Abstract

In many scenarios, human decisions are explained based on some high-level concepts. In this work, we take a step in the interpretability of neural networks by examining their internal representation or neuron’s activations against concepts. A concept is characterized by a set of samples that have specific features in common. We propose a framework to check the existence of a causal relationship between a concept (or its negation) and task classes. While the previous methods focus on the importance of a concept to a task class, we go further and introduce four measures to quantitatively determine the order of causality. Through experiments, we demonstrate the effectiveness of the proposed method in explaining the relationship between a concept and the predictive behaviour of a neural network.

Cause and Effect: Concept-based Explanation of Neural Networks

by

Mohammad Nokhbeh Zaeem

A Thesis submitted to
the Faculty of Graduate Studies and Research
in partial fulfilment of
the requirements for the degree of
Master of Computer Science

Department of Computer Science

Carleton University

Ottawa, Ontario, Canada

2021

Copyright ©

2021 - Mohammad Nokhbeh Zaeem

The undersigned recommend to
the Faculty of Graduate Studies and Research
acceptance of the Thesis

Cause and Effect: Concept-based Explanation of Neural Networks

Submitted by **Mohammad Nokhbeh Zaeem**
in partial fulfilment of the requirements for the degree of
Master of Computer Science

Majid Komeili, Thesis Supervisor

Yongyi Mao, External Examiner

Yuhong Guo, Internal Examiner

Elizabeth Stobert, Chair of Defence

Carleton University

2021

Table of Contents

Abstract	1
Table of Contents	iii
List of Tables	vi
List of Figures	vii
1 Introduction	1
1.1 Importance of Interpretability	2
1.2 Problem Statement	3
1.3 Previous Works Shortcomings	4
1.3.1 Nonlinear Concepts	5
1.3.2 Causality	6
1.4 Contributions	7
1.5 Organization of Thesis	8
2 Background	9
2.1 Logic and Causal Expressions	9
2.2 Neural Networks	11
2.2.1 Neurons	11
2.3 Structures of Neural Networks	12

2.3.1	Recurrent Neural Network	12
2.3.2	Feed-Forward Neural Network	13
2.4	Training Neural Networks	15
2.4.1	Cost Functions	17
2.5	Accuracy Measurements	18
2.5.1	Confusion Matrix	18
2.5.2	Accuracy	19
2.5.3	Precision and Recall	19
2.5.4	F1 Score	20
2.5.5	ROC Curver	20
2.6	Split of Samples	21
2.7	Overfitting and Prevention	22
2.7.1	Early Stopping	22
2.7.2	Batch Normalization	23
2.7.3	Dropout	23
2.8	Commonly Used Architectures	23
2.8.1	Alexnet	23
2.8.2	VGG16	24
2.8.3	Resnet18	24
2.8.4	InceptionV3	26
3	Literature Review	30
3.1	Intrinsic Interpretability	30
3.2	Local Interpretability	31
3.3	Model-agnostic Interpretability	31
3.4	Difference or Gradient-based Methods	32
3.5	Example-Based Interpretability	32

3.6	Post Hoc Concept-based Interpretability of Neural Networks	33
3.6.1	Summary	35
4	Proposed Method	38
4.1	Representing Concepts	38
4.2	Quantifying the Relationships	41
4.3	Discussions	44
4.3.1	Distribution Sample Set	44
4.3.2	Inspection Layer Choice	44
4.3.3	Trustworthiness of a Network	45
5	Experiments and Results	47
5.1	Coloured MNIST Dataset	47
5.2	Images with Captions Dataset	48
5.3	Analysis of Concept Influence	49
5.4	Informativeness of the Proposed Relationship Measures	52
5.5	Comparison with Linear Based Classifier	53
5.6	Comparison with TCAV method	54
5.7	Comparison with IBD method	56
6	Conclusion and Future Work	73
	List of References	75

List of Tables

1	Structure of Alexnet neural network.	24
2	Structure of VGG16 neural network.	25
3	Structure of Resnet18 neural network.	27
4	Structure of InceptionV3 neural network.	28
5	mixed-a blocks. The parameter n is the number of input channels and the parameter m depends on which inceptionA block is being processes.	29
6	Comparison of concept-based methods. Y: Yes, N: No, +: global ex- planation but contribution of concept for each input can be calculated.	36
7	Different datasets that include image annotations.	37
8	Explanations for several classes using our method and IBD method. The concepts labelled as irrelevant by the majority of three annotators are in bold.	61
9	Explanations for several classes using our method and IBD method. The concepts labelled as irrelevant by the majority of three annotators are in bold.	62

List of Figures

1	(a) There are four different subsets that might potentially be empty. Depending on which subset is empty, one of the four conditions will happen. (b) 3 relative positions of two sets. With respect to (T) Task class, (S) Sufficient condition, (N) Necessary condition, and (I) Negative necessary.	10
2	Visual structure of an artificial neuron.	12
3	How each calculation in a convolutional layer works. The weights for the red calculation is the same as the green calculation.	14
4	How each calculation in a pooling layer with stride 2 works. Each neuron depends on 4 neurons of the previous layer as shown.	15
5	Gradient Descend in a two dimensional space. The initial condition has a role in the quality of the local minimal point found.	16
6	ROC curve. While the classifier of the green curve is better than both blue and purple, the blue and purple classifiers are not comparable. Blue is better for application that recall is more important while purple is better for applications that FPR is important.	21
7	Overfitting in training.	22
8	Structure of residual block used in Resnet.	25

9	(a) samples of positive and negative concepts, (b) distribution sample set, (c) representation of task class and concept in network, (d) analysis of the relationship between the task class and concept showing the concept is a necessary condition for the task class.	39
10	The two sets might not have a relationship but on the manifold of data they might have a strong relationship.	42
11	The process of creating quantification curve for necessary score. In this process we check if the task class is true with probability at least t then the concept is true with probability of at least $1 - t$. The highest value is achieved when there is no counter-example (F) samples. . . .	43
12	How the two hint colours are chosen for each figure in ColorDataset1. Each figure is associated with two different colours that have the highest distance in colour space.	48
13	Two samples from CaptionDataset1.	49
14	Comparison of two networks with the same structure, while one has been trained on CaptionDataset1 (left side), and the other has been trained on CaptionDataset2 (right side). The area of one under the necessary curve shows that the concept is necessary for the task class. The values for each AUC are mentioned at the bottom of the figure. These results are extracted from the last layers of a neural network with an alexnet structure.	51
15	Analysis for class 0 (left) and class 1 (right). The causal relationship of a colour concept with two different classes. This concept is necessary for class 1 (right) but the concept is inconsistent with class 0 (left) so the negative of concept is necessary for class 0.	53

16	Shortcoming of the linear layers to pick up a concept in earlier sections of the neural network (right) while it can be easily detected with the proposed method (left). If a concept is not present, knowing where in the network the concept is disappearing can be a helpful tool. For instance in transfer learning.	55
17	Results of the caption experiment for different linear (right) and non-linear (left) concept classifiers show that directional derivative can be misleading in both cases. The joint distribution of concept and class decisions for the nonlinear and linear models (first row). Changes of relationship measures with respect to different thresholding (second row). Distribution of directional derivative (third row). Moreover, the new measures give us more information about the type of the relationship between concept and task class.	57
18	Results of caption experiment for different linear (right) and nonlinear (left) concept classifiers show that directional derivative can be misleading in both cases.	58
19	Some minimal and maximal samples of concepts.	63
20	Continued.	64
21	Some minimal and maximal samples of animal related concepts. . . .	65
22	Continued.	66
23	Some minimal and maximal samples of bathroom related concepts. . .	67
24	Continued.	68
25	Some minimal and maximal samples of kitchen related concepts. . . .	69
26	Continued.	70
27	Deepdream results of the concept maximal inputs for the top 12 most common concepts, while some of them make sense, most of them are unexpected.	71

28	Deepdream results of the concept maximal inputs for the second top 12 most common concepts, while some of them make sense, most of them are unexpected.	72
----	---	----

Chapter 1

Introduction

Applications of Machine Learning (ML) and Artificial Intelligence (AI) as methods to help with automatic decision-making have grown to the extent that it has raised concerns about the trustworthiness of these methods. There have been rules and regulations all around the world that organizations should provide explanations for decisions made by their automated decision-making systems [14]. These concerns often exist whenever the problem at hand is not fully understood, explored or our knowledge of the problem is not complete. Knowing the reasoning of machine learning methods may also help with catching their unwanted behaviours by comparing the reasoning to experts' understanding of problems. An example of such unwanted behaviours is biases in decision-making. On the other hand, explanations can be used to extract the knowledge gained by these black boxes as well. Knowledge extraction can help with a better understanding of the AI view of the problem and the machine learning methods.

Neural networks as one of the most promising forms of AI with high performance on classification problems like ImageNet challenge¹ have been criticized for their black box decision-making process.

¹ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [7, 34].

1.1 Importance of Interpretability

Since the adaptation of machine learning in many domains, there have been some concerns about the reliability of the machine learning models. Some of the most important questions are whether these models consistent with the current knowledge at hand, and if not which one is at fault? Can we extract knowledge from these models and put the knowledge into theoretical tests for verification? What are the key concepts of input for the classification task and how these concepts influence the decision of the machine learning method?

In more sensitive applications like medical diagnosis based on patient symptoms, it's not enough to just have a prediction for the patient's disease. Often a good explanation is required for these applications that can justify the decision and check that the decision making is consistent with the subject expert knowledge.

In other applications, the stakes are not that high but a justification might be required for convincing the humans involved in the decision-making process. For instance, for denying a loan application, the applicant may ask for an explanation and the loan offering institute might be legally obliged to offer answers to such questions.

In applications with high dimensional input spaces, like image, audio and video, the relation between inputs and outputs do not form an easy-to-understand relation. For instance, explaining an image is classified as doctor because "its specific values of pixels have a property" does not form a good explanation. In these applications, a middle high-level meaningful abstraction is a necessity to explain the decision to humans. Humans have been using high-level abstraction or concepts for the simplification of explanations and definitions. For instance, in a definition, a doctor is a person probably wearing a lab coat in a clinic examining patients with medical instruments and treating them with medicine. Lab coat, clinic, medical instruments are all examples of high-level concepts. The explanations in high-level concepts make the

explanation more understandable and easier to examine. For example, the relevance of different concepts to the doctor class is intuitive for humans.

1.2 Problem Statement

One of the most important questions asked about neural network’s decisions is how a certain *concept* influences the internal representation and eventually the output of the neural network. Here, a concept is a representation of a feature and is defined by a set of samples with that feature, against a random set [19]. For example, for predicting the job title of a person from their image, this feature can be as simple as the colour of uniform (e.g. white or pink), background (e.g. office, ambulance or clinic), or objects (e.g. stethoscope) in the image.

Breaking down of the decision, output or *task class* of a given pretrained neural network into high-level humanly meaningful *concepts* presented in the input (post-hoc analysis) have been an active area of research in the past few years. This process is done by inspecting the internal representations or *activations* of the neural network. This approach may be called *concept-based explanation* of neural networks.

