TECHNISCHE HOCHSCHULE NÜRNBERG
GEORG SIMON OHM

Faculty of Computer Science

Automatic image recognition in upload filters - computing of transparent decisions (XAI), with the help of Deep Learning methods

Bachelor thesis submitted in partial fulfillment of the requirements for the degree of Bachelor of Science in Business Information Systems and Management

Author:

Timo Bohnstedt

Student ID: 301 8484

Assesor:	Prof. Dr. Alfred Holl
Supervisor:	Prof. Dr. Florian Gallwitz

Hinweis: Diese Erklärung ist in alle Exemplare der Abschlussarbeit fest einzubinden. (Keine Spiralbindung)

TECHNISCHE HOCHSCHULE NÜRNBERG
GEORG SIMON OHM

Prüfungsrechtliche Erklärung der/des Studierenden

Angaben des bzw. der Studierenden:

Name: Vorname: Matrikel-Nr.:

Fakultät: Studiengang:

Semester:

Titel der Abschlussarbeit:

Ich versichere, dass ich die Arbeit selbständig verfasst, nicht anderweitig für Prüfungszwecke vorgelegt, alle benutzten Quellen und Hilfsmittel angegeben sowie wörtliche und sinngemäße Zitate als solche gekennzeichnet habe.

Ort, Datum, Unterschrift Studierende/Studierender

Erklärung zur Veröffentlichung der vorstehend bezeichneten Abschlussarbeit

Die Entscheidung über die vollständige oder auszugsweise Veröffentlichung der Abschlussarbeit liegt grundsätzlich erst einmal allein in der Zuständigkeit der/des studentischen Verfasserin/Verfassers. Nach dem Urheberrechtsgesetz (UrhG) erwirbt die Verfasserin/der Verfasser einer Abschlussarbeit mit Anfertigung ihrer/seiner Arbeit das alleinige Urheberrecht und grundsätzlich auch die hieraus resultierenden Nutzungsrechte wie z.B. Erstveröffentlichung (§ 12 UrhG), Verbreitung (§ 17 UrhG), Vervielfältigung (§ 16 UrhG), Online-Nutzung usw., also alle Rechte, die die nicht-kommerzielle oder kommerzielle Verwertung betreffen.

Die Hochschule und deren Beschäftigte werden Abschlussarbeiten oder Teile davon nicht ohne Zustimmung der/des studentischen Verfasserin/Verfassers veröffentlichen, insbesondere nicht öffentlich zugänglich in die Bibliothek der Hochschule einstellen.

Hiermit ☐ genehmige ich, wenn und soweit keine entgegenstehenden Vereinbarungen mit Dritten getroffen worden sind,

☐ genehmige ich nicht,

dass die oben genannte Abschlussarbeit durch die Technische Hochschule Nürnberg Georg Simon Ohm, ggf. nach Ablauf einer mittels eines auf der Abschlussarbeit aufgebrachten Sperrvermerks kenntlich gemachten Sperrfrist

von Jahren (0 - 5 Jahren ab Datum der Abgabe der Arbeit),

der Öffentlichkeit zugänglich gemacht wird. Im Falle der Genehmigung erfolgt diese unwiderruflich; hierzu wird der Abschlussarbeit ein Exemplar im digitalisierten PDF-Format auf einem Datenträger beigefügt. Bestimmungen der jeweils geltenden Studien- und Prüfungsordnung über Art und Umfang der im Rahmen der Arbeit abzugebenden Exemplare und Materialien werden hierdurch nicht berührt.

Ort, Datum, Unterschrift Studierende/Studierender

Abstract

Nowadays, the decisions made by Deep Learning Algorithms influence our everyday life. Since the Discussion around the "Directive on Copyright in the EU Digital Single Market" had started, more and more people are interested in the decisions which are made by algorithms. In the following, this thesis provides an theoretical introduction about Machine Learning, Deep Learning and Explainable Artificial Intelligence. Moreover, a prototype gives a realistic insight into how Deep Learning and Explainable Artificial Intelligence can be implemented within a Jupyter Notebook. Finally, the achieved results get evaluated and generalized among the topics of Upload Filter.

Preface I

My work at the Siemens AG had given me tremendous insight into the business analytics field. But since my interested is going more into the statistics Field, I was highly motivated to work on topics related to Machine Learning and Deep Learning. In the next few years, I think of a position in the Data Science Field, and so I committed myself to a final year project which covers this field as well. The Explainable Artificial Intelligence area made me even more curious to find out what we can learn from machine learning algorithms. To work on this was a tough challenge, but at least it was a useful insight into scientific writing.

Preface II

The first time I came into contact with Machine Learning was through an IT project. The intention was to develop a self-driving car, controlled by a deep neural network. The whole project was very challenging, but I had a lot of fun, and my motivation was even higher. My motivation was immediately put a rough test in my exchange semester. In Hong Kong, I focused on machine learning during the entire semester. Mathematical proofs that were required to can pass one of the courses were an extremely challenging task. Mainly because of lacking prior knowledge of statistics, I had to work twice as hard. During the semester, I was able to close the gaps, master the tasks and finally did an excellent job at the programming part. This experience has strengthened my wish to concentrate further in this direction, and so I was highly motived to put everything that I have learned into my final year project. Since then, I have learned a lot and even if the work was not as perfect as it could it helped me a lot particularly since I am participating in the computer science master program with a focus on data science. I am sure there is more to learn. So, I think this thesis was a tiny step in my life long learning journey.

Contents

Chapter 1

Introduction

> "Most times, the way isn't clear, but you want to start anyway. It is in starting with the first step that other steps become clearer."
>
> — Israelmore Ayivor, Leaders' Frontpage: Leadership Insights from 21 Martin Luther King Jr. Thoughts

1.1 Upload Filter in the public discussion

The "Directive on Copyright in the EU Digital Single Market", which came into force on the 7th of June in 2019, introduced a requirement for an automated Upload Filter. The discussion refers to Article 17 (previously Article 13) because it forces content-sharing services such as Facebook, Google and YouTube to be responsible for copyright violations. From this point on, it is no longer possible to check contents manually [Woollacott, 2019].

In Figure 1.1 you can see Protesters in Berlin. For them, the fear of an Upload Filter is real. They are concerned that these filters can harm their business as small content creators or their freedom of speech.

Different parties, organizations and scientists are arguing for and against the copyright directive. For example, the Christian Democratic Union (CDU) and its representative at the European Parliament Axel Foss support the directive. He said that it protects the freedom of expression on the internet and a diverse media landscape [Europarl, 2017]. Critics as Julia Reda from th e Pirate Party (also a member of the European Parliament) denies this statement. She described it as a change into a dark future for internet freedom, when upload filters would check every content automatically [Woollacott, 2019].

Figure (1.1): In Berlin, opponents of the copyright directive are protesting. They fear that Article 13 will lead to upload filters, which they see as a danger considering different aspects [Tagesspiegel, 2019].

More precisely, she assumes that technology does not know the nuances of particular copyright law yet. And even more, she fears that small content creators are not able to interpret the decisions of the upload filters [Reda, 2019]. Dr Gallwitz is an associate professor at the Technical University of Nuremberg where he is researching in the field of pattern recognition. He criticized that there is no smart software that could recognize quotations from other sources [Gallwitz, 2019].

Instead of using smart software, a real-world Upload Filter is realized through fingerprinting. This technique allows comparing blocked content with the latest user-generated content. The hardware requirements which are needed to compare the contents are proportionally small. Less high hardware requirements are one of the reasons why most companies still use older techniques (e.g. hash values), even they can build smarter Machine Learning models [Spoerri, 2019] [Wagner, 1983].

The Microsoft software Photo-DNA identifies child pornography with the fingerprinting method, e.g. if an image in Microsoft's database is marked as pornographic. That means YouTube can detect when someone wants to upload such an image to the platform [Microsoft, 2013]. The Software Content ID uses the same technique to identify copyright offences. It is used and developed by YouTube. That shows how Technique Companies are responsible for which method is used. Even scientific research has shown that different approaches are performing better, the companies have to see this decision from a business

point of view [YouTube, 2010].

Using Machine Learning models as upload filters in the future could be possible, but among this usage are many problems. One of them is to get labelled data (necessary to train a model) which can detect patterns. Another problem is to define the patterns or the labels itself [Waltermann and Hess, 2019]. Labels are categories which you could assign to an image [Goodfellow et al., 2016] e. g. attach the label cat to an image with a cat. In the case of copyright violations or offensive content, we block any item that has labels with copyright content. A machine learning algorithm which can detect all assaulting labels alone must be complex because the complexity from machine learning algorithms is increasing with increasing input features and an increasing amount of expected outputs [Yao et al., 2017]. In machine learning terms, a feature is any input we can put into a machine learning algorithm, e.g. an image with 32 x 32 pixels has 1024 input features because every pixel counts as a single feature [Goodfellow et al., 2016].

1.2 Upload filters and Explainable Artificial Intelligence

As mentioned in the introduction to the Directive on EU Copyright and Upload Filters (1.1), a Machine Learning Algorithm, used to detect copyright infringements would be too complex. Joel Dudley, director of the Institute for Next-Generation Healthcare at the Ichan School of Medicine, said: "We can build these models, but we don't know how they work." [Knight, 2019]. His statement was part of an interview with the journalist Will Knight from MIT's Technology Review about Explainable Artificial Intelligence (XAI). A precise definition of what exactly Explainable Artificial Intelligence is does not exist. The global idea is that we use a Machine Learning Algorithmto make decisions which must be in natural language. For example, if we use an image as an input for a Machine Learning Algorithm, the task would be to label the images with a label for cancer or no cancer. A doctor who wants to use this information could be interested in the question of why the algorithm, predicts a cretin label for this image. [Samek et al., 2017].

Apart from the Technology Review, newspapers like The New York Times, Financial Times and The Register have also reported that an Explainable Artificial Intelligence is necessary an Machine Learning Algorithm is used for an automatic decision making [Kuang, 2017] [Robinson, 2017] [Waters, 2017]. David Gunning explains in his research project about Explainable Artificial Intelligence that Machine Learning can bring a huge benefit to the transportation sector, as well as to the finance , security , legal , medicine and military sector. Gunning claims that algorithms are limited, because they can not explain their actions and decisions to users in an understandable way. He assumes Explainable Artificial Intelligence

will be essential if users want to understand, trust, and effectively manage this incoming new Machine Learning Algorithms[Gunning, 2019]. Figure 1.2 shows an example of how transparent Upload Filter can be applied to uploaded images. Even this look not intuitive for a non-technical user, if we build an Upload Filter with Deep Learning methods, it has to be an Explainable Artificial Intelligence [Waltermann and Hess, 2019]. Otherwise, there will be no explanation of how the decision from an Upload Filter was made.

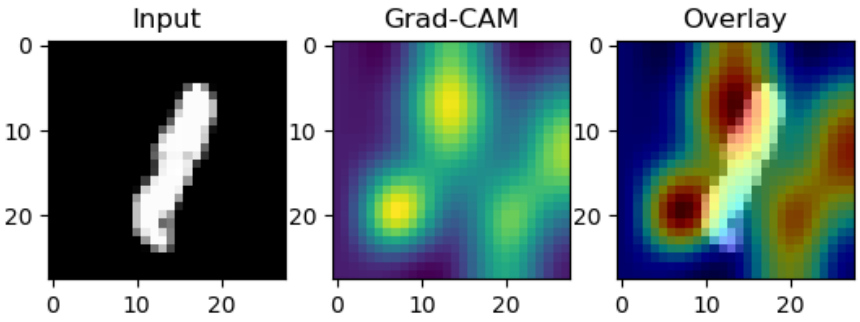

Figure (1.2): Example of how transparent Upload Filter can be applied to uploaded images [Versloot, 2019]

1.3 The Goal of this Thesis

Even automatic image recognition in Explainable Artificial Intelligence with Deep Learning procedures is just a research topic, it is important to evaluate how we could generate comprehensible decisions (XAI). So, the main objective is to find an answer to the following question:

> *"How can automatic image recognition be implemented in such a way that the resulting decisions are transparent?"*

For this purpose, I would like to develop an algorithm which should be able to recognize patterns on images (automatic image recognition) automatically. If a pattern is recognized by the algorithm, it should assign a label to the image. In the pre-field, I define which markings correspond to appropriate content and which to inappropriate content. This way, the algorithm should finally decide which images are appropriate or inappropriate according to the criteria I defined before (Upload Filter).

Afterwards, the results of the algorithm are presented in such a way that the decisions become transparent (Explainable Artificial Intelligence). What transparent actually means is hard to tell at this point of my thesis, because it depends on the algorithms and techniques, which are used to implement the filter. Anyway to make things more clear, transparent could explained as the following:

> *Imagine a social media platform like Instagram, where everybody could share its images. If someone shares an image and it is getting blocked by a filter (e.g. because of copyright violation), the user should be able to understand why the decision would be made. There could be an automatically generated text, which says that the algorithm had detected a copyright issue and it highlights specific parts in the blocked image. The highlighted details should tell the user why the algorithm came to decision (content is protected by copyright law).*

In contrast to my example, an application in real life would be much more complicated. First, in such a real-world application, there is a much more extensive range of labels that the algorithm could set. Second, it is not only possible to upload images, but also text and sound which would have to be checked. Therefore my example application is only a prototype. Anyway, the question is asked on top of this section will be answered anyways. At least from a theoretical perspective. In the end, I would like to use my prototype to transfer the knowledge gained to a more general scenario.

1.4 Thesis Structure

A brief Introduction (1) about the two main terms Upload Filter and Explainable Artificial Intelligence is already addressed in the previous introduction section. Preface I and II is about the motivation for this thesis and about the prior knowledge which in order to work on this task. The "Requirement Chapter"(2 focuses on details which are necessary to answer the question which is given at the Goal of this Thesis (1.3). The "Requirements"(3 section defines properties and demands of a Upload Filter prototype which makes transparent decisions with the help of Deep Learning and Explainable Artificial Intelligence methods. The next two chapters (introduction (4),(introduction (5), are presenting an analytical approach o how such a prototype can be implemented. The next chapters are applying the answers to a broader use case scenario like described in section "The Goal of this Thesis" (1.3). It starts with an evaluation followed by a generalization and ends with a summary. Before the work on this thesis has actually started, to finish a structured plan in the methodological table (Figure 1.3.) was mandatory. Here it is presented in a translated and shortened version. Here can be seen a detailed explanation of the thesis structure.

Chapter	Research Quesitons	Methods	Objektives
Introduciton	Why XAI is important for Upload Filters and what publications are published among this Topic?	Literature research	A short introduction in the topics around Upload Filters and Explainable Artificial Intelligence and insights of prior work from which focus on this particular area
Background and Theory	What are the theortical fundamental which are needed to understand an Upload Filter Prototype implementaitn which uses XAI to presst ist predictions on image classes?	Literature research	A detailed explanation of how exactly XAI works within a predictive model works, and therefore an introduction of the topics which are needed in order to understand XAI (machine learning, deep learning, etc.)
Requirenments	What are the requirements if the question "Is XAI in Upload Filters useful?" will be answered?	Literature research, Brain Storming	A first idea of what precisely a prototype must do if it should be used to generate knowledge about XAI in Upload Filters which estiamtes the probability of an image to belongs to a certain class
Functional Specifications	What are the functional specifications of the mentioned prototype and in which use case szenario will this be a useful application?	Literature research, Use-Case Szenario	Functional specifications and a use case scenario for an Upload Filter implementation.
IT-Specifications	How exactly is the prototype build, which software libraries are going to be used, and how will the knowledge be presented transparently?	Programming, Use-Case Szenario, Use Programming Language Dokumentation	The prototype and a presentation of the crucial parts of its implementation
Evaluation	How good is the Prototyp in predicting classes of images, how does it present its results?	Variante analysis (comapre the actual targets of the achived performacne9	An overview of the prototpye implementation
Generalization	What do the results from the evaluation mean in a broader scenario?	Varianz analyis from the evaluation the expections found in literature in the introdrucion	A generic answer to the question if it is useful in image recognition
Conclusion	What can I conclude while working on this topics?	Presenting my own Opinon	An explanation of my thoughts around this topics while working on them for a particular time

Figure (1.3): Methodological table which was prepared in the run-up to the bachelor thesis in cooperation with the assesor Dr Prof Alfred Holl.

1.5 Related Work

Katharina Blandina Weitz wrote in her master thesis "Applying Explainable Artificial Intelligence for Deep Learning Networks to Decode Facial Expressions of Pain and Emotions" about programmes used in hospitals, which detect pain in facial expressions. According to her, this software should use deep learning to support human practitioners. Towards her thesis, Ms Weitz found out that these deep learning methods work well in recognising pain in human facial expressions, but they are hard to interpret. From her abstract, you can learn that XAI does not depend on a specific data set that was used to perform a machine learning task. Rather, according to their statement, the methods can be applied to any dataset that was previously used for a task that was processed by deep learning [Weitz, 2018]. As described in the Goal of this Thesis (1.3) the contexted will be changed among this thesis.

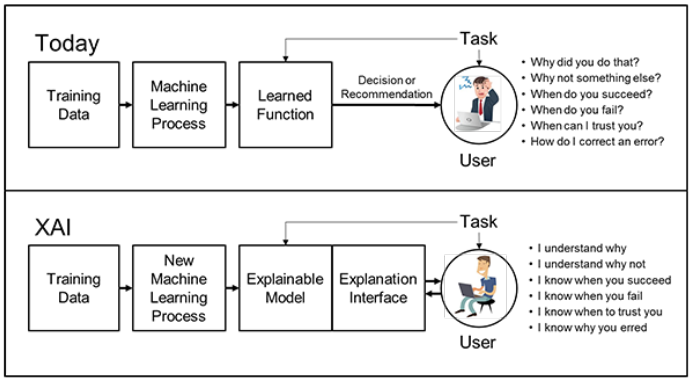

Figure (1.4): Explainable Artificial Intelligence Concept from the Defense Advanced Research Projects Agency [Robinson, 2017]

The Defence Advanced Research Projects Agency (DARPA)is an agency of the United States Department of Defence. They focus on the development of emerging technologies that the military could use. Because both the number of applications and the complexity of Machine Learning models have increased, a project has been started to specialize in Explainable Artificial Intelligence. It aims for more transparent models while maintaining a high level of prediction accuracy. Furthermore, the program wants to enable human users to understand, appropriately trust, effectively manage the output a Machine Learning Algorithm [Robinson, 2017]. Figure 1.4 shows how the Defense Advanced Research Projects Agency presents its projects as a concept. As I have also described in the Goal of this Thesis (1.3), the concept wants to move away from a model that human users cannot understand towards to a more transparent model.

There are more scientific papers on , but a complete bibliography is unfortunately not possible within this work.

Chapter 2

Background and Theory

> "Math is like water. It has a lot of difficult theories, of course, but its basic logic is very simple."
>
> - Haruki Murakami, IQ84

This chapter is a summary of all fundamental aspects of Machine Learning. More specifically, the crucial elements for an understanding of the prototype-functionality, which is presented later on. First of all, the historical development of Machine Learning will be covered briefly. As soon as the reader has an understanding of the historical development of Machine Learning, its categories will be discussed. Therefore, not every detail is covered. Only aspects and algorithms of Machine Learning which are relevant in order to talk about the topic of this thesis. In the final part of this chapter, two different approaches of Explainable Artificial Intelligence are introduced. Furthermore, a specific kind Explainable Artificial Intelligence implementation is mentioned, such that the reader can understand how predictions can be made transparent.

2.1 History of Machine Learning

An example of Machine Learning would be to take a lot of images and try to recognize cats (patterns) on them. The Machine Learning Algorithm would be able to predict - for new images in the future - whether it is a cat or not. In this example, Machine Learning can be seen as

> *"[...] a set of methods which can automatically detect patterns in data, and then use the uncovered patterns to predict future data or to perform other kinds of decision making under uncertainty" [Murphy, 2012, p. 1].*

But there is more than one definition which describes Machine Learning. Arthur Samuel was the first researcher who used this term and defined it as "Machine Learning is a field of study that gives computers the ability to learn without being explicitly programmed" [Samuel, 1959]. The computer scientist Mitchel provided a more formal definition of Machine Learning, which is quoted in hundreds of papers[1]. In his book from 1997, he says "A computer program is said to learn from experience E concerning some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E" [Mitchell, 1997].

