1 score for Tiny-ImageNet dataset

Integrated Solution for Interpretable And Reliable DNNs

In this chapter, we explore a potential integrated solution for interpretable and reliable deep neural networks. Experiments and analyses are conducted on visual entailment, which is a novel visual inference task we propose. Prior to introducing the integrated solution, the demand for such solutions for interpretable and reliable DNNs is first discussed in Section 7.1. Section 7.2 describes how the proposed techniques (Chapter 4, 5, 6) holistically realize a contribution to interpretable and reliable deep neural networks for visual entailment. We now elaborate on each section below in detail.

7.1 The Demand for Integrated Interpretable And Reliable Solution

To the best of our knowledge, existing methods on interpretable and reliable DNNs are mainly singular works that promote explainability and trustworthiness in deep learning out-of-context from other modes of explanation, and their uses in practice. This has yielded a body of techniques for deep learning that may be contradictory in intent, incompatible with each other, and with limited practical utility. We consider a practical interpretable and reliable deep learning demands a collection of integrated techniques that include the following aspects:

1. The capability to represent network mechanisms at a technical level to support model auditing and debugging. An example of such interpretation could be capturing the interactions between input features across different modals. This aspect provides model details in depth for the expert who can audit model quality control, the model's perceived biases, opportunities to improve the technical operation of the model, and to efficiently "debug" a poor performing model.

2. An interpretation of an output that is clear and layman-friendly to the users. An ideal interpretation needs to be grounded in human understandable semantics associated with the domain of the data. This aspect assures the general acceptance of explanations to generic model users in society, since there is no requirement for user expertise.

3. The endowment of a measure of confidence, self-assurance, or trust with model decisions. As undesired model decisions may cause a potential catastrophe, trust is an important criterion to measure to properly accept or reject a model decision. This aspect secures the model reliability by providing a confidence score along with model decisions.

Each of these three aspects is indispensable towards building a transparent and reliable DNN system for myriad real life applications, and in particular for visual intelligence. Society's awareness of the acceleration of computer vision technologies for surveillance, autonomous self driving, robotics, and medical image analysis puts downward pressure on researchers to create solutions that are not only performant but also explainable to users and auditors. Face recognition systems should be audited that they are not biased against a particular population, for example, and vision algorithms for self driving cars need to be auditable in the event of an accident or other situation involving liability.

7.2 An Integrated Solution for Visual Entailment

Our categorized literature review in Part I indicates the lack of systematic interpretable and reliable DNN solutions. In this dissertation, we make a novel contribution by presenting a set of solutions that, together, establish an interpretable and reliable deep learning system satisfying the aspects mentioned in Section 7.1. We demonstrate the utility of the system for *visual entailment*, which is a novel visual inference task we propose [Xie et al., 2019], that offers technically useful and layman-friendly explanations, alongside a discussion of potentially leveraging the proposed certainty measurement (Chapter 6) to quantify the level of trust a user could apply to model decisions[1].

Our integrated interpretable and reliable DNN solution for visual entailment task include the following components,

- **Technician-Oriented Explanation**. We develop a deep learning model that solves a new, challenging formulation of the visual entailment problem (Chapter 5). The key architecture ingredient enabling explanations are attention mechanisms, which highlight the interaction between features from different modals (vision and text), and could be further useful for a technician on model debugging.

- **Layman-Friendly Explanation**. We develop a technique to refine the layman-friendly explanations offered by the attention mechanisms with recognition scores revised from Chapter 4.

- **Model Certainty Measurement**. We discuss a potential usage of a novel measurement of the trustworthiness on model decisions (introduced in Chapter 6), providing human users an indicator to properly accept or reject a model decision for the visual entailment problem based on quantitative measures.

[1]Due to the lack of access to our original trained model for visual entailment task (Chapter 5), the experiments and analyses in this section are conducted on a newly trained EVE-Image model based on the original code. The new EVE-Image model has similar performance as the one reported in Chapter 5, and the overall accuracy for test split is 71.24%, the test accuracy for contradiction, neutral, and entailment are 71.55%, 67.44%, and 74.72% respectively.

7.2.1 Technician-Oriented Explanation for Visual Entailment

As presented in Section 5.5, the text-image attention visualization shows based on the input text, which areas in the input image are highly involved in generating the given model decision. Such attention visualization offers a "probe" for the model technicians on how the hidden states of text and image features interact to generate a model output. Different visualized text-image attention patterns can be leveraged by the technicians for multi-modal feature interaction studies, model error analysis, and potentially for model improvement.

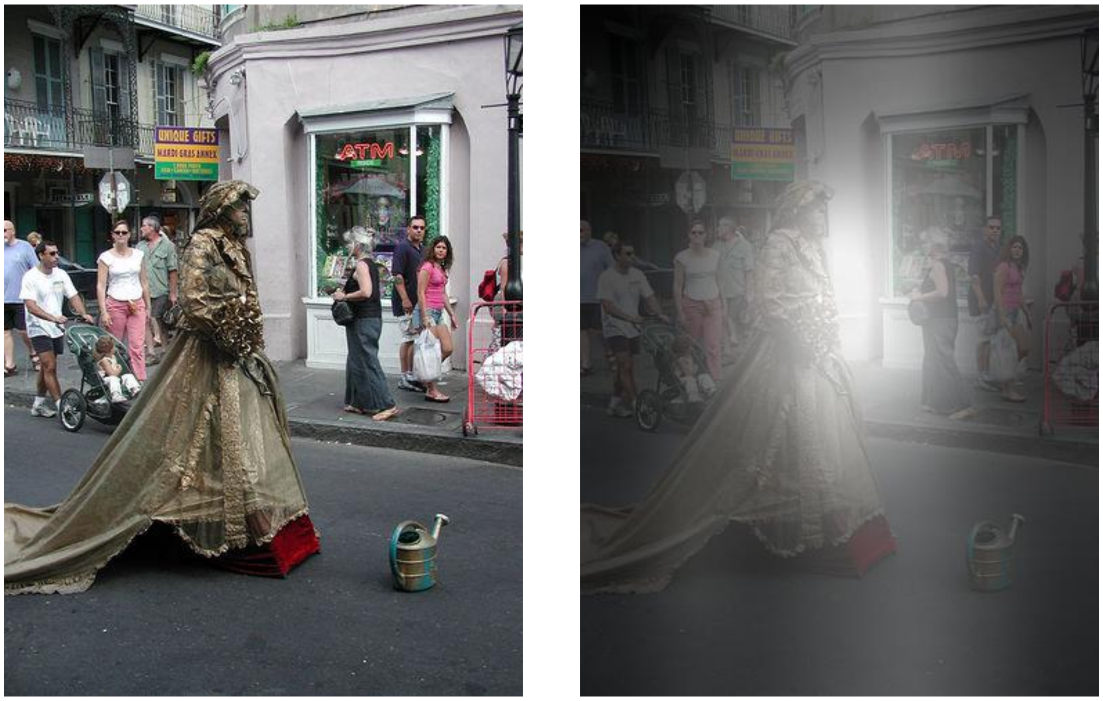


Figure 7.1: Attention visualization for multi-modal feature interaction studies.

Figure 7.1 is the attention visualization based on given hypothesis *"Someone wearing a costume is standing on the street."*, which is correctly predicted as *"entailment"* by the model. Based on the illustrated attention, technicians can observe the model is able to properly focus on the person and the person's surroundings in the image with respect to the text information for the model prediction.

Figure 7.2 is the attention visualization for hypothesis *"A man is reading newspaper*

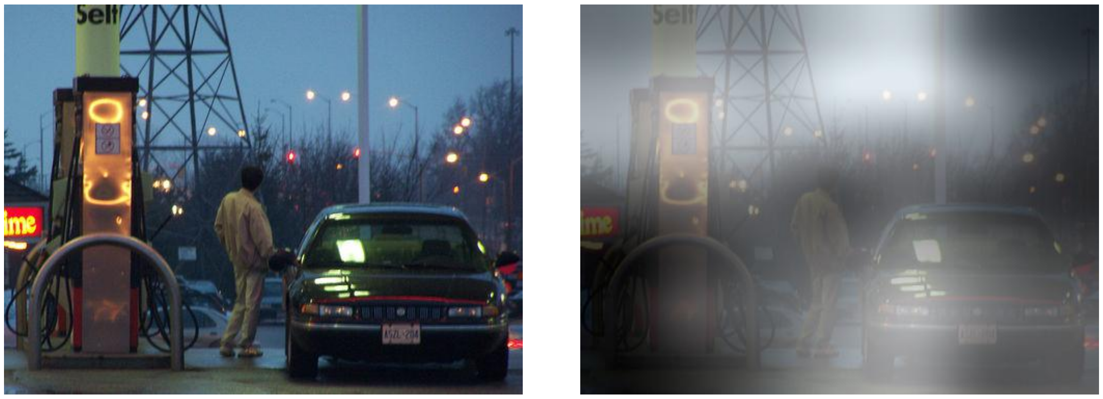


Figure 7.2: Attention visualization for model error analysis.

and drinking soda.”, which is labeled *“contradiction”* but predicted as *“neutral”* by the model. The prediction error can be tracked by technicians as the model’s concentration is wandering in the image, and mostly on irrelevant image areas.