The goal of this analysis is to describe the internal state of a neural network in a simple manner. Since in most cases the decision of the network is comparably much more complex than what can be easily defined in terms of input space, the concepts become an important factor. A concept can be extracted much easier than the higher-level information such as task classification. By comparing these concepts with the higher-level decision of the neural network, we can show how the neural network operates at a high level. On the other hand, simpler techniques might be used to show the trustworthiness of these simpler concepts. For instance, checking the colour of a uniform concept with simpler techniques is much easier than a job title.

The training phase of the original network and the explanation phase can be completely separate with different datasets (one for task classes and one for concept classes) and done by different people. For instance, for predicting the job title of a person from their image, the goal is to determine whether having a clinic as an image’s background affects the prediction of the job title to be a doctor. Or how the presence of a stethoscope around the neck changes the prediction of the neural network. These methods do not require concept labels and task labels to be from the same set of samples. For example, the task of predicting if the job title of a person is doctor can be represented by a set of physician images. But, the concept clinic may be represented by a set of clinic images and the concept stethoscope may be represented by a set of stethoscope images.

1.3 Previous Works Shortcomings

There have been several previous works on quantitatively measuring the importance of a concept at different layers of a neural network [4, 15, 19, 55].

Most of these methods rely highly on simplification of the problem and many assumptions. Early works tried to visualize what phenomenon in the input maximizes a certain neuron in a layer by generating the pattern that increases a neuron’s output or subsequently, a classification [4, 11, 29, 30]. While these methods might be useful in some situations, due to the fact that the samples are generated, their samples lack a higher input structure.

Other methods try to learn a certain concept by inspecting all neurons of a layer [19, 55]. The goal is to extract a specific concept from activations of a layer, so some extraction structure must be assumed to extract the concept from the layer outputs (concept classifier). Most previous works have had problems with defining such structures, and have just stuck to the simplest design (like linear combinations

of all layer’s neurons). This choice adds unreliable assumptions to the explanations.

Another problem that arises is how to measure the importance and relevance of a concept. Most methods involve very restrictive assumptions and resort to theoretical methods. Some methods assume that the space of activations is linear, and use dot product or directional derivative as a similarity measure [19] or in an extended version linear decomposition for mapping a vector into a concept basis [55]. As we show in our experiments, there are cases that the linear assumption doesn’t hold. In the following sections, we discuss this assumption in more detail.

1.3.1 Nonlinear Concepts

Most concept-based methods often assume that a concept, if present in an activation space, should be linearly separable from non-concept samples [4, 19, 15, 55]. This assumption, however, does not necessarily hold, especially in the earlier layers of a network where the learned features are often not abstract enough to linearly separate concepts [21] or in later layers when they fuse to form higher-level concepts. This hinders these methods’ ability in tracking the presence of a concept throughout the network. Another limitation comes from the assumption that the gradient of a section of a network with respect to the input is a good representation of that section [19, 55]. Such a first-order approximation might be misleading. This issue has been extensively discussed for saliency maps—which are also based on gradients approximations—and have been proven misleading [1, 20].

In our method, we check the presence of a concept in a layer’s activations by training a concept classifier—a network with the same structure as the task classifier from that layer onward but trained to detect the concept. The accuracy of concept classification gives us a good understanding of the importance and possible influence of the concept on a task class. For instance, the colour of shoes might not be relevant in the job classification task and such information is likely to be discarded by the network

after the first few layers. If in a particular layer, a concept cannot be detected by the concept classifier, it is safe to say that the network cannot recall the concept –i.e. the concept is forgotten (not necessarily universally but to the capacity and power of the given network). Such a conclusion can be made only if the concept classifier shares the same structure as the task classifier since the network structure is the upper limit for extraction power of the network. Moreover, the concept classifier is initialized by the weights of the network under inspection. This initialization will reduce the number of concept samples required for training the concept classifier. This particular choice of the concept classifier’s structure allows us to track the concept information across the original network’s layers.

1.3.2 Causality

We aim to capture and quantify the *causal relationship* between a concept and prediction of a class. For example, we want to quantify the extent that the presence of concept fever (or its absence) is a necessary or sufficient condition for the prediction of class flu. Another shortcoming of the previous methods is that most methods yield a score that captures the correlation between concepts and output and cannot give any further details about the nature of such a relation [4, 19, 55]. Following the above example about job title classification from images, the correlation between wearing a lab coat and being classified as a doctor cannot answer questions like do all images classified as doctor include lab coats or all people wearing lab coats are classified as doctors. This problem is sometimes referred to as causality confusion. Note that the goal is not to investigate causal relationships in the training dataset. We aim to investigate the causal relations “learned” by a neural network.

In a medical diagnosis setting, existing methods have difficulty answering questions like, do all patients that are classified as having flu have fever symptoms (fever necessary for predicting flu)? Are all patients with fever classified as having flu (fever

sufficient for predicting flu)? Are all patients classified as flu do not have a fever (absence of fever necessary for predicting flu)? Are all patients with fever not classified as flu (fever sufficient for negative of predicting flu)?

Based on the trained concept classifier and the existing network for task classification, we evaluate whether a concept is necessary, sufficient, or irrelevant for a specific task class. To avoid unnecessary assumptions like linear assumption or first-order approximation, we use a distribution sample set –i.e. a set of samples representing the distribution of data manifold. This set is a representative of the likely inputs of the network. Then we directly measure four relationship scores based on concept and target values of the distribution sample set. The four measures will be extracted in terms of causal expressions, showing whether a concept causes a task class or vice-versa. Unlike the previous works in [4, 55], which are limited to specific network structures like convolutional layers, the proposed method can be applied to a wide range of network structures.

1.4 Contributions

Contributions of this work are as follows:

1. We propose a framework to capture the existence of a given concept in a layer of a neural network without the linear assumption or first-order approximation.
2. We also propose a set of scores to capture the nature of the relationship between the concept and the network decisions in the form of causal expressions. In other words, we determine which one logically follows the other.
3. We then show practical applications of our method based on several experiments. Some of our experiments is designed to investigate linear and first-order

approximation assumptions (individually to isolate them from other factors).

The results have shown that these assumptions are not reliable.

4. Through experiments, we also compare our method with two existing methods, namely TCAV [19] and IBD [55], in determining relationships of concepts and tasks. The results show that our method succeeds in cases previous methods fail. Moreover, we demonstrate that our method outperforms the IBD method in explaining an image classification of a real-world dataset [54].

1.5 Organization of Thesis

The rest of this thesis is organized as follows. In chapter 2 we go over the basic backgrounds needed for a better understanding of this thesis including logical expressions and neural network structures. In chapter 3, we review what has been done in this area of research in more detail. In chapter 4, we present the proposed method and discuss its advantages compared to previous methods. In chapter 5, we demonstrate our claims through several experiments. Some of the experiments are designed to examine a certain assumption while others are a comparison of existing methods with the proposed approach. In chapter 6 we conclude and discuss some directions for possible future work.

Chapter 2

Background

In this chapter, we go over the background information needed for a better understanding of this work. In section 2.1 we cover the logical expression and how they can be considered as causality. In sections 2.2 and 2.3 we review the design of neural networks and their architectures. In sections 2.4, 2.5 and 2.7 we go over the process of training neural networks, their accuracy measures and the overfitting problem. Finally, in section 2.8 we go over some of the commonly used neural network structures used in the thesis.

2.1 Logic and Causal Expressions

Logical expressions are usually expressed as a causality clause in the mathematical notation form of $A \Rightarrow B$. In this notation, phenomenon A is the reason for the phenomenon B and whenever A happens, B will follow. To understand the clause, both A and B should be understood. Meaning that both condition and consequence should be familiar for humans so that the clause can be understandable.

In fundamental math, concepts are represented by sets. We use the same representation to visualize the relation between concept and task in a neural network. Being in a set means that the corresponding feature is present, and not being in the

Figure 1: (a) There are four different subsets that might potentially be empty. Depending on which subset is empty, one of the four conditions will happen. (b) 3 relative positions of two sets. With respect to (T) Task class, (S) Sufficient condition, (N) Necessary condition, and (I) Negative necessary.

set means the feature is not present. Two arbitrary sets are usually demonstrated by a Venn diagram (as seen in Figure 1a).

There can be several possible relations between the set C and the set T . Each of these relations can also be represented as a causality clause.

1. $C \subseteq T$ meaning C is a sufficient condition for T ($C \Rightarrow T$).
2. $C \cap T = \emptyset$ meaning C and T are inconsistent ($C \Rightarrow \neg T$ or $T \Rightarrow \neg C$). We call C a negative necessary condition for T .
3. $T \subseteq C$ meaning C is a necessary condition for T ($T \Rightarrow C$).
4. $C \cup T = M$ meaning either C or T or both should happen ($\neg C \Rightarrow T$ or $\neg T \Rightarrow C$).

We call C a negative sufficient condition for T (M is a set that contains all the elements).

Figure 1b shows a visualization of some of these conditions.

2.2 Neural Networks

Inspired by the natural neural networks, the basic building blocks of the human brain, Artificial Neural Networks was introduced to mimic the same behaviour in a loose manner [26].

2.2.1 Neurons

To mimic the actions of a neural network, we have to start from the smallest building blocks or the neural cells. Neural cells in the brain consist of three main sections, Dendrites, Axon and Soma.

Dendrites have the responsibility to collect the information from other neurons. As we will discuss later, this information is passed as the frequency that those cells are fired. Soma is like a summation and converts all of the information gathered to a single activation which is passed to the axon. Axon creates a pulse of activation (fires) whenever the signal in the soma reaches a certain level (firing threshold) and reduces the value accumulated to a non-firing value. This value is reduced over time as well if activation does not reach the firing threshold.

The measure of activation of a neuron is measured by the rate or the frequency at which the axon fires. The connections between dendrites of the cell and soma of other cells have special adjustable parameters that change over time to adapt.

Based on the description of neurons provided we can model a neuron by the model provided in Figure 2. In a more formal way the artificial neuron or unit is simulated as a weighted sum of its inputs from other neurons with an activation function. the adjustability of soma-dendrites connections are modeled by weights w_i and a bias term w_0 , and the function of Axon is considered as a nonlinear function $f(.)$.

$$o = f\left(\sum_i w_i x_i\right) \quad (1)$$

Figure 2: Visual structure of an artificial neuron.

Activation Function

The choice of activation functions $f(\cdot)$ has been a question throughout the history of the development of neural networks. In some older works, sigmoid is used since it is close to the real modelling.

$$f(x) = \frac{1}{1 + e^{-x}} \quad (2)$$

In more recent works, Rectified Linear Unit (relu) is more common due to simpler gradient calculations and faster training.

$$f(x) = \max(0, x) \quad (3)$$

2.3 Structures of Neural Networks

Artificial Neural Networks or Neural Networks are classified by their structure of connectivity into two main groups, first Recurrent Neural Network, and second Feed Forward Neural Networks.

2.3.1 Recurrent Neural Network

In Recurrent Neural Networks (RNNs), there is no limitation of connectivity. Neurons outputs can be connected to neurons that with one or more steps will connect to the input of the same neuron (like a feedback loop). Of course, a delay is assumed

between calculations. This type of neural network is more suitable for applications with sequences and causal behaviours.