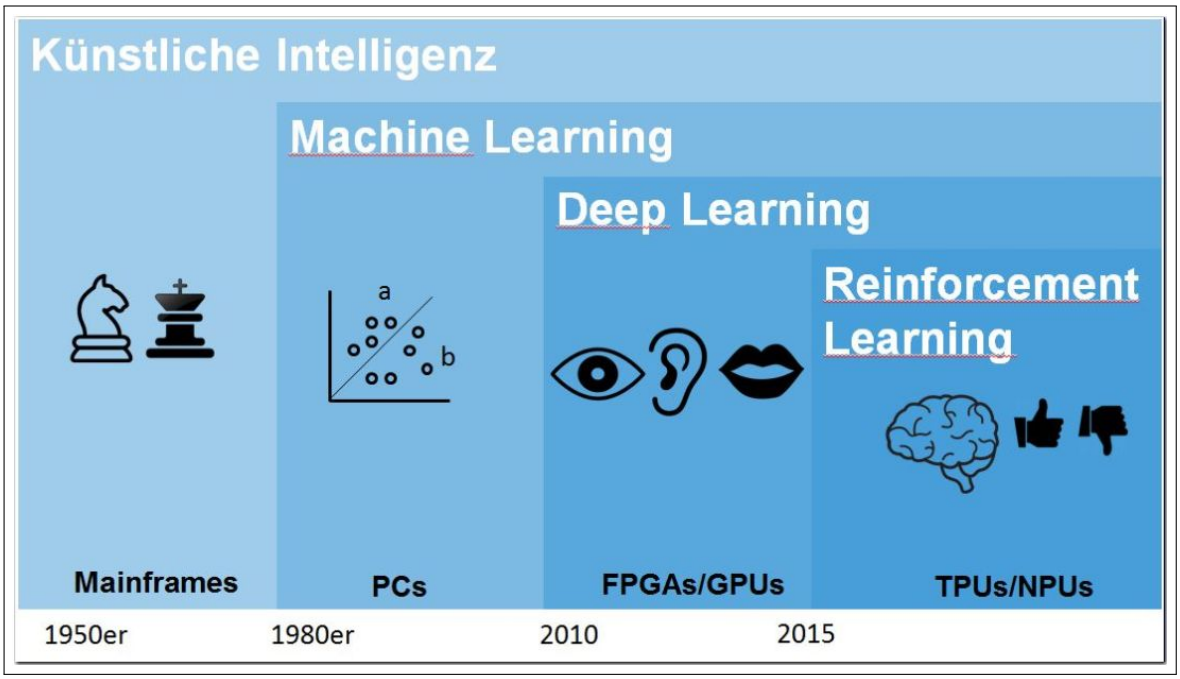

Figure (2.1): Determining critical technologies within the domains of Artifical Intelligence (extended representation by Dmitri Gross according to a presentation from Michael Copeland) [Gross, 2017] [COPELAND, 2017]

According to the science magazine, the most influential computer scientist, in the field of pattern recognition, Michael Jordan wants to give another modern definition. His objective was to give a neutral statement, which defines the word Artifical Intelligence. He says:

> *"It is one of today's rapidly growing technical fields, lying at the intersection of computer science and statistics, and at the core of artificial intelligence and data science" [Bohannon, 2016] [Jordan and Mitchell, 2015]*

Furthermore, with this definition, Jordan wants to connect the terms of Machine Learning and statistics. In his opinion, these two terms are the essential components of Artifical Intelligence [Michael Jordan, 2018]. Figure 2.1 shows how machine learning has changed

[1] 998 results on google scholar while searching for the original meaning from Mitchel (date: 24 Jan 2020)

over the last decades. It shows that machine learning became popular in the eighties, as a result of the increasing use of personal computers (PC). Furthermore, you can see deep learning - which is one of the crucial parts of my thesis - gained on popularity until 2010, caused by the development of more powerful Graphical Processing Units (GPU). Besides Figure 2.1 is criticized as well. Mainly because of the unclear definitions terminology which is used in it. Anyways it can be used to get an overview of the historical evolution of the field.

Figure (2.2): The Netflix documentary Alphago is part of the hype around Deep Learning Methods. Critics claim until the problem of transparency is not solved this results are not crucial because those models can not be used in a real-world application [Rocke, 2019]

Alpha Go, which is developed by the Google Research Laboratory, is a good example of the popularity of Deep Learning Methods. It shows how an enormous amount of time and hardware resources can lead to success as the artificial player has been the world ranked number one in the board game go. But this success is criticized as well, because it is not a real-world application and gots pushed by a Netflix Documentary, Scientist Claim that it is just an overhyped topic as long as the models could not be explained (Figure 2.2).

2.2 Suprvised Machine Learning for Upload Filters

By doing the following steps, a Cost Function (Definition 2.2.5) will be optimized to get a prediction on unseen data, as defined by Mitchel (Section 2.1). An example application is an Upload Filter which determines if a dog or a cat is on an image:

1. Eliminate empty entries and statistically irrelevant data (e.g. duplicates)
2. Put input data in the required form (defined the form of implementation)
3. Split the Data into a test and a training set (to test on unseen data)
4. Initialize the parameters of the model
5. Learn the parameters for the model by minimizing the Cost Function(Definition 2.2.5)
6. Use the learned parameters to make predictions (Definition 2.2.1))
7. Test the ability of the model to predict unseen data correctly

In the following, these steps are described while focusing on the details of the current step. The example of an Instagram Upload filter is a Classification Task which can be solved by a Supervised Machine Learning Algorithm. It is designed to learn by example (cat images) [Murphy, 2012, p. 3 - 4]. As can be seen in Figure 2.3, examples of cats and suitable counterexamples are assumed to be given before we start to design an Supervised Machine Learning Algorithm.

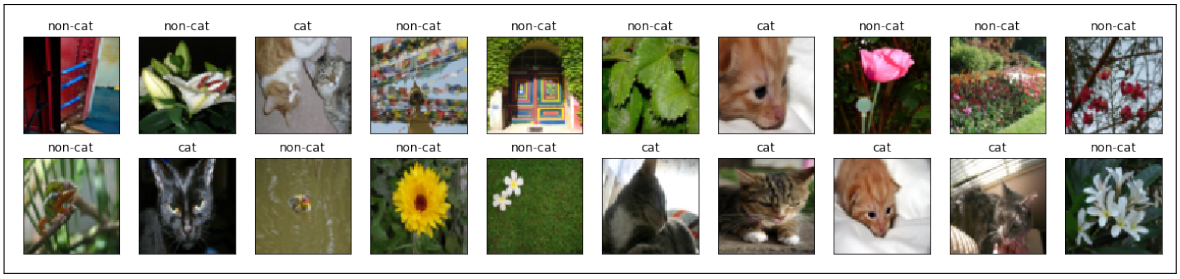

Figure (2.3): Images of detected cats, examples sampled from the from the "ImaegNet" data set[Chollet, 2016]

The name "Supervised Learning" comes from the idea that training this type of algorithm is like having a supervisor which observes the whole learning process, e.g. recognize cats on images after having seen enough examples and counterexamples. At the same time, someone leads the algorithm to the correct result [Goodfellow et al., 2016, p. 103] [Murphy, 2012, p. 3]. Technically spoken, these input data is called Training Data. They consist of input data (for example, values that represent an image) and output data (for example, the class which an image belongs to). During training, the algorithm will search for patterns in the input data that correlate with the desired outputs. After training, the supervised learning algorithm will take in new unseen inputs (cats, no cats or both). Afterwards, the algorithm determines which labels, are used to classify the original inputs, based on prior training data. Such an algorithm can be expressed as follows [Murphy, 2012, p. 3]:

Definition 2.2.1

$$f(X) = \hat{y}$$

where

f() = mapping function e.g. assigns the label cat to a cat picture
X = (several) input values e.g. images with cats
$\hat{y}$ = predicted output e.g. a label with cat or no cat

Where $\hat{y}$ is the predicted output, which is determined by a mapping function f that assigns a label to a single value or a set of multiple-input values donated by X, the function connects an input (features) to a predicted output (probability). The logic behind this function is also called machine learning model [Murphy, 2012, p. 3].

Before Machine Learning Algorithm (model) which detects cats on images gets trained, the Training Data has to be prepared. This step is called Preprocessing. Training data consists of inputs paired with the correct outputs [Brownlee, 2019]. For example, the inputs are the samples of cat images, where the outputs are the right labels of our examples. When talking about the training of a Machine Learning Algorithm (Model), the optimization of a function is always meant by that. Further, the function to be optimized is the mapping function which assigns an output value to the given input. Therefore the parameters of the function will be optimized during the training process, such that the output values match the actual output values as closely as possible. In other words, such that the real values as close as possible to the predicted values (Definition 2.2.5). Usually, that is an optimization problem and is solved with different techniques. The technique also determines the required form of the input and output data (Step 2). If we used a Logistic Regression to classify cats, a structure as can be seen in Definition 2.2.2 is necessary. Logistic Regression is a model which is applied to determine the probability of a certain class or event exists such as pass/fail, win/lose, alive/dead or healthy/sick. It can be extended to predict several classes of events such as determining whether an image contains a cat, dog, lion, etc.

Definition 2.2.2

$$X \in \mathbb{R}^{n_x \times m}$$
$$y \in \mathbb{R}^{m}$$

where

XX = an input matrix e.g. each column represents the values of an (cat) images
Y = a single row matrix e.g. each column holds a binary value (yes/ no)
n_x = number of values which represents one example e.g. pixels of an image
m = number of examples e.g. (cat) images
$\mathbb{R}$ = all values are in real number space

A pixel image of a cat, as you can see on the left-hand side in Figure 2.4, is represented as three matrices (Figure 2.4, centre). With the help of mathematical functions from the

field of linear algebra, the images are transformed from the matrix form into a vector form (Figure 2.4, right) [Brownlee, 2019, p. 276]. Even if it is not necessary, it is recommended to normalize the values. Otherwise, the calculations could become very slow and very memory intensive. [Goodfellow et al., 2016, p. 57].

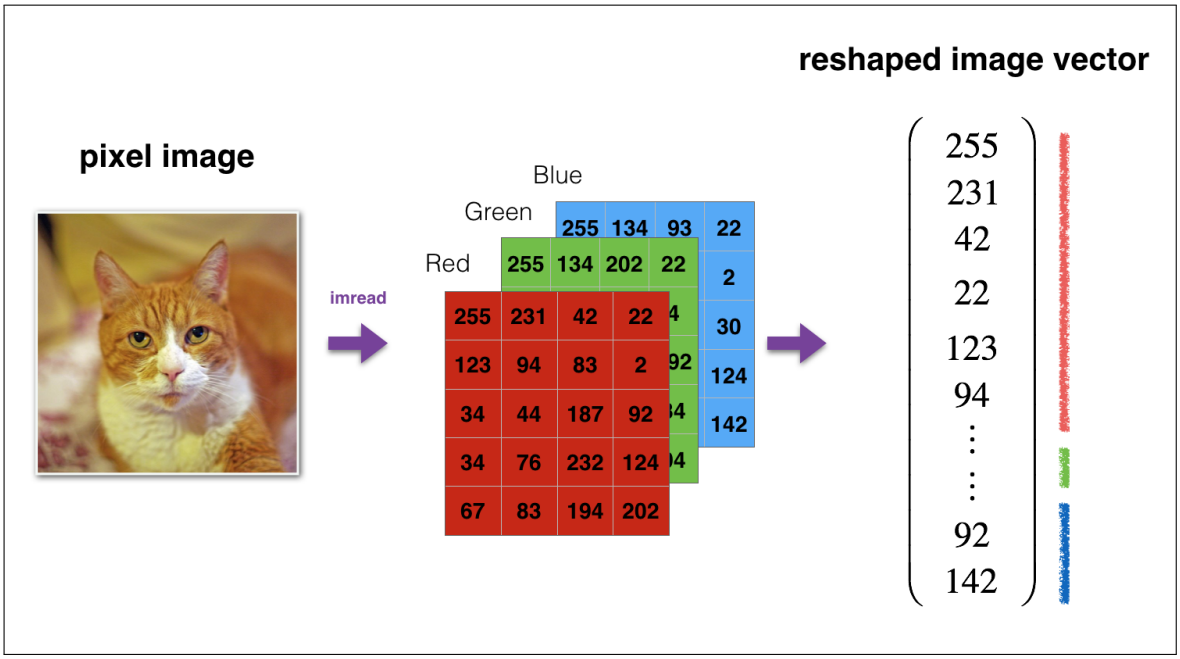

Figure (2.4): At the beginning a picture is represented by three matrices. Each for one colour channel (red, green and blue). After transforming it with linear algebra we get a vector.

Because we want to optimize the Cost Function (Definition 2.2.5) for many examples, instead of just one, we store the input in a matrix (Definition 2.2.2). After transforming a picture, we get a vector, so it is mandatory to store every single vector in the column of a matrix. Matrix representations are one of the reasons why machine learning algorithms are much more efficient because nowadays, memory is cheaper than computing power. Without matrix multiplication, more loops would be needed. Loops, on the other hand, require a relatively large amount of computing power, especially for extensive data. Matrix multiplication requires a lot of memory but requires fewer loops and therefore, less computing power. Due to the extreme amounts of data that are processed during the calculation of machine learning algorithms, these effects are multiplied by each other, which makes an efficient computation even more important [Ng, 2008].

The next step in designing a Supervised Machine Learning Algorithm(after the preprocessing is done) is to create a specific function which can be optimized. At first, a linear function which looks like the following is used:

Definition 2.2.3

$$z = WX + b$$

where

z = Results of the linear function (one for each example)
W = A matrix with parameters assigned to every row of the input matrix X
X = Matrix as a result of the preprocessing
b = Intercept added in a linear equation called Bias parameter.

The output values of the linear function can then be passed to a Sigmoid Function. Every function which is used after calculating the linear function itself is an so-called Activation Functions. They are mathematical equations to determine the output of a neural network. An Activation Functions can be used for different tasks. So the Sigmoid Function (Definition 2.2.4) is one specific example which maps a value between 0 and 1 to each output. In such a way, that this value can be interpreted as a probability (Definition 2.2.4). Finally, a label can be assigned to the input by using a rule-based approach. For example, it could be the case that if the probability is greater than 50% that a certain case will occur, a positive label will also be assigned (the probability that a cat is on the image is greater than 50%, then the image has the label cat).

Definition 2.2.4

$$\hat{y} = sigmoid(z)$$

where

z = The results of the linear function (one for each example)
$\hat{y}$ = Calculated labels (one for each example)

The next step for the Instagram Upload Filter is to get a criteria, which we can use to optimize the parameters. In other words to quantify the success of our training process. In order to get such a criteria a so-called error value is formed. This is calculated from the actual label (y)and the predicted label ($\hat{y}$). How exactly this calculation is done is determined by the loss function. For example the loss function could be defined by the cross-entropy function $\mathcal{L}$, which compares the given values with the calculated values. Every function

in machine learning that could be used to compute such an error would be called Loss Function. Finally, all error values of the individual examples are summed up and divided by the total number of examples (images). This way, we get an error function $\mathcal{J}$ which maps a single value to all errors. This error function can be optimized during the training process so that we get parameters ($\mathcal{W}$ and b) of the function which leads to a global minimum of the function. While in the first step, the so-called Forward Propagation, the error values and calculations are performed, in the second step the weights of the function will be optimized. This step uses the chain rule to form the gradient of the weights and adjust them according to this gradient. In the literature, the second step is commonly called Backward Propagation. I will go into this procedure in more detail in section Gradient Decent and the Gradient (2.3). Because it shows the overall cost in Machine Learning Literature, the function which sums up every loss of each input is name Cost Function:

Definition 2.2.5

$$\mathcal{J} = \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}(\hat{y}^{(i)}, y^{(i)})$$

where

m = Number of examples e.g. number of cat and no cat images
$\mathcal{L}$ = Cross-Entropy function
$\hat{y}^{\text{i}}$ = Calculated label for the i^{th} example
y^{i} = Correct label for the i^{th} example
$\mathcal{J}$ = Cost Function which can be optimized s. t. it is at its global minimum

The Cost Function should not be optimized as close as possible to the training data. A function which would fit perfectly is called over-fitted.

Overfitting means models perform well on the training data but don't generalize well for new data. It happens when the model is too complex relative to the amount and noisiness of the training data. So, how do we know the parameters are overfitted? If a test against the training set gets excellent results but, a test against the test set has very low accuracy, t model is likely overfitted. So, it's time to take corrective measure [Goodfellow et al., 2016, p. 110]. Anyway, the goal of supervised learning as in the Instagram Upload Filter is to achieve high performance on unseen data. To do that the data has to be divided into a test and a training set. The training set is used like described before (preprocessing the data and approximate the cost function to its global minimum). In contrast, the test data set would only be preprocessed and then used to make predictions about unseen cat images. So it is possible to test the neural network of its ability to predict unseen data (History of Machine Learning (2.1)).

In other words, every step is necessary to train a machine-learning algorithm to predict if it is a cat or not. In figure 2.5 where

- $\boldsymbol{x}_{\mathrm{x}}$ is x^{th} pixel of an image representation,
- $\boldsymbol{W}_{\mathrm{x}}$ is x^{th} parameter assigned to each pixel ofx^{th},
- $\mathbf{W\ X + b}$ is the linear function which computes a first output and
- $\boldsymbol{\sigma}$ is the sigmoid function

is the process of predicting an 64×64 image visualized.

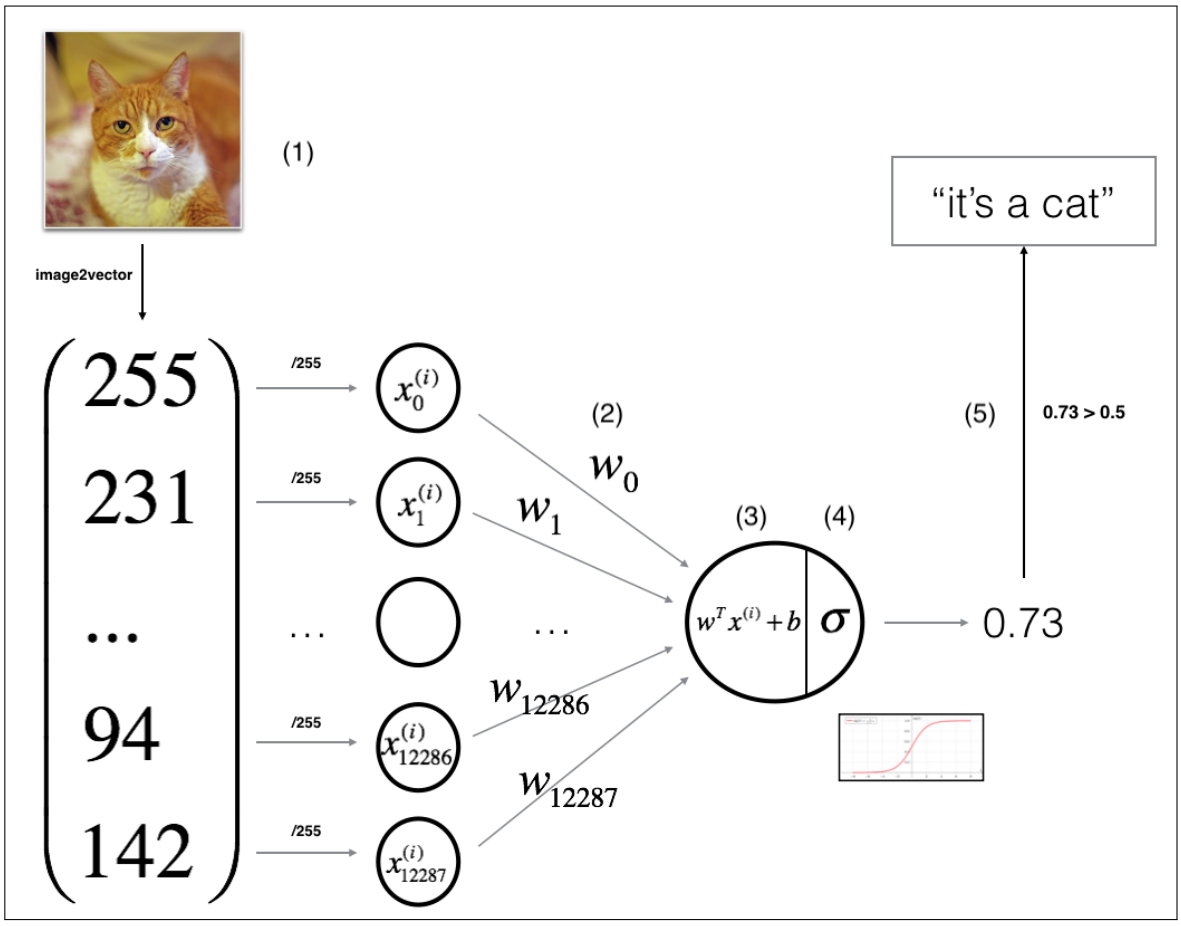

Figure (2.5): The image visualizes the whole process of using an image (1) and putting this into a single neuron (4). First, the neuron calculates a linear function and uses the result as input to a sigmoid function (becomes an interpretable value between zero and one). The network can make a prediction (5) while using a decision boundary (e.g. if the probability is higher as 0.5 that it is a cat) if the forecast is not correct, the parameters of the function (2, 3) getting adjusted as long as the function is optimized. Finally, we hat an approximation that fits (as close as possible) to all training examples.