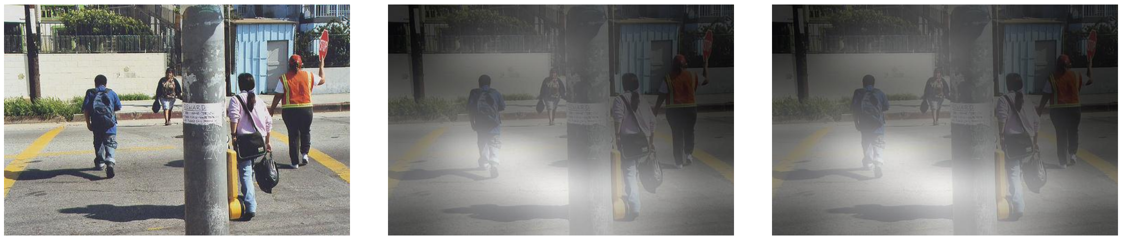


Figure 7.3: Attention visualization for model improvement.

For potential model improvement, technicians can analyze error cases (as illustrated in Figure 7.2) for reasons that cause incorrect predictions, and revise the model accordingly. Besides, model improvement may also be derived from correctly predicted cases. Figure 7.3 shows the attention visualization for two distinct hypotheses. The middle sub-figure is for hypothesis *“One person is sitting on the bench.”* and is correctly predicted as *“contradiction”* by the model. The right sub-figure is for hypothesis *“People are crossing the road.”* and is correctly predicted as *“entailment”*. Although the two cases are correctly predicted, the two attention visualizations are almost identical, implying a more effective way of multi-modal feature interaction is needed for model performance improvement.

7.2.2 Layman-Friendly Explanation for Visual Entailment

Based on the generated attention visualizations, technicians are looking for insights on model debugging and potential improvement directions, while non-technical users, on the other hand, are seeking explanations presented in an easy-understanding way. In order to provide layman-friendly explanations for visual entailment task, here we extend the concept recognition scoring technique (introduced in Chapter 4) to attention mechanisms. The revised scoring technique is conducted on the pre-generated attention visualization, identifying how well each semantic concept (which is also referred to as *entities* below) is recognized by the model, providing non-technical users a straightforward way to comprehend model behaviors. We now elaborate on how we extend the scoring technique to attention visualizations, followed by some generated layman-friendly explanations below.

Applying Recognition Scoring to Attention Visualization

In order to apply the concept recognition scoring technique (Chapter 4) to the attention visualizations, we leverage Flickr30k Entities [Plummer et al., 2015], which is a dataset containing image annotations for all entities presented in given Flickr30k [Young et al., 2014] image captions. For each entity, the corresponding image area is annotated using a bounding box, as illustrated in Figure 7.4. The bounding box annotation in the image can be used as the ground truth area for a given entity towards calculating the concept recognition score.

In contrast to Chapter 4, where multiple model visualizations are generated with respect to a model prediction, here for each model prediction there is only one attention visualization correspondingly. Thus greedy selection (Algorithm 1) is omitted, and only the recognition score is leveraged. Based on Chapter 4, the recognition score is defined as a Jaccard like similarity measure,

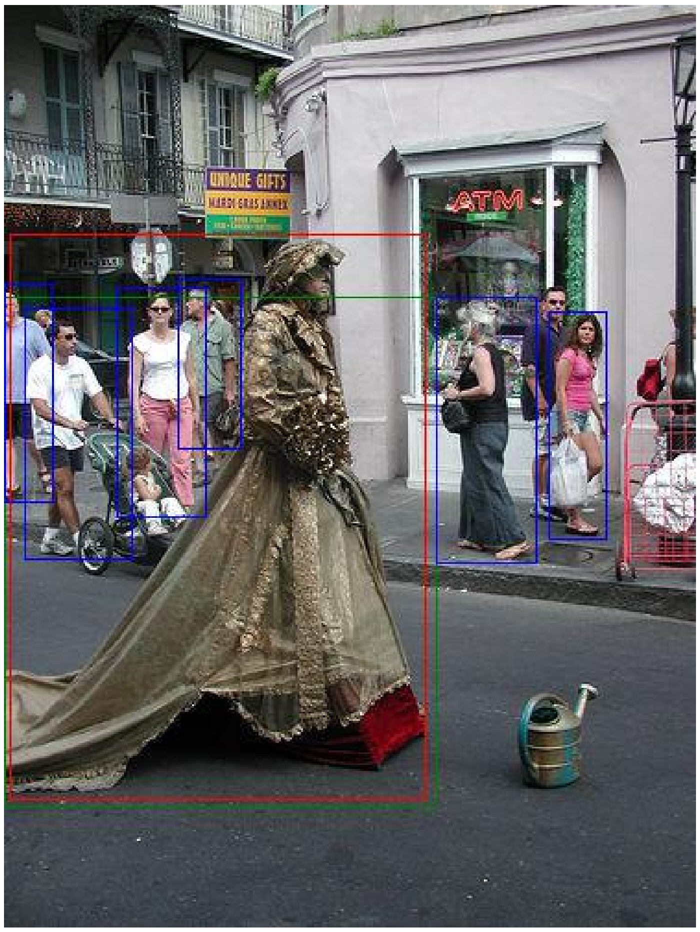


Figure 7.4: An image annotation example from Flickr30k Entities dataset

$$S_e(A, \xi) = \frac{|M_e(\xi) \cap A|}{|M_e(\xi) \cup A|}$$

where A is the text-image attention visualization in our case, ξ is the given input image, e is an entity or concept, $M_e(\xi)$ is a binary segmentation mask (can be generated based on the entity bounding box) corresponding to the entity e. As illustrated in Figure 7.5, the "people" entity class is annotated using a bounding box in the given image, and the recognition score of entity "people" can be calculated based on how well the highlighted attention area matches with the given bounding box.

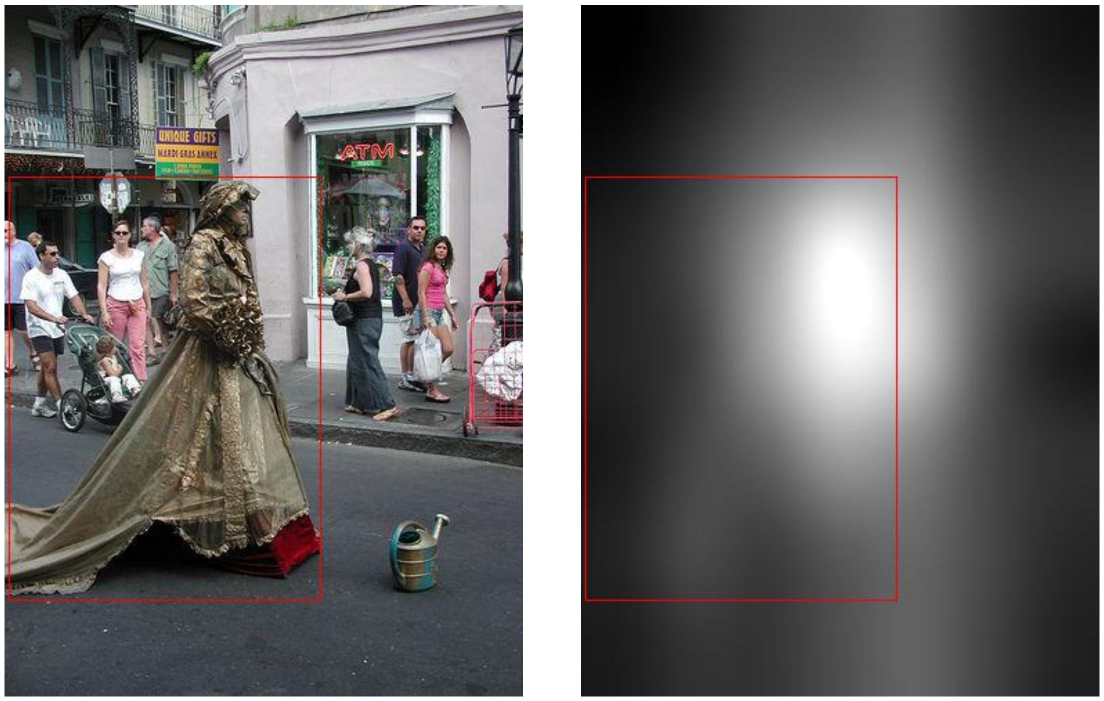


Figure 7.5: Applying recognition scoring to attention visualization.

Examples of Layman-Friendly Explanations

Figure 7.6, 7.7, and 7.8 illustrate some examples of the calculated entity recognition score. As shown in Figure 7.6, the recognition score for "people" is 0.37, indicating to non-technical users that the "people" area is relatively well-focused by the model. While in Figure 7.7, and 7.8, the recognition score is relatively low, implying the model is not mainly focusing on the given entity ("vehicles" and "instruments" respectively). The visualized attention with entity bounding box annotation also provides non-technical users a straightforward way to understand where entities locate and whether or not the model attends to such entities.