2.3.2 Feed-Forward Neural Network

In contrast to recurrent neural networks, Feed-Forward Networks, each neuron's output is only connected to neurons that are not involved in the calculation of the input of the mentioned neuron. In such a structure, neurons can sometimes be divided into layers that each layer output is only connected to the neurons of the next layers. On top of that, there are sets of neurons that all the layers after are calculated only based on their output.

Fully Connected Layers

A layer that is connected to all neurons of the previous layer is sometimes called a fully connected layer or dense layer. This type of layer has a lot of expression power. Though it is a very expressive representation, learning the weights of a fully connected layer needs more data than the counterpart layers.

Convolutional Layers

In convolutional layers, each layer is connected only to a small portion of the previous layers' neurons (this size is sometimes referred to as kernel size) [12]. In these layers, each layer has one or more continuous dimensions, which is in most cases corresponding with the input space. For instance, for images, 2-dimensional convolutional layers are used while for timed input sequences, 1-dimensional convolutional layers are usually more appropriate. It is assumed that anywhere in the continuous dimension the extraction of the information is possible in the same manner.

On top of that, the weights of the connections are shared between different neurons

Figure 3: How each calculation in a convolutional layer works. The weights for the red calculation is the same as the green calculation.

in such a way that the weight is the same for all the neurons that have a specific relative position in the continuous dimensions of inputs. Figure 3 shows an example of how calculations are done in a 2-dimensional convolutional layer. In the Figure, the weights 3×3 weights for the calculation of red and green neurons on the output are the same. Each output is just calculated by using the same set of weights on different locations of the input space.

If we need the output space to have the same number of neurons in continuous dimensions as the input space, we can add zeros to the edges of the input and treat them as regular neurons. The number of these zeros added to each edge is called padding.

Pooling Layers

Since in convolutional layers the space of input and output is almost the same size in continuous dimensions, for lowering the number of computations throughout the network, pooling layers are used. Pooling layers apply a simple operation like maximum or averaging over windows of input. For removing the redundancy only a portion of the calculation outputs is kept, one every couple of units. How many of the outputs are skipped over is sometimes called strip, a stride of two would mean that one neuron is kept in every two outputs. Figure 4 shows how the units are used in a pooling layer of stride two. Each of the output neurons is calculated based on a 2×2 window of inputs. This size is also sometimes referred to as the kernel size of the pooling layer.

Figure 4: How each calculation in a pooling layer with stride 2 works. Each neuron depends on 4 neurons of the previous layer as shown.

Softmax Layers

In many applications, the resulting classes are mutually exclusive, in those cases the output of the classifier is best expressed as a probability distribution over classes. If the output of the neural network is treated as a probability distribution, there is a need that all the values be positive and sum up to one. These properties are achieved by a softmax layer. In a softmax layer natural exponent of all parameters is calculated and they are averaged to sum up to one.

$$f(\vec{x})_i = \frac{e^{x_i}}{\sum_j e^{x_j}} \quad (4)$$

2.4 Training Neural Networks

Training neural networks is usually defined as an optimization problem over a set of inputs and outputs. The training process tries to maximize the similarity or minimize the difference between the output of the network and the desired output given the input.

Considering the weights of all neurons in the network, we want to find a vector of values that optimizes a certain cost function. As we know, the network can be expressed as a multi-variable function $n(\cdot)$ with weights and input as its input.

$$o = n(\vec{x}, \vec{w}) \quad (5)$$

Figure 5: Gradient Descend in a two dimensional space. The initial condition has a role in the quality of the local minimal point found.

Then based on some cost function $c(\cdot)$ we aim to find a vector of weights that optimizes

$$\min_{\vec{w}} \sum_{\vec{x}, \vec{y} \in T} c(n(\vec{x}, \vec{w}), \vec{y}) \quad (6)$$

Since the behaviour of the function being minimized is continuous in terms of optimizing parameters $(\vec{w})$, optimization can be done by continuous optimization methods. The basis of most of these methods is the gradient descend approach. In gradient descend, it is assumed that small steps in the reverse direction of the gradient of a function will decrease it gradually and if repeated the function can be minimized over enough steps. Figure 5 shows an example of how gradient descent works in a two-dimensional optimizing a function. It also demonstrates the importance of the initial condition for the type of optimal point found. Gradient descent guarantees to converge to a local optimum but there is no guarantee to reach the global optimum and the optimum point reached depends on the initial condition.

Backpropagation

Since the calculation of the gradient of the cost function with respect to the weights is not straightforward, a method is needed to organize the calculations. Backpropagation [35] is such a method to organize and rewrite the derivation chain rule.

$$\frac{\partial e}{\partial w_j} = \frac{\partial e}{\partial o} \frac{\partial o}{\partial w_j} \quad (7)$$

now if we assume $o = f(\sum_i w_i x_i)$,

$$\frac{\partial o}{\partial w_j} = f'(\sum_i w_i x_i) x_j \quad (8)$$

But if the weight w_j is in previous layers we have to continue the derivation:

$$\frac{\partial o}{\partial w_j} = f'(\sum_i w_i x_i) \sum_i w_i \frac{\partial x_i}{\partial w_j} \quad (9)$$

The term $\frac{\partial x_i}{\partial w_j}$ can be recursively calculated in the same manner.

Gradient Descent

After the calculation of the gradient of cost function, the value is expected to go down if we take small steps in the reverse direction of it.

$$\vec{w}_{i+1} = \vec{w}_i - \delta \nabla_w(e) \quad (10)$$

The length of step δ can be a fixed value or optimized in each step for the best decrease of the cost function.

2.4.1 Cost Functions

Many functions have been used as cost functions for measuring the deviating from the desired output of a neural network. in many cases, Mean Square Error (MSE) is

a good measure. It's the average squared difference between the desired output and the current output.

$$e = \frac{1}{n} \sum_{i=1}^n (o_i - y_i)^2 \quad (11)$$

In other cases, Cross-Entropy Loss is sometimes used to characterize a classification problem. Cross-entropy loss is a measure of the distance between probability distributions, and if we assume that the outputs are probability distribution (usually after softmax or sigmoid for single output), this metric is used.

$$e = - \sum o_i \log y_i \quad (12)$$

2.5 Accuracy Measurements

There are many acceptable measures to capture the correctness of a prediction system [43]. One of the most important ones is the confusion matrix.

2.5.1 Confusion Matrix

Confusion Matrix is calculated by checking how true samples of a class are classified. Each column of a confusion matrix shows the distribution of prediction of the classifier for the samples of a true class.

There are two types of errors that a classifier can make, type I error or false alarm is when the classifier claims some sample to be of this class while in fact, it is not (False Positive). Type II error or miss detection is when the classifier classifies the sample as negative while it is in fact of this class (False Negative). The following table shows a simple confusion matrix for two classes:

		True labels		
		Positive	Negative	
Prediction	Positive	TP	FP	P
	Negative	FN	TN	N
Total		TP+FN	FP+TN	#samples

2.5.2 Accuracy

Accuracy is the most commonly used measure of correctness of a classifier. It is defined as how much of the samples are classified correctly (regardless of their true label or prediction). In terms of the confusion matrix accuracy can be calculated by the following equation:

$$Acc = \frac{TP + TN}{TP + TN + FP + FN} \quad (13)$$

2.5.3 Precision and Recall

In many cases, the False Negative error and miss detection of the classifier are not as important as each other. For this purpose several measures are defined to quantify the importance of these two mistakes: True positive rate (TPR) or Recall or Sensitivity is defined as

$$Rec = \frac{TP}{TP + FN} \quad (14)$$

Positive predictive value (PPV), or Precision is defined as:

$$Prec = \frac{TP}{TP + FP} \quad (15)$$

Specificity (SPC) or Selectivity or True negative rate (TNR) is also defined as

$$spc = \frac{TN}{TN + FP} \quad (16)$$

and have a close connection to accuracy and recall. False Positive Rate (FPR) is calculated by subtracting one from Specificity.

2.5.4 F1 Score

F1 score is usually used to balance the importance of Precision and Recall in a problem. F1 score is defined in terms of precision and recall as the following:

$$F1 = 2 \cdot \frac{Prec \cdot Rec}{Prec + Rec} \quad (17)$$

2.5.5 ROC Curver

The Receiver Operating chart or ROC curve is created by plotting the Recall against the false positive rate. Since in harder problems minimizing both error types at the same time is not easy, the ROC curve tries to visualize the balance between these two errors. The performance of each classifier is a point on the ROC curve. Classifiers that provide soft-output (or probabilities) will be converted to hard outputs by adding a threshold, depending on the threshold the error types are calculated. By connecting all the points achieved by a certain soft classifier its ROC curve is plotted, which characterizes the Recall and FPR that can be achieved at the same time.

The best point in the ROC curve is the left top corner, which means there is no false negative (recall of one) and false positives (FPR of zero). The importance of ROC is more obvious in applications that the error types are not as equally important. So the ROC can show what is the recall of the classifier when we want the classifier to have an FPR of a specific small value or what is the FPR when we want a recall of almost one.

Figure 6: ROC curve. While the classifier of the green curve is better than both blue and purple, the blue and purple classifiers are not comparable. Blue is better for application that recall is more important while purple is better for applications that FPR is important.

2.6 Split of Samples

Since the process of training is defined as an optimization problem based on samples, it is not reasonable to report the final optimized value as a function of the same samples. Moreover, it is observed if the training is done using another set of samples, the cost on other samples is not monotonically decreasing. We will talk about this phenomenon later. Because of these reasons, in the training of neural networks, usually samples are split into two or three sets, training, evaluation and test sets.

The training samples are the samples used in the process of training the neural network. the evaluation samples are the samples that are used to evaluate the training accuracy during the training process, but the cost function is not directly optimized on them. They are mostly used as quality control of the training. Since they are also used in the process of training, it's a good practice to not fully trust the accuracy achieve on this set. The test samples are the samples that are not involved in training or the design in any way. The best practice is that they are not even available to the designer of the neural network. The accuracy measures in such circumstances are the most reliable quality measure of the final neural network.

Figure 7: Overfitting in training.

2.7 Overfitting and Prevention

Overfitting is a problem that often arises due to the expressiveness of the network. After the split of data into training and evaluation, it seems reasonable to expect the same accuracy on the evaluation samples as training samples, but that is not the case. In many cases, the network tries to improve the accuracy based on training examples so much that it no longer makes sense. For instance, if there were a small change in the colour of different classes, the network will try and extract it and correlate it with classes to improve the accuracy. Because of such behaviours, the model who is trying to learn the function will be more fit to the training points rather than all possible points. This behaviour usually appears after extraction of all relevant information, figure 7 shows how the error function for evaluation data increases after some iterations while the training error function is monotonically decreasing.

For stopping such behaviour several methods are used in the structure of the neural network so that it cannot extract very detailed information.

2.7.1 Early Stopping

The easiest way to stop overfitting is to observe the behaviour of the evaluation data error function and make sure at the end of the training we keep the weights that minimize this error function. We also can stop training if we feel the error function

is not decreasing, hence the name.