An algorithm that works like this can be considered as simple neural network [Britz, 2015]. The term neural came from the fact that at first scientists tried to recreate the functionality of neurons in the human brain. Besides the early beginning, neural networks haven't much in common with the brain because actually, it seems the brain is much more complicated as it seems before [Kriesel, 2007]. However, it is called a network because usually different neurons working together. In the simple scenario from Figure 2.5 we had just one neuron (Figure 2.6)

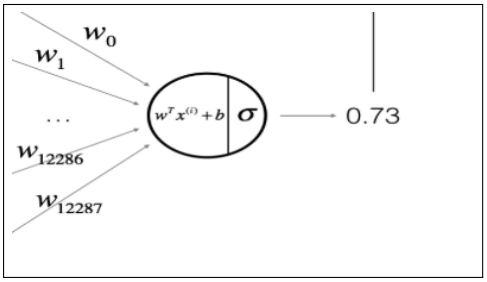

Figure (2.6): An Example for a single neuron. First, a linear the output will be calculated by a simple linear function with the parameters W and b. Afterwards the output will be normalized to get probabilities (values between 0 and 1).

For the Instagram Upload Filter, such a Artificial Neural Network[2] achieves a test accuracy of about 70% (Listing 2.1).

```
1  Cost after iteration 0: 0.693147
2  Cost after iteration 1000: 0.214820
3  Cost after iteration 1100: 0.203078
4  Cost after iteration 1200: 0.192544
5  Cost after iteration 1300: 0.183033
6  Cost after iteration 1400: 0.174399
7  Cost after iteration 1500: 0.166521
8  Cost after iteration 1600: 0.159305
9  Cost after iteration 1700: 0.152667
10 Cost after iteration 1800: 0.146542
11 Cost after iteration 1900: 0.140872
12 train accuracy: 99.04306220095694 %
13 test accuracy: 70.0 %
```

Code Listing (2.1): Test accuracy is 70% after iteration 2000 times and using 209 examples with 12287 features (64×64 pixels). This is not state of the art but very good if considering that this is a linear classifier on a high dimensional feature space.

[2]The achieved accuracy is not always precisely the same, because the local minimum is reached during optimization instead of the global minimum. In practice, this change is hardly relevant. Besides, the results can differ depending on the implementation. The results presented here were obtained with the programming language Python using frameworks from the field of data science.

It is crucial to achieve a higher accuracy before creating a transparent neural network which is easy to understand. To introduce more precise methods is the goal of the next section.

2.2.1 Deep Learning Neural Networks

Instead of just using one neuron to predict if an image shows a cat, the Artificial Neural Network can look like in Figure 2.7, where

- $\boldsymbol{x}_{\mathrm{x}}$ is the x_{th} pixel of an image representation,
- $\boldsymbol{a}_{[1]}$ is calculated input values from the first layer,
- $\boldsymbol{W}_{[1]}$ is the parameters assigned to each pixel of x_{x},
- $\boldsymbol{W}_{[2]}$ is the parameters assigned to each input $a_{[1]\mathrm{n}}$ input, calculated by layer one
- $\mathbf{W\ X + b}$ is the linear function which computes a first output and
- $\boldsymbol{\sigma}$ is the sigmoid function,

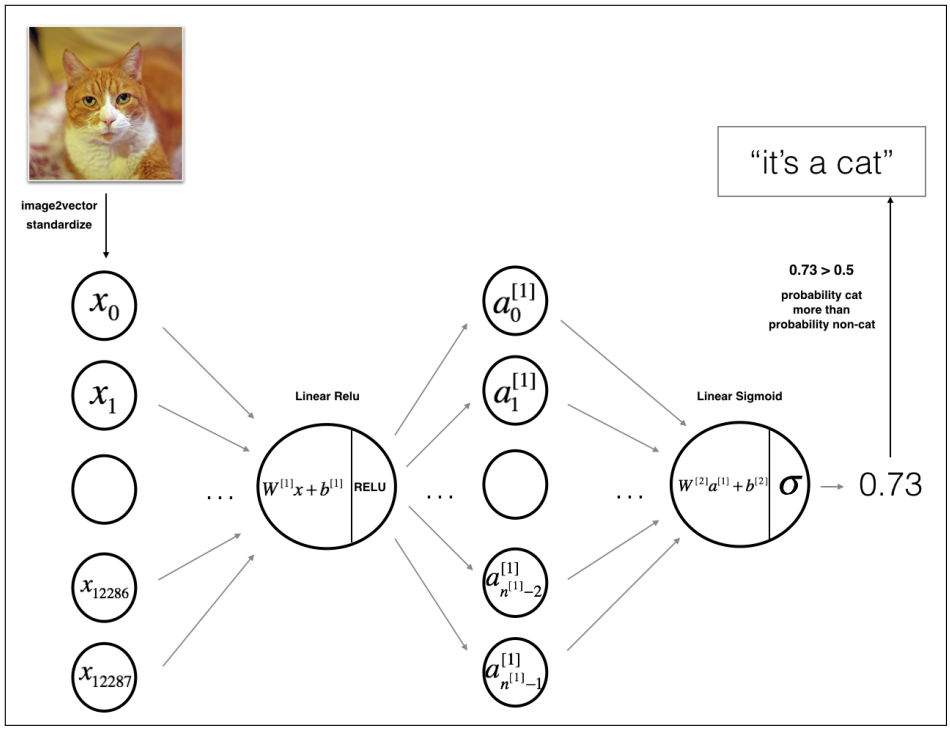

Figure (2.7): The picture visualizes the whole process of using an image (1) and putting this into two neurons (1,2). The first calculates a linear function and put the result into a function so that it can be forwarded normalized to the second neuron. The next takes the values and calculates (as before) the result of a linear function which would be made interpretable while using the sigmoid function (between zero and 1). Last but not least, the result can be interpreted so that the network can make a prediction (5). If the forecast is not correct, the parameters of the function (2, 3, 4) getting adjusted. It will be adjusted (fitted) as long as the function is optimized to match as best as possible to all training examples.

shows how the same network as before would look like if we would use two neurons (1, 3) in two layers (4,5). On the one hand, more neurons resulting in an increasing amount of parameters which the algorithm has to optimize [Goodfellow et al., 2016, p. 21]. So, for example, to find a global minimum for a Cost Function in our Instagram Upload Filter example, it would take significantly more time and requires more cat images. But on the other hand (if correctly implemented) a neural network with more neurons and multiply layers would outstand a neural network with just one neuron by far. Listing 2.2 shows that the algorithm detects up to 80% of the pictures (if it shows a cat example or not). So, the deep neuronal network, compared with a single layer neural network, has increased by 10%.

As I mentioned before, the results could be much better if more examples were used as input data. The required time and computing capacity would be comparatively low with such an implementation. Nevertheless, I will use the algorithm with more input at this place, because the given amount is sufficient to present the mentioned advantages [Goodfellow et al., 2016, p.167] [Murphy, 2012, p.995 - 997].

In general, the term is related to the number of layers. Andrew Ng suggests that neural nets with more than one layer should be called deep [Ng, 2008]. Furthermore, every layer which is not the output and not the output layer itself would be called the hidden layer. Whereas the output layer counts to the total amount of layers, the input layer is usually not considered as a so-called layer. There is not an exact definition of what a layer exactly is. Most sources are excluding the input and including the output layer. Steps in which an activation function activates the neurons do not count as independent layers. They belong to the previous layer (to which the neuron also belongs which input this function receives to standardize it)[Ng, 2008] [Kriesel, 2007] [Goodfellow et al., 2016].

A problem with the neural network, which is not deep (introduced above) would be its capability of training with larger images. So, the network was built to recognize patterns on images with 1288 input features ((64 $\times$ 64 pixels). Suppose the input is a (300 $\times$ 300 RGB image, the first layer of the network has 100 neurons, and each one is fully connected (fully connected layer) to the input. Fully connected means that each neuron in the previous layer is connected to each node in the following layer.The number of parameters which the training process has to optimize would be calculated with the following formula [Vasudev, 2019][Trask et al., 2015].

```
...
Cost after iteration 1600: 0.148214
Cost after iteration 1700: 0.137775
Cost after iteration 1800: 0.129740
Cost after iteration 1900: 0.121225
Cost after iteration 2000: 0.113821
Cost after iteration 2100: 0.107839
Cost after iteration 2200: 0.102855
Cost after iteration 2300: 0.100897
Cost after iteration 2400: 0.092878
train accuracy: 98.5645933014 %
test accuracy: 80.0 %
```

Code Listing (2.2): Test accuracy is 80% after iteration 2400 times and using 209 examples with 12288 features (64×64 pixels). This is not state of the art but very good if considering that this is an algorihm wich is not spezialised to recoginize images.

Definition 2.2.6

$$W_{\text{ff}} = F_{\text{-1}} \times F$$
$$B_{\text{ff}} = F$$
$$P_{\text{ff}} = W_{\text{ff}} \times + B^{\text{ff}}$$

where

W_{ff} = Number of weights of a FC Layer which is connected to an FC Layer
B_{ff} = Number of biases of a FC Layer which is connected to an FC Layer
P_{ff} = Number of parameters of a FC Layer which is connected to an FC Layer
F = Number of neurons in the FC Layer
$F_{\text{-1}}$ = Number of neurons in the previous FC Layer

In the equation above, $F_{-1} \times F$is the total number of weights (connections between layers) from neurons of the previous fully connected layer, to the neurons of the current fully connected layer. The total number of biases is the same as the number of neurons (F). So, for the example given above (300×300 pixels, 100 Layers in the first hidden layer, RGB and fully connected) the calculation would look like this:

$$W_{\text{ff}} = 270.000 \times 100 = 27.000.000$$
$$B_{\text{ff}} = 100$$
$$P_{\text{ff}} = W_{\text{ff}} \times + B_{\text{ff}} = 27.000.000 + 100 = 27.000.100$$
$$F := 100$$
$$F_{\text{-1}} = 270.000 \text{ (300 x 300 pixels x 3 colour channels)}$$

If the picture gets even bigger and the layers get more, you can't optimize the weights with standard hardware, because if you look at the calculation it becomes clear that the number of parameters is much bigger. As a guideline, there are about 14.000.000 parameters where it is still possible to train without special hardware (e.g. on your own computer). Everything above this is difficult to realize on a personal computer without any hardware adjustments. Besides, the models could not predict in any case in a satisfactory time [Trask et al., 2015]. Instead of the introduced neural network, there are particular implementations of deep neural networks that can calculate a forecast more efficiently.

The subject of the next section is to determine how to achieve excellent results in a reasonable time, even if the input image is greater than or equal to 300×300 pixels.

2.2.2 Convolutional Neural Networks

The most popular deep learning models leveraged for computer vision problems are convolutional neural networks. To present the Convolutional Neural Networks (which will be made transparent), the example given at the beginning of this chapter will be changed. Instead of cats, the network shall now recognize ten different digits which are simulated with the hands as you can see in Figure 2.9, where:

- $\mathbf{y}$ is a vector with the length of the possible outputs, e.g. five-digit imitating means the length of a single output vector would be five

- $\boldsymbol{y}_{\text{i}}$ is a binary value with determines if the input belongs to a class on the i^{th} position within the vector

This is a more realistic case of an upload filter because similar to this, the "Hiterlergruß" gesture or a another forbidden gesticulation can be identified. Even if the strength of a

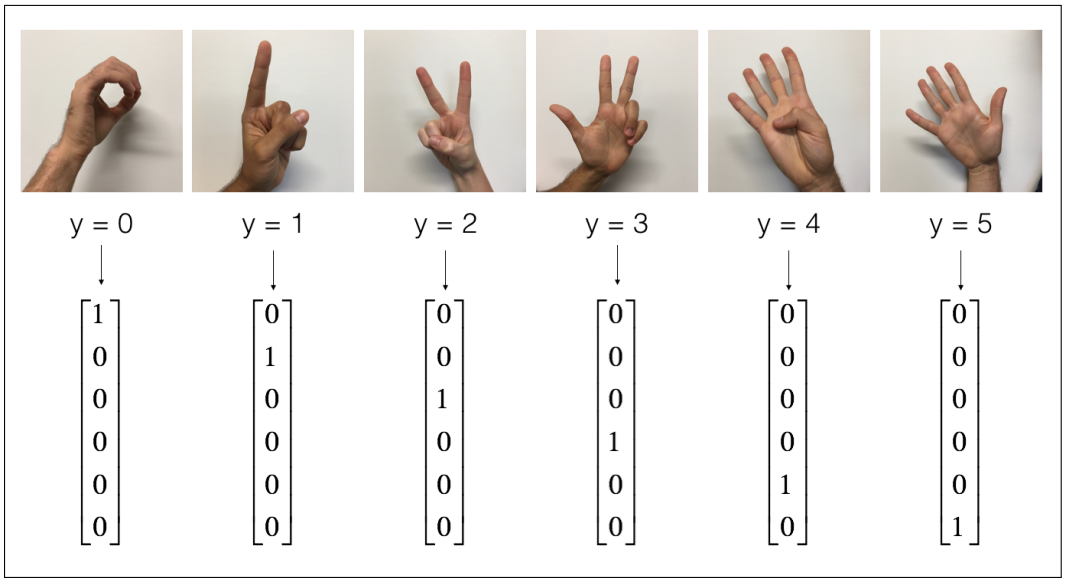

Figure (2.8): Examples of gestures which are imitating digits

convolutional neural network is to recognize images with high dimensions, the dimensions will be low as in the example before. The calculation with inputs of higher dimensions takes to long such that it can be trained along this thesis. [Trask et al., 2015]

Before I used a so-called densely (entirely, fully) connected neural network (Figure 2.7). The network consists of a certain amount of neurons which are arranged in different layers (Figure 2.5). Each neuron is fully connected with the neurons in the previous layer. To see how complex such a network can be, I have visualized a neural network in Figure 2.9 . Here you can see that such a network can become complex. Consider that each connection is represented by a weight. This weight has to be initialized and adjusted. If it comes to understand when and why which weight was adjusted to ensure transparency, it becomes clear that such a network can be a confusing issue.

More neurons would result in a high amount of parameters if the picture has too many dimensions [Goodfellow et al., 2016, p. 324]. So, an alternative approach would be to use the mathematical operation of convolution [3]. The convolution makes it possible to detect patterns like edges which could be used to classify an image. If the pictures have edges (patters) which are similar to the kind of edges from another image, the probability is high that it is in the same class. For example, if the algorithms detect a set of edges which are typically for cat ears, the algorithm will probably "think" this picture is a cat. Other cases how extracted patters while using filters can look like you can see in the following:

[3] Technically I skip the narrowing operation, so this would be a cross-correlation instead of convolution. Still, as in literature and by convention, I call this a convolutional operation anyway[Ng, 2008]

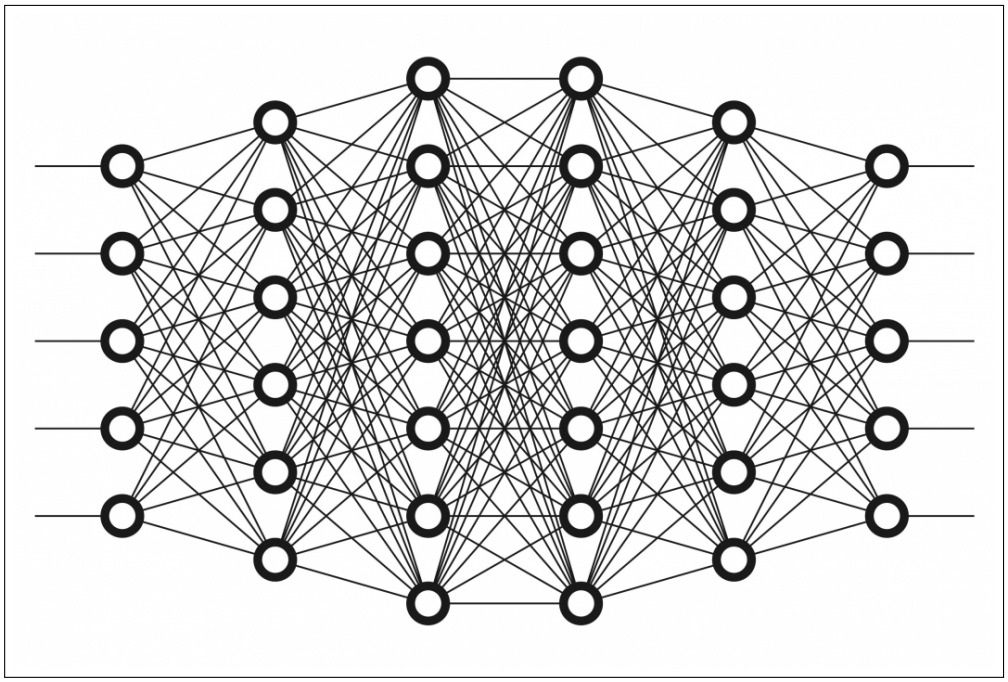

Figure (2.9): Examples of a fully connected neuronal network

Convolution operations are widely spread in computer vision algorithms, so it is not unique for Convolutional Neural Networks. It is a mathematical operation where a small matrix of numbers (filter, kernel, is passed through the matrix image representation. Every colour channel would need its filter. The name filter comes from the fact that it filters specific features from the input image, e.g. horizontal and vertical images. In Figure 2.11, where

- ***** is the mathematical operation of convolution
- **X** represents an image with a horizontal edge in the middle of the image
- **A** = the output matrix (feature map) which shows where the edge is[4]
- **f** = a filter for detecting horizontal edges,

you can see how a vertical edge, which is represented by a 6×6 matrix X is filtered and represented by a feature map (4×4 matrix) while using a 3×3 matrix as a filter.

For a convolution operation, a filter would be placed over a selected pixel. Afterwards, each value from the filter has to be multiplied with the corresponding values from the image.