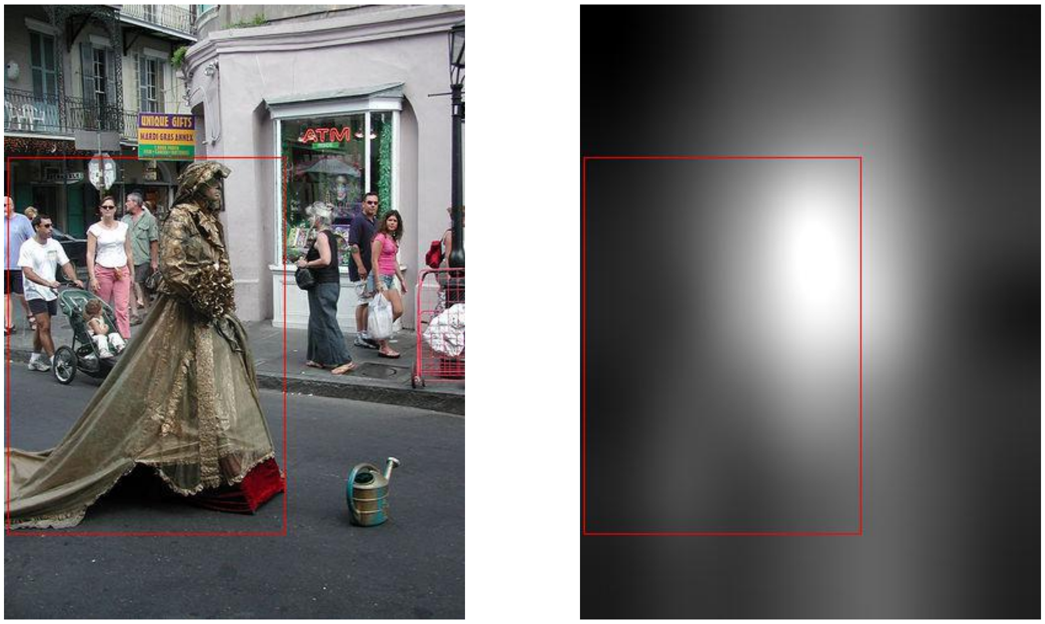


Figure 7.6: Recognition score for entity “people”, $S = 0.37$.

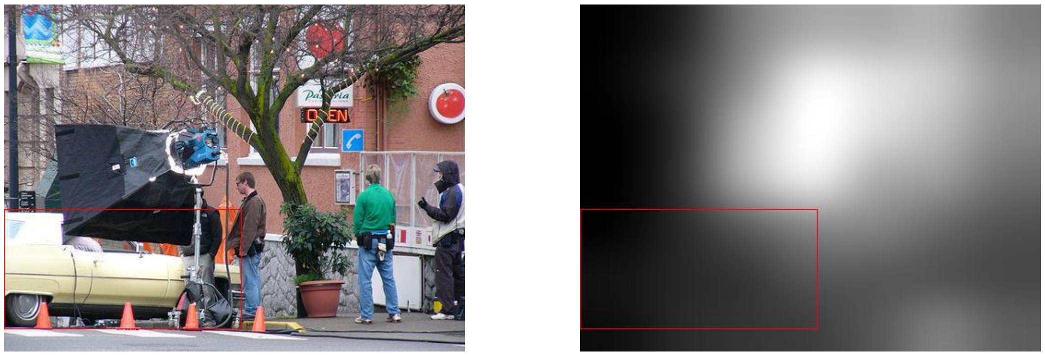


Figure 7.7: Recognition score for entity “vehicles”, $S = 0.18$.

7.2.3 Model Certainty Exploration for Visual Entailment

The intrinsic measurement for DNN decisions introduced in Chapter 6 offers users a potential solution towards building a robust and reliable deep neural network. As model robustness and reliability is not the main focus of this dissertation study, here we only discuss the feasibility of leveraging the introduced certainty measurement for the visual entailment task[2].

Based on Chapter 6, the model certainty score for visual entailment task can be calcu-

[2]For technical details for the introduced model certainty measurement, please refer to Chapter 6.

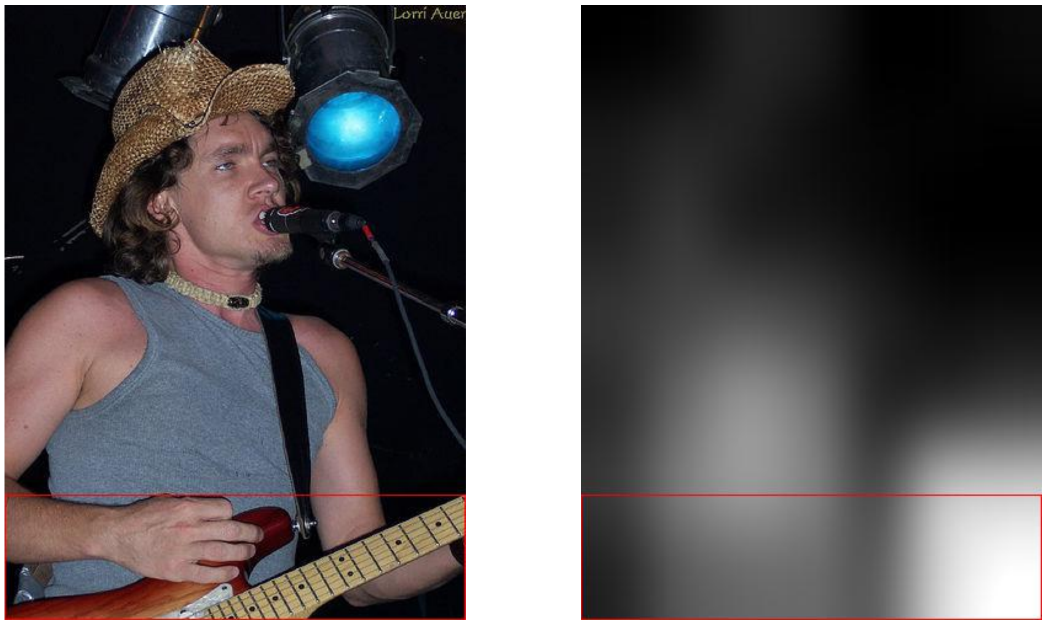


Figure 7.8: Recognition score for entity “instruments”, $S = 0.21$.

lated in the following steps,

1. **Activation Extraction**. This is a prerequisite step for model certainty generation, which is to extract and store the hidden states of the well-trained visual entailment model for all or sampled training examples from selected layers.

2. **KNN Purity and Certainty Calculation**. For a query example from visual entailment dataset, conduct K nearest neighbor algorithm in a hidden activation space spanned by all/sampled visual entailment training examples. The **purity** score measures the portion of training neighbors that has the same label as the predicted query class. The weighted sum of purity score across multiple layers is the **certainty** score for the query example.

3. **Decision Judgement**. Based on the calculated certainty score for the query example, the users decide whether or not to trust this prediction using a pre-defined certainty threshold.

Towards making our visual entailment model more robust, patterns from generated adversarial examples may also be included for the selection of a proper threshold.

Future Work And Conclusion

8.1 Future Work

In this section, we discuss potential future studies based on the techniques proposed in this dissertation. The future work is categorized into the following two aspects,

- **Methodological Improvements**: This introduces ways to *improve* of our proposed techniques on interpretable and reliable DNNs.
- **New Area Applications**: This discusses how we may *apply* our proposed techniques to other application areas and research topics.

We now elaborate in detail below.

8.1.1 Methodological Improvements

Here we discuss some future directions for our proposed techniques for potential improvement, towards better solving the problem of interpretable and reliable deep neural networks.

Relating Input Concepts to CNN Decisions (Chapter 4) For this work, the effects of "unique", "misleading", and "sparse" concepts can be explored in more detail in future studies. In particular, common misclassifications for a given scene class remain to be investigated and model decision explanations can be concluded in depth using the recognized concepts that are (not) common between those misclassified examples. The effect of "sparse" concepts on CNN classification is also left to be explored, and a possible method for this direction is to occlude the "sparse" concept areas in an image and check its influence on the CNN's decision. It is also worth exploring the mechanics of how concept recognitions impact downstream network activations leading to a decision and to devise a measure of the importance of concept recognition for CNN decision making.

Achieving Intrinsic DNN Explanations via Attention Mechanism (Chapter 5) In this work we propose a novel visual reasoning task, *Visual Entailment*, and design an intrinsic explainable model, *EVE*, to solve it. The intrinsic explanation is achieved via the visualization of text-image attention. Based on the error analysis for some wrongly predicted examples in Chapter 7, the attention module can be further explored in order to align features from text and image branches more efficiently. Specifically, the text-image attention in EVE model is conducted upon the feature representation for the whole text sentence, instead of the features for every single word. Thus the features for some critical words may get lost in the whole sentence representation, resulting in inaccurate attention on image areas. A recent follow-up work [Chen et al., 2019b] addresses this issue by proposing a new model, UNiversal Image-TExt Representation (UNITER), which leverages a quasi-BERT [Devlin et al., 2019] model structure that is built upon Transformers [Vaswani et al., 2017] to extract representations for each text input, image input and also capture the fine-grained interaction between text and image. The text-image attention in UNITER offers attention visualizations for the related image areas with respect to each *single* word. UNITER achieves about 79% accuracy for both validation and testing splits of SNLI-VE dataset, exceeding the EVE performance (about 71% accuracy) by a large margin. However, error analysis of UNITER for SNLI-VE dataset is not provided, and based on 79% we believe there still exists a large room for performance improvement. For the next step, we look forward to more follow-up research on SNLI-VE towards a better performed and more transparent solution.

An Intrinsic Certainty Measure for Deep Neural Network Decisions (Chapter 6) In this work, we introduce a quantitative intrinsic notion of the certainty for DNN decisions. The certainty measurement is built upon the internal activation patterns in the intermediate layer of the DNN, and is calculated based on the Euclidean distance between the activation of a query example and the activation of each training example. For the next step, since the intermediate activation space is of high dimension, the certainty measurement can be potentially improved by leveraging more advanced distance metrics. Besides, the manifold of complex activation space is another interesting direction for further exploration.