2.7.2 Batch Normalization

Batch normalization is the method of using the batches to add a variation to training so the network cannot extract too many irrelevant details. In each batch of training, the values of the network activations for the layer are normalized so the average of activations of all samples of a batch have a fixed value.

2.7.3 Dropout

Another idea is to omit neurons randomly in some layers to prevent information to pass [45]. For compensation of the omitted neuron's effect, other neurons' values are scaled up.

2.8 Commonly Used Architectures

Many famous architectures have been successful in different image classification problems. Most methods developed are inspired by these methods or exactly use the same concepts. Here we cover four of these architectures.

2.8.1 Alexnet

Alexnet network structure first proposed in [22] is one of the first deep structures to get a good accuracy on Imagenet challenge. the network consists of five convolutions layers and three dense layers, the Table 1 shows the parameters of this structure.

Layer No	Layer Type	Input Size	Output Size	Kernel Size	Stide	Padding
1	conv1	$227 \times 227 \times 3$	$55 \times 55 \times 96$	11×11	4	0
2	maxpool1	$55 \times 55 \times 96$	$27 \times 27 \times 96$	3×3	2	0
3	conv2	$27 \times 27 \times 96$	$27 \times 27 \times 256$	5×5	1	2
4	maxpool2	$27 \times 27 \times 256$	$13 \times 13 \times 256$	3×3	2	0
5	conv3	$13 \times 13 \times 256$	$13 \times 13 \times 384$	3×3	1	1
6	conv4	$13 \times 13 \times 384$	$13 \times 13 \times 384$	3×3	1	1
7	conv5	$13 \times 13 \times 384$	$13 \times 13 \times 256$	3×3	1	1
8	maxpool3	$13 \times 13 \times 256$	$6 \times 6 \times 256$	3×3	2	0
9	fc1	9216	4096			
10	fc2	4096	4096			
11	fc3	4096	1000			

Table 1: Structure of Alexnet neural network.

2.8.2 VGG16

VGG16 was first introduced in [41]. In this design for the first time padding is used in a way that the size of the output of convolutional layers is the same as their input, so more layers can be stacked on top of each other. In total fifteen convolutional layers and three dense layers are joined together to create VGG16 model (see Table 2).

2.8.3 Resnet18

In another design, the problem of gradient vanishing is dealt with. For solving the gradient vanishing problem, a skip connection is introduced to make the gradient reach earlier layers easier [16]. Every two convolutional layers (starting with convolutional second layer) are linked together with a skip link (like Figure 8).

With this design, the speed of training is much faster than in previous designs (the weights converge to the desired value much faster). This structure is summarized in

Layer No	Layer Type	Input Size	Output Size	Kernel Size	Stide	Padding
1	conv1	$224 \times 224 \times 3$	$224 \times 224 \times 64$	3×3	1	1
2	conv2	$224 \times 224 \times 64$	$224 \times 224 \times 64$	3×3	1	1
3	maxpool1	$224 \times 224 \times 64$	$112 \times 112 \times 64$	2×2	2	0
4	conv3	$112 \times 112 \times 64$	$112 \times 112 \times 128$	3×3	1	1
5	conv4	$112 \times 112 \times 128$	$112 \times 112 \times 128$	3×3	1	1
6	maxpool2	$112 \times 112 \times 128$	$56 \times 56 \times 128$	2×2	2	0
7	conv5	$56 \times 56 \times 128$	$56 \times 56 \times 256$	3×3	1	1
8	conv6	$56 \times 56 \times 256$	$56 \times 56 \times 256$	3×3	1	1
9	conv7	$56 \times 56 \times 256$	$56 \times 56 \times 256$	3×3	1	1
10	maxpool3	$56 \times 56 \times 256$	$28 \times 28 \times 256$	2×2	2	0
11	conv8	$28 \times 28 \times 256$	$28 \times 28 \times 512$	3×3	1	1
12	conv9	$28 \times 28 \times 512$	$28 \times 28 \times 512$	3×3	1	1
13	conv10	$28 \times 28 \times 512$	$28 \times 28 \times 512$	3×3	1	1
14	maxpool4	$28 \times 28 \times 512$	$14 \times 14 \times 512$	2×2	2	0
15	conv11	$14 \times 14 \times 512$	$14 \times 14 \times 512$	3×3	1	1
16	conv12	$14 \times 14 \times 512$	$14 \times 14 \times 512$	3×3	1	1
17	conv13	$14 \times 14 \times 512$	$14 \times 14 \times 512$	3×3	1	1
18	maxpool5	$14 \times 14 \times 512$	$7 \times 7 \times 512$	2×2	2	0
19	fc1	25088	4096			
20	fc2	4096	4096			
21	fc3	4096	1000			

Table 2: Structure of VGG16 neural network.

Figure 8: Structure of residual block used in Resnet.

Table 3.

2.8.4 InceptionV3

Another design that achieved high accuracy in the Imagenet challenge is the inception designs [48]. These designs are more complex than previous ones to remove the redundant weight and to make the training process easier and more robust. So after a few couples of initial layers (as shown in Table 4), the layers are replaced by mixed layers that have fewer parameters but more complex internal designs.

The designs include multiple branches from input to output with different expressiveness powers (since not all the channels need the full expressiveness a big convolutional layer). One of the sub-modules is described in the Tables 5. These sub-modules are designed to reduce the number of parameters in the network, as it is obvious, concatenating two convolutional layers with kernels of 3×1 and 1×3 have fewer parameters than a single 3×3 kernel layer. This process is referred to as factorization and the number of parameters is reduced more when the kernel sizes are bigger.

Though factorization reduces the number of parameters, the expressiveness of the module created is limited, so multiple different layers are used in parallel to cope with the changes made and the results are concatenated in the channel space at the last layer. Of course, more factorization and parallelism can be utilized when the number of parameters is higher as block-e are.

Layer No	Layer Type	Input Size	Output Size	Kernel Size	Stide	Padding
1	conv1	$227 \times 227 \times 3$	$112 \times 112 \times 64$	7×7	2	1
2	maxpool1	$112 \times 112 \times 64$	$56 \times 56 \times 128$	3×3	2	0.5
3	conv2	$56 \times 56 \times 64$	$56 \times 56 \times 64$	3×3	1	1
4	conv3	$56 \times 56 \times 64$	$56 \times 56 \times 64$	3×3	1	1
5	conv4	$56 \times 56 \times 64$	$56 \times 56 \times 64$	3×3	1	1
6	conv5	$56 \times 56 \times 64$	$56 \times 56 \times 64$	3×3	1	1
7	conv6	$56 \times 56 \times 64$	$28 \times 28 \times 128$	3×3	2	0.5
8	conv7	$28 \times 28 \times 128$	$28 \times 28 \times 128$	3×3	1	1
9	conv8	$28 \times 28 \times 128$	$28 \times 28 \times 128$	3×3	1	1
10	conv9	$28 \times 28 \times 128$	$28 \times 28 \times 128$	3×3	1	1
11	conv10	$28 \times 28 \times 128$	$14 \times 14 \times 256$	3×3	2	0.5
12	conv11	$14 \times 14 \times 256$	$14 \times 14 \times 256$	3×3	1	1
13	conv12	$14 \times 14 \times 256$	$14 \times 14 \times 256$	3×3	1	1
14	conv13	$14 \times 14 \times 256$	$14 \times 14 \times 256$	3×3	1	1
15	conv14	$14 \times 14 \times 256$	$7 \times 7 \times 512$	3×3	2	0.5
16	conv15	$7 \times 7 \times 512$	$7 \times 7 \times 512$	3×3	1	1
17	conv16	$7 \times 7 \times 512$	$7 \times 7 \times 512$	3×3	1	1
18	conv17	$7 \times 7 \times 512$	$7 \times 7 \times 512$	3×3	1	1
19	avgpool1	$7 \times 7 \times 512$	$1 \times 1 \times 512$	7×7	7	0
21	fc1	512	1000			

Table 3: Structure of Resnet18 neural network.

Layer No	Layer Type	Input Size	Output Size	Kernel Size	Stide	Padding
1	conv1	$299 \times 299 \times 3$	$149 \times 149 \times 32$	3×3	2	0
2	conv2	$149 \times 149 \times 32$	$147 \times 147 \times 32$	3×3	1	0
3	conv3	$147 \times 147 \times 32$	$147 \times 147 \times 64$	3×3	1	1
4	maxpool1	$147 \times 147 \times 64$	$73 \times 73 \times 64$	3×3	1	0
5	conv4	$73 \times 73 \times 64$	$73 \times 73 \times 80$	1×1	1	0
6	conv5	$73 \times 73 \times 80$	$71 \times 71 \times 192$	3×3	1	0
7	maxpool2	$71 \times 71 \times 192$	$35 \times 35 \times 192$	3×3	2	0
8	mixed-a1	$35 \times 35 \times 192$	$35 \times 35 \times 256$			
9	mixed-a2	$35 \times 35 \times 256$	$35 \times 35 \times 288$			
10	mixed-a3	$35 \times 35 \times 288$	$35 \times 35 \times 288$			
11	mixed-b1	$35 \times 35 \times 288$	$17 \times 17 \times 768$			
12	mixed-c1	$17 \times 17 \times 768$	$17 \times 17 \times 768$			
13	mixed-c2	$17 \times 17 \times 768$	$17 \times 17 \times 768$			
14	mixed-c3	$17 \times 17 \times 768$	$17 \times 17 \times 768$			
15	mixed-c4	$17 \times 17 \times 768$	$17 \times 17 \times 768$			
17	mixed-d1	$17 \times 17 \times 768$	$8 \times 8 \times 1280$			
18	mixed-e1	$8 \times 8 \times 1280$	$8 \times 8 \times 2048$			
19	mixed-e2	$8 \times 8 \times 2048$	$8 \times 8 \times 2048$			
20	avgpool	$8 \times 8 \times 2048$	$1 \times 1 \times 2048$			
21	fc1	2048	1000			

Table 4: Structure of InceptionV3 neural network.

Layer No	Layer Type	Input Size	Output Size	Input From	Kernel Size	Stide	Padding
1	conv	$35 \times 35 \times n$	$35 \times 35 \times 64$	input	1×1	1	0
2	conv	$35 \times 35 \times n$	$35 \times 35 \times 48$	input	1×1	1	0
3	conv	$35 \times 35 \times 48$	$35 \times 35 \times 64$	2	5×5	1	2
3	conv	$35 \times 35 \times n$	$35 \times 35 \times 64$	input	1×1	1	0
4	conv	$35 \times 35 \times 64$	$35 \times 35 \times 96$	3	3×3	1	1
5	conv	$35 \times 35 \times 96$	$35 \times 35 \times 96$	4	3×3	1	1
6	avgpool	$35 \times 35 \times n$	$35 \times 35 \times n$	input	3×3	1	1
7	conv	$35 \times 35 \times n$	$35 \times 35 \times m$	6	1×1	1	0
8	concat		$35 \times 35 \times (m+224)$	1,3,5,7			

Table 5: mixed-a blocks. The parameter n is the number of input channels and the parameter m depends on which inceptionA block is being processes.

Chapter 3

Literature Review

Interpretability is defined as “the degree to which a human can understand the cause of a decision” [27] or “the degree to which a human can consistently predict the model’s result” [18].