[4]because this pictures are very small the dimensions of the edge which is shown in A are not accurate and unprofessionally. While using matrices of higher dimensions the proportions would be more accurate

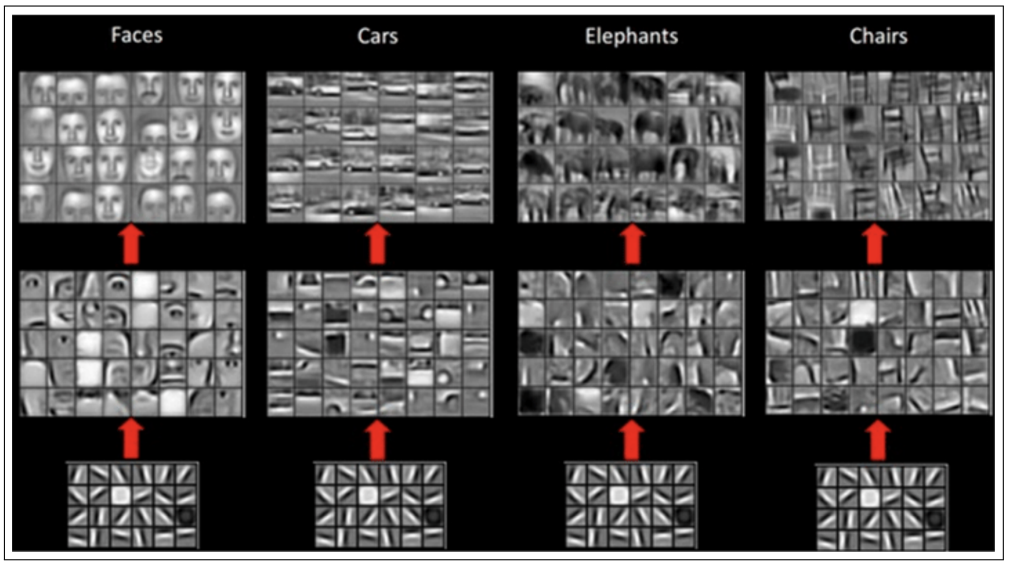

Figure (2.10): Patterns identified while using convolutional neural networks [Stewart, 2019]

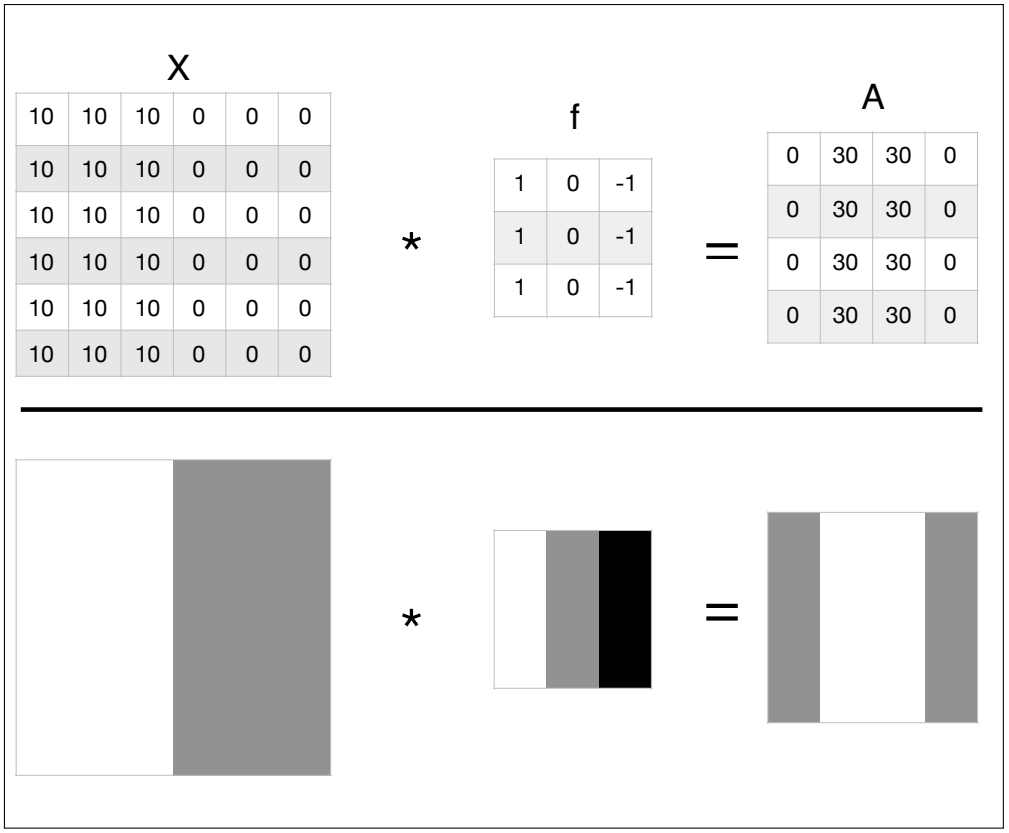

Figure (2.11): Using Convolution for edge detection

Finally, the sum would be placed in the right place in the feature map as to be shown in Figure 2.12.

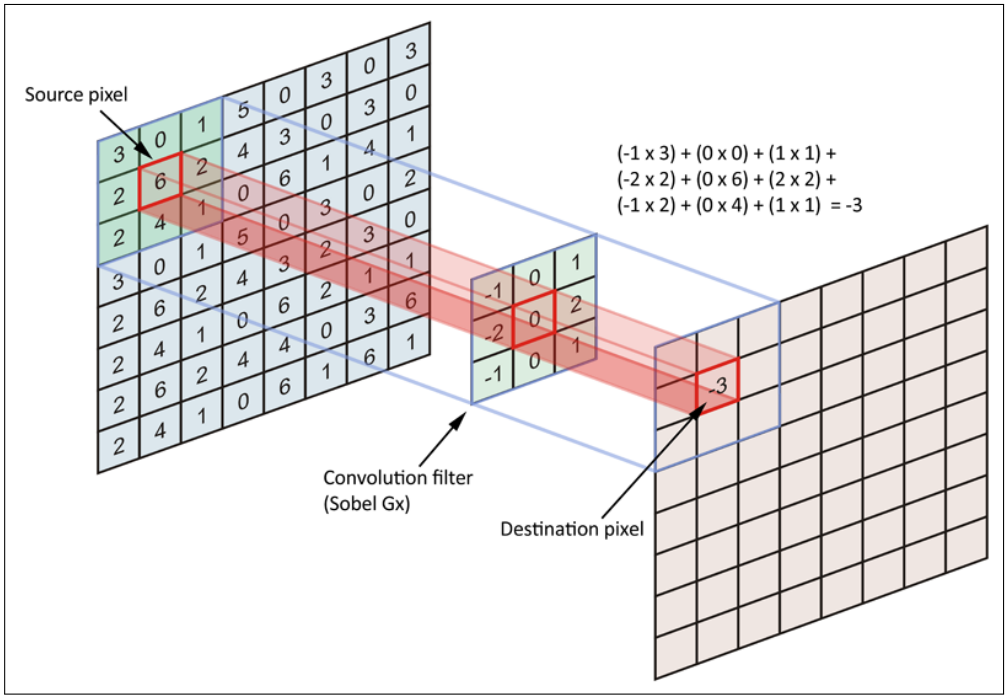

Figure (2.12): Convolution Operation over a matrix

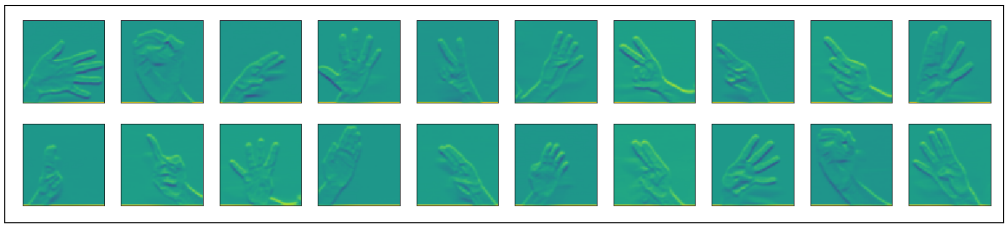

Figure (2.13): Convolution Operation on hand gestures, using a filter for horizontal edges and projecting them on the original image (without changing the size of it)

The algorithms had to use a specific filter matrix, to detect of edges of every kind (e.g. vertical and horizontal edges). Two filters are used to identify at first horizontal edges (Figure 2.13)and second vertical edges (Figure 2.14). Finally, both edges get projected to the original image and colour values were normalized to show the vertical and horizontal edges more clearly (2.15). To figure out which values (filters) should be used to detect patterns is challenging because there are almost endless opportunities. That is where the neural network could be used. Instead of setting the filter values manually, they are the parameters of a neural network. These parameters can now be optimized by using the cost function of the network (Definition 2.2.5). That means the total amount of parameters comes no longer from the size of an image it came now from the filter size [Ng, 2008])

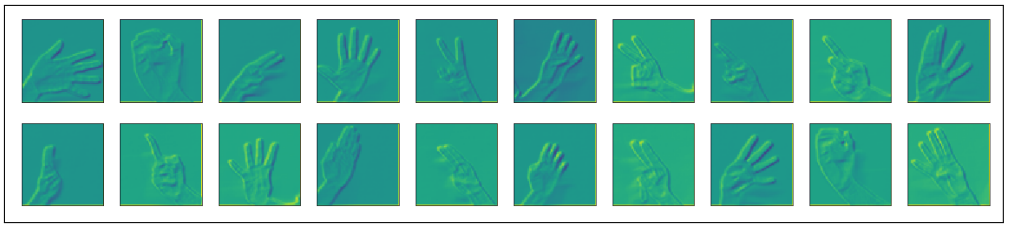

Figure (2.14): Convolution Operation on hand gestures, using a filter for horizontal edges and projecting them on the original image (without changing the size of it)

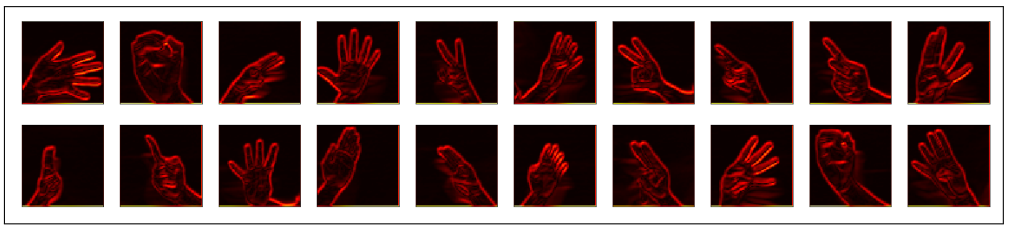

Figure (2.15): Convolution Operation on hand gestures, using a filter for horizontal and vertical edges and projecting them on the original image (without changing the size of it but while using normalization to make the edges more clear)

In Figure 2.16, you can see the architecture of such a network which we could use to identify filter values which are finally helping us to classify.

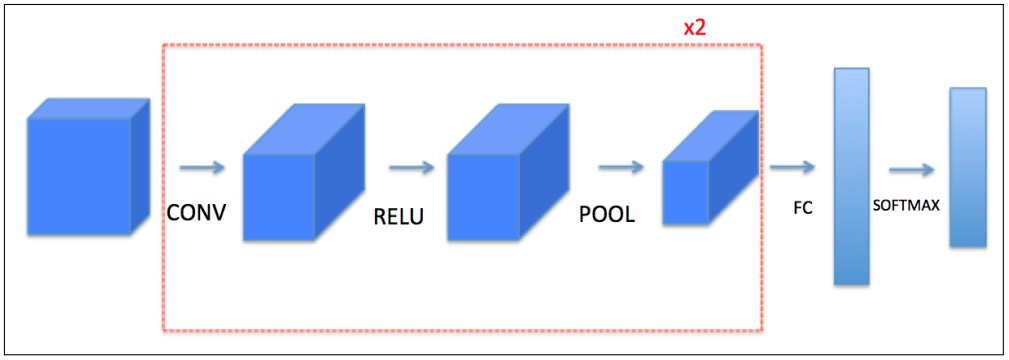

Figure (2.16): A typical architecture of convolutional neural network made from different building blocks

As could be seen in the picture, a convolutional neuronal network contains more than just convolution operations. One example is the Pooling Layer which is an Activation Function and is used to standardize values between the layers. The softmax Activation Function is used to get probabilities for each possible class. Afterwards, a decision boundary can be decided whether the image belongs to a class or not. The Pooling Layer reduces the height and width of the input. It helps minimize computations, as well as it helps to make feature detectors more invariant to its position in the input data. Typically a Pooling Layer is one of the two following types:

- Max-pooling uses another matrix which slides over the input and stores the max value of the window in the output.

```
Cost after epoch 0 =      1.917929
Cost after epoch 5 =      1.506757
Train Accuracy =    0.940741
Test Accuracy =     0.783333
```

Code Listing (2.3): Test accuracy is 80% after iterations 2400 times and using 209 examples with 12288 features (64×64 pixels). That accuracy is not state of the art but very good if considering that this is an algorithm which is not specialized to recognize images.

- Average-pooling uses another matrix which slides over the input and stores the average value in the output.

In the Figure 2.18 2.17 you can see how this would be done, where:

- **Stride** is the value which determines for how many pixels the filter get shifted each time.

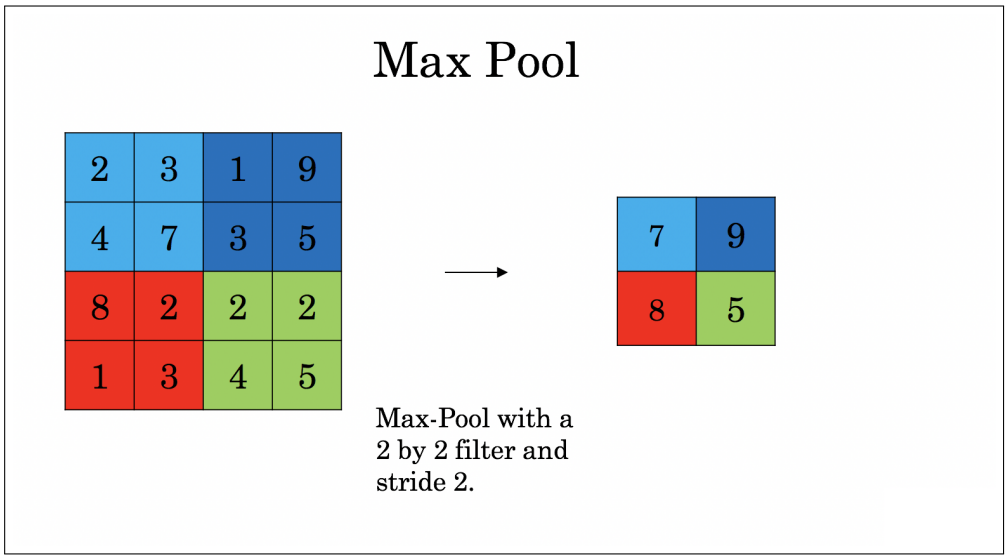

Figure (2.17): Maximum pooling, or max pooling, is a pooling operation that calculates the maximum, or largest, value in each patch of each feature map. The results are down sampled or pooled feature maps that highlight the most present feature in the patch, not the average presence of the feature in the case of average pooling [].

Using this kind of network with a dataset of ten different classes, the accuracy is close to 80%[5].

The state of the art results in image recognition, e.g. on the cifar10[6] dataset, are impressive examples of how this technique has increased the performance of machine learning in the last year. They were achieve results of 90% and can be calculated in a pleasing amount of

[5]The result could be even better if we used more data for the training process and a bigger network so that the difference to a usual Artificial Neural Network would be more outsanding

[6]CIFAR-10 is an established computer-vision dataset used for object recognition. It is a subset of the 80 million tiny images dataset and consists of 60,000 32×32 colour images containing one of 10 object classes, with 6000 images per class. Alex Krizhevsky, Vinod Nair collected it, and Geoffrey Hinton.

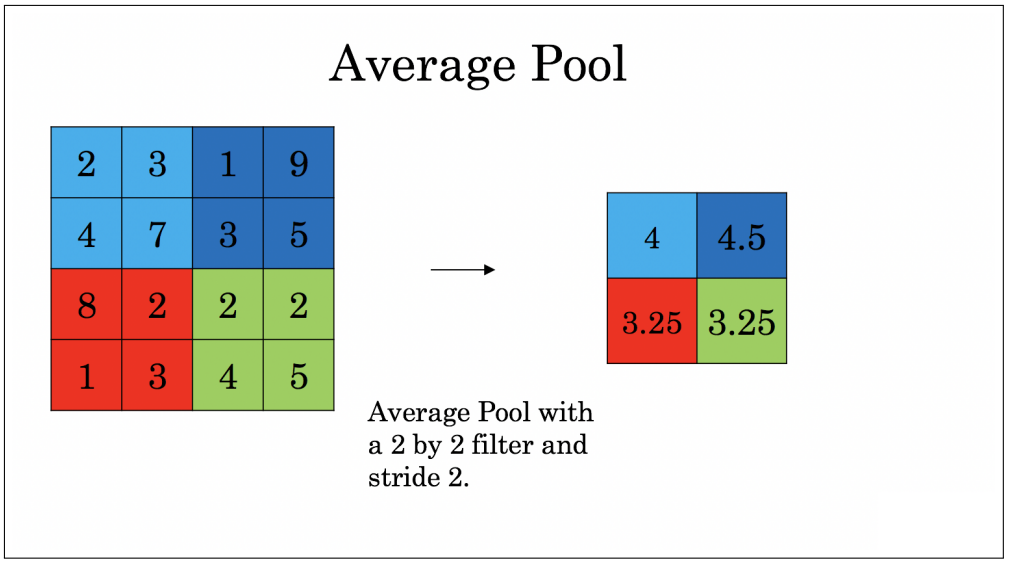

Figure (2.18): Average pooling involves calculating the average for each patch of the feature map. This means that each 2×2 square of the feature map is down sampled to the average value in the square.

time.

However, because of their non-linear structure, convolutional neural network algorithms with outstanding overall performance are usually seen as a black box. That means no information is provided about what exactly lead the networks to their conclusion. Transferred to the Upload Filter example, this would imply that 10% of the users are blocked more falsely during the uploading. The user could not be informed about why the algorithm came to its decision.

The goal of the next section is to present the theory of different methods of how a convolutional neural network is no longer a black box but a transparent algorithm. In other words, a decision support system from which the user can be informed of why the algorithm came to a specific decision.

2.3 Explainable Artificial Intelligence

Most Deep Learning Models today are a black-box. Especially no counteraction is set to make them transparent. To in non-transparent models, techniques have arisen to understand feature importance. That means for a given prediction, how important is each input feature value to that prediction? Before two of these techniques are explained in detail, why this is important is summarized.

Since Deep Learning has become more and more successful, science is also increasingly concerned with how to make Deep Learning models more transparent. The reason for that is that for business leaders, machine learning engineers and the users who are confronted by the results produced by machine learning algorithms, it is essential to know why a decision was made. Unless we were not able to learn from the model and they will become a black box [Michael Jordan, 2018] [Kuang, 2017].

In Figure 2.19 you can see what the difference between a traditional explanation approach within a Machine Learning System and an Explainable Artificial Intelligence explanation is. Whereas the traditional approach focuses on the output itself (e.g. this is a cat) the Explainable Artificial Intelligence approach combines some features to an explanation which can be understood by humans (e.g. has fur, whiskers and claws).