8.1.2 New Area Applications

Here we discuss potential applications of our proposed techniques on some real life cases and other research topics.

Real Life Applications Model interpretability and reliability are important for decision critical scenarios, where improper model decisions may cause severe consequences. Our proposed techniques can be applied to such real life scenarios via constructing a more desirable DNN system. Here we elaborate on one example in medical application where our proposed techniques can be jointly leveraged.

Imagine a scenario where a doctor uses a DNN system to make diagnostic decisions. The diagnosis is based on *hybrid* data from both image, such as CT scan or ultrasound image, and text, such as anamnesis, symptom description, etc. The decentness of the model decision is crucial, as it may affect the next medical treatment, and even critical succeed decisions. In this case, we list three possible questions raised by the doctor and how our techniques can be leveraged to potentially tackle them below, i) *How the model decision is addressed?* This question can potentially be solved by revising the proposed method in Chapter 4 to relate model decisions with human-understandable medical evidence in the given input data. ii) *How data from different resources (text and image) interact with* *each other* during the model decision making process, towards generating the final model prediction? The text-image attention module from Chapter 5 can be used to capture such interaction patterns. iii) As a diagnostic decision has critical potential influences, thus should not be released unless verified. Given the generated model decision, *how would a model user know if the decision is reliable or not*, what post-hoc procedures can be taken for trustworthiness verification? The certainty measurement designed in Chapter 6 can be applied to generate a certainty score, indicating the reliability of a model decision.

Other Research Topics Our proposed techniques can be leveraged to other existing research topics, enabling their methods with interpretability and reliability characteristics. Recently, the combination between model explainability and **Reinforcement Learning** (RL) has been actively studied [Puiutta and Veith, 2020, Madumal, 2020, Madumal et al., 2019, Sequeira et al., 2019, Cruz et al., 2019, Shi et al., 2020, Cruz et al., 2020, Akrour et al., 2020]. According to a recent survey paper on explainable reinforcement learning [Puiutta and Veith, 2020], most current methods provide explanations for RL systems in a post-hoc fashion, that is to "mimic" or "simplify" a well-trained RL system for explainability. The intrinsic design for RL explanations, which is to generate explanations inherently from the trained RL system remains to be explored. While our intrinsic technique proposed in Chapter 5 can be potentially leveraged to address this gap, by adding attention component to the RL system and visualize how intermediate features interact with each other. Another finding from the survey is most existing explanation solutions for RL are targeting expert users instead of normal users who are not equipped with machine learning knowledge. While our proposed technique introduced in Chapter 4 can potentially be adapted for RL systems to solve this issue by, first extracting human-understandable semantic concepts from the input or intermediate representations of the RL system, and then relate those concepts to RL decision. By providing layman-friendly explanations, the confidence of normal users towards leveraging RL systems to solve their real life problems would potentially increase.

Our proposed techniques can also be leveraged to some closely related topics for their field promotions. One topic that is closely related to interpretable and reliable DNNs is **Model Debugging**, as discussed in Section 3.2. According to our systematic literature review, there exists a limited amount of work on solving the model debugging problem for deep neural networks. However, we consider model debugging is an indispensable component for future DNN systems, without which the improvement of model performance would be limited, thus hinders the prosperity of such DNN systems applications. Our proposed work can be leveraged for model debugging. Specifically, for Chapter 4, the relationships between recognized concepts, which could be "misleading", "unique", or "sparse" concept, and DNN decision can be further explored to find strong associations between a group of concepts and a specific type of model decisions. Those concepts can then serve as a signature, and if the DNN model is not able to properly capture such "signature concepts", further training is required to gain more accuracy. In this way, the DNN model can be fine-tuned more efficiently towards achieving better performance. Another closely related topic is **Adversarial Defense**, as discussed in Section 3.3. Our proposed method in Chapter 6 provides a novel direction towards building a robust DNN system. The manifold of the model's intermediate patterns can be further explored, and the certainty measurement can be improved to not only justify normal examples that are wrongly predicted, but also capture adversarial examples since their hidden representation could reveal abnormal patterns.

8.2 Conclusion

Building a real life artificial intelligence system that is equipped with human-competitive performance, robust against abnormal examples, and transparent and reliable for the decisions made is a long term goal existing in the community. In many real life applications, such as face recognition [Masi et al., 2018], voice assistance [Tulshan and Dhage, 2018], machine translation [Yang et al., 2020], etc, the unprecedented representation power of deep neural networks, although opaque in nature, enables the system to achieve excellent performance that is competitive to human behaviors. However, in some cases, for instance the application in driving assistant or disease diagnostics, the consequences of incorrect model decisions may cause potential catastrophes, raising the demand for the transparency and reliability characteristics of models applying to those scenarios.

Towards achieving the transparency and reliability goal, this dissertation explores the topic of interpretability and reliability of deep neural networks, focusing on a specific domain, visual intelligence. Two types of visual intelligence tasks, image classification and visual inference, are explored. Towards interpretable and reliable DNNs for visual intelligence, a categorized literature review is first provided, depicting the field in a systematic and comprehensive way. Following the literature review, two types of methods are proposed to generate explanations for model decisions, i) through hidden state visualization and ii) by leveraging attention to make a DNN intrinsically interpretable. In the former work, a Convolutional Neural Network (CNN) is utilized as an image scene classifier and our proposed explanation method is to explain why the CNN classifier predicts an image as a specific scene class by relating the CNN decisions with input concepts via a visualization technique called deconvolution. In the latter work, our proposed method achieves the goal of interpretability from an intrinsic perspective. An attention-based DNN is designed to solve visual entailment, which is a novel visual inference task we proposed that involves the interaction between computer vision and natural language processing. The attention heatmap is visualized to depict how the input image interacts with the input natural sentence towards generating the model decision. To achieve DNN reliability, a certainty measurement for DNN decisions is designed, based on which the users could decide whether or not to adopt the given DNN decision. At the end of this dissertation, further experiments and analyses are conducted for the visual entailment task to depict how the introduced techniques holistically realize a contribution to interpretable and reliable deep neural networks for visual intelligence.

Our proposed methods can be jointly leveraged to offer technician-oriented explanations which potentially for model debugging on performance improvement, generate layman-friendly explanations for normal users without technical background, and also provide the system with a reliability measurement such that the system users own the freedom to properly decline doublable model decisions. The field of interpretable and reliable deep learning is being vividly explored. For the next step, we look forward to future studies on extending our methods towards building a more powerful and trustworthy integrated DNN system for a wide range of real life applications.

Bibliography

Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Süsstrunk. SLIC superpixels compared to state-of-the-art superpixel methods. *IEEE transactions on pattern analysis and machine intelligence*, 34(11):2274–2282, 2012.

Julius Adebayo and Lalana Kagal. Iterative orthogonal feature projection for diagnosing bias in black-box models. *arXiv preprint arXiv:1611.04967*, 2016.

Alekh Agarwal, Alina Beygelzimer, Miroslav Dudik, John Langford, and Hanna Wallach. A reductions approach to fair classification. In *International Conference on Machine Learning*, pages 60–69, 2018.

Naveed Akhtar and Ajmal Mian. Threat of adversarial attacks on deep learning in computer vision: A survey. *arXiv preprint arXiv:1801.00553*, 2018.

Riad Akrour, Davide Tateo, and Jan Peters. Reinforcement learning from a mixture of interpretable experts. *arXiv preprint arXiv:2006.05911*, 2020.

Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes. *arXiv preprint arXiv:1610.01644*, 2016.

Saleema Amershi, Max Chickering, Steven M Drucker, Bongshin Lee, Patrice Simard, and Jina Suh. Modeltracker: Redesigning performance analysis tools for machine learning.

In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pages 337–346. ACM, 2015.

Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete problems in AI safety. *arXiv preprint arXiv:1606.06565*, 2016.

Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei 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.

Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. VQA: Visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pages 2425–2433, 2015.

Devansh Arpit, Stanisław Jastrzębski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, et al. A closer look at memorization in deep networks. In *International Conference on Machine Learning*, pages 233–242, 2017.

Leila Arras, Franziska Horn, Grégoire Montavon, Klaus-Robert Müller, and Wojciech Samek. Explaining predictions of non-Linear classifiers in NLP. In *Proceedings of the 1st Workshop on Representation Learning for NLP*, pages 1–7, 2016.

Leila Arras, Franziska Horn, Grégoire Montavon, Klaus-Robert Müller, and Wojciech Samek. "What is relevant in a text document?": An interpretable machine learning approach. *PloS one*, 12(8):e0181142, 2017.

Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. *arXiv preprint arXiv:1802.00420*, 2018.

Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, 10(7):e0130140, 2015.

David Baehrens, Timon Schroeter, Stefan Harmeling, Motoaki Kawanabe, Katja Hansen, and Klaus-Robert MÃžller. How to explain individual classification decisions. *Journal of Machine Learning Research*, 11(Jun):1803–1831, 2010.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

Yasaman Bahri, Jonathan Kadmon, Jeffrey Pennington, Sam S Schoenholz, Jascha Sohl-Dickstein, and Surya Ganguli. Statistical mechanics of deep learning. *Annual Review of Condensed Matter Physics*, 2020.

Osbert Bastani, Carolyn Kim, and Hamsa Bastani. Interpreting blackbox models via model extraction. *arXiv preprint arXiv:1705.08504*, 2017.