Many methods have been proposed in recent years to deal with the problem of interpretability but there is not a general categorization of them. Only some general categories have been discussed about how each method approaches the problem and how they seek or construct explanations. Several articles have tried to study the problem of interpretability in a general way [8] and some surveys [5, 28] have been gathered about existing methods.

3.1 Intrinsic Interpretability

Interpretability approaches in terms of how they try to explain the machine learning model are divided into two groups. Intrinsic methods try to simplify the machine learning model to make it understandable (or interpretable) to humans (as is). They either make the machine learning inherently interpretable or convert the method to an interpretable representation. Linear models, decision trees, rules sets are among machine learning methods, considered intrinsically interpretable. For instance, the

anchors method [33] tries to simplify the model by locally approximating it by an IF-THEN rule.

Other methods try to make the model more interpretable without approximating it with simple models. For example, neural network-based concept bottleneck methods [21] insert a concept space between the task classifier and the input.

Post Hoc methods try to explain an already trained model without changing the model. These methods are called post hoc methods, as the problem of interpretability is dealt with after model training.

3.2 Local Interpretability

Interpretability of machine learning methods in terms of what they explain is divided into two main branches. Global explanations try to explain the whole behaviour of the machine learning model. Local explanations provide an explanation for individual predictions. For example, methods based on saliency maps like Grad-cam [36], examine the sensitivity of individual predictions to changes of input pixels.

3.3 Model-agnostic Interpretability

Another categorization of the interpretability methods is based on whether the method depends on the internal states of the machine learning method. If it is dependent on the internal states, it can no longer be used for models that lack that specific state. For example, TCAV [19] and IBD [55] are specific to neural networks. Or [38] tries to visualize the activation differences in a neural network (not applicable to models that do not have activations). Such interpretability methods are called model-specific interpretability, while the approaches that do not depend on the internal states are called model agnostic methods. For instance, Interpretable

Model-agnostic Explanations (LIME) [32] explains the model in the local area of the sample linearly. Shapley-based methods [17, 25, 46] borrow some concepts from cooperative game theory to measure the importance of each factor towards the detection of a class are also model-agnostic methods. Anchors [33] that tries to give a local approximation of the model by rules, is also model-agnostic.

3.4 Difference or Gradient-based Methods

There have been many methods that have tried to explain the decision of machine learning methods by perturbation of the input space especially in cases that the machine learning model is a continuous function like neural networks. Some rely on differences [38], while many methods use gradient as an explanation of the model [3, 6, 9, 37, 39, 40, 42, 44, 47]. For explanation of neural networks, sometimes these methods are referred to as saliency maps which as we mentioned have been shown to be not reliable.

3.5 Example-Based Interpretability

Some of the methods use examples from the training set to justify their decision-making. These methods sometimes find representative examples to represent the important sections of the input space and to make methods interpretable. These samples are sometimes called prototypes. On the other hand, negative samples (samples of other classes) are used to show where the class decision boundary stops are called criticisms [18].

3.6 Post Hoc Concept-based Interpretability of Neural Networks

There have been several works on explaining the intermediate activations of a neural network based on human-friendly concepts. Most notably, Kim et al. [19] proposed a percentage measure, called TCAV score, to measure how much a concept interacts with the task classifier. TCAV works based on whether the gradient of the neural network is in the direction of the concept. The direction of the concept is defined as the direction orthogonal to the linear classification decision boundary between concept and non-concept samples. The TCAV score captures the correlation between the network output and the concept and lacks detailed information about the nature of the relationship. Moreover, it assumes that concepts can be represented linearly in the activations space, an assumption that does not necessarily hold [21]. They also represent a section of the network only by its gradient (first-order approximation), which might be misleading. A similar approach has also been explored in methods Net2Vec [11] and Network dissection [4] methods but they assume that the concepts are aligned with single neurons’ activation.

In another work, Interpretable Basis Decomposition for visual explanation (IBD) [55], the authors tried to explain the activations of a neural network by greedily decomposing it into some concept directions. They use the resulting decomposition as explanations for the image classification task. One of the drawbacks of such an approach is its linear assumption which comes from the usage of linear decomposition of the gradient in the activation space. Using greedy methods can also potentially result in inaccurate and unstable results. Another limitation of the IBD method [55] is that it can only explain convolutional layers and therefore for neural networks that include dense layers, they have to modify the network. In their experiments, they have replaced each dense layer with a global average pooling layer and a linear layer.

The linear assumption indicates that a concept in hidden layers corresponds to a vector and the representation of data in each layer is a vector space. Such methods assume that addition, subtraction, scalar product and inner product (as projecting an activation to a concept vector) operations in an activations space are always meaningful. The linear assumption is originated in feature visualization methods. Most feature visualization methods optimize for inputs that maximally activate certain neurons or directions. Early studies on neural network activation space tried to find samples that maximally activate a single neuron, and associate a concept to the neuron. In [49] the authors argued that random linear combinations of neurons may also correspond to interpretable meaningful concepts. The general idea of using a linear classifier to check the information of intermediate layers originated in [2]. They proposed to use linear probes – trainable linear classifiers independent of the network – to get an insight into the network representations. In contrast to what was mentioned in [49], in [4, 30] the authors reported that the basis (each neuron) direction activation is more often corresponding to a meaningful concept than just random vectors. Still feature visualization methods, ignore the distribution of the input data which results in inputs that are not consistent with real samples.

Linear interaction of concepts has been even less studied in feature visualization methods. In [30] the authors showed in some cases the addition of two concepts’ activations will result in inputs with both concepts present. But they cast doubt on whether this finding is always true. Linear assumption lacks enough evidence to be considered reliable for being the basis of interpretability methods that try to gain the trust of humans and justify neural network decision making.

Some other methods have tried to automatically discover new concepts from neural networks, namely Automatic Concept-based Explanations (ACE) [13] and Completeness-aware Concept-Based Explanations (CCE) [52], rather than taking a concept as input. ACE tries to automatically extract concepts based on TCAV while

CCE extracts concepts based on convolutional layers continuity. CCE also tries to avoid first-order approximation by measuring the importance of concepts using Shapley score. Though these methods can help with cases that no principle exists for rational behaviour of the network, in many cases, the experts have a good principle about the problem at hand and the principle’s concepts are predefined. So they want to check the consistency of the neural network with the existing principles. For instance, in the detection of a certain disease, the experts check all the related symptoms and are not interested in other concepts that the machine learning method might introduce. For example for the prediction of a patient having flu, medical experts know that fever is a symptom, and we want to know exactly what is the relation between the fever and being classified as having flu.

Our work relates to CACE [15] in that, both try to address the shortcoming of TCAV [19] by capturing causal expressions. The CACE method [15] measures the influence of concept by the difference of conditional expected values. This requires highly controllable datasets or very accurate generative models that may not be available in practice.

Our work relates to [51] in that both use a specific visual method to examine the influence of different input features on the output of a machine learning model. But our method goes further and inspects the nature of the relationship and quantifies these visualizations. We also consider high-level concepts instead of raw input features.

3.6.1 Summary

Since there are many similarities and assumptions between different methods, we put them into a table to make the comparison easier.

Table 6 compares the assumptions of different concept-based methods for explaining neural networks. This list is not exhaustive and only includes the most prominent

works for comparison reasons. The methods can be divided into two main groups, the ones that concept is defined as a dataset and the ones that try to automatically extract new concept definitions based on the neural network.

	method	linear	first order	covolutional layer	global
concept dataset	TCAV [19]	Y	Y	N	Y
	IBD [55]	Y	Y	Y	+
	Net dissection [4]	N	N	Y	+
concept discovery	ACE [13]	Y	Y	Y	Y
	CCE [53]	Y	N	Y	Y

Table 6: Comparison of concept-based methods. Y: Yes, N: No, +: global explanation but contribution of concept for each input can be calculated.

The first two columns show the assumptions of each method, linear assumption and first-order approximation, the Y value means that the method is using that assumption. The convolutional layer column shows the dependency of a method to the existence of convolutional layers, i.e. some methods cannot be used without convolutional layers. And the last column, global, shows whether a method gives global or local explanations for neural network decision, a value of + means that the method is global but the contribution of each concept can be calculated.

Datasets

In our review of the datasets for concept-based explanations of neural networks, we only include the image datasets that have attribute-level annotations. These datasets are summarized in the Table 7.

Annotation level column have different values, class denotes that each class has

annotations (not each image). Image denotes that each image has annotations but no other information is provided. Pixel level annotation means that each pixel is annotated to whether they have a certain concept or not. Segment annotation means that parts of the image are annotated to be a certain concept, this type of annotation might be in form of a polygon or a bounding box that includes the concept.

dataset	samples	classes	attributes	annotation level
AwA [23]	30475	50	85	class
Birds [50]	11788	200	112	image
aPascal-aYahoo [10]	15339	32	64	segment (bounding box)
SUN [31]	14340	707	102	segment (polygon)
Broden [4]	-	-	1376	pixel

Table 7: Different datasets that include image annotations.

Since not all the samples are annotated in Broden dataset for all groups of attributes, we gave a value of “-” in the table.

In the next chapter, we will propose a framework for a concept-based explanation for neural networks, which simultaneously addresses the linear assumption, first-order approximation and causality confusion issues discussed above.

Chapter 4

Proposed Method

4.1 Representing Concepts

In our method, we base the explanations on a certain layer’s activations and explain whether and how the concept interacts with a task class based on the activations of the layer. We break up the neural network into two sections, the section before the hidden layer (denoted by $f(x)$) and the section after the hidden layer (denoted by $g_w(z)$). w denoted the trainable parameters of the second section and the whole network can be expressed as $g_w(f(x))$ (see Figure 9c).

As shown in Figure 9, we only need two sets of samples for our analysis. (1) Concept set labelled on the concept information only (Figure 9a). (2) Distribution sample set, without any labelling (Figure 9b). Note that access to the original training task data is not required.

For the sake of explaining the proposed method, let us consider a neural network trained on colour-coded hand-written digits. In the training set, a unique colour was assigned to each class, and samples within each class were coloured accordingly. For instance, all 0’s in the training set are red, and all 1’s are blue. One would expect the network decision to be influenced by the colour as well as the digit itself; but the challenge is how to measure this influence. We aim to determine, how the concept,

Figure 9: (a) samples of positive and negative concepts, (b) distribution sample set, (c) representation of task class and concept in network, (d) analysis of the relationship between the task class and concept showing the concept is a necessary condition for the task class.

i.e. red, influences the decision-making of this neural network.

The first step in our analysis is to check whether the concept is present in the layer. In other words, can the classifier on activations of the layer achieve acceptable concept detection accuracy. We check if the second section of the network has adequate power and capacity to distinguish the concept set (colour red) in that layer.

For representing the concept, a set of positive and negative concept examples are used, in our case red samples against other colours (Figure 9a). Then the concept classifier with the structure of the second section of the neural network is trained ($g(z)$) to distinguish the concept from non-concept activations (z values). As a result, we will have two networks with identical structures but different parameters. w denotes the parameters of the original network trained for task classification (digit classification) and w' denotes the parameters learned to distinguish a concept from non-concept samples (red vs. other colours). $g_w(z)$ is the task classifier, whereas $g_{w'}(z)$ is the concept classifier. For learning $g_{w'}$, we initialize the trainable parameters of g as w . Note that the parameters of the first section ($f(x)$) do not change while w' is learned.