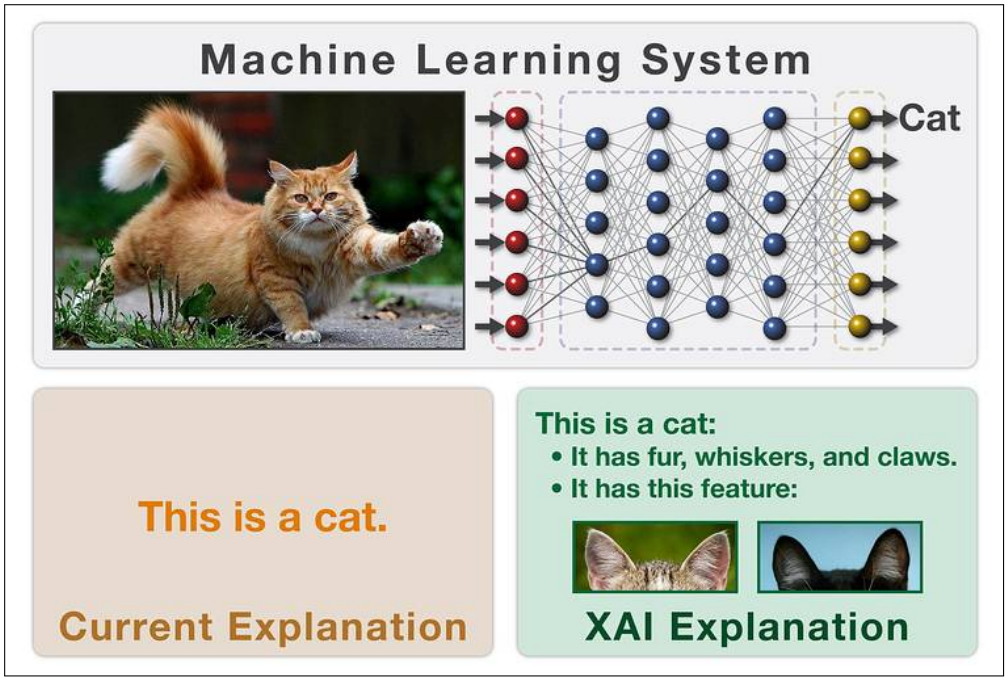

Figure (2.19): This is a cat, but how do you know? Explainable Artificial Intelligence seeks to develop systems that can translate complex algorithmic decision-making into language humans can understand.[Robinson, 2017]

Managers with a high responsibility for there department find it challenging to use these complex algorithms. Although they are aware that the results could increase the success of their company, they are afraid of using these models, because they are not transparent. If something goes wrong, they can hardly explain why the algorithm made the wrong decision. The results created by deep learning in image recognition could be beneficial for the healthcare sector as well, but if it is unclear why decisions were made, doctors will not give those approach any chance. As well as the Managers doctors are responsible for their decisions,

and if they get supported by an algorithm, they must understand the algorithm in the first place. Otherwise, they could not trust them, because they would not be able to distinguish between a right and a wrong decision.

A lack of transparency is responsible for the rejection of deep learning models in the finance industry as well. Especially since the last big financial crisis in 2009 banks und financial companies must make their decisions transparent to different stakeholders. And as already introduced in my introduction, small business owners could be affected by semitransparent models as well. If they run their business with the help of social media, it would harm them if their uploads get rejected automatically, without even knowing why this decision would be made.

More specifically a small shop owner wants to upload a picture which shows him, introducing a new product. The algorithm was trained to detect the "Hitlergruß", and so he predicted the user performs this gesture. So, Explainable Artificial Intelligence would help to explain automatically why this prediction would be made. Thus, the shop owner could change the crucial factors immediately.

Now I will dive deeper into two techniques which could be used in an upload filter. This could help the shop owner from the example given in the paragraph above. These two techniques are [Kuang, 2017]:

1. Shapely Additive exPlanations (SHAP) and
2. Integrated Gradients (IG)

The Shapley Additive exPlanations and the Integrated Gradients (IG) belong to two different categories for Explainable Artificial Intelligence.

1. Shapley-value-based algorithms and
2. gradient-based algorithms

Two factors can easily distinguish these models. Firstly, the assumptions and secondly, the underlying mathematical principles used in the models. Before I am going to explain how they differ, I present the fundamentals of categories. Next chapters focus on the question which approaches used in the prototype, based on the requirements for the prototype [Michael Jordan, 2018] [Kuang, 2017].

The two fundamentals of the shapely-value-based algorithms and gradient-based algorithms are:

1. Shapley Values
2. The Gradient

Shapley Values

Let assume that the algorithm behind an upload filter predicts the price for a painting. For a particular painting, it predicts € 300,000, and the prediction will be explained through shapely values. The art has an age of 50 years, is part of a private collection, is not mentioned in literature and is in good shape (Figure 2.21).

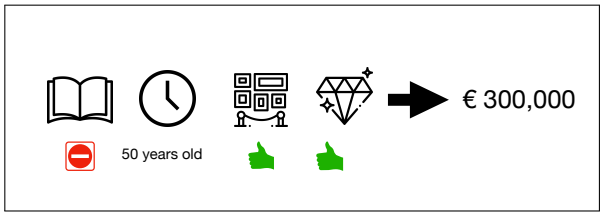

Figure (2.20): A concrete visualization of the features from painting for which the hapley values will be calculated.The painting has an age of 50 years, is part of a private collection, is not mentioned in literature and and is in a good shape.

The average prediction for all paintings is €310,000. Now the question "How much has each feature value contributed to the prediction compared to the average prediction?" For that, quantitative criteria will be used.

For a linear regression model, the answer is quite simple. The attribute of each feature is the weight of the feature times the feature value. It works because linear regression is a linear model. So, to make more complex models (e.g. deep learning, convolutional neural networks) transparent, a different solution is required. The Shapley value, coined by Shapley (1953), is a method for assigning payouts to players depending on their contribution to the total payout. Players cooperate in a coalition and receive a certain profit from this cooperation [Aas et al., 2019] [Lundberg and Lee, 2017] [Lundberg et al., 2018].

As we had not a real "game", the "game" is the prediction task for a single instance of the dataset. The "gain" is the actual prediction for this instance, minus the average prediction for all instances. The "players" are the feature values of the instance that collaborate to receive the gain (= predict the price for the painting). In the example, the feature values "Not mentioned in the literature, "painting is in good shape", "painting is 50 years old" and "painting is part of a private collection" worked together to achieve the prediction of

€300,000. Our goal is to explain the difference between the actual prediction (€300,000) and the average prediction (€310,000): a difference of -€10,000.

An answer looks like the following:

- "Not mentioned in literature" feature contributed € 30,000
- "50 years old" feature contributed € 10,000
- "Part of a private collection" feature contributed € 0
- "Paining is in a good shape" feature contributed -€ 50,000

The contributions add up to -€10,000, the final prediction minus the average predicted apartment price.

How do we calculate The Shapley value for one feature gets calculated the follows. The Shapley value is the average minimal contribution of a feature value among all possible coalitions.

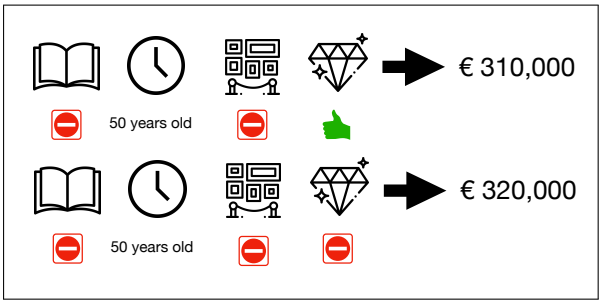

Figure (2.21): One sample in order to calculate the contribution of "is in a good shape" feature value to the prediction when added to the coalition of age, mentioned in literature and within a private collection

In the following we evaluate the contribution of the "Painting is in good shape" feature value when it is added to a coalition of "Not mentioned in literature" and "50 years old"."Not mentioned in literature", "Painting is in good shape", and 50-years old" feature values are from the given predicted instance (painting with a worth of €300,000). The feature value for whether a painting is or is not in a private collection came from a random choice among all possible paintings which are in the same distribution. So, the value "in private collection" was replaced from yes by no. If the price of the painting gets now predicted again (with this combination), the predicted price is €310,000. In a second step, the "in good shape" feature value get doped from the coalition by replacing it with a random value among all other possible paintings which can be predicted. In the example, it was not in good shape, but it could have been good shape again. The prediction of the painting price for the coalition of "mentioned in literature" and "50 years old" is €320,000. The contribution of "In Good Shape" was:

$$€ 310{,}000 - €320{,}000 = -€10.000$$

This estimate depends on the values of the randomly drawn painting that served as a “donor” for the "shape" and "private" feature values. We will get better estimates if we repeat this sampling step and average the contributions. The result is even better if the sampling step gets repeated and calculate the average of all contributions. Furthermore, it is done repeatedly computing all coalitions. Because of the repeatedly computing for every combination, the calculation time increases exponentially. One solution to keep the computation time small is to calculate contributions for only a few samples of the possible coalitions.

If we estimate the Shapley values for all feature values, we get the complete distribution of the prediction (minus the average) among the feature values.

SHalpey Additive exPlanations (shap-software-library)

If Shapley values are a method which maps a contribution to each player, a Shapley-value-based explanation method will aim to get estimated shapely values which are close to the precisely calculable ones. It is achieved because the Shapley Additive Explanations are randomly dropping out some of the features. Potentially any combination of features can be left out. As long as you look at each feature individually, there is an almost infinite number of combinations. A calculation would take too long. As a counteraction, pairs of features are formed and in the following considered as one player. The individual success of a player is then estimated. Let us look again at the example of the shop owner. If a group of pixels (subset) does not contribute to the overall result of the game (Accuracy), it will not change even if you drop out the whole group [Lundberg et al., 2018] [Molnar, 2019] [Lundberg and Lee, 2017].

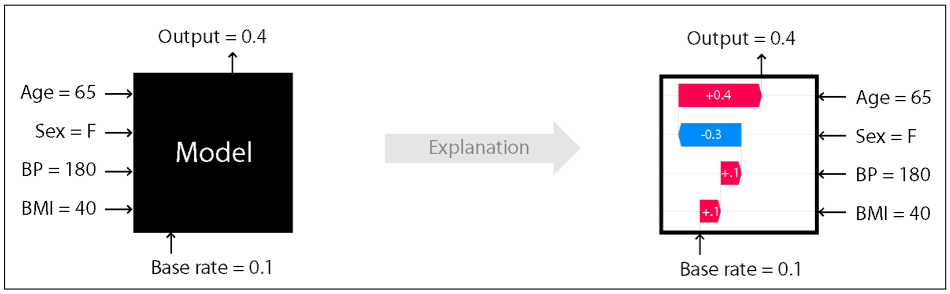

Figure (2.22): The Header of the SHAP Github Repoitory shows an example of a Machine Learning Algorithm is explained through the library. It takes a model with age, sex, bp and bmi as feature examples and maps an attribute to this values. Each values showsthe average marginal contribution of a feature value across all possible coalitions of features.

Figure 2.22 shows an easy example of how the shap library explains the feature importance with a Machine Learning Algorithm. The colours help to get an intuitive understanding of which features is more important, so red stand for significant contribution to the overall prediction (on average) there as blue is less important to it.

Gradient and Gradient Decent

Even dough I said before I will not go into details about optimization I have to give some theory about the gradient and the gradient descent if it comes to a Explainable Artificial Intelligence method which is built on top of the gradient and the gradient descent.

Gradient Descent is possibly the most used optimization algorithm in the field of deep learning and machine learning. If there would be a cost function as introduced in the section about Machine Learning (e.g. which maps a value of how good a model can predict if an image belongs to the class "Hiterlergruß"). The goal would be to find the optimum for every weight in the model. So, the cost value would be as small as possible (value determines the percentage of failure) as long as the value is not adapting to a specific set of examples (remembering, overfitting). The gradient descent is an algorithm that optimizes the cost value by making changes to the weights. The difference is determined by the gradient, which shows the direction of the deepest decent. In other words, it is a value which determines a direction which minimizes the cost as few Backward Propagation steps as possible. So, whereas in the so-called forward propagation the actual cost would be calculated the backpropagation would use the chain rule to get the gradient at first and at second adjusts the weights by subtracting the gradient from them [Goodfellow et al., 2016].

This method is quite old, and one of the reasons why this method got popular in the last years was the increase in computation power. Until a few years, a new generation of computational processing units and graphical procession units are fast enough to work on such tasks efficiently.

In Figure 2.23, you can see the graph of a cost function in three-dimensional space. The line shows the line from a starting point by going down to a local minimum. It is crucial to know that gradient descent can not distinguish between a local and a global minimum. Therefore it would need different steps of gradient descent. In practice, this is not important because it works for most cases just fine enough. For a field where very high accuracy is a must, machine learning would be the wrong approach anyway. Therefore a logic-based attempt would lead to a state of the art results [Russell and Norvig, 2009] [Kohlhase et al., 2017].

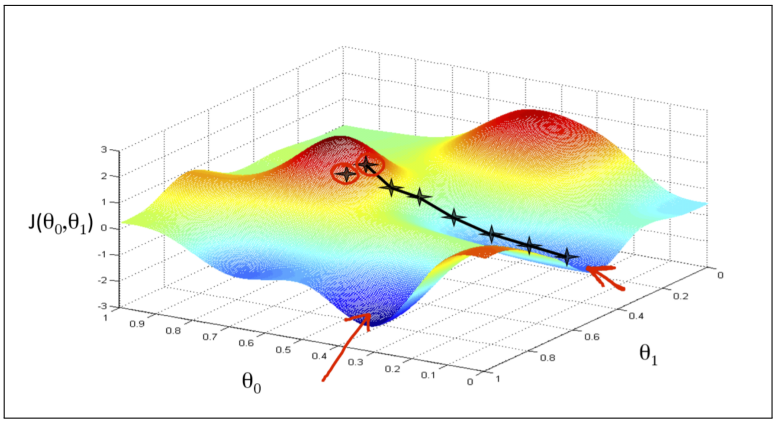

Figure (2.23): Shows the path to a local minimum while using the parameters gradients to compute the path [Wikipedia, 2020]

2.3.1 Integated Gradient

A gradient-based explanation method tries to explain a given prediction by using the gradient of the output concerning the input features. The gradient is not only used here to optimize the weights of the parameters but also to see which features have been adjusted the most and are therefore most important.

The goal is to know how a decision was made. So it can be concluded afterwards why this decision was made, so the gradient is used again. As already mentioned, the gradient points (for each parameter) in the direction of the next local minimum. So if we know which parameters are adjusted more than others, we can make a statement which parameters are more important than others. Since each parameter is assigned to an input value, it can be concluded how significant this value is to reach the desired minimum as fast as possible. As also mentioned above, an efficient implementation of Machine Learning Algorithm is essential to save computing power. An exact calculation costs an enormous amount of resources, which is why the gradients are not calculated in this step but estimated [Molnar, 2019] [Tjoa and Guan, 2019] [Sundararajan et al., 2017].

Figure 2.24 shows three paths between a baseline (r1 , r2) and an input (s1, s2). Path P2, used by Integrated Gradients, simultaneously moves all features from off to on. Path P1 moves along the edges, turning features on in sequence. Other paths like P1 along different edges correspond to different sequences. SHAP computes the expected attribution over all such edge paths like P1.

The Integrated Gradient tries to estimate Aumann-Shapley values which are as close as possible to the calculated values. The Integrated Gradient works by assuming there is a

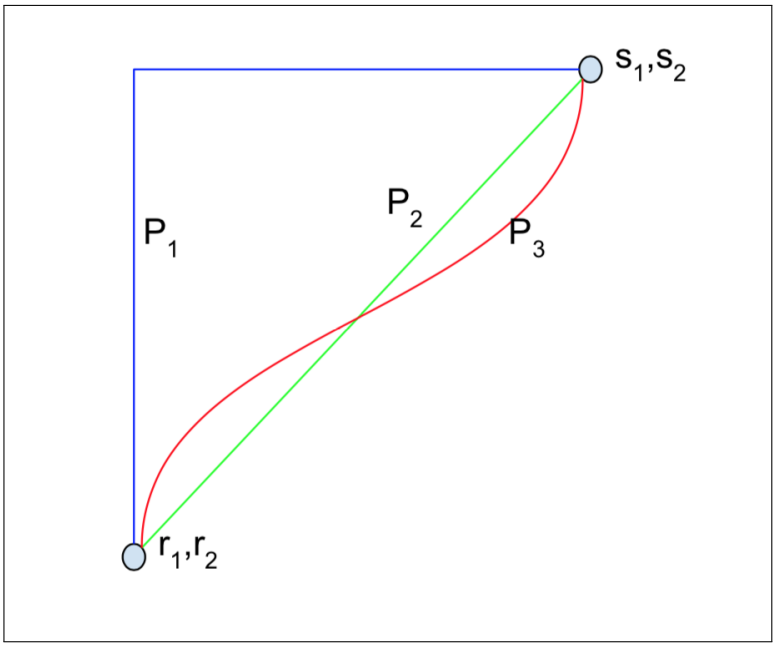

Figure (2.24): Shows different paths between a baseline and an input to compare it the Integrated Gradient (P2) with the SHalpey Additive exPlanations[Tjoa and Guan, 2019]

straight line, from the actual input (e.g. the image of a shop owner which tires of uploading it to Instagram) to a specific baseline input (e.g. a black image). The gradient of the prediction concerning input features would be integrated along this path [Sundararajan et al., 2017]. Integrated Gradient is also a method where the input varies along a straight line between the baseline and the input. At the same time, the prediction moves from uncertainty to certainty (the final result, e.g. for 98% the image shows the "Hiterlergruß"). At each point on this path, the gradient is used to attribute the change in the prediction probability back to the input features. So, Integrated Gradient aggregates these gradients along the using the paths integral [Mudrakarta et al., 2018]. Figure 2.28 visualizes who this method would look like after applying it to a few examples. To make it more clear how this would be done, it now it will be described it step by step :

1. Choose any image as a baseline (e.g. black picture with each pixel 0)
2. Make images brighter as long as they become the input again (Figure 2.25)
3. Compare the final output (as moving from certainty to uncertainty) to the path of the images (from the black baseline to the input, Figure 2.26)
4. We want to know then the slope of the score vs intensity graph doesn't remain stagnant (Interesting Gradients)
5. The Input pictures get changed such that the interesting gradients can be seen in there (Figure 2.27)

Figure (2.25): Three different steps along the path from the baseline to the input

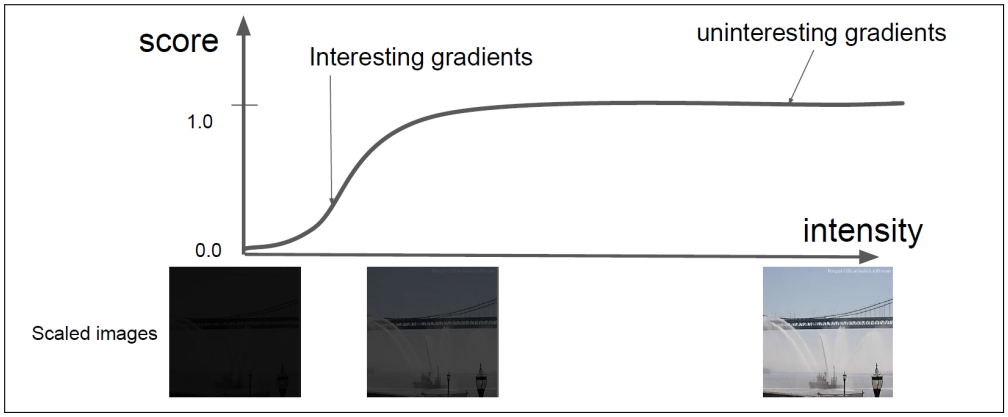

Figure (2.26): compression from the two paths (along the result and along the baseline to output)

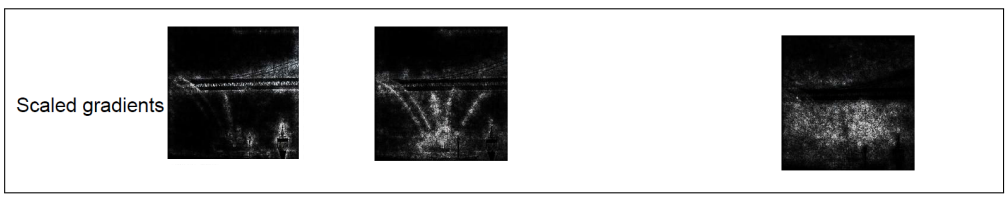

Figure (2.27): result of the Integrated Gradient

As the Integrated Gradient approximates Aumann-Sapley values the function which estimates the values must be a piecewise differentiable function of the input features (compare Shapley Values).

Because the Method should make sensible feature attributions to chose a suited baseline is crucial. For example, if a black image is selected as a baseline, integrated Gradient would not choose attribute importance to a completely black pixel in an actual image. If black pixels are not necessary, e.g. because just the frame of the shop owners image is black and the rest of the image is only bright, this will work perfectly fine. But in general (because such an assumption could not be made in a real-world-scenario) the baseline value should both have a near-zero prediction, and also faithfully represent a complete absence of signal.