David Bau, Bolei Zhou, Aditya Khosla, Aude Oliva, and Antonio 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.

Kevin Baum, Maximilian A Köhl, and Eva Schmidt. Two challenges for CI trustworthiness and how to address them. In *Proceedings of the 1st Workshop on Explainable Computational Intelligence (XCI 2017)*, 2017.

Yahav Bechavod and Katrina Ligett. Penalizing unfairness in binary classification. *arXiv preprint arXiv:1707.00044*, 2017.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155, 2003.

Richard Berk, Hoda Heidari, Shahin Jabbari, Matthew Joseph, Michael Kearns, Jamie Morgenstern, Seth Neel, and Aaron Roth. A convex framework for fair regression. *arXiv preprint arXiv:1706.02409*, 2017.

Alex Beutel, Jilin Chen, Zhe Zhao, and Ed H Chi. Data decisions and theoretical implications when adversarially learning fair representations. *arXiv preprint arXiv:1707.00075*, 2017.

Jacob Bien and Robert Tibshirani. Prototype selection for interpretable classification. *The Annals of Applied Statistics*, pages 2403–2424, 2011.

Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. A large annotated corpus for learning natural language inference. *arXiv preprint arXiv:1508.05326*, 2015.

Wieland Brendel, Jonas Rauber, and Matthias Bethge. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. *arXiv preprint arXiv:1712.04248*, 2017.

Toon Calders, Faisal Kamiran, and Mykola Pechenizkiy. Building classifiers with independency constraints. In *2009 IEEE International Conference on Data Mining Workshops*, pages 13–18. IEEE, 2009.

Flavio Calmon, Dennis Wei, Bhanukiran Vinzamuri, Karthikeyan Natesan Ramamurthy, and Kush R Varshney. Optimized pre-processing for discrimination prevention. In *Advances in Neural Information Processing Systems*, pages 3992–4001, 2017.

Oana-Maria Camburu, Tim Rocktäschel, Thomas Lukasiewicz, and Phil Blunsom. e-SNLI:

Natural language inference with natural language explanations. In *Advances in Neural Information Processing Systems*, pages 9560–9572, 2018.

Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 39–57. IEEE, 2017.

Nicholas Carlini, Guy Katz, Clark Barrett, and David L Dill. Ground-truth adversarial examples. 2018.

Gail A. Carpenter and Stephen Grossberg. The art of adaptive pattern recognition by a self-organizing neural network. *Computer*, 21(3):77–88, 1988.

Fabio Carrara, Fabrizio Falchi, Roberto Caldelli, Giuseppe Amato, Roberta Fumarola, and Rudy Becarelli. Detecting adversarial example attacks to deep neural networks. In *Proceedings of the 15th International Workshop on Content-Based Multimedia Indexing*, page 38. ACM, 2017.

Shan Carter, Zan Armstrong, Ludwig Schubert, Ian Johnson, and Chris Olah. Activation atlas. *Distill*, 2019. doi: 10.23915/distill.00015. https://distill.pub/2019/activation-atlas.

Chaofan Chen, Oscar Li, Daniel Tao, Alina Barnett, Cynthia Rudin, and Jonathan K Su. This looks like that: Deep learning for interpretable image recognition. In *Advances in Neural Information Processing Systems*, pages 8928–8939, 2019a.

Kan Chen, Rama Kovvuri, and Ram Nevatia. Query-guided regression network with context policy for phrase grounding. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017a.

Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *arXiv preprint arXiv:1606.00915*, 2016.

Pin-Yu Chen, Huan Zhang, Yash Sharma, Jinfeng Yi, and Cho-Jui Hsieh. Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pages 15–26. ACM, 2017b.

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. Uniter: Learning universal image-text representations. *arXiv preprint arXiv:1909.11740*, 2019b.

Yunjey Choi. Image captioning pytorch implementation. https://github.com/yunjey/pytorch-tutorial/tree/master/tutorials/03-advanced/image_captioning. Accessed: 2018-10-30.

Chung-Hua Chu and Yu-Kai Feng. Study of eye blinking to improve face recognition for screen unlock on mobile devices. *Journal of Electrical Engineering & Technology*, 13 (2):953–960, 2018.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

Moustapha Cisse, Yossi Adi, Natalia Neverova, and Joseph Keshet. Houdini: Fooling deep structured prediction models. *arXiv preprint arXiv:1707.05373*, 2017.

Isidro Cortes Ciriano and Andreas Bender. Reliable prediction errors for deep neural networks using test-time dropout. *Journal of chemical information and modeling*, 2019.

Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, et al. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *arXiv preprint arXiv:1805.10190*, 2018.

Mark Craven and Jude W Shavlik. Extracting tree-structured representations of trained networks. In *Advances in neural information processing systems*, pages 24–30, 1996.

Francisco Cruz, Richard Dazeley, and Peter Vamplew. Memory-based explainable reinforcement learning. In *Australasian Joint Conference on Artificial Intelligence*, pages 66–77. Springer, 2019.

Francisco Cruz, Richard Dazeley, and Peter Vamplew. Explainable robotic systems: Interpreting outcome-focused actions in a reinforcement learning scenario. *arXiv preprint arXiv:2006.13615*, 2020.

Ido Dagan, Oren Glickman, and Bernardo Magnini. The pascal recognising textual entailment challenge. In *Machine learning challenges. evaluating predictive uncertainty, visual object classification, and recognising tectual entailment*, pages 177–190. Springer, 2006.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proc. of IEEE Computer Vision and Pattern Recognition Conference*, pages 248–255. IEEE, 2009.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, 2019.

Phoebe MR DeVries, Fernanda Viégas, Martin Wattenberg, and Brendan J Meade. Deep learning of aftershock patterns following large earthquakes. *Nature*, 560(7720):632, 2018.

Yanzhuo Ding, Yang Liu, Huanbo Luan, and Maosong Sun. Visualizing and understanding neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association* *for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1150–1159, 2017.

Yinpeng Dong, Hang Su, Jun Zhu, and Bo Zhang. Improving interpretability of deep neural networks with semantic information. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4306–4314, 2017.

Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9185–9193, 2018.

Michele Donini, Luca Oneto, Shai Ben-David, John S Shawe-Taylor, and Massimiliano Pontil. Empirical risk minimization under fairness constraints. In *Advances in Neural Information Processing Systems*, pages 2791–2801, 2018.

Filip Karlo Došilović, Mario Brčić, and Nikica Hlupić. Explainable artificial intelligence: A survey. In *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pages 0210–0215. IEEE, 2018.

Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. Fairness through awareness. In *Proceedings of the 3rd innovations in theoretical computer science conference*, pages 214–226. ACM, 2012.

Cynthia Dwork, Nicole Immorlica, Adam Tauman Kalai, and Max Leiserson. Decoupled classifiers for group-fair and efficient machine learning. In *Conference on Fairness, Accountability and Transparency*, pages 119–133, 2018.

Ethan Elenberg, Alexandros G Dimakis, Moran Feldman, and Amin Karbasi. Streaming weak submodularity: Interpreting neural networks on the fly. In *Advances in Neural Information Processing Systems*, pages 4044–4054, 2017.

Dumitru Erhan, Yoshua Bengio, Aaron Courville, and Pascal Vincent. Visualizing higher-layer features of a deep network. *Départment d'Informatique et Recherche Opérationnelle, University of Montreal, QC, Canada, Tech. Rep*, 1341, 2009.

Dumitru Erhan, Aaron Courville, and Yoshua Bengio. Understanding representations learned in deep architectures. *Départment d'Informatique et Recherche Opérationnelle, University of Montreal, QC, Canada, Tech. Rep*, 1355, 2010.

Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song. Robust physical-world attacks on deep learning visual classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1625–1634, 2018.

Michael Feldman, Sorelle A Friedler, John Moeller, Carlos Scheidegger, and Suresh Venkatasubramanian. Certifying and removing disparate impact. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 259–268. ACM, 2015.

Ruth C Fong and Andrea Vedaldi. Interpretable explanations of black boxes by meaningful perturbation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3429–3437, 2017.

Nicholas Frosst and Geoffrey Hinton. Distilling a neural network into a soft decision tree. *arXiv preprint arXiv:1711.09784*, 2017.

Fabian B Fuchs, Oliver Groth, Adam R Kosoriek, Alex Bewley, Markus Wulfmeier, Andrea Vedaldi, and Ingmar Posner. Neural stethoscopes: Unifying analytic, auxiliary and adversarial network probing. *arXiv preprint arXiv:1806.05502*, 2018.

Akira Fukui, Dong Huk Park, Daylen Yang, Anna Rohrbach, Trevor Darrell, and Marcus Rohrbach. Multimodal compact bilinear pooling for visual question answering and visual grounding. *arXiv preprint arXiv:1606.01847*, 2016.

Artur S d'Avila Garcez, Krysia B Broda, and Dov M Gabbay. *Neural-symbolic learning systems: Foundations and applications*. Springer Science & Business Media, 2012.

Yunchao Gong, Liwei Wang, Micah Hodosh, Julia Hockenmaier, and Svetlana Lazebnik. Improving image-sentence embeddings using large weakly annotated photo collections. In *European Conference on Computer Vision*, pages 529–545. Springer, 2014.

Abel Gonzalez-Garcia, Davide Modolo, and Vittorio Ferrari. Do semantic parts emerge in convolutional neural networks? *International Journal of Computer Vision*, 126(5): 476–494, 2018.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples (2014). *arXiv preprint arXiv:1412.6572*, 2015.