Other than showing the concept is present and extractable by the network, training

another network with the same structure gives us a way to generalize over concept samples. And since now we have generalizable representations of both task classes and concepts, we can proceed with our causal analysis.

Now we can check the possible relationship between this concept representation and the task class. Checking if a set is a subset of another, can be easily done by checking the definition. Since we cannot sample every possible instance in our input space, we only check the relationship on a distribution sample set (Figure 9b). Note that this sample set is chosen randomly and it is not specifically selected like prototyping methods. As we mentioned earlier, the distribution sample set, the samples used for causal analysis evaluation, does not have labels. The set T is a subset of set C if every sample in T is also in C , which is equivalent to C being a necessary condition for T or $T \Rightarrow C$. Checking the negation of this definition is much easier (just checking that no counter-example exists). For this purpose, a scatterplot is generated by evaluating the task classifier and concept classifier on the distribution sample set (each point in the scatterplot is a sample of this set). A counter-example, in this case, is a sample in C and not in T (e.g. a sampled classified as 0 and not red). So the clause correctness corresponds to the case where the top left corner of the evaluation graph is empty – equivalently no counter-examples found in our distribution sample set (Figure 9d). Note that the points of the scatter plots are only outputs of the task classifier ($g_w(z)$) and concept classifiers ($g_{w'}(z)$) based on the layer and are not necessarily close to true labels, this is a positive point since we want to measure the relationship based on network information and not the true labels.

Two observations support our choice for using the same structure for the detection of concepts. First, if the concept is present and the network is using it, the network has to extract information using its structure so the network structure should be able to detect it. Second, if the concept is not detectable by the existing structure there is no way of it being involved in the network decision. Of course, if the network is

not using the concept but it's present in the layer, the evaluation analysis will detect the concept not being involved in task class decision making.

Though concepts like colour can be easily learned by much simpler network structures, more complex concepts like the presence of objects (a stethoscope) might not be as simple to detect. Since this network is pre-trained (on task classification), it is less likely to overfit given the fact that all overfit prevention of the original network is at work. These mechanisms include all the batch normalization, drop out, etc layers in $g(z)$ and train, evaluation, test split of concept samples. Moreover, the structural coherence of the network is kept intact. In other words, the limitations, powers and local behaviour of the network (as initial parameters) are considered in the detection of concept, keeping the convolutional activations as convolution representations (with the spatial information preserved).

Of course, this evaluation is only done using the samples and does not necessarily mean that the real sets defined by the decision boundary of the concept and target classifier networks have such a relation. In fact, we are only interested in the samples that are on the manifold (see Figure 10). And decision boundary behaviour outside the data manifold distribution does not have any effect on the practical relationship and hence is not important to us. For example in Figure 10, the concept (C) is a sufficient condition for the task class (T) since all samples predicted as concept red are classified as target class.

4.2 Quantifying the Relationships

Since the concept classifier and task class are represented by soft decisions (outputs of the two networks), we propose a method to quantify the absence of counter-examples similar to the ROC curve (see Figure 11).

Consider the fact that the logical expression $C \Rightarrow T$ is equivalent to $C \vee \neg T$. For

Figure 10: The two sets might not have a relationship but on the manifold of data they might have a strong relationship.

the expression to be true either C or $\neg T$ has to be true. Assuming the threshold t for both expressions and calculating $\neg T$ by $1 - T$, we get the fact that the expression holds for any sample that is not in the F section.

For better handling of imbalanced classes we use the adapted F1 score instead of accuracy:

$$F1 = \frac{2PR}{P + R}. \quad (18)$$

where P and R are adapted precision and recall defined as:

$$P = \frac{TP}{TP + F}, \quad (19)$$

$$R = \frac{TN}{TN + F}. \quad (20)$$

TP, TN and F denote the number of samples in the corresponding part of Figure 11.

Based on the introduced parameters, for each threshold t , an F1 score can be calculated. The measure of the strength of a necessary relationship is then calculated as the area under the F1 versus threshold curve. Intuitively the strongest relationships in this measure, hold with stronger accuracy for smaller thresholds t .

For simplicity here, we assumed that the threshold for C and $\neg T$ are equal, but in general, these thresholds can be considered as threshold t_C for C and threshold t_T for T . In that case, the quantitative curve will be a 3d surf (the F1 score vs. t_C and

Figure 11: The process of creating quantification curve for necessary score. In this process we check if the task class is true with probability at least t then the concept is true with probability of at least $1 - t$. The highest value is achieved when there is no counter-example (F) samples.

t_T) and the volume under the curve should be used as the measure of the strength of the relationship.

The four possible relationships mentioned above, respectively, correspond to (1) bottom right, (2) top right, (3) top left, (4) bottom left corners of the Figure 9d being empty. The same numbering is indicated in the Figure 1a –i.e. relationship holds when they are empty. Each of these relationships can logically be converted to an OR ($\vee$) expression and evaluated in the same manner we evaluated the necessary relationship. A simpler way is to logically convert them to a necessary evaluation and quantified with the mentioned process. For instance C being sufficient for T is equivalent to T being necessary for C . The negation of concepts and tasks ($\neg C$) is calculated by just subtracting them from one, i.e. $(1 - C)$, so for negative necessary measurement, we do the same calculation with the negative of concept values.

For any of our experiments, we create four quantification curves based on the scatterplot. It helps quantify the relationship between the concept and task class. Each quantification curve shows the F1 score against different thresholds. We further use the area under curve (AUC) to summarize each curve into a real-valued score between 0 and 1. The area under the curve is a good estimate of how strong the

relationship is. For instance, an area under the curve close to one is a very strong relationship while an area of zero means there is no basis for that relationship. In the next chapter, we demonstrate the proposed methods over several experiments.

4.3 Discussions

4.3.1 Distribution Sample Set

The distribution sample set, the set that represents the distribution of likely inputs of the network plays an important role in our analysis as all measure evaluations are based on the samples of this set. Checking the network with the same set that it was trained on, will most likely confirm the network decisions. Most methods that predict the behaviour of the network need such sample sets, for instance, TCAV [19] need samples from the task class.

The distribution sample set represents likely cases of input and should be a good representation of the inputs that the network will be tested on. For instance, using the job title classification example, if we expect most but not all doctors to wear lab coats, it should be reflected in the distribution sample set. On the other hand, if we don't expect the network to be tested on images of doctors with stethoscopes in a garage fixing cars, distribution sample set should not contain such images. The fact that distribution sample set does not need any kind of labelling, enables us to use any set of inputs like a held-out part of data or even inputs recorded from other sources, as long as they are a good representation of likely task classification inputs.

4.3.2 Inspection Layer Choice

The choice of which layer to inspect is not a straightforward decision. Of course, the inspection of later layers is computationally cheaper since the training of the smaller

concept classifier is cheaper. But there is no guarantee that the concepts are still present in those layers since the network might have traded them with a combination of concepts more useful for the task classification. As we show in our experiments, sometimes higher level concepts replace lower level concepts. For instance presence of a stethoscope might not be detectable in the last layer, but the presence of a medical instrument might be possible. So distinguishing medical instruments from each other might not be easy for the network but distinguishing images that include a medical instrument from the ones that don't might be possible. For this reason, we start our analysis from the last layer in the network and work our way back till we reach a layer that the concept is present (classifiable with good accuracy) or reach the first layer (which will guarantee that the concept is too hard to be detected by the network).

If we reach a the input layer and we realize the whole network cannot distinguish the concept samples from non-concept samples, we realize the network does not have the capacity to extract the concept information. On the other hand, if a layer is found to have the concept present, then the causal analysis detects the relationship between the concept and the classification. Note that the concept might be present in a layer but not used, meaning there is no relation between classification and that concept. Specifically, in the input layer, for a expressive enough network, most concepts are detectable, but there is no guarantee that the network would use all possible concepts.

4.3.3 Trustworthiness of a Network

For measuring the trustworthiness of a neural network we examine the concepts proposed by the experts to determine the relation of the concept with the neural network behaviour. After the relation was quantized and determined the knowledge extracted is compared to experts points and the trustworthiness of the network is determined. For instance, a network is trained to distinguish between patients that have the flu, a cold and allergies. Experts propose that in most cases of flu patients have a fever,

and allergy patients usually don't have fever symptoms. So the network is examined using the concept fever. If the network reflects this knowledge: (1) fever is necessary for flu classification and (2) fever is negative necessary for allergy classification then the network is trustworthy (at least about the fever symptom).

Of course, if all layers are inspected and the concept suggested by the experts is not found we can conclude that the concept does not affect the output. If the experts believe the concept should be involved in the classification, then we can conclude that concerning this concept, the network is not trustworthy.

Chapter 5

Experiments and Results

In this section, we explore the application of the proposed method in the evaluation of the relationship of neural network task classes and concepts in a controlled setting and a real-world setting. The controlled settings explain neural networks with alexnet structure and the real-world setting uses a pretrained Resnet18. We compare our results with TCAV [19] and IBD [55] methods as they are the most related works to the proposed method.

We have constructed two controlled datasets to simulate the scenario where certain concepts have a positive or negative correlation with the task classes.

5.1 Coloured MNIST Dataset

MNIST is an image recognition benchmark dataset consisting of ten classes of hand-written digit images [24]. We modify this dataset to add useful or useless additional hints (as colour) to the samples for the neural network to use. Each class may correspond with multiple colours. The colour of each digit in a class is chosen randomly from a set of two colours. We choose the colours of each class (its colour set) in a way that no two classes (colour sets) in the colour space can be linearly separated, see Figure 12. We simulate two scenarios: 1) each digit is coloured using its own set

of colours, –i.e. the colour concept and task class are fully correlated. We call this dataset ColorDataset1. 2) all colourings are random (from the same set) –i.e. no relation between the colour concept and task class. We call this dataset ColorDataset2.

Figure 12: How the two hint colours are chosen for each figure in ColorDataset1. Each figure is associated with two different colours that have the highest distance in colour space.

For generating the concept samples (for training the concept classifier), we shuffle pixels of images (to wipe out the image information) and then add colours. Each colour set (two colours) is considered a concept.

5.2 Images with Captions Dataset

In this dataset, we add a hint caption to two classes of the Imagenet dataset [7, 34], namely class dog and class cat. The hint is added as a white text on the image (by changing the pixels of the image). So part of the sample pixels has some extra information about the class. We consider two scenarios: 1) The caption always reads the same as the image (the word cat for cat images, and dog for dog images). We call this dataset CaptionDataset1. 2) The caption is always a random word and hence

does not include any information about the classification task (dogs vs. cats). We call this dataset CaptionDataset2. The captions have random rotation and scaling associated with them. Figure 13 shows two samples of the CaptionDataset1 images.

Figure 13: Two samples from CaptionDataset1.

For generating the concept samples (caption concept), we shuffled pixels of images (to wipe out the image information) and then add a caption to the resulting shuffled image. This technique makes sure that the concept is only present in these samples and our representation of the concept is the most accurate.