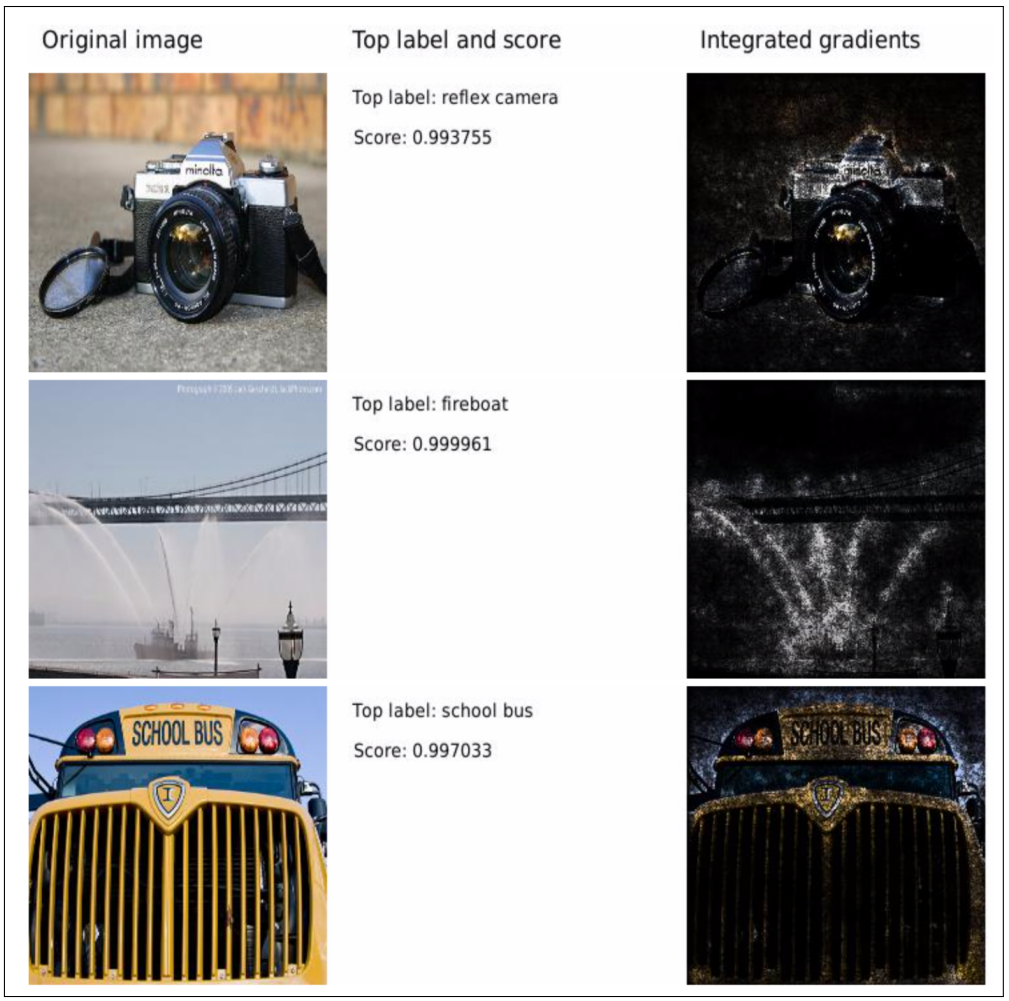

Figure (2.28): The Integrated Gradient applied to sample of images and visualized to demonstrate which pixels are important (brighter) and which pixels are less important (darker)

2.3.2 Expected Gradient Approach

The Expected Gradient Approach combines the two methods which were described above. It is useful because the two approaches above generate noisy data as can be seen later on in the prototype. While in the two methods above, one single example is used as a reference value. This approach allows using the whole dataset. It makes the resulting values easier to interpret.

It tries to combine a multitude of ideas from Integrated Gradients, SHapley Additive exPlanations (SHAP) and libraries such as SHAP-Library implement some brilliant approximations and samplings. To cover up all their work fils an entries Mater Thesis with ease so this part is left out. Important to understand is the result which is provided by Software-Library such that it can be interpreted later on the course of this thesis.

An intuitive way to understand the Expected Gradient Values is following illustration: The feature values are chef which are entering a hotel kitchen step by step in random order. All chefs (feature values) in the room contributing to the lunch (= contribute to the prediction, which is predicted by a machine learning model). The Expected Gradient Values are marks for the meal which got prepared in the kitchen. This value is the average change in the grading. That mentioned change is defined as follow: Grade revived by a combination of cooks which are already in the kitchen if the actual chef is entering. More precisely, the average of all different combinations of chefs witch is in the kitchen, while the specific chef comes into the kitchen.

Figure 2.29 shows how this can be visualized on an overlay on image predictions. Especially, it can be seen which regions on the image are responsible for the predicted probability of bing part of a class (red) or not (blue).

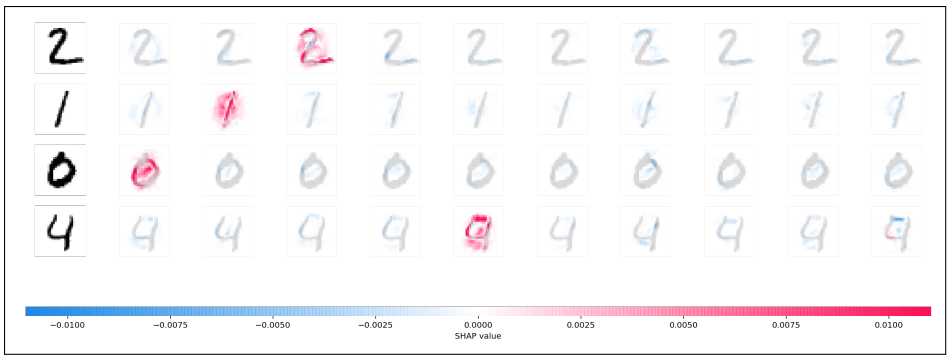

Figure (2.29): The shap applied to sample of images and visualized to demonstrate which pixels are important (red) and which pixels are less important (blue) [Slundberg, 01]

Chapter 3

Requirements

> "Big shots are only little shots who keep shooting."
>
> - Christopher Morley

Explainability is the primary cause of this thesis, and until now, the development of explainable deep neural networks has been considered as a theoretical issue (Chapter 2). Within the next few chapters, a prototype is set to become an essential part, when it comes to transform this academic issue into a real-world problem.

The previous chapters focused on systematic literature research around the following topics:

- Supervised Machine Learning (Section 2.2)
- Deep Learning (Section 2.2.1)
- Convolutional Neural Networks (section 2.2.2)
- Explainable Artificial Intelligence (Section 2.3.2)

Therefore my prototype has to be an implementation of a supervised machine learning algorithm. More precisely a deep neural network of a convolutional architecture. Moreover, the implementation of the prototype has to present the results clearly and transparently. If the prototype can not do this, an application will not help to apply the given knowledge into practice. This Section can be seen as a preliminary explanation of what a prototype has to accomplish. So, the requirements can be discussed in more detail.

3.1 Objective

The prototypes objective is to make predictions on a given input and present these predictions (output) transparently. The details of what input, output and transparency mean will be described in the following sections.

3.2 Input Data

In her cutting edge master thesis, Katarina Blandina Weitz presented a prototype to decode facial expressions of pain and emotions. For example, images of human's emotions as input were notably valuable. In order to identify the worth of explainable deep learning methods in upload filters, the subject does not matter. Besides, a limitation of a specific subject (e.g. human's emotions) is to use the results for a generalization. This thesis uses a variation of Weitz input. In fact, the subject of the images will change, but the specifications are similar. This means the prototypes input consist of photographs which show a random subject except for human expressions. As given in the preliminary explanation above, the prototype uses deep learning with a convolutional architecture. I have already mentioned the technical detail of input data in the theory part of convolutional neural networks (Section 2.2.2). Within this chapter, the Definition 2.2.2 specified the input data when it comes to training a machine learning algorithm as

> *a vector with probabilities for each image which gots predicted. Every probability stands for the likelihood to belong to a specific class.*

In addition to Definition 2.2.2, another input has to be defined. The objective of the prototype is to make predictions. Some information has to be given, to enable an algorithm to do this. The need of a second input was also explained in before (Chapter 2), as the concept of training and test data was introduced.

A problem with input data is the computation time. In a nutshell, if more data and classes are given, the computation time increases exponentially. So, it makes sense to keep the input dataset and its possible outcomes as small as possible. This relation is explained in Chapter 2). As a summary, the prototypes input data will fulfil the following requirements:

- A dataset which can be divided into a training and test set
- A dataset which consists of images with any subject except for human facial expressions
- The size of the input dataset shall not be too big, otherwise it can not be trained on a personal computer

3.3 Data processing

As mentioned in the introduction of this chapter, the prototype is useful to turn the theory into applied knowledge. The prototype exploits the introduced methods to make predictions on unseen data (test data), in particular convolutional neural networks. The principal

characteristic of the prototype is its transparency, which will be achieved by using Shapley values and the integrated gradient. So, the required process looks as follows:

1. The images (input) will be loaded
2. The input must be divided into a training and test set
3. Parameters of a convolutional neural network got trained while using the test set
4. The model will be evaluated during the test set
5. A few examples of the test set are used to show a transparent decision

This process is required to implement my prototype successfully. Figure 3.1 visualizes the processes within a circle. It means that the processes can be repeated several times, as long as the output fulfils the required form. The symbols above each step shall provide a better understanding of the process. For example, step four aims for a high accuracy. In other words, it is import that the prediction hits the right label of the image. It can be illustrated with an arrow that is supposed to hit a target. The requirements of step five, on the other hand, are less clear. It is expected that results are transparent but everyone can have a unique understanding of what transparency means. For example, in the eyes of non-technical user, transparency is something different, compared to technical user. So, this step is visualized through an eye.

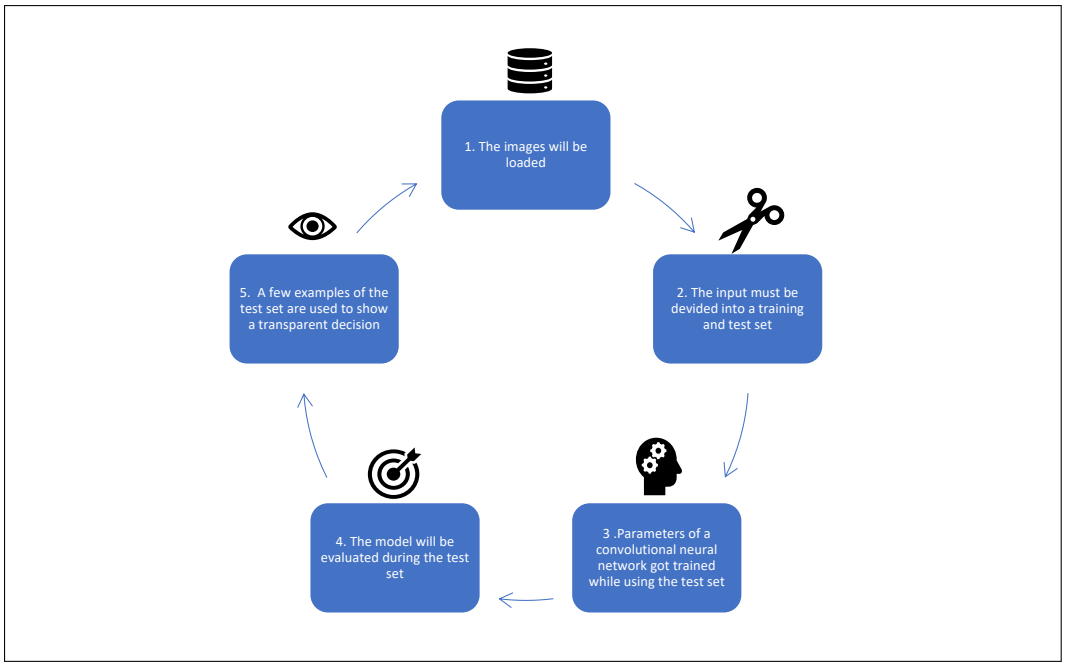

Figure (3.1): The Figure shows the required process of my prototype. Every symbol visualizes the main objective and the focus of each step. Step five points out that it is expected that the results are transparent. Everyone has a unique understanding of what transparency means. For example, in the eyes of non-technical users, transparency is different, compared to technical users. So, this step is visualized through an eye.

3.4 Output Data

The predictions were made as part of an explainable deep learning model. A classifier to make predictions on unseen data (step four of the previous section), is the first step along the road to transparent decisions. This step of making predictions on unseen data has already been achieved through the use of a convolutional neural network. How this works in general, is also described in Section 2.2.2. It defines the output as follows:

> *A vector with probabilities for each image which gots predicted. Every probability stands for the likelihood to belong to a specific class.*

Transparency is a crucial aspect of the prototype. That is why everything that explains how the algorithm came to the probabilities will be seen as a second output. The best method for this investigation is to use the expected gradient approach. Understanding the output of this method is absolutely crucial if it comes to make transparent decisions. An intuitive way to understand the output of the expected gradient approach is the following illustration:

> *The input values enter a room in random order. All input values in the room contribute to the final prediction. The Shapley value of a feature value is the average change in the prediction that the coalition which is already in the room receives, when the feature value joins them.*

Nevertheless, this intuition is not appropriate for people without an information technology background. For those people it is beneficial to visualize these values on the given input. This leads to the following requirements:

1. The prototype returns a prediction for each image in the test data set
2. The prototype returns additional values for a subset of the predictions. More precisely, a value is mapped to each input variable of the examined image. This mapping reflects its contribution to the overall result
3. The final output is an image. Therefore the output of Step 2 gets projected on the examined image. The pixels in the image should be colored in such a way that it is clearly visible how valuable the surrounding image area was for a prediction.

3.5 Evaluation

Transparency is a qualitative value. Therefore it is hard to evaluate the results. Instead, it is possible to make assumptions about how transparent and clear the results are. But without a well-designed survey, such an unjustified assumption can lead to wrong findings

and a false conclusion. To avoid this scenario, only the output as explained before gets evaluated. If the output is equal to this explanation, the transparency will be regarded as useful.

Nevertheless, there is also a quantifiable characteristic that has to be evaluated. If the accuracy (Section 2.2) on the test set is below 90% the prototype will not be useful for making predictions. Instead, the expected gradient approach can be used to investigate why the classifier fails.

Chapter 4

Functional Specifications

"Details matter. It's worth waiting to get it right."

- Steve Jobs, Found of Apple

4.1 Input Data

4.2 Scale the Data

4.3 Fit a Model

4.4 Make Predictions

4.5 Make Transparent Decisions

Chapter 5

Implementation

"What would life be if we had no courage to attempt anything?"

-Vincent Van Gogh

The first steps in the pseudocode example, which defines the functionalities of the prototype is not about Explainable Artificial Intelligence. It focuses on the development of a predictive model which uses deep learning. So first of all, the scope of this chapter is on the technical details of the prototype. The most popular deep learning models leveraged for computer vision problems are convolutional neural networks (CNN). The topic is already addressed in the theory chapter, but Figure 5.1 you can see a short overview of the most important key points.

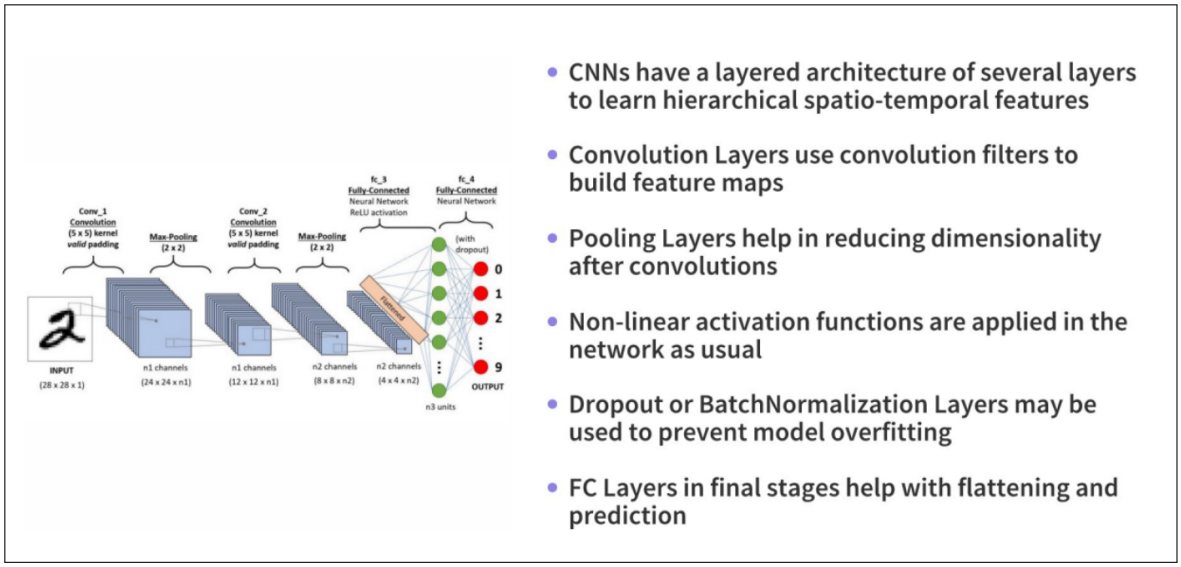

Figure (5.1): Keypoints of an example cnn architecture (VGG16)

5.1 Developing a predictive deep learning model with transfer learning

In the context of this thesis, the implementation of transparent Convolutional Neural Networks Models is the main objective. Therefore the exact architecture and the achieved accuracy of the predictions will not be explained in the tiniest detail. Furthermore, developing and training a model costs a lot of time and effort. Instead of starting from scratch, it makes sense to build on an already existing architecture, because the effort will be less huge. Besides, it saves a lot of time to use pre-trained network.

An opportunity is to leverage the power of transfer learning and retrained models. This model was trained on advanced hardware, with a tremendous amount of data and time to adjust the weights as close as possible to a massive amount of different claes. Considering this fact, the model has already learnt a robust hierarchy of features. Consequently, the model is useful for extract representations of features for over a million images to 1,000 different categories. So it can act as a feature extractor for new images suitable for a computer vision problem.

The model which is used in the prototype is the VGG16. His architecture can be seen in Figure 5.2. The name is determined by its 16 different layers which consist of Convolutional layers, max-pooling layers, activation layers and fully connected layers.

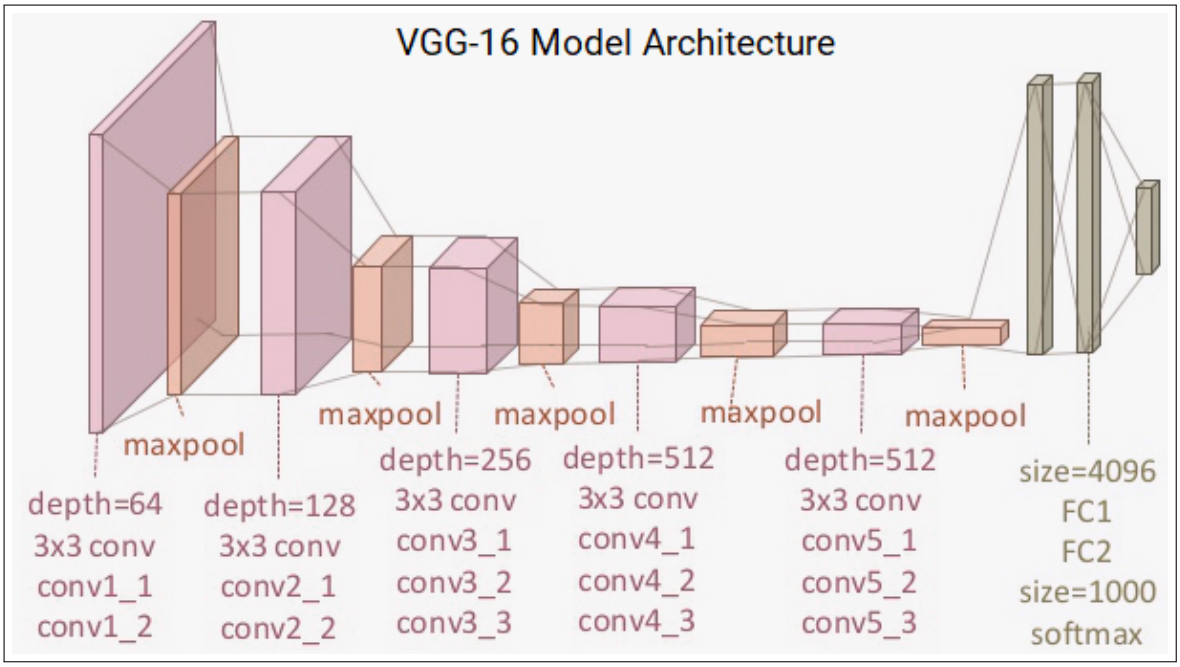

Figure (5.2): Keypoints of an example cnn architecture (VGG16)

There are 13 convolutional layers, 5 Max Pooling layers and 3 Dense layers which sum up to 21 layers but only 16 weight layers. Conv 1 has 64 filters while Conv 2 has 128 filters, Conv 3 has 256 filters while Conv 4 and Conv 5 has 512 filters. VGG-16 network is trained on ImageNet dataset, which has over 14 million images and 1000 classes, and archives 92.7% top-5 accuracy. It exceeds AlexNet (another Convolutional Neural Networks Architecture) by replacing large filters of size 11 and 5 in the first and second convolution layers with small size 3x3 filters.