Bryce Goodman and Seth Flaxman. European union regulations on algorithmic decision-making and a âĂIJright to explanationâĂİ. *AI Magazine*, 38(3):50–57, 2017.

Paula Gordaliza, Eustasio Del Barrio, Gamboa Fabrice, and Jean-Michel Loubes. Obtaining fairness using optimal transport theory. In *International Conference on Machine Learning*, pages 2357–2365, 2019.

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the V in VQA matter: Elevating the role of image understanding in visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6904–6913, 2017.

Alex Graves, Greg Wayne, and Ivo Danihelka. Neural turing machines. *arXiv preprint arXiv:1410.5401*, 2014.

Oliver Groth, Fabian B Fuchs, Ingmar Posner, and Andrea Vedaldi. Shapestacks: Learning vision-based physical intuition for generalised object stacking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 702–717, 2018.

Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Franco Turini, Fosca Giannotti, and Dino Pedreschi. A survey of methods for explaining black box models. *ACM Computing Surveys (CSUR)*, 51(5):93, 2018.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel R Bowman, and Noah A Smith. Annotation artifacts in natural language inference data. *arXiv preprint arXiv:1803.02324*, 2018.

Moritz Hardt, Eric Price, Nati Srebro, et al. Equality of opportunity in supervised learning. In *Advances in neural information processing systems*, pages 3315–3323, 2016.

Michael Harradon, Jeff Druce, and Brian Ruttenberg. Causal learning and explanation of deep neural networks via autoencoded activations. *arXiv preprint arXiv:1802.00541*, 2018.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, pages 2980–2988. IEEE, 2017.

Ruidan He, Wee Sun Lee, Hwee Tou Ng, and Daniel Dahlmeier. Effective attention modeling for aspect-level sentiment classification. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1121–1131, 2018.

Hoda Heidari, Claudio Ferrari, Krishna Gummadi, and Andreas Krause. Fairness behind a veil of ignorance: A welfare analysis for automated decision making. In *Advances in Neural Information Processing Systems*, pages 1265–1276, 2018.

Lisa Anne Hendricks, Zeynep Akata, Marcus Rohrbach, Jeff Donahue, Bernt Schiele, and Trevor Darrell. Generating visual explanations. In *European Conference on Computer Vision*, pages 3–19. Springer, 2016.

Dan Hendrycks and Kevin Gimpel. Early methods for detecting adversarial images. *arXiv preprint arXiv:1608.00530*, 2016.

Jay Heo, Hae Beom Lee, Saehoon Kim, Juho Lee, Kwang Joon Kim, Eunho Yang, and Sung Ju Hwang. Uncertainty-aware attention for reliable interpretation and prediction. In *Advances in Neural Information Processing Systems*, pages 909–918, 2018.

Michael Hind, Dennis Wei, Murray Campbell, Noel CF Codella, Amit Dhurandhar, Aleksandra Mojsilović, Karthikeyan Natesan Ramamurthy, and Kush R Varshney. TED: Teaching AI to explain its decisions. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, pages 123–129. ACM, 2019.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Geoffrey E Hinton. Learning distributed representations of concepts. In *Proc. of the Annual Conference of the Cognitive Science Society*, volume 1, page 12. Amherst, MA, 1986.

Giles Hooker. Discovering additive structure in black box functions. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 575–580, 2004.

Giles Hooker. Generalized functional anova diagnostics for high-dimensional functions of dependent variables. *Journal of Computational and Graphical Statistics*, 16(3):709–732, 2007.

Holger Hoos and Kevin Leyton-Brown. An efficient approach for assessing hyperparameter importance. In *International Conference on Machine Learning*, pages 754–762, 2014.

Bo-Jian Hou and Zhi-Hua Zhou. Learning With interpretable structure from gated RNN. *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

Drew A Hudson and Christopher D Manning. Compositional attention networks for machine reasoning. *arXiv preprint arXiv:1803.03067*, 2018.

Rahul Iyer, Yuezhang Li, Huao Li, Michael Lewis, Ramitha Sundar, and Katia Sycara. Transparency and explanation in deep reinforcement learning neural networks. In *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, pages 144–150. ACM, 2018.

Sarthak Jain and Byron C Wallace. Attention is not explanation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3543–3556, 2019.

Heinrich Jiang, Been Kim, Melody Guan, and Maya Gupta. To trust or not to trust a classifier. In *Advances in neural information processing systems*, pages 5541–5552, 2018a.

Yu Jiang, Vivek Natarajan, Xinlei Chen, Marcus Rohrbach, Dhruv Batra, and Devi Parikh. Pythia v0. 1: the winning entry to the vqa challenge 2018. *arXiv preprint arXiv:1807.09956*, 2018b.

Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. CLEVR: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2901–2910, 2017a.

Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Judy Hoffman, Li Fei-Fei, C Lawrence Zitnick, and Ross B Girshick. Inferring and executing programs for visual reasoning. In *ICCV*, pages 3008–3017, 2017b.

Faisal Kamiran and Toon Calders. Classification with no discrimination by preferential sampling. In *Proc. 19th Machine Learning Conf. Belgium and The Netherlands*, pages 1–6. Citeseer, 2010.

Faisal Kamiran and Toon Calders. Data preprocessing techniques for classification without discrimination. *Knowledge and Information Systems*, 33(1):1–33, 2012.

Toshihiro Kamishima, Shotaro Akaho, and Jun Sakuma. Fairness-aware learning through regularization approach. In *2011 IEEE 11th International Conference on Data Mining Workshops*, pages 643–650. IEEE, 2011.

Daniel Kang, Deepti Raghavan, Peter Bailis, and Matei Zaharia. Model assertions for debugging machine learning. In *NeurIPS MLSys Workshop*, 2018.

Michael Kearns, Seth Neel, Aaron Roth, and Zhiwei Steven Wu. Preventing fairness gerrymandering: Auditing and learning for subgroup fairness. In *International Conference on Machine Learning*, pages 2569–2577, 2018.

Been Kim, Cynthia Rudin, and Julie A Shah. The Bayesian case model: A generative approach for case-based reasoning and prototype classification. In *Advances in Neural Information Processing Systems*, pages 1952–1960, 2014.

Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas, et al. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (TCAV). In *International Conference on Machine Learning*, pages 2673–2682, 2018a.

Jin-Hwa Kim, Jaehyun Jun, and Byoung-Tak Zhang. Bilinear attention networks. *arXiv preprint arXiv:1805.07932*, 2018b.

Jinkyu Kim, Anna Rohrbach, Trevor Darrell, John Canny, and Zeynep Akata. Textual explanations for self-driving vehicles. In *Proceedings of the European conference on computer vision (ECCV)*, pages 563–578, 2018c.

Pieter-Jan Kindermans, Sara Hooker, Julius Adebayo, Maximilian Alber, Kristof T Schütt, Sven Dähne, Dumitru Erhan, and Been Kim. The (un)reliability of saliency methods. In *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, pages 267–280. Springer, 2019.

Janet L Kolodner. An introduction to case-based reasoning. *Artificial intelligence review*, 6(1):3–34, 1992.

Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, Michael Bernstein, and Li Fei-Fei. Visual genome: Connecting language and vision using crowdsourced dense image annotations. 2016. URL https://arxiv.org/abs/1602.07332.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 1097–1105, 2012.

Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*, 2016.

Sebastian Lapuschkin, Alexander Binder, Grégoire Montavon, Klaus-Robert Muller, and Wojciech Samek. Analyzing classifiers: Fisher vectors and deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2912–2920, 2016.

Yann LeCun. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

Yann LeCun, LD Jackel, Leon Bottou, A Brunot, Corinna Cortes, JS Denker, Harris Drucker, I Guyon, UA Muller, Eduard Sackinger, et al. Comparison of learning algorithms for handwritten digit recognition. In *International conference on artificial neural networks*, volume 60, pages 53–60. Perth, Australia, 1995.

Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Yann LeCun, Kilian Weinberger, Patrice Simard, and Rich Caruana. Debate: Interpretability is necessary in machine learning. `https://www.youtube.com/watch?v=2hW05ZfsUUo`, 2017.

John D Lee and Katrina A See. Trust in automation: Designing for appropriate reliance. *Human factors*, 46(1):50–80, 2004.

Tao Lei, Regina Barzilay, and Tommi Jaakkola. Rationalizing neural predictions. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 107–117, 2016.

Gaël Letarte, Frédérik Paradis, Philippe Giguère, and François Laviolette. Importance of self-attention for sentiment analysis. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 267–275, 2018.

Jiwei Li, Will Monroe, and Dan Jurafsky. Understanding neural networks through representation erasure. *arXiv preprint arXiv:1612.08220*, 2016.

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 *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018a.

Yao Li, Martin Renqiang Min, Wenchao Yu, Cho-Jui Hsieh, Thomas Lee, and Erik Kruus. Optimal transport classifier: Defending against adversarial attacks by regularized deep embedding. *arXiv preprint arXiv:1811.07950*, 2018b.

Min Lin, Qiang Chen, and Shuicheng Yan. Network in network. *arXiv preprint arXiv:1312.4400*, 2013.

Zachary C Lipton. The mythos of model interpretability. *Communications of the ACM*, 61 (10):36–43, 2018.

Hui Liu, Qingyu Yin, and William Yang Wang. Towards explainable NLP: A generative explanation framework for text classification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5570–5581, 2019.