5.3 Analysis of Concept Influence

In this experiment, we show the effectiveness of the proposed method in detecting the causal relationship between a concept and task classes of neural networks. We train a neural network on each of our datasets. The results for CaptionDataset1 and CaptionDataset2 are shown in Figure 14. Similar results were obtained on the ColorDataset1 and ColorDataset2. On the left side, it can be seen that the concept (caption dog) was detected to have a 98% necessary relationship with the class dog. All samples predicted by g_w as the dog class (above the red horizontal line) are predicted by $g_{w'}$ to have the concept (i.e. they are on the right of the red vertical line). Plotting the modified F1 score for different values of thresholds t gives the necessary quantification curve (blue), which has an area under the curve of close to

one. The sufficient quantification curve (orange) can be obtained from the scatter plot of C vs. T (instead of T vs. C) because C being sufficient for T is equivalent to T being necessary for C . The negative necessary quantification curve (green), can be obtained from the scatter plot of T vs. $1 - C$ and the negative sufficient quantification curve (red), can be obtained from the scatter plot of $1 - C$ vs. T . The results for CaptionDataset2 are shown on the right of Figure 14. It can be seen that there is no tangible relationship between the dog class and the caption concept i.e. the AUC of all four measures are small. This confirms that the proposed method detects the causal relationship between the concept and the task classification.

Figure 14: Comparison of two networks with the same structure, while one has been trained on CaptionDataset1 (left side), and the other has been trained on CaptionDataset2 (right side). The area under the necessary curve shows that the concept is necessary for the task class. The values for each AUC are mentioned at the bottom of the figure. These results are extracted from the last layers of a neural network with an alexnet structure.

Now that we have established that the method can detect the usefulness of the hints, from now on we only use the CaptionDataset1 and ColorDataset1.

5.4 Informativeness of the Proposed Relationship Measures

In this experiment, we show how the proposed relationship measures can reveal more than just a correlation between a concept and a task class. We examine several classes of a neural network with ten classes, trained on our ColorDataset1 to check the causal relationships between different target classes and a particular concept. In particular, we inspect for the colour set associated with class handwritten figure 1 (see Figure 12). So, we expect the concept (the corresponding colour set of class 1) to be a necessary condition for class 1 and negative necessary for other classes. Figure 15 shows the results of this experiment for two task classes 0 and 1. It can be seen that the presence of this colour set is a 86% necessary condition for class 1 (see the blue solid curve on the bottom-right sub-figure), while it is a 93% negative necessary condition for class 0 (see the dashed green curve on the bottom-left sub-figure). This confirms that for class 1, it is necessary to have the colour set (if not present, the task is not classified as class 1), but for other classes, like class 0, it is necessary not to have this colour set (if the colours are present, the output will not be class 0).

Figure 15: Analysis for class 0 (left) and class 1 (right). The causal relationship of a colour concept with two different classes. This concept is necessary for class 1 (right) but the concept is inconsistent with class 0 (left) so the negative of concept is necessary for class 0.

5.5 Comparison with Linear Based Classifier

Many methods use linear classifiers to extract concept information from the middle layers of neural networks. In this experiment, we check the implications of using a linear classifier to detect concepts in network activations. We check the effectiveness of such assumption in the detection of concept in earlier layers of a neural network trained on our CaptionDataset1. As we know by design the concept dog and the caption dog have a positive correlation (all dog images have the dog caption). Figure

16 shows the two scenarios, one with our design ($g_{w'}(z)$ as the concept classifier) on left, and one with linear design (a linear classifier as the concept classifier) on right and shows the effectiveness of the proposed measures compared with directional derivation measures in capturing the relationship between task class and concept.

While our method, shows that for this layer, the negative sufficient (red curve) and negative necessary (green curve) are both good descriptions of the relationship. Since the area under the green (78%) and red curve (89%) is high. With linear classifiers, the four relationship area measures are almost the same, and the designed relationship cannot be detected. These findings are consistent with the points mentioned in [21].

This experiment shows that the linear classifiers have failed to extract concept information specifically in earlier network layers. The same phenomenon might be observed on the final layers of a neural when a concept is merged with other concepts to create a higher-level concept.

5.6 Comparison with TCAV method

In many methods, the directional derivation of a classifier with respect to activations or input is considered a good representation of the model’s local behaviour (first-order approximation) [19, 55]. As it was shown in TCAV [19], the directional derivative can be calculated by the dot product of concept gradient and task gradient. These methods use the dot product of concept classifier gradient and task classifier gradient as an agreement score on different class samples (linear assumption). Here we demonstrate that this score is not accurate. We test the score on a neural network trained on our CaptionDataset1. We test the relationship between the dog class and the caption cat. As we know, by design, these two information are inconsistent in the training data, -i.e. no training data have both. Figure 17 shows that the proposed measures capture the negative relationship between the class dog and the concept caption cat correctly.

Figure 16: Shortcoming of the linear layers to pick up a concept in earlier sections of the neural network (right) while it can be easily detected with the proposed method (left). If a concept is not present, knowing where in the network the concept is disappearing can be a helpful tool. For instance in transfer learning.

Our method captures the correct relationship (Negative Necessary) of the concept cat for the class dog, as it is seen in Figure 17 the area under the negative necessary (green) curve is very close to one. The third row of Figure 17 shows the distribution of directional derivatives. Though the concept and class are by design inconsistent, directional derivatives are positive on all samples of distribution sample set, showing it is not a reliable explanation. Distribution sample set (points that the evaluations were done) consists of dog and cat images with both dog and cat captions.

Further investigating the relationship between directional derivative, we evaluated the difference of the task classifier output and the concept classifier output and plotted it against the directional derivative. Figure 18 shows that there is no positive or negative correlation between the directional derivative value and the agreement of the concept and classifier in both linear and non-linear cases.

5.7 Comparision with IBD method

Since the IBD method [55] is limited to convolutional networks, in this experiment we change our method to be comparable to IBD. We examine the last hidden layer of a Resnet18 trained on the Places365 dataset [54] – a dataset where each class is a place. This network was the benchmark of the IBD method. We use the same set of concept classifiers they trained (with the parameters they provided). We use 10,000 samples from the places365 validation set without their labels as the distribution sample set. Our concepts come from the same dataset IBD method used as their benchmark, Broden [4] – a dataset with segmentation annotations. For a better comparison, we use only use the concepts originally used in the IBD benchmark.

This experiment is designed to find the most important concepts for classifying each class of the Places365 dataset. For each concept, a concept classifier is trained, and then each task class (a class of Places365) is examined against each concept. The

Figure 17: Results of the caption experiment for different linear (right) and nonlinear (left) concept classifiers show that directional derivative can be misleading in both cases. The joint distribution of concept and class decisions for the nonlinear and linear models (first row). Changes of relationship measures with respect to different thresholding (second row). Distribution of directional derivative (third row). Moreover, the new measures give us more information about the type of the relationship between concept and task class.

necessary scores of concepts for each task class are sorted and the highest values are reported as the most necessary concepts for the class. The most necessary concepts are then compared against IBD recommended concepts, by decomposition of the

Figure 18: Results of caption experiment for different linear (right) and nonlinear (left) concept classifiers show that directional derivative can be misleading in both cases.

decisions into concept space. The top seven are reported for both methods. Then three different annotators were asked to highlight concepts that are not relevant to the class, their majority vote is considered as irrelevant concepts (highlighted in the Tables 8 and 9). The concepts are from left to right in decreasing importance.

Examining the results of the experiment (by a majority of three annotators), it is apparent that our method assigns more reasonable values of necessary scores to the concepts (compared to what IBD calculates based on its decomposition process). For instance for their benchmark topiary garden, in IBD’s top 7 concepts, IBD suggested tail and sheep (among five others) which are irrelevant to the class of topiary garden. On the other hand, our method suggests plant and tree which are quite relevant concepts to the topiary garden class. For soccer field class our method proposes grass, pitch, grandstand, court, person, post and goal which are all are relevant. But IBD suggests pitch, field, cage, ice rink, tennis court, grass, and telephone booth. Among these concepts, cage, ice rink, tennis court, and telephone booth are irrelevant to the soccer field class (see Tables 8 and 9).

We also realized that the quality of the IBD benchmark concepts is not verified. So we checked in which samples of distribution sample set every concept is maximized (the output of the concept classifier network is maximal) and where it is minimized.

Figures 19 and 20 shows examples of this experiment. In each section, in the first row, the five test images of the minimal concept classifier output are shown in the increasing relevance order from left to right. In the second row, the five test images that maximize the concept are shown in the increasing relevance order from left to right. This test will do a sanity check on concept learning, (since the IBD method does not check the accuracy of its learned concepts) and shows that what is considered as each of the concepts, and shows if some of the concepts are not reliable.

For instance, we found that the concepts sheep, cow, hay and elephant are not accurate and maximized whenever a dirt field is present, (like a horse race track) as shown in Figures 21 and 22. These figures follow the same structure as Figures 19 and 20. Their second row shows that even some images maximize more than one of these concepts, meaning the network cannot even distinguish between these concepts. Or in other words, in this layer, these individual animal concepts are merged to form a higher-level “animal” concept while the original ones are forgotten (apparently the type of animal is not important for Places365 classes). Usage of concepts without verification decreases the quality of explanations of the IBD method.

The same phenomenon is observed for the set of concepts of towel, bathtub, toilet tissue, soap dispenser and bidet (Figures 23 and 24) and the set of concepts of refrigerator, microwave, knife, kitchen island, and kettle (Figures 25 and 26). This will result in the degradation of IBD’s accuracy of concepts which propagates into their decomposition process. These results emphasize the importance of checking accuracy of a concept prediction, which is indeed one of the very first steps in the proposed method. Our method allows verifying concept classification accuracy and prevents such pitfalls.

For further investigation, we also calculate the deepdream maximal output for each class [29]. Deepdream is a method to calculate an input that maximally activates a certain neural network, by optimizing the input on several scales (or octaves).