Once again, the general idea behind transfer learning for image recognition is that if a model is trained (pre-trained, transfer learned) on a large and broad enough dataset, this model can be seen as a generic model approach of the real world. So it is easier to take advantage of these learned feature maps. It is much more productive instead of training such a model from scratch. To do so a massive amount of hardware resources (computation power) have to be available.

Usually, there are two approaches which can be used to customize a pre-trained model:

1. Feature Extraction: Use the representations learned by a previous network to extract important features from new samples. You add a new classifier, which will be trained from scratch, on top of the pre-trained model so that you can repurpose the feature maps learned previously for the dataset.

   There is no need to train the entire model again. The base of a convolutional network already contains generically useful features. It is therefore used to make predictions on images. However, the final, classification part of the pre-trained model is specific to the original classification task, and subsequently particular to the set of classes on which the model was trained.

2. Fine-Tuning: This is a more involved technique, where we do not just replace the final layer (for classification/regression), but we also selectively retrain some of the previous layers. Deep neural networks are highly configurable architectures with various hyperparameters. As discussed earlier, the initial layers have been seen to capture generic features, while the later ones focus more on the specific task at hand. An example is depicted in the following figure on a face-recognition problem, where initial lower layers of the network learn very generic features and the higher layers learn very task-specific features.

While keeping the functional specifications in mind, the prototype can use the machine learning framework TensorFlow. So, it can fulfil the technical requirements with state-of-the-art technology. TensorFlow is an end-to-end open-source platform for machine learning. It has a comprehensive, flexible ecosystem of tools, libraries and community resources that makes it easier to implement the prototype while not losing the scope of my focus. It will

not work to explain all details of every single software library which is used in the following. Even more, since those libraries have huge documentations on the Internet. Instead the next paragraph aim on the implementation process.

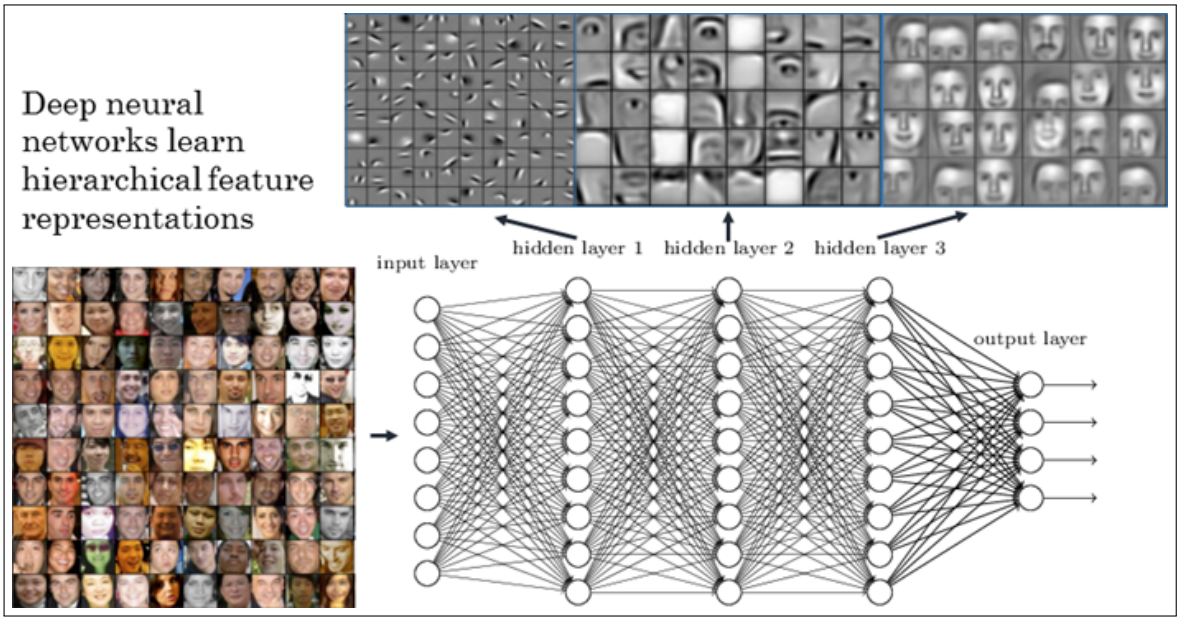

Figure (5.3): An example is depicted in the following figure on a face-recognition problem, where initial lower layers of the network learn very generic features and the higher layers learn very task-specific features.

An example of why transfer learning does work is described in Figure 5.3. The example shows that Deep Learning Systems and Models are in a layered architectures that learn different features at different layers (hierarchical representations of layered features). These layers are then finally connected to the last layer (usually a fully connected layer, in the case of supervised learning) to get the final output. This layered architecture allows utilizing a pre-trained network without its final layer as a fixed feature extractor for other tasks.

The implementation of the transfer learning model follows along with these steps:

- Examine and understand the data
- Build an input pipeline, in this case using TensorFlow ImageDataGenerator
- Compose the model
- Load in the pre-trained base model (and pretrained weights)
- Stack the classification layers on top
- Train, the model
- Evaluate model

Even though the dataset can be downloaded from kaggle.com the machine learning framework Tensor Flow provides an own class which held the datasets. This software package is called tfds and can be used direly to get the data on your local machine. Another advantage is that data can be split into a train and a test datasets. So, with the following lines of code step one is entirely solved:

```
1
2  # TensorFlow Function to laod a Dataset
3
4  (raw_train, raw_validation, raw_test), metadata = tfds.load(
5  'cats_vs_dogs',
6
7  # function which split the data into two sets,
8  # such that results on unseen data can be predicted
9
10 split=['train[:80%]', 'train[80%:90%]', 'train[90%:]'],
11 with_info=True,
12 as_supervised=True,)
```

It means the first milestones from the functional specifications are set, and step two can be set in focused now. Since the data is not just loaded onto the local machine, even more, it's split in train and test datasets a first part of the second Milestone (Data processing)is solved. Moreover, the next steps can be solved almost with the entire library and a few more dependencies, e.g. numpy-framework. A listing of the entire code and each implementation step or more detailed explanations would end in poor documentation of how the code was written. So, instead, a small summary is enough to present the most important insights from the transfer learning process.

When working with a small dataset, it is a common practice to take advantage of features learned by a model trained on a larger dataset in the same domain. This is done by instantiating the pre-trained model and adding a fully-connected classifier on top. In the following you can see the summary of the Convolutional Neural Networks model as it was before and for comparison how the summary looks like after adding the dataset-specific classifier layers on top. The pre-trained model is "frozen", and only the weights of the classifier get updated during training. In this case, the convolutional base extracted all the features associated with each image, and you just trained a classifier that determines the image class given.

```
Layer (type)                 Output Shape              Param #     Connected
                                                                   to
==================================================================
input_1 (InputLayer)         [(None, 160, 160, 3) 0
------------------------------------------------------------------
Conv1_pad (ZeroPadding2D)    (None, 161, 161, 3)  0       input_1
------------------------------------------------------------------
Conv1 (Conv2D)               (None, 80, 80, 32)   864       Conv1_pad
------------------------------------------------------------------
...
--------------------
block_5_depthwise_BN (BatchNorm (None, 20, 20, 192)  768
                                                   block_5_depthwise
------------------------------------------------------------------
block_5_depthwise_relu (ReLU)   (None, 20, 20, 192)  0
                                                   block_5_depthwise

Conv_1_bn (BatchNormalization)  (None, 5, 5, 1280)   5120    Conv_1
------------------------------------------------------------------
out_relu (ReLU)              (None, 5, 5, 1280)   0          Conv_1_bn
==================================================================
Total params: 2,257,984
Non-trainable params: 2,257,984
```

```
Layer (type)                 Output Shape              Param #
==================================================================
mobilenetv2_1.00_160 (Model) (None, 5, 5, 1280)        2257984
------------------------------------------------------------------
global_average_pooling2d (Gl (None, 1280)              0
------------------------------------------------------------------
...
------------------------------------------------------------------
dense_1 (Dense)              (None, 1024)              1311744
------------------------------------------------------------------
dropout_1 (Dropout)          (None, 1024)              0
------------------------------------------------------------------
dense_2 (Dense)              (None, 512)               524800
------------------------------------------------------------------
dropout_2 (Dropout)          (None, 512)               0
------------------------------------------------------------------
dense_3 (Dense)              (None, 1)                 513
==================================================================
Total params: 4,095,041
Trainable params: 1,837,057
Non-trainable params: 2,257,984
```

To present a complete overview of the implemented model in Figure 5.4 you can see four images which got predicted correct:

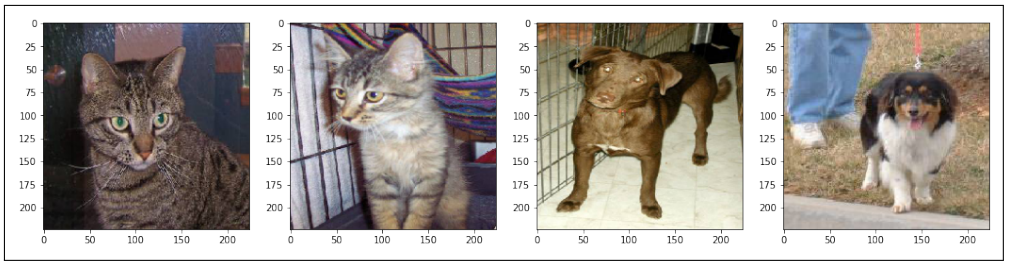

Figure (5.4): Images which got into an machine learning algorithm (CNN-VGG16 Architecture) and got predicted correct

```
[['0', 'cat'],
['0', 'cat'],
['1', 'dog'],
['1', 'dog']]
```

5.2 Making Transparent Decicions

To make the predictions transparent, the Prototype uses the shap technique. It combines different ideas from Integrated Gradient, SHapley Additive exPlanations (SHAP). This technique tries to explain model decisions using expected gradients (an extension of integrated gradients). This is a feature attribution method designed for differentiable models based on an extension of Shapley values to infinite player games. The Prototype uses the shap framework here for this technique. Integrated gradients values are a bit different from SHAP values and require a single reference value to be integrated. However, in SHAP Gradient Explainer, expected gradients reformulate the integral as an expectation and combine that expectation with sampling reference values from the background dataset. Therefore this technique uses an entire dataset as the background distribution versus just a single reference value.

After the model has been loaded, the necessary layers have been added and a set of examples have been recognized correctly, the algorithm behind the prototype calculates the attributes for every single image example. As you can see in Listing 5.1 der total amount of lines of code is pretty low and straight forward. Whereas the code itself is simple, the logic behind it is quite complex. In the Theory Chapter the fundamentals of this approach where described the following. Moreover a intuitive way to interpreted to results was presented. So, in the next chapter this intuition will be used to evaluate the results.

```
# utility function to pass inputs to specific model layers
# def map2layer(x, layer):

feed_dict = dict(zip([model.layers[0].input],
[preprocess_input(x.copy())]))

return K.get_session().run(model.layers[layer].input, feed_dict)

# focus on the 7th layer of CNN model

print(model.layers[7].input)
Out [46]:
<tf.Tensor 'block2_pool_2/MaxPool:0' shape=(1, 56, 56, 128)
dtype=float32>

# make model predictions

e = shap.GradientExplainer((model.layers[7].input,
model.layers[-1].output), map2layer(
preprocess_input(X.copy()), 7))
shap_values, indexes = e.shap_values(map2layer(to_predict, 7),
ranked_outputs=2)
index_names = np.vectorize(lambda x:
class_names[str(x)][1])(indexes)

print(index_names)

Out [47]: array([['chain', 'chain_mail'],
['great_grey_owl', 'prairie_chicken'],
['desktop_computer', 'screen'],
['Egyptian_cat', 'tabby']], dtype='<U16')

# visualize model decisions
visualize_model_decisions(shap_values=shap_values, x=to_predict,
labels=index_names, figsize=(20, 40))
```

Code Listing (5.1): asadd

Chapter 6

Evaluation

> "Small daily improvements over time lead to stunning results?"
>
> - Robin Sharma

Discussion and evaluation of accuracy are not useful since it focuses on Explainable Artificial Intelligence. Nevertheless, the train and test curves of the accuracy are documented in Figure 6.1 to give the reader a complete overview of the model. It can be seen that even the results are nor outstanding or state of the art among all the classifier it does a good job. It shall be mentioned that if a model does "a good job" or not is mostly seen from a personal perspective. Furthermore, the field in which the algorithm should be sued is important. So, whereas in prediction Cat and Dogs for fun, 97% of accuracy looks good enough. As mentioned in the Introduction, use within an Upload filter can be critical. Especially since that 3% can be still a lot. Let us take Instagram, for example. Assume that this or a similar classifier is used as an upload filter. Millions of people and thousands on Bussines realize on the Instagram Application. So, if the classifier makes 3% false decisions, this can harm many people. Maybe it is not as worse for private use, but since busies models are built on top of this application, it is critical.

As mentioned in the Expected Gradient Approach section, classical approaches as the shapely-values lead to results which are hard to interpret. Even an interpretation is a qualitative basis, and for a user with a deeper technical understanding, those values can mean something Figure 6.2 shoes how noisy those values are.

This becomes even more clear in comparison with images from the same distribution which uses the Expected Gradient Approach instead of pure shapely values

In Figure 6.3, the expected gradient values from the 7th layer of the VGG-16 CNN Model can be seen. Here it is more likely to get a conclusion of what features are essential. A possible outcome is that the pixels which show a cat or a dog are more critical to the prediction.

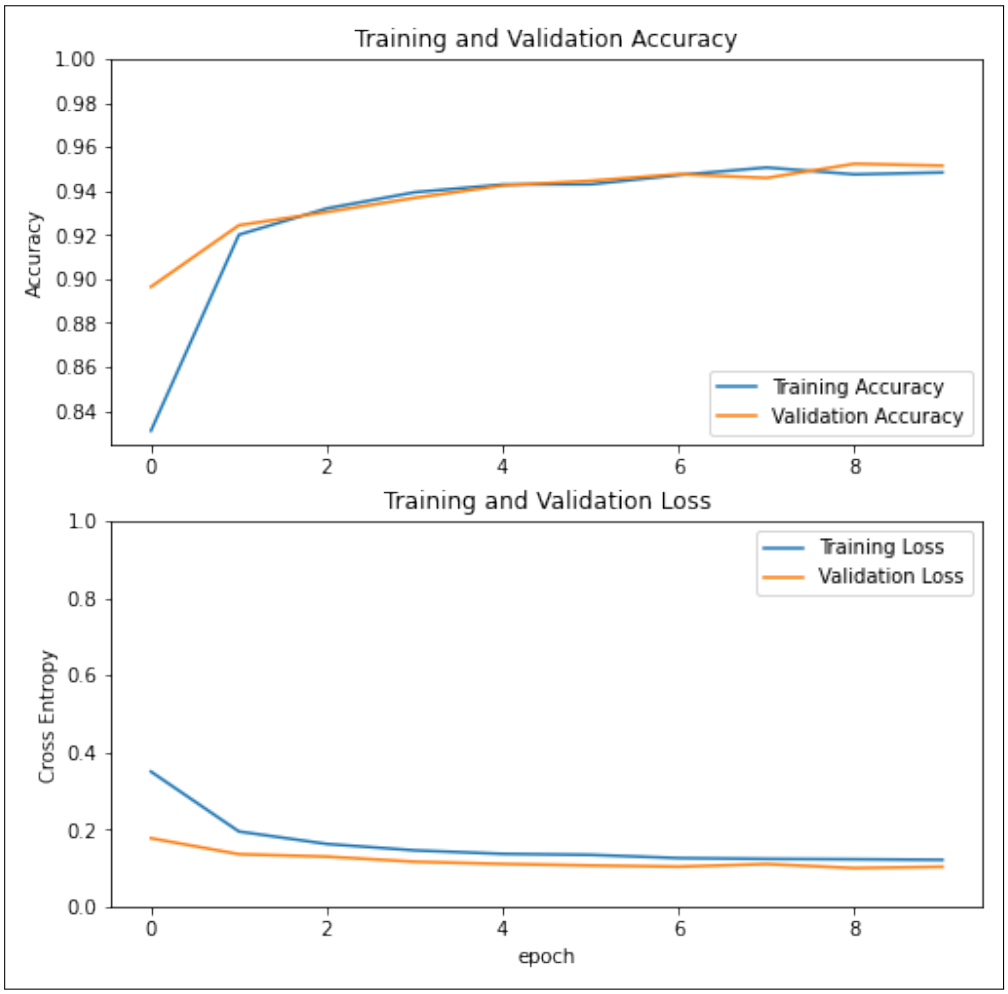

Figure (6.1): Line Plots of Loss and Accuracy Learning Curves for the Dogs and Cats Dataset (VGG16-Architecture)

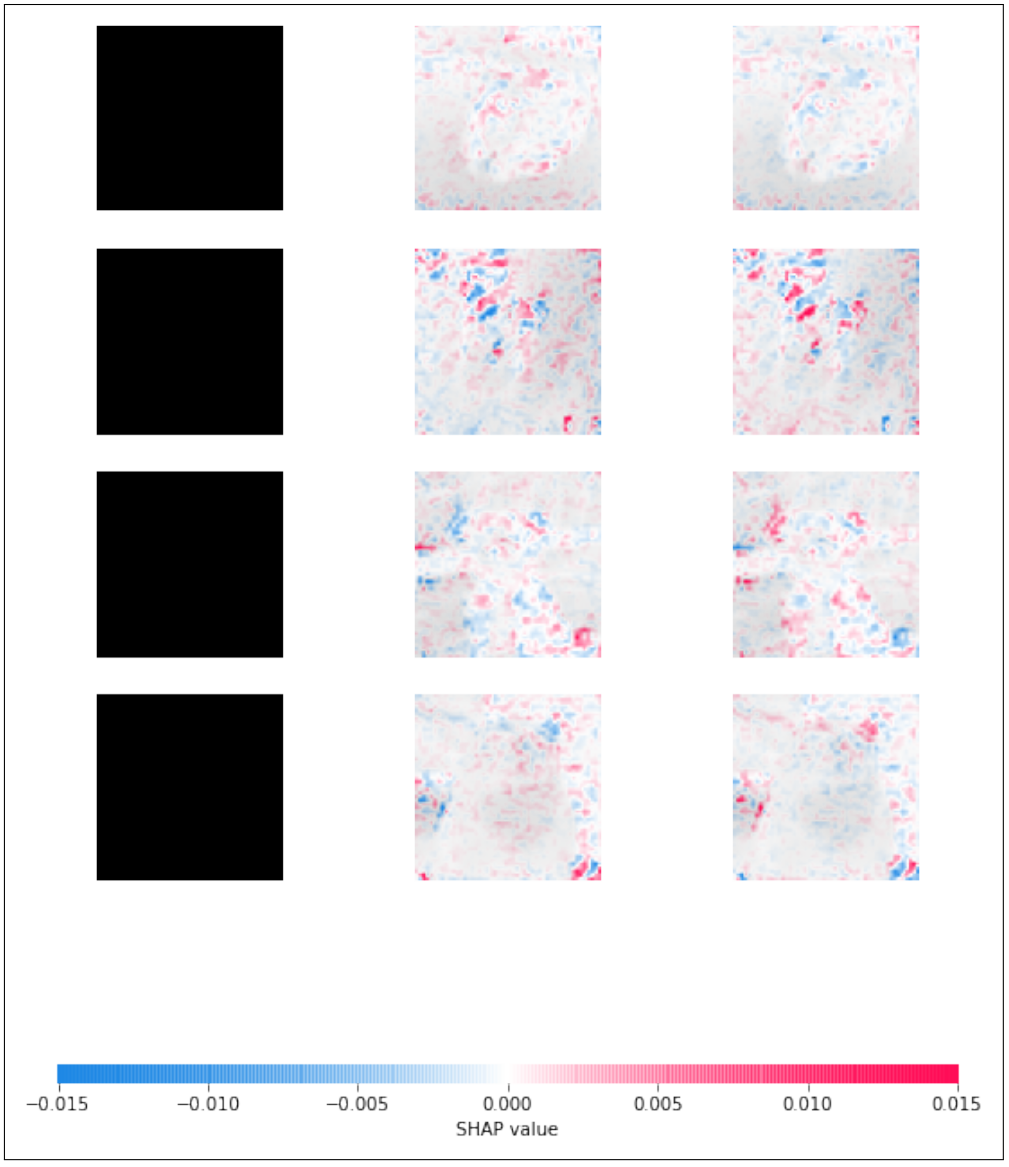

Figure (6.2): Shapley Values

So, far so good. But since the objective is to use this approach as an upload filter to help people to overthink their decisions, this is not enough.