Shixia Liu, Xiting Wang, Mengchen Liu, and Jun Zhu. Towards better analysis of machine learning models: A visual analytics perspective. *Visual Informatics*, 1(1):48–56, 2017.

Xuanqing Liu, Yao Li, Chongruo Wu, and Cho-Jui Hsieh. Adv-bnn: Improved adversarial defense through robust bayesian neural network. *arXiv preprint arXiv:1810.01279*, 2018.

Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song. Delving into transferable adversarial examples and black-box attacks. *arXiv preprint arXiv:1611.02770*, 2016.

Christos Louizos, Kevin Swersky, Yujia Li, Max Welling, and Richard Zemel. The variational fair autoencoder. *arXiv preprint arXiv:1511.00830*, 2015.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015.

Dongdong Lv, Shuhan Yuan, Meizi Li, and Yang Xiang. An empirical study of machine learning algorithms for stock daily trading strategy. *Mathematical Problems in Engineering*, 2019, 2019.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

Prashan Madumal. Explainable agency in reinforcement learning agents. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 13724–13725, 2020.

Prashan Madumal, Tim Miller, Liz Sonenberg, and Frank Vetere. Explainable reinforcement learning through a causal lens. *arXiv preprint arXiv:1905.10958*, 2019.

Carey E Priebe David J Marchette and Jason G DeVinney Diego A Socolinsky. Classification using class cover catch digraphs. *Journal of Classification*, 20:3–23, 2003.

David Mascharka, Philip Tran, Ryan Soklaski, and Arjun Majumdar. Transparency by design: Closing the gap between performance and interpretability in visual reasoning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4942–4950, 2018.

Iacopo Masi, Yue Wu, Tal Hassner, and Prem Natarajan. Deep face recognition: A survey. In *2018 31st SIBGRAPI conference on graphics, patterns and images (SIBGRAPI)*, pages 471–478. IEEE, 2018.

Inc Matterport. Mask rcnn pytorch implementation. https://github.com/multimodallearning/pytorch-mask-rcnn. Accessed: 2018-10-30.

David Alvarez Melis and Tommi Jaakkola. Towards robust interpretability with self-explaining neural networks. In *Advances in Neural Information Processing Systems*, pages 7775–7784, 2018.

Dongyu Meng and Hao Chen. Magnet: A two-pronged defense against adversarial examples. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 135–147. ACM, 2017.

Aditya Krishna Menon and Robert C Williamson. The cost of fairness in binary classification. In *Conference on Fairness, Accountability and Transparency*, pages 107–118, 2018.

Tim Miller. Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence*, 2018.

Grégoire Montavon, Sebastian Lapuschkin, Alexander Binder, Wojciech Samek, and Klaus-Robert Müller. Explaining nonlinear classification decisions with deep Taylor decomposition. *Pattern Recognition*, 65:211–222, 2017.

Grégoire Montavon, Wojciech Samek, and Klaus-Robert Müller. Methods for interpreting and understanding deep neural networks. *Digital Signal Processing*, 73:1–15, 2018.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: A simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2574–2582, 2016.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1765–1773, 2017.

Khushboo Munir, Hassan Elahi, Afsheen Ayub, Fabrizio Frezza, and Antonello Rizzi. Cancer diagnosis using deep learning: A bibliographic review. *Cancers*, 11(9):1235, 2019.

W James Murdoch and Arthur Szlam. Automatic rule extraction from long short term memory networks. *International Conference on Learning Representations*, 2017.

Anh Nguyen, Jason Yosinski, and Jeff 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.

Camburu Oana-Maria, Shillingford Brendan, Minervini Pasquale, Lukasiewicz Thomas, and Blunsom Phil. Make up your mind! Adversarial generation of inconsistent natural language explanations. *arXiv preprint arXiv:1910.03065*, 2019.

Oluwatobi Olabiyi, Eric Martinson, Vijay Chintalapudi, and Rui Guo. Driver action prediction using deep (bidirectional) recurrent neural network. *arXiv preprint arXiv:1706.02257*, 2017.

Chris Olah, Alexander Mordvintsev, and Ludwig Schubert. Feature visualization. *Distill*, 2017. doi: 10.23915/distill.00007. https://distill.pub/2017/feature-visualization.

Chris Olah, Arvind Satyanarayan, Ian Johnson, Shan Carter, Ludwig Schubert, Katherine Ye, and Alexander Mordvintsev. The building blocks of interpretability. *Distill*, 2018. doi: 10.23915/distill.00010. https://distill.pub/2018/building-blocks.

Mahbod Olfat and Anil Aswani. Spectral algorithms for computing fair support vector machines. In *International Conference on Artificial Intelligence and Statistics*, pages 1933–1942, 2018.

Utku Ozbulak. PyTorch CNN visualizations. `https://github.com/utkuozbulak/pytorch-cnn-visualizations`, 2019.

Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami. The limitations of deep learning in adversarial settings. In *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 372–387. IEEE, 2016a.

Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. Distillation as a defense to adversarial perturbations against deep neural networks. In *2016 IEEE Symposium on Security and Privacy (SP)*, pages 582–597. IEEE, 2016b.

Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia conference on computer and communications security*, pages 506–519. ACM, 2017.

Dong Huk Park, Lisa Anne Hendricks, Zeynep Akata, Bernt Schiele, Trevor Darrell, and Marcus Rohrbach. Attentive explanations: Justifying decisions and pointing to the evidence. *arXiv preprint arXiv:1612.04757*, 2016.

Dong Huk Park, Lisa Anne Hendricks, Zeynep Akata, Anna Rohrbach, Bernt Schiele, Trevor Darrell, and Marcus Rohrbach. Multimodal explanations: Justifying decisions and pointing to the evidence. In *31st IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

Adrián Pérez-Suay, Valero Laparra, Gonzalo Mateo-García, Jordi Muñoz-Marí, Luis Gómez-Chova, and Gustau Camps-Valls. Fair kernel learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 339–355. Springer, 2017.

Wolter Pieters. Explanation and trust: what to tell the user in security and AI? *Ethics and information technology*, 13(1):53–64, 2011.

Geoff Pleiss, Manish Raghavan, Felix Wu, Jon Kleinberg, and Kilian Q Weinberger. On fairness and calibration. In *Advances in Neural Information Processing Systems*, pages 5680–5689, 2017.

Bryan A Plummer, Liwei Wang, Chris M Cervantes, Juan C Caicedo, Julia Hockenmaier, and Svetlana Lazebnik. Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. In *Proceedings of the IEEE international conference on computer vision*, pages 2641–2649, 2015.

Erika Puiutta and Eric Veith. Explainable reinforcement learning: A survey. *arXiv preprint arXiv:2005.06247*, 2020.

Chuan Qin, Hengshu Zhu, Tong Xu, Chen Zhu, Liang Jiang, Enhong Chen, and Hui Xiong. Enhancing person-job fit for talent recruitment: An ability-aware neural network approach. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 25–34. ACM, 2018.

Maithra Raghu, Justin Gilmer, Jason Yosinski, and Jascha Sohl-Dickstein. SVCCA: Singular vector canonical correlation analysis for deep learning dynamics and interpretability. In *Advances in Neural Information Processing Systems*, pages 6076–6085, 2017.

Gabriëlle Ras, Marcel van Gerven, and Pim Haselager. Explanation methods in deep learning: Users, values, concerns and challenges. In *Explainable and Interpretable Models in Computer Vision and Machine Learning*, pages 19–36. Springer, 2018.

Priyanka P Raut, Namrata R Borkar, and ME Student. Techniques and implementation of face spoof recognition: Perspectives and prospects. *IJESC*, 8(1), 2018.

Mengye Ren, Ryan Kiros, and Richard Zemel. Exploring models and data for image question answering. In *Advances in neural information processing systems*, pages 2953–2961, 2015.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Model-agnostic interpretability of machine learning. *arXiv preprint arXiv:1606.05386*, 2016a.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Nothing else matters: Model-agnostic explanations by identifying prediction invariance. *arXiv preprint arXiv:1611.05817*, 2016b.

Marco Tulio Ribeiro, Sameer Singh, and Carlos 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. ACM, 2016c.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Anchors: High-precision model-agnostic explanations. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

Marko Robnik-Šikonja and Igor Kononenko. Explaining classifications for individual instances. *IEEE Transactions on Knowledge and Data Engineering*, 20(5):589–600, 2008.

Andras Rozsa, Ethan M Rudd, and Terrance E Boult. Adversarial diversity and hard positive generation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 25–32, 2016.

Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206, 2019.

Sara Sabour, Yanshuai Cao, Fartash Faghri, and David J Fleet. Adversarial manipulation of deep representations. *arXiv preprint arXiv:1511.05122*, 2015.

Pouya Samangouei, Maya Kabkab, and Rama Chellappa. Defense-gan: Protecting classifiers against adversarial attacks using generative models. *arXiv preprint arXiv:1805.06605*, 2018.

Wojciech Samek, Thomas Wiegand, and Klaus-Robert Müller. Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models. *arXiv preprint arXiv:1708.08296*, 2017.

Adam Santoro, David Raposo, David G Barrett, Mateusz Malinowski, Razvan Pascanu, Peter Battaglia, and Tim Lillicrap. A simple neural network module for relational reasoning. In *Advances in neural information processing systems*, pages 4967–4976, 2017.

Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv 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.