While deepdream is sometimes a good visualization method to figure out what maximally activates a certain classifier, the method fails to capture higher structures. For instance in Figure 27, while concepts like sky, tree or fabric make sense, the concepts with a higher structure like wall, floor, windowpane, building, person, or head are nothing like what a human may expect.

topiary_garden							
Ours	plant	hedge	tree	brush	flower	bush	sculpture
IBD	hedge	brush	tail	palm	flower	sheep	sculpture
crosswalk							
Ours	crosswalk	road	sidewalk	post	container	streetlight	traffic light
IBD	crosswalk	minibike	pole	rim	porch	central reservation	van
living_room							
Ours	armchair	sofa	back	cushion	back pillow	coffee table	ottoman
IBD	armchair	fireplace	inside arm	shade	sofa	frame	back pillow
market/indoor							
Ours	pedestal	sales booth	shop	case	bag	bulletin board	food
IBD	sales booth	pedestal	food	fluorescent	shop	shops	apparel
soccer_field							
Ours	grass	pitch	grandstand	court	person	post	goal
IBD	pitch	field	cage	ice rink	tennis court	grass	telephone booth
forest/broadleaf							
Ours	tree	bush	trunk	cactus	brush	fire	leaves
IBD	tree	trunk	bush	leaves	semidesert	grid	clouds
art_school							
Ours	person	hand	paper	plaything	fabric	bag	board
IBD	paper	drawing	plaything	painting	hand	board	figurine

Table 8: Explanations for several classes using our method and IBD method. The concepts labelled as irrelevant by the majority of three annotators are in bold.

dining_hall						
Ours	drinking glass	stool	table	spindle	menu	person
IBD	plate	light	stool	sash	napkin	display board
						plate
						spindle
shoe_shop						
Ours	shoe	bottle	shelf	box	gym shoe	boot
IBD	shoe	gym shoe	handbag	hat	catwalk	shop window
						bag
						minibike
butte						
Ours	mountain	hill	desert	badlands	rock	valley
IBD	hill	badlands	desert	cliff	cloud	mountain
						land
						diffusor
canyon						
Ours	mountain	rock	cliff	hill	badlands	land
IBD	cliff	mountain	badlands	desert	pond	bumper
						hill
coast						
Ours	sea	sand	land	embankment	rock	mountain
IBD	sea	wave	land	mountain pass	sand	cliff
						water
						cloud
creek						
Ours	bush	river	rock	land	cliff	tree
IBD	river	waterfall	land	pond	leaf	ice
						earth
						fire

Table 9: Explanations for several classes using our method and IBD method. The concepts labelled as irrelevant by the majority of three annotators are in bold.

object.part.tree

object.wood

object.table

Figure 19: Some minimal and maximal samples of concepts.

object_part.road

object.grass

Figure 20: Continued.

object.sheep

object.cow

object.hay

Figure 21: Some minimal and maximal samples of animal related concepts.

object.elephant

Figure 22: Continued.

object.towel

object.bathtub

object.toilet tissue

Figure 23: Some minimal and maximal samples of bathroom related concepts.

object.soap dispenser

object.bidet

Figure 24: Continued.

object.refrigerator

object_part.microwave

object.knife

Figure 25: Some minimal and maximal samples of kitchen related concepts.

object.kitchen island

object.kettle

Figure 26: Continued.

Figure 27: Deepdream results of the concept maximal inputs for the top 12 most common concepts, while some of them make sense, most of them are unexpected.

Figure 28: Deepdream results of the concept maximal inputs for the second top 12 most common concepts, while some of them make sense, most of them are unexpected.

Chapter 6

Conclusion and Future Work

We proposed a framework for verifying the presence of high-level concepts in the activations of the intermediate layers of neural networks. We also determine the type or nature of the causal relationship between a concept and the neural network task classes by quantifying the causal relationship between the task classes and the concept. We showed the effectiveness of the proposed measurements through several comparative experiments, demonstrating improved performance compared with the two previous methods.

The proposed method enables tracking a certain concept across layers of a neural network. This analysis can show either a concept was too difficult for the whole network to begin with, so the model didn't have the extraction power for it, or it got lost during intermediate layers. Moreover, the analysis shows how much influence a concept has on each layer and potentially at which layer the concept was forgotten. The main reason this analysis is possible is the particular choice of the concept classifier. Moreover, this particular choice of concept classifier helps avoid unnecessary simplification made by previous works, like first-order approximation and linear assumption.

Through experiments, we showed that our framework can deal with situations that the concept is detectable non-linearly while previous methods (TCAV and IBD)

struggled in detecting such concepts. Moreover, through several experiments, we showed the drawbacks of the first-order approximation used in [19] and [55]. We also evaluate the quality of the concept detected by the proposed method and IBD using human judgment. The results confirm that in most cases the proposed method results in more relevant concepts.

Most concepts influencing a certain decision are necessary conditions, but for finding equivalence conditions we need to find a concept or a set of concepts that is both necessary and sufficient. A combination of necessary conditions to generate equivalence conditions can be the subject of future work.

List of References

- [1] Julius Adebayo, Justin Gilmer, Ian Goodfellow, and Been Kim. Local explanation methods for deep neural networks lack sensitivity to parameter values. *arXiv preprint arXiv:1810.03307*, 2018.
- [2] Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes. 2016.
- [3] David Baehrens, Timon Schroeter, Stefan Harmeling, Motoaki Kawanabe, Katja Hansen, and Klaus-Robert Müller. How to explain individual classification decisions. *arXiv preprint arXiv:0912.1128*, 2009.
- [4] David Bau, Bolei Zhou, Aditya Khosla, Aude Oliva, and Antonio Torralba. Network dissection: Quantifying interpretability of deep visual representations. In *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, volume 2017-Janua, pages 3319–3327, 2017.
- [5] Diogo V Carvalho, Eduardo M Pereira, and Jaime S Cardoso. Machine learning interpretability: A survey on methods and metrics. *Electronics*, 8(8):832, 2019.
- [6] Piotr Dabkowski and Yarin Gal. Real time image saliency for black box classifiers. *arXiv preprint arXiv:1705.07857*, 2017.
- [7] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.
- [8] Finale Doshi-Velez and Been Kim. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*, 2017.
- [9] Dumitru Erhan, Yoshua Bengio, Aaron Courville, and Pascal Vincent. Visualizing higher-layer features of a deep network. *University of Montreal*, 1341(3):1, 2009.
- [10] Ali Farhadi, Ian Endres, Derek Hoiem, and David Forsyth. Describing objects by their attributes. *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2009*, 2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition:1778–1785, 2009.
- [11] Ruth Fong and Andrea Vedaldi. Net2vec: Quantifying and explaining how concepts

- are encoded by filters in deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8730–8738, 2018.
- [12] Kunihiko Fukushima, Sei Miyake, and Takayuki Ito. Neocognitron: A neural network model for a mechanism of visual pattern recognition. *IEEE transactions on systems, man, and cybernetics*, (5):826–834, 1983.
 - [13] Amirata Ghorbani, James Wexler, James Y Zou, and Been Kim. Towards automatic concept-based explanations. In *Advances in Neural Information Processing Systems*, pages 9277–9286, 2019.
 - [14] Bryce Goodman and Seth Flaxman. European union regulations on algorithmic decision-making and a “right to explanation”. *AI Magazine*, 38(3):50–57, 2017.
 - [15] Yash Goyal, Amir Feder, Uri Shalit, and Been Kim. Explaining classifiers with causal concept effect (cace). *arXiv preprint arXiv:1907.07165*, 2019.
 - [16] 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.
 - [17] Dominik Janzing, Lenon Minorics, and Patrick Blöbaum. Feature relevance quantification in explainable ai: A causal problem. In *International Conference on Artificial Intelligence and Statistics*, pages 2907–2916. PMLR, 2020.
 - [18] Been Kim, Oluwasanmi Koyejo, Rajiv Khanna, et al. Examples are not enough, learn to criticize! criticism for interpretability. In *NIPS*, pages 2280–2288, 2016.
 - [19] Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas, and Rory Sayres. Interpretability beyond feature attribution: Quantitative Testing with Concept Activation Vectors (TCAV). *35th International Conference on Machine Learning, ICML 2018*, 6:4186–4195, 2018.
 - [20] 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. *arXiv preprint arXiv:1711.00867*, 2017.
 - [21] Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang. Concept bottleneck models. In *International Conference on Machine Learning*, pages 5338–5348. PMLR, 2020.
 - [22] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012.
 - [23] Christoph H. Lampert, Hannes Nickisch, and Stefan Harmeling. Attribute-based classification for zero-shot visual object categorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(3):453–465, mar 2014.
 - [24] Yann LeCun. The mnist database of handwritten digits. <http://yann.lecun.com/exdb/mnist/>, 1998.

- [25] Scott Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *arXiv preprint arXiv:1705.07874*, 2017.
- [26] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.
- [27] Tim Miller. Explanation in artificial intelligence: Insights from the social sciences. *Artificial intelligence*, 267:1–38, 2019.
- [28] Christoph Molnar. *Interpretable machine learning*. Lulu. com, 2020.
- [29] Alexander Mordvintsev, Christopher Olah, and Mike Tyka. Inceptionism: Going deeper into neural networks. 2015.
- [30] Chris Olah, Alexander Mordvintsev, and Ludwig Schubert. Feature visualization. *Distill*, 2(11):e7, 2017.
- [31] Genevieve Patterson, Chen Xu, Hang Su, and James Hays. The SUN attribute database: Beyond categories for deeper scene understanding. *International Journal of Computer Vision*, 108(1-2):59–81, 2014.
- [32] 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, 2016.
- [33] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Anchors: High-precision model-agnostic explanations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.
- [34] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.
- [35] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117, 2015.
- [36] 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.
- [37] Ramprasaath R Selvaraju, Abhishek Das, Ramakrishna Vedantam, Michael Cogswell, Devi Parikh, and Dhruv Batra. Grad-cam: Why did you say that? *arXiv preprint arXiv:1611.07450*, 2016.
- [38] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In *International Conference on Machine Learning*, pages 3145–3153. PMLR, 2017.

- [39] Avanti Shrikumar, Peyton Greenside, Anna Shcherbina, and Anshul Kundaje. Not just a black box: Learning important features through propagating activation differences. *arXiv preprint arXiv:1605.01713*, 2016.
- [40] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.
- [41] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.
- [42] Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. Smoothgrad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825*, 2017.
- [43] Marina Sokolova, Nathalie Japkowicz, and Stan Szpakowicz. Beyond accuracy, f-score and roc: a family of discriminant measures for performance evaluation. In *Australasian joint conference on artificial intelligence*, pages 1015–1021. Springer, 2006.
- [44] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*, 2014.
- [45] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.
- [46] Erik Štrumbelj and Igor Kononenko. Explaining prediction models and individual predictions with feature contributions. *Knowledge and information systems*, 41(3):647–665, 2014.
- [47] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *International Conference on Machine Learning*, pages 3319–3328. PMLR, 2017.
- [48] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.
- [49] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *2nd International Conference on Learning Representations, ICLR 2014 - Conference Track Proceedings*, pages 1–10, 2014.
- [50] Peter Welinder, Steve Branson, Takeshi Mita, Catherine Wah, Florian Schroff, Serge Belongie, and Pietro Perona. Caltech-ucsd birds 200. 2010.
- [51] James Wexler, Mahima Pushkarna, Tolga Bolukbasi, Martin Wattenberg, Fernanda Viégas, and Jimbo Wilson. The what-if tool: Interactive probing of machine learning models. *IEEE transactions on visualization and computer graphics*, 26(1):56–65, 2019.

- [52] Chih-Kuan Yeh, Been Kim, Sercan Arik, Chun-Liang Li, Tomas Pfister, and Pradeep Ravikumar. On completeness-aware concept-based explanations in deep neural networks. *Advances in Neural Information Processing Systems*, 33, 2020.
- [53] Chih-Kuan Yeh, Been Kim, Sercan O. Arik, Chun-Liang Li, Tomas Pfister, and Pradeep Ravikumar. On Completeness-aware Concept-Based Explanations in Deep Neural Networks. pages 1–27, 2019.
- [54] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.
- [55] Bolei Zhou, Yiyou Sun, David Bau, and Antonio Torralba. Interpretable basis decomposition for visual explanation. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 11212 LNCS:122–138, 2018.