Figure (6.3): Shapley Values

Another approach can be seen in Figure 6.4. Here the 14th layers gradients are calculated and visualized. No, a few specific cafeterias stand out from the visualization. Especially the regions around the nose and the ears seems as important for the algorithms. This interpretation came from the fact that these regions are red. It stands for a higher value and stands therefore for higher importance.

At the very end of the Theory (2) Chapter an intuitive way of how Shapley values can be interpreted was given. So, now this intuition will be used to interred the values. For a better

Figure (6.4): Shapley Values

understanding compare this interpretation with the explains in the Theory (2) Chapter. The metaphorical the pixels were replaced by chefs in a kitchen. Every chef enters the kitchen randomly such that different constellations of the chef are in the kitchen. Average increase/ decrease in the grading of meal which got prepared before and after a chef is entering the kitchen is the Shapley value. So, the overall grade would be higher if the chef (pixels) around the nose are in the kitchen. Note that this is not an absolute value (Is in or Is not in the kitchen). Moreover, the other chef, who is in the kitchen in different combinations is crucial since this is an average value.

Chapter 7

Generalization

> "The big picture doesn't just come
> from distance; it also comes from time."
>
> - Simon Sinek

Now it is time to summarize the current results and insights from the theory chapter and the chapter around the prototype to get an idea if it is possible to use AI as a transparent Upload Filter.

I think no matter what companies say, Artifical Intelligence can not replace humans in their role of a moderator for online contend. Most companies use a combination of automatic systems and human resources to check the content from users. Even the first part (automated systems) can be replaced by deep learning algorithms to make a forecast of which content humans should observe. First, it still is up to human employees to check uploaded content. The next few insights from this thesis are the main reasons this is true.

- Even the results in usual image recognition are outstanding have a high accuracy ($>$ 95%) the expiations make this technology an easy target for special cases. It can harm many users while running their business on social Networks.
- The hardware resources which were needed to train those results are huge. This leads to the fact that if a real-world application wants to cover every single class, there must be tremendous resources. In a Nutshell, it costs a lot, and human resources are cheaper in most cases. Especially since outsourcing is pretty easy because a high-speed internet connection is available in almost every single country.
- Even the results from the prototype have shown that it is possible to make deep learning models more transparent it still costs a lot of effort to describe its interpretation.

If we got back to the start-up company which got introduced as a use-case scenario, the team would maybe conclude that the technique is too expensive to use it as a complete replacement. Even more than the method is still a topic which needs final research to make

interpretation more useful.

Instead, the company could think of an implementation which uses deep learning to help with moderation. For example, pushing suspect content (as cats which got uploaded to the platform) to a human moderator. Even more, if real-world examples would replace the use-case scenario a classifier which sends every image which could show offensive content (guns, nudity) to a moderator can save a tremendous amount of time and effort.

But what has all this to do with Explainable Artificial Intelligence? Let's take the example of a cat image which looks very close to a dog or vice versa. Then it can help to give the moderator a hint what the machine learning model has thought. Or better on what feature it had a closer look before it decides for or against to assign a label to an image.

Chapter 8

Conclusion

> "It is beyond a doubt that all our knowledge begins with experience."
>
> - Immanuel Kant

Even the opportunities are great which were provided by explainable Deep Learning Models, an upload filter as it is mentioned in the introduction is still not possible. More research is necessary to ensure a higher interpretability. Even though the prototype leaves much room for more advanced and clear Explainable Artificial Intelligence Models. Moreover a realistic scenario with more classes provides a much petter picture of how far can transparency be developed.

Modern technique companies face a tough challenge since the amount of data which got uploaded to there platforms got more and more. So it is in their response to check if new technologies can help us and to save our privacy while not harming the business opportunities which get established in the last years.

Instead of implementing new algorithms, those companies should at least try if Explainable Artificial Intelligence, Deep Learning to explain their algorithms. Even though, more research is needed to give an intuitive interpretation of deep learning decision boundaries this thesis had shown that with just a few lines of code, a deep learning model could be made more transparent.

I hope that I can be part of development of more transparent Deep Learning Models and so I will focus at least a few more times on this topic, while I'm participating at the master in computer science. My goal is to implement even more of those methods to find out if they could help humans to understand deep learning models. So, probably we can all participate in the significant opportunities which were provided from those machine learning algorithms.

List of Figures

List of Listings

References

[Aas et al., 2019] Aas, K., Jullum, M., and Løland, A. (2019). Explaining individual predictions when features are dependent: More accurate approximations to shapley values. cite arxiv:1903.10464.

[Bohannon, 2016] Bohannon, J. (2016). Who's the Michael Jordan of computer science? New tool ranks researchers' influence. *Science*.

[Britz, 2015] Britz, D. (2015). Implementing a Neural Network from Scratch in Python – An Introduction – WildML.

[Brownlee, 2019] Brownlee, J. (2019). *Deep Learning for Computer Vision: Image Classification, Object Detection, and Face Recognition in Python*. Machine Learning Mastery.

[Chollet, 2016] Chollet, F. (2016). Building powerful image classification models using very little data. https://blog.keras.io/building-powerful-image-classification-models-using-very-little-data.html. (Accessed on 05/01/2020).

[COPELAND, 2017] COPELAND, M. V. (2017). DEEP LEARNING EXPLAINED WHAT IT IS, AND HOW IT CAN DELIVER BUSINESS VALUE TO YOUR ORGANIZATION CHAPTER 1 |. Available at https://www.nvidia.com/content/dam/en-zz/Solutions/deep-learning/home/DeepLearning_eBook_FINAL.pdf.

[Europarl, 2017] Europarl, M. (2017). Copyright directive: statement by Axel VOSS (EPP,DE), rapporteur - Multimedia Centre. Available at https://multimedia.europarl.europa.eu/en/copyright-directive-statement-by-axel-voss-eppde-rapporteur-{_}I158298-V{_}v.

[Gallwitz, 2019] Gallwitz, F. (2019). EU-Urheberrechtsreform: Experten zu Upload-Filtern | Science Media Center Germany. Available at https://www.sciencemediacenter.de/alle-angebote/rapid-reaction/details/news/eu-urheberrechtsreform-experten-zu-upload-filtern/?fbclid=IwAR1E0AqgLK8566U8VkGBo6e6YcV2QBZn5L4d3rb1dib{_}ik9T7iJk0augXzs.

[Goodfellow et al., 2016] Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press. http://www.deeplearningbook.org.

[Gross, 2017] Gross, D. (2017). 2. Was leisten die Domänen der Künstlichen Intelligenz? Available at https://www.sigs-datacom.de/ots/2017/ki/2-was-leisten-die-domaenen-der-kuenstlichen-intelligenz.html.

[Gunning, 2019] Gunning, D. (2019). Darpa's explainable artificial intelligence program. *AI Magazine*, 40(2):44–58.

[Jordan and Mitchell, 2015] Jordan, M. I. and Mitchell, T. M. (2015). Machine learning: Trends, perspectives, and prospects. *Science*, 349(6245):255–260.

[Knight, 2019] Knight, W. (2019). The Dark Secret at the Heart of AI - MIT Technology Review. Available at https://www.technologyreview.com/s/604087/the-dark-secret-at-the-heart-of-ai/.

[Kohlhase et al., 2017] Kohlhase, M., Koprucki, T., Müller, D., and Tabelow, K. (2017). Mathematical models as research data via flexiformal theory graphs. In Geuvers, H., England, M., Hasan, O., Rabe, F., and Teschke, O., editors, *CICM*, volume 10383 of *Lecture Notes in Computer Science*, pages 224–238. Springer.

[Kriesel, 2007] Kriesel, D. (2007). *A Brief Introduction to Neural Networks*. Available at http://www.dkriesel.co.

[Kuang, 2017] Kuang, C. (2017). Can A.I. Be Taught to Explain Itself? - The New York Times. Available at https://www.nytimes.com/2017/11/21/magazine/can-ai-be-taught-to-explain-itself.html.

[Lundberg and Lee, 2017] Lundberg, S. and Lee, S. (2017). A unified approach to interpreting model predictions. *CoRR*, abs/1705.07874.

[Lundberg et al., 2018] Lundberg, S. M., Erion, G. G., and Lee, S. (2018). Consistent individualized feature attribution for tree ensembles. *CoRR*, abs/1802.03888.

[Michael Jordan, 2018] Michael Jordan (2018). Michael I. Jordan Interview: Clarity of Thought on AI. Available at https://medium.com/syncedreview/michael-i-jordan-interview-clarity-of-thought-on-ai-ed936d0dc421.

[Microsoft, 2013] Microsoft (2013). PhotoDNA | Microsoft. Available at https://www.microsoft.com/en-us/photodna.

[Mitchell, 1997] Mitchell, T. M. (1997). *Machine Learning*. McGraw-Hill, New York.

[Molnar, 2019] Molnar, C. (2019). *Interpretable Machine Learning*. https://christophm.github.io/interpretable-ml-book/.

[Mudrakarta et al., 2018] Mudrakarta, P. K., Taly, A., Sundararajan, M., and Dhamdhere, K. (2018). Did the model understand the question? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1896–1906, Melbourne, Australia. Association for Computational Linguistics.

[Murphy, 2012] Murphy, K. P. (2012). *Machine Learning: A Probabilistic Perspective*. The MIT Press.

[Ng, 2008] Ng, A. (2008). Stanford cs229 - machine learning - andrew ng.

[Reda, 2019] Reda, J. (2019). Julia Reda – Upload filters. Available at https://juliareda.eu/eu-copyright-reform/censorship-machines/.

[Robinson, 2017] Robinson, D. (2017). You better explain yourself, mister: DARPA's mission to make an accountable AI • The Register. Available at https://www.theregister.co.uk/2017/09/28/inside{_}explainable{_}ai/.

[Rocke, 2019] Rocke, A. (2019). The true cost of alphago zero. https://keplerlounge.com/artificial/intelligence/2019/03/24/alpha-go-zero.html. (Accessed on 05/01/2020).

[Russell and Norvig, 2009] Russell, S. and Norvig, P. (2009). *Artificial Intelligence: A Modern Approach*. Prentice Hall Press, USA, 3rd edition.

[Samek et al., 2017] Samek, W., Wiegand, T., and Müller, K. (2017). Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models. *CoRR*, abs/1708.08296.

[Samuel, 1959] Samuel, A. L. (1959). Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development*, 3:210–229.

[Slundberg, 01] Slundberg (01). shaoshanglqy/shap-shapley. https://github.com/shaoshanglqy/shap-shapley. (Accessed on 05/01/2020).

[Spoerri, 2019] Spoerri, T. (2019). On Upload-Filters and other Competitive Advantages for Big Tech Companies under Article 17 of the Directive on Copyright in the Digital Single Market — jipitec. Available at https://www.jipitec.eu/issues/jipitec-10-2-2019/4914.

[Stewart, 2019] Stewart, M. (2019). Simple introduction to convolutional neural networks. https://towardsdatascience.com/simple-introduction-to-convolutional-neural-networks-cdf8d3077bac. (Accessed on 05/01/2020).

[Sundararajan et al., 2017] Sundararajan, M., Taly, A., and Yan, Q. (2017). Axiomatic attribution for deep networks. *CoRR*, abs/1703.01365.

[Tagesspiegel, 2019] Tagesspiegel, D. (2019). Protest gegen upload-filter: Aufstand der generation youtube - medien - gesellschaft - tagesspiegel. https://www.tagesspiegel.de/gesellschaft/medien/protest-gegen-uploadfilter-aufstand-der-generation-youtube/24082412.html. (Accessed on 05/01/2020).

[Tjoa and Guan, 2019] Tjoa, E. and Guan, C. (2019). A survey on explainable artificial intelligence (XAI): towards medical XAI. *CoRR*, abs/1907.07374.

[Trask et al., 2015] Trask, A., Gilmore, D., and Russell, M. (2015). Modeling order in neural word embeddings at scale. *CoRR*, abs/1506.02338.

[Vasudev, 2019] Vasudev, R. (2019). Understanding and Calculating the number of Parameters in Convolution Neural Networks (CNNs).

[Versloot, 2019] Versloot, C. (2019). Visualizing keras cnn attention: Grad-cam class activation maps – machinecurve. https://www.machinecurve.com/index.php/2019/11/28/visualizing-keras-cnn-attention-grad-cam-class-activation-maps/#. (Accessed on 05/01/2020).

[Wagner, 1983] Wagner, N. R. (1983). Fingerprinting. In *1983 IEEE Symposium on Security and Privacy*, pages 18–18.

[Waltermann and Hess, 2019] Waltermann, H.-M. and Hess, T. (2019). Upload-filter für content. *MedienWirtschaft*, 16(2):16–21.

[Waters, 2017] Waters, R. (2017). Intelligent machines are asked to explain how their minds work | Financial Times. Available at https://www.ft.com/content/92e3f296-646c-11e7-8526-7b38dcaef614.

[Weitz, 2018] Weitz, K. (2018). Available at http://www.cogsys.wiai.uni-bamberg.de/theses/weitz/Masterarbeit_Weitz.pdf.

[Wikipedia, 2020] Wikipedia (2020). Gradient descent - wikipedia. https://en.wikipedia.org/wiki/Gradient_descent. (Accessed on 05/01/2020).

[Woollacott, 2019] Woollacott, E. (2019). EU Copyright Directive Passed - Upload Filters And All. Available at https://www.forbes.com/sites/emmawoollacott/2019/03/26/eu-copyright-directive-passed-upload-filters-and-all.

[Yao et al., 2017] Yao, Y., Xiao, Z., Wang, B., Viswanath, B., Zheng, H., and Zhao, B. (2017). Complexity vs. performance: empirical analysis of machine learning as a service. pages 384–397.

[YouTube, 2010] YouTube (2010). How Content ID works - YouTube Help. Available at https://support.google.com/youtube/answer/2797370?hl=en.

List of Definitions

Glossary

Activation Functions Activation functions are mathematical equations that determine the output of a neural network. The function is attached to each neuron in the network, and determines whether it should be activated (“fired”) or not, based on whether each neuron’s input is relevant for the model’s prediction. . 14, 26

Artifical Intelligence Deep learning is part of a broader family of machine learning methods based on artificial neural networks with representation learning. Learning can be supervised, semi-supervised or unsupervised. 9, 60, 63

Artificial Neural Network Artificial neural networks or connectionist systems are computing systems vaguely inspired by the biological neural networks that constitute animal brains. Such systems "learn" to perform tasks by considering examples, generally without being programmed with task-specific rules.. 17, 18, 27

Backward Propagation Second step while callculation an Machine Learning Algorithm which callculates the gradients of the weights in order to optimize the error value (Forward Propagation) . 15, 34

Bias Bias is just like an intercept added in a linear equation. It is an additional parameter in the Neural Network which is used to adjust the output along with the weighted sum of the inputs to the neuron. Moreover, bias value allows you to shift the activation function to either right or left. . 14

Classification Task Classification is the process of predicting the class of given data points. Classes are sometimes called as targets/ labels or categories. Classification predictive modeling is the task of approximating a mapping function (f) from input variables (X) to discrete output variables (y). . 11

Convolutional Neural Networks In deep learning, a convolutional neural network is a class of deep neural networks, most commonly applied to analyzing visual imagery. They are also known as shift invariant or space invariant artificial neural networks, based on their shared-weights architecture and translation invariance characteristics.. 21, 23, 47, 48, 50

Cost Function Sum of all single values which came from a Loss Function. 10, 13, 15, 19

Deep Learning Deep learning is part of a broader family of machine learning methods based on artificial neural networks with representation learning. Learning can be supervised, semi-supervised or unsupervised. iii, 4, 5, 10, 28, 49, 62, 63

Explainable Artificial Intelligence Explainable AI refers to methods and techniques in the application of artificial intelligence technology such that the results of the solution can be understood by human experts. . iii, 3–5, 7, 8, 29, 30, 34, 46, 54, 61, 62, 64

Forward Propagation First step while callculation an Machine Learning Algorithm which callculates the losses of the single outputs and sum them up to a final error value . 15, 72

Graphical Processing Units A graphics processing unit is a specialized electronic circuit designed to rapidly manipulate and alter memory to accelerate the creation of images in a frame buffer intended for output to a display device. GPUs are used in embedded systems, mobile phones, personal computers, workstations, and game consoles. . 10

Logistic Regression In statistics, the logistic model is used to model the probability of a certain class or event existing such as pass/fail, win/lose, alive/dead or healthy/sick. This can be extended to model several classes of events such as determining whether an image contains a cat, dog, lion, etc.. 12

Loss Function In mathematical optimization and decision theory, a loss function or cost function is a function that maps an event or values of one or more variables onto a real number intuitively representing some "cost" associated with the event.. 15, 72

Machine Learning Machine learning is the scientific study of algorithms and statistical models that computer systems use to perform a specific task without using explicit instructions, relying on patterns and inference instead. It is seen as a subset of artificial intelligence. . iii, 2–4, 7–9, 15, 29, 73

Machine Learning Algorithm see Machine Learning. . 3, 7, 8, 12, 33–35, 65, 72, 73

Pooling Layer A pooling layer is another building block of a CNN. Its function is to progressively reduce the spatial size of the representation to reduce the amount of parameters and computation in the network. . 26

Pooling Layer n the context of artificial neural networks, the rectifier is an activation function defined as the positive part of its argument: where x is the input to a neuron. This is also known as a ramp function and is analogous to half-wave rectification in electrical engineering.. 26

Preprocessing In any Machine Learning process, Data Preprocessing is that step in which the data gets transformed, or Encoded, to bring it to such a state that now the machine can easily parse it. In other words, the features of the data can now be easily interpreted by the algorithm. . 12

Sigmoid Function A sigmoid function is a mathematical function having a characteristic "S"-shaped curve or sigmoid curve. A common example of a sigmoid function is the logistic function shown in the first figure and defined by the formula: Other standard sigmoid functions are given in the Examples section. 14

Supervised Learning see Supervised Machine Learning Algorithm. . 11

Supervised Machine Learning Algorithm Supervised learning is the machine learning task of learning a function that maps an input to an output based on example input-output pairs. . 11, 13, 74

Training Data The training data is an initial set of data used to help a program understand how to apply technologies like neural networks to learn and produce sophisticated results. It may be complemented by subsequent sets of data called validation and testing sets. . 11, 12

Upload Filter An upload filter is server-side software that checks files during upload, rejects them if necessary, changes them or initiates other measures. . iii, 1, 2, 4, 5, 10, 28, 63