Pedro Sequeira, Eric Yeh, and Melinda T Gervasio. Interestingness elements for explainable reinforcement learning through introspection. In *IUI Workshops*, page 7, 2019.

Wenjie Shi, Zhuoyuan Wang, Shiji Song, and Gao Huang. Self-supervised discovering of causal features: Towards interpretable reinforcement learning. *arXiv preprint arXiv:2003.07069*, 2020.

Avanti Shrikumar, Peyton Greenside, and Anshul 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.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

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

Ilya M Sobol. Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates. *Mathematics and computers in simulation*, 55(1-3):271–280, 2001.

Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*, 2014.

Siddhant Srivastava, Anupam Shukla, and Ritu Tiwari. Machine translation: From statistical to modern deep-learning practices. *arXiv preprint arXiv:1812.04238*, 2018.

Jiawei Su, Danilo Vasconcellos Vargas, and Sakurai Kouichi. One pixel attack for fooling deep neural networks. *arXiv preprint arXiv:1710.08864*, 2017.

Jiawei Su, Danilo Vasconcellos Vargas, and Kouichi Sakurai. One pixel attack for fooling deep neural networks. *IEEE Transactions on Evolutionary Computation*, 2019.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Gradients of counterfactuals. *arXiv preprint arXiv:1611.02639*, 2016.

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

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

Pedro Tabacof and Eduardo Valle. Exploring the space of adversarial images. In *2016 International Joint Conference on Neural Networks (IJCNN)*, pages 426–433. IEEE, 2016.

Sarah Tan, Rich Caruana, Giles Hooker, Paul Koch, and Albert Gordo. Learning global additive explanations for neural nets using model distillation. *arXiv preprint arXiv:1801.08640*, 2018.

Damien Teney, Peter Anderson, Xiaodong He, and Anton van den Hengel. Tips and tricks for visual question answering: Learnings from the 2017 challenge. *arXiv preprint arXiv:1708.02711*, 2017.

Damien Teney, Peter Anderson, Xiaodong He, and Anton van den Hengel. Tips and tricks for visual question answering: Learnings from the 2017 challenge. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4223–4232, 2018.

Tijmen Tieleman and Geoffrey Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2):26–31, 2012.

Tatiana Tommasi, Novi Patricia, Barbara Caputo, and Tinne Tuytelaars. A deeper look at dataset bias. In *Domain adaptation in computer vision applications*, pages 37–55. Springer, 2017.

Amrita S Tulshan and Sudhir Namdeorao Dhage. Survey on virtual assistant: Google Assistant, Siri, Cortana, Alexa. In *International Symposium on Signal Processing and Intelligent Recognition Systems*, pages 190–201. Springer, 2018.

Amazon Mechanical Turk. Amazon mechanical turk. *Retrieved August*, 17:2012, 2012.

Kush R Varshney and Homa Alemzadeh. On the safety of machine learning: Cyber-physical systems, decision sciences, and data products. *Big data*, 5(3):246–255, 2017.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3156–3164, 2015.

Hoa Trong Vu, Claudio Greco, Aliia Erofeeva, Somayeh Jafaritazehjan, Guido Linders, Marc Tanti, Alberto Testoni, Raffaella Bernardi, and Albert Gatt. Grounded textual entailment. *arXiv preprint arXiv:1806.05645*, 2018.

Yequan Wang, Minlie Huang, Li Zhao, et al. Attention-based LSTM for aspect-level sentiment classification. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pages 606–615, 2016.

Sarah Wiegreffe and Yuval Pinter. Attention is not not explanation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 11–20, 2019.

Blake Woodworth, Suriya Gunasekar, Mesrob I Ohannessian, and Nathan Srebro. Learning non-discriminatory predictors. *arXiv preprint arXiv:1702.06081*, 2017.

Qi Wu, Damien Teney, Peng Wang, Chunhua Shen, Anthony Dick, and Anton van den Hengel. Visual question answering: A survey of methods and datasets. *Computer Vision and Image Understanding*, 163:21–40, 2017.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Cihang Xie, Jianyu Wang, Zhishuai Zhang, Zhou Ren, and Alan Yuille. Mitigating adversarial effects through randomization. *arXiv preprint arXiv:1711.01991*, 2017a.

Ning Xie, Md Kamruzzaman Sarker, Derek Doran, Pascal Hitzler, and Michael Raymer. Relating input concepts to convolutional neural network decisions. *arXiv preprint arXiv:1711.08006*, 2017b.

Ning Xie, Farley Lai, Derek Doran, and Asim Kadav. Visual entailment task for visually-grounded language learning. *arXiv preprint arXiv:1811.10582*, 2018.

Ning Xie, Farley Lai, Derek Doran, and Asim Kadav. Visual entailment: A novel task for fine-grained image understanding. *arXiv preprint arXiv:1901.06706*, 2019.

Ning Xie, Gabrielle Ras, Marcel van Gerven, and Derek Doran. Explainable deep learning: A field guide for the uninitiated. *arXiv preprint arXiv:2004.14545*, 2020.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pages 2048–2057, 2015.

Shuoheng Yang, Yuxin Wang, and Xiaowen Chu. A survey of deep learning techniques for neural machine translation. *arXiv preprint arXiv:2002.07526*, 2020.

B Yegnanarayana. *Artificial neural networks*. PHI Learning Pvt. Ltd., 2009.

Junho Yim, Donggyu Joo, Jihoon Bae, and Junmo Kim. A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. In *The IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

Jason Yosinski, Jeff Clune, Anh Nguyen, Thomas Fuchs, and Hod Lipson. Understanding neural networks through deep visualization. *arXiv preprint arXiv:1506.06579*, 2015.

Peter Young, Alice Lai, Micah Hodosh, and Julia Hockenmaier. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *Transactions of the Association for Computational Linguistics*, 2:67–78, 2014.

Xiaoyong Yuan, Pan He, Qile Zhu, Rajendra Rana Bhat, and Xiaolin Li. Adversarial examples: Attacks and defenses for deep learning. *arXiv preprint arXiv:1712.07107*, 2017.

Xiaoyong Yuan, Pan He, Qile Zhu, and Xiaolin Li. Adversarial examples: Attacks and defenses for deep learning. *IEEE transactions on neural networks and learning systems*, 2019.

Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rodriguez, and Krishna P Gummadi. Fairness beyond disparate treatment & disparate impact: Learning classification without disparate mistreatment. In *Proceedings of the 26th International Conference on World Wide Web*, pages 1171–1180. International World Wide Web Conferences Steering Committee, 2017a.

Muhammad Bilal Zafar, Isabel Valera, Manuel Rodriguez, Krishna Gummadi, and Adrian Weller. From parity to preference-based notions of fairness in classification. In *Advances in Neural Information Processing Systems*, pages 229–239, 2017b.

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

Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European Conference on Computer Vision*, pages 818–833. Springer, 2014b.

Matthew D Zeiler, Graham W Taylor, Rob Fergus, et al. Adaptive deconvolutional networks for mid and high level feature learning. In *ICCV*, volume 1, page 6, 2011.

Rowan Zellers, Yonatan Bisk, Ali Farhadi, and Yejin Choi. From recognition to cognition: Visual commonsense reasoning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6720–6731, 2019.

Rich Zemel, Yu Wu, Kevin Swersky, Toni Pitassi, and Cynthia Dwork. Learning fair representations. In *International Conference on Machine Learning*, pages 325–333, 2013.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*, 2016.

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, Ruiming Cao, Ying Nian Wu, and Song-Chun Zhu. Growing interpretable part graphs on ConvNets via multi-shot learning. In *Proc. of AAAI Conference on Artificial Intelligence*, pages 2898–2906, 2017.

Quanshi Zhang, Ruiming Cao, Feng Shi, Ying Nian Wu, and Song-Chun Zhu. Interpreting CNN knowledge via an explanatory graph. *Proc. of AAAI Conference on Artificial Intelligence*, 2018.

Quanshi Zhang, Yu Yang, Haotian Ma, and Ying Nian Wu. Interpreting CNNs via decision trees. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6261–6270, 2019a.

Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay. Deep learning based recommender system: A survey and new perspectives. *ACM Computing Surveys (CSUR)*, 52(1):5, 2019b.

Wei Emma Zhang, Quan Z Sheng, and Ahoud Abdulrahmn F Alhazmi. Generating textual adversarial examples for deep learning models: A survey. *arXiv preprint arXiv:1901.06796*, 2019c.

Yue Zhao, Maciej K Hryniewicki, Francesca Cheng, Boyang Fu, and Xiaoyu Zhu. Employee turnover prediction with machine learning: A reliable approach. In *Proceedings of SAI intelligent systems conference*, pages 737–758. Springer, 2018.

Zhengli Zhao, Dheeru Dua, and Sameer Singh. Generating natural adversarial examples. *arXiv preprint arXiv:1710.11342*, 2017.

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

Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2921–2929, 2016a.

Bolei Zhou, Aditya Khosla, Agata Lapedriza, Antonio Torralba, and Aude Oliva. Places: An image database for deep scene understanding. *arXiv preprint arXiv:1610.02055*, 2016b.

Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

Yuke Zhu, Oliver Groth, Michael Bernstein, and Li Fei-Fei. Visual7w: Grounded question answering in images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4995–5004, 2016.

Luisa M Zintgraf, Taco S Cohen, Tameem Adel, and Max Welling. Visualizing deep neural network decisions: Prediction difference analysis. *International Conference on Learning Representations*, 2